Actes de la 22e Conférence Nationale en Intelligence Artificielle

Jérôme Lang

AfIA

Association française
pour l'Intelligence Artificielle

CNIA

Conférence Nationale en Intelligence Artificielle

PFIA 2019

Table des matières

Éditorial

Ce siècle a dix-neuf ans. Toulouse était tranquille,
La chaleur s'abattait lourdement sur la ville,
Ni canal ni Garonne n'avaient l'humeur à rire
Ni à pleurer d'ailleurs, ç'aurait pu être pire.

Et soudain, tout s'agite : on voit de toutes parts
Accourir des chercheurs de France et de Navarre,
Sous le bras un poster, dans leur sac des reliures,
Avançant d'un bon pas vers la Manufacture.

Sont-ce des spécialistes de culture du pastel ?
Non : ces gens sont venus pour une conférence
Qui porte, m'a-t-on dit, sur les intelligences
Qui sont, le croirez-vous, toutes artificielles.

Dans la Manufacture bruissent tous les salons,
De problèmes bien durs de planification,
De représentation et de raisonnement,
D'apprentissage et de systèmes multi-agents.

Devant un auditoire curieux et assidu,
Pas moins de quinze articles y seront débattus !
Car ces grands scientifiques, ils sont ce que nous sommes :
Ils peuvent se tromper comme toutes femmes et hommes.

Jérôme Lang

Comité de programme

Présidents

- Jérôme Lang

Membres

- Nathalie Aussenac
- Meghyn Bienvenu
- Tristan Cazenave
- Séverine Dubuisson
- Hélène Fargier
- Béatrice Fuchs
- Andreas Herzig
- Frédéric Koriche
- Pierre Marquis
- Nicolas Maudet
- Marie-Laure Mugnier
- Philippe Muller
- Christine Solnon
- Serena Villata
- Bruno Zanuttini

Nouvelle distance de graphe pour la reconnaissance d'objets 3D déformables basée sur la décomposition en étoiles-triangles *

Kamel Madi [1, 3] Eric Paquet [2] Hamamache Kheddouci [1]

[1] Université de Lyon, CNRS, Université Lyon 1, LIRIS, UMR5205, F-69622, France
[2] Conseil national de recherches, Ottawa, Canada
[3] Umanis, Levallois-Perret, 92300, France

kamel.madi@liris.cnrs.fr

Résumé

Nous traitons le problème de la comparaison des objets 3D déformables représentés par des graphes tels que les tessellations triangulaires. Nous proposons une nouvelle technique d'appariement de graphes pour mesurer la distance entre ces graphes. L'approche proposée est basée sur une nouvelle décomposition de tessellations triangulaires en étoiles-triangles. L'algorithme garantit un nombre minimum d'étoiles-triangles disjointes, offre une meilleur dissimilarité en couvrant un voisinage plus large et permet la création de descripteurs invariants face aux déformations les plus courantes. L'approche proposée est basée sur une approximation de la distance d'édition de graphes, qui est tolérante aux bruit et à la distorsion, ce qui rend notre technique particulièrement adaptée à la comparaison des objets déformables. La classification est effectuée en utilisant des techniques d'apprentissage automatique supervisées. Notre approche définit un espace de métrique en utilisant des techniques de plongement et de noyaux de graphes. Il est prouvé que la distance proposée est pseudo-métrique. Sa complexité temporelle est déterminée et la méthode est évaluée en utilisant des bases de données de référence. Nos résultats expérimentaux confirment les performances et l'exactitude de notre système.

Mots Clés

Appariement de graphes, Distance d'édition de graphes, Décomposition de graphes, Plongement de graphes, Métrique de graphes, Classification de graphes, Reconnaissance de motifs, Reconnaissance d'objets 3D, Reconnaissance d'objets déformables, Apprentissage de métrique.

1 Introduction

La comparaison des objets 3D est l'une des tâches les plus importantes en vision artificielle. Les objets représentés par des graphes tels que des maillages triangulaires, peuvent être comparés en utilisant des techniques d'appariement de graphes. Dans un graphe, les propriétés sont associées à des sommets tandis que les relations sont représentées par des arêtes. Les sommets, les arêtes et leurs attributs sont spécifiés en fonction de l'application sous-jacente ; par exemple, les sommets peuvent représenter des points, des régions d'intérêt ou toutes autres sous-structures obtenues en appliquant un processus de réduction de données tel que la segmentation. Les arêtes sont associés à la connectivité entre sommets, définissant ainsi une relation topologique entre eux, tels que la proximité, l'adjacence, etc. L'appariement de graphes est le processus permettant de trouver une correspondance entre les sommets et les arêtes de deux graphes qui satisfait un ensemble de contraintes, garantissant que les sous-structures d'un des graphes correspond à des sous-structures similaires dans l'autre graphe. Plusieurs approches ont été proposées pour résoudre le problème d'appariement de graphes [2, 3]. La distance d'édition de graphes est l'une des mesures les plus célèbres pour déterminer une telle distance [4]. Elle est définie comme étant le coût minimal de séquences d'opérations d'éditions qui transforment un graphe en un autre. La tolérance au bruit est l'un des avantages de la distance d'édition. Malheureusement, cette dernière a une complexité très élevée qui augmente exponentiellement en fonction du nombre de sommets [5]. Dans cet article, nous abordons le problème de la comparaison des objets 3D déformables. Les objets sont représentés par des graphes. Nous proposons une nouvelle distance pour comparer des objets 3D déformables. Cette distance est basée sur la décomposition de tessellations triangulaires en un ensemble de nouvelles sous-structures appelées *étoiles-triangles*. Une étoile-triangles est une composante connexe formée par l'union d'un triangle et de l'ensemble des triangles voisins, en fonction du degré de voisinage considéré. La décomposition proposée offre une représentation paramétrable des étoiles-triangles, laquelle est déterminée en fonction du degré de voisinage attribué (Définition 4). Le nombre d'étoiles-triangles résultantes est bien inférieur au nombre de sommets et au nombre d'étoiles classiques [6]. Par conséquent, la com-

*Cet article est une adaptation en langue française de notre article "*New graph distance for deformable 3D objects recognition based on triangle-stars decomposition*" [1] publié le 28 janvier 2019 dans la revue internationale *Pattern Recognition*

plexité temporelle est réduite. La mesure de dissimilarité proposée assure une meilleure approximation de la distance d'édition de graphes. En effet, considérer des structures en étoiles-triangles permet de couvrir un voisinage plus large en plus de conférer un descripteur local plus riche comparé aux étoiles classiques [6]. Avec la distance proposée, il est possible de construire un ensemble de descripteurs qui soient invariants et robustes face aux déformations les plus courantes. La classification est réalisée en utilisant des techniques d'apprentissage supervisées. Notre approche définit un espace de métriques décrivant les différents objets. Cinq classificateurs ont été utilisés, à savoir : le classificateur naïf de Bayes, le classificateur des forêts aléatoires, l'approche des arbres boostés par gradient, le classificateur à vaste marge (SVM) et la régression logistique. Le reste de l'article est organisé comme suit : dans la Section 2, nous passons brièvement en revue certains travaux connexes. La décomposition proposée est décrite dans la Section 3 tandis que la distance proposée est introduite dans la Section 4. Dans la Section 5, nous présentons et discutons nos résultats expérimentaux en les comparant avec des algorithmes de comparaison d'objets déformables, pour une base de données de référence. Enfin, la Section 6 conclut l'article.

2 État de l'art

Dans cette section, nous passons brièvement en revue certaines méthodes de reconnaissance des objets 3D. Pour plus de détails, nous invitons les lecteurs à s'orienter vers [7] pour les méthodes de reconnaissance des objets 3D et vers [5] pour les techniques de reconnaissance de formes et d'appariement de graphes. Nous présentons ensuite, brièvement, les algorithmes avec lesquels nous comparons notre approche.

Les techniques de comparaison des objets 3D peuvent être divisées en trois grandes classes [7] : les méthodes à base de caractéristiques, les méthodes à base de graphes et les autres méthodes. Concernant les méthodes à base de caractéristiques, les objets sont comparés en utilisant des caractéristiques associées à leurs propriétés géométriques et topologiques. Ces caractéristiques peuvent être des cartes globales, locales ou spatiales [8]. Les méthodes à base de graphes sont des outils puissants pour établir des correspondances entre des objets ainsi que pour produire des descripteurs invariants. Selon le type de graphe considéré, plusieurs techniques à base de graphes ont été proposées pour la comparaison des objets 3D [7]. Par exemple, dans certaines approches, les formes sont réduites à des squelettes par un processus d'amincissement [9]. Les squelettes ainsi obtenus sont comparés en utilisant des techniques d'appariement de graphes. D'autres approches s'appuient sur des graphes de Reeb, construits à partir des fonctions de correspondances définies sur les variétés correspondant aux formes [10]. Diverses techniques de segmentation ont été proposées dans la littérature, dans lesquelles les formes sont segmentées en un ensemble fini de composantes à partir desquelles un graphe est construit. Ces graphes peuvent être comparés en utilisant des techniques d'appariement de graphes [11]. D'autres méthodes ont été proposées [7], telles que : la similarité à base des vues [12], la similarité à base des erreurs volumétriques [13] et la similarité à partir d'un ensemble de points pondérés [14].

Nous comparons l'approche proposée avec divers algorithmes de l'état de l'art associés aux compétitions SHREC. Ces algorithmes correspondent à la base de données utilisée pour leurs évaluations, à savoir la base de données TOSCA [15]. Par conséquent, notre méthode a été comparée aux algorithmes suivants : **CAM** [16] : une approche dans laquelle les surfaces sont représentées par des courbes 3D extraites autour de points caractéristiques. **GeodesicD2** : Une description globale consistant en la distribution des distances géodésiques associées à une forme 3D donnée [17]. **DSR** [18] : le vecteur de caractéristiques hybride est une combinaison de deux descripteurs basés sur les vues : un tampon de profondeur pour la silhouette et une fonction radiale d'étendue. **RSH** [19] : L'approche par rayons avec représentation harmonique sphérique est une méthode qui aligne les modèles sur une position canonique, détermine les étendues maximales à laquelle elle applique une décomposition en harmoniques sphériques. **TD** [20] : le descripteur de distribution de température est un descripteur de forme basé sur le noyau de chaleur. La norme L2 est utilisée pour évaluer la distance entre les descripteurs. **Shape-DNA** [21] : Shape-DNA est une empreinte numérique obtenue en évaluant les valeurs propres de l'opérateur de Laplace-Beltrami associé à la variété. La correspondance entre deux objets est obtenue en comparant leurs valeurs propres respectives. **SRCP-TD** [22] : Le SRCP-TD est une méthode basée sur une représentation fragmentée d'un noyau de chaleur invariant par rapport à l'échelle. Les auteurs utilisent les fonctions propres de Laplace-Beltrami pour détecter des points critiques sur la variété. Le descripteur est construit à partir des valeurs du noyau de chaleur sur ces points. Une représentation fragmentée est utilisée pour réduire la dimensionnalité du descripteur.

3 Une nouvelle approche de décomposition en étoiles-triangles

Dans cette section, nous proposons une nouvelle décomposition de maillages triangulaires en composantes connexes que nous appelons *étoiles-triangles*. Cette décomposition a comme objectif de réduire le nombre de composants tout en couvrant un voisinage plus large. L'étendue du voisinage associé à une étoile-triangles est déterminée par son ordre N_k. A partir de cette représentation, il est possible de définir un ensemble de descripteurs qui soient invariants et robustes aux déformations les plus courantes. Un ordre total strict sur les triangles doit être établi avant de procéder à la décomposition. Cet ordre vise à réduire le nombre

d'étoiles-triangles générées tout en garantissant l'unicité de la décomposition.

3.1 Étoiles-triangles

Nous proposons de décomposer des graphes tels que des tessellations triangulaires en un ensemble de composants connectés que nous appelons étoiles-triangles (TS). Les étoiles-triangles sont définies comme suit :

Définition 1 (Voisinage d'un triangle) Deux triangles sont voisins s'ils partagent au moins un sommet commun. Soient t_1 et t_2 deux triangles et $V(t_1)$ et $V(t_2)$ leurs sommets respectifs. t_1 et t_2 sont $voisins \Leftrightarrow \|V(t_1) \cap V(t_2)\| > 0$. En d'autres termes, le voisinage (N) d'un triangle t est constitué de tous les triangles partageant au moins un sommet commun avec t.

Définition 2 (N_k-voisinage d'un triangle) Deux triangles t_0 et t_k sont N_k-voisins si entre t_0 et t_k il y a, au plus, une chaîne de $(k-1)$ triangles distincts, qui sont consécutivement deux à deux voisins. Formellement, t_0 et t_k sont N_k-$voisins \Leftrightarrow \exists\ t_{i=1...k-1}$ où : $\forall\ i \in 1...(k-1),\ \ t_i\ et\ t_{i+1}\ \ sont\ voisins$. Dans le cas de $k = 1$, le N_k-voisinage est réduit à un voisinage simple (Définition 1).

Définition 3 (Étoile-triangles) Une étoile-triangles ts est un sous-graphe étiqueté, défini par un triangle et l'ensemble formé par ses voisins. Formellement, une étoile-triangles ts est un triplet $ts = (t_r, T', \theta)$, où : t_r est le triangle racine, T' est l'ensemble des triangles voisins et $\theta : T \rightarrow L_T$ est la fonction d'étiquetage des triangles, tandis que L_T représente l'ensemble des étiquettes.

Définition 4 (N_k-étoile-triangles) Une N_k-étoile-triangles N_k-ts est une étoile-triangles définie par un triangle et l'ensemble de ses N_k voisins. Dans le cas de $k = 1$, la N_k-étoile-triangles est une simple étoile-triangles (Définition 3).

Les caractéristiques de l'étoile-triangles un six-uplet $t_j = (v_1, v_2, v_3,\ e_1, e_2, e_3)$ est associé à chaque triangle t_j. Les sommets v_i sont étiquetés par leurs coordonnées cartésiennes respectives $v_i = (x, y, z)$, tandis que les arêtes $e_k = (v_p, v_w)$ sont étiquetées (pondérées) avec la distance euclidienne entre leurs sommets respectifs (v_p, v_w). Les triangles sont étiquetés avec un triplet $t_j = (id, Area, Perimeter)$, où id est un nombre. Chaque étoile-triangles est caractérisée par un ensemble de descripteurs permettant d'évaluer la dissimilarité entre étoiles-triangles. Nous considérons les descripteurs suivants : la surface de l'étoile-triangles, le périmètre de l'étoile-triangles, la surface des triangles formant les étoiles-triangles, leurs périmètres, les poids associés à leurs arêtes et les degrés de leurs sommets. Notre choix de descripteurs est justifié par le fait que ces quantités sont invariantes face aux déformations les plus courantes.

La représentation en vecteurs des étoiles-triangles Un vecteur est associé à chaque étoile-triangles. Ce vecteur comprend la surface globale AG et le périmètre global PG d'une étoile-triangles, la surface A et le périmètre P de chaque triangle appartenant à l'étoile-triangles, les poids associés à leurs arêtes W, ainsi que les degrés Deg associés à leurs sommets. Ce vecteur est défini comme suit :
$\{AG(ts),\ \ PG(ts),\ \ \{A(t_i),\ \ P(t_i),\ \ W(t_{i,\ j=1...3}),\ Deg(t_{i,\ j=1...3})\}_{i=1}^{i=\|T(ts)\|}\}$
Les différentes variables sont décrites dans la Table 1. Les triangles appartenant à l'étoile-triangles ts sont classés par ordre décroissant de leurs surfaces (A). Les poids et les degrés sont également classés par ordre décroissant. Tous les vecteurs des étoiles-triangles TS ont la même taille : $size = 2 + (8\ \ \Gamma)$, où Γ est le nombre maximum des triangles dans les étoiles-triangles. Si une étoile-triangles ts a un nombre de triangles inférieur à Γ, les entrées non attribuées sont complétées par des zéros.

Définition 5 (Étoiles-triangles disjointes) Deux étoiles-triangles ts_i et ts_j sont $disjointes$ s'ils ne partagent pas, au moins, un triangle commun. Soit $i \neq j,\ si\ ts_i$ et ts_j sont $disjointes \Rightarrow T(ts_i) \cap T(ts_j) = \varnothing$.

Symbole	Description
$t_{i,l}$	Triangle t_l appartenant à l'étoile-triangles $ts_i : t_l \in ts_i$
$W_{i,l,k}$	Poids (distance euclidienne) de l'arête e_k appartenant au triangle $t_l \in ts_i$
$Deg_{i,l,k}$	Degré du sommet v_k appartenant au triangle $t_l \in ts_i$
Γ	le nombre maximum des triangles dans les étoiles-triangles
$\alpha \in \mathbb{R}_+^6$	Les paramètres associés avec les descripteurs où $\|\alpha\|_1 = 1$
$A(t_i)$	Surface du triangle i.
$P(t_i)$	Périmètre du triangle i.
$AG(ts_i)$	Surface de l'étoile-triangles i. $AG(ts_i) = \sum_{j=1}^{j=\|T(ts_j)\|} A(t_j)$
$PG(ts_i)$	Périmètre de l'étoile-triangles i. $PG(ts_i) = \sum_{j=1}^{j=\|T(ts_j)\|} P(t_j)$

TABLE 1 – Les symboles associés à la mesure de dissimilarité ainsi que leur description.

3.2 Ordonnancement des triangles

La décomposition proposée génère des étoiles-triangles disjointes (Définition 5), ce qui réduit considérablement le nombre de composants ($\|TS\| \ll \|V\| < \|T\|$) ainsi que le nombre de comparaisons impliquées dans le processus d'appariement des deux graphes. Cependant, en fonction de l'ordre considéré, les étoiles-triangles obtenues peuvent différer. En effet, une même tessellation peut générer différents ensembles d'étoiles-triangles si l'ordre des triangles considéré n'est pas identique.

Afin de garantir l'**unicité** de la décomposition et de réduire davantage le nombre d'étoiles-triangles, un ordre total strict décroissant sur l'ensemble des triangles doit être établi avant leur décomposition en étoiles-triangles. En considérant $\|Voisins\ (Triangles)\|$ avec ordre décroissant, nous générons un nombre plus petit d'étoiles-triangles TS (comme nous le montrons dans les expérimentations), contribuant ainsi à la réduction de la complexité temporelle.

Afin d'établir un ordre total strict décroissant sur l'ensemble des triangles, chaque triangle t_i est représenté par

un vecteur $\{\|N(t_i)\|, \{x_{i,j}, y_{i,j}, z_{i,j}\}_{j=1}^{j=3}\} \in \mathbb{R}^{10}$ qui correspond au nombre des voisins $\|N(t_i)\|$ et aux coordonnées cartésiennes x, y, z associées aux sommets formant le triangle t_i. Il est clair que les coordonnées et par conséquent l'ordre des triangles peuvent être affectés par une rotation de l'objet. Pour résoudre ce problème, les coordonnées sont exprimées dans le repère défini par les vecteurs propres du tenseur d'inertie associé aux sommets. Le nombre de voisins $\|N(t_i)\|$ est utilisé pour réduire davantage le nombre d'étoiles-triangles. Si deux triangles ont le même nombre de voisins, les coordonnées du sommet sont utilisées pour garantir l'unicité de la décomposition. Les sommets du vecteur (10-uplet) sont classés lexicographiquement en fonction de leurs coordonnées.

3.3 La décomposition en étoiles-triangles

Une fois que l'ordre total strict des triangles a été établi, la décomposition du graphe en étoiles-triangles peut être effectuée. Ce processus est décrit dans l'Algorithme 1. Selon l'ordre établi pour les triangles (ordre total strict décroissant), la première N_k-étoile-triangles est construite à partir du premier triangle et de ses N_k-voisins (Définition 4) qui n'appartiennent à aucune autre N_k-étoile-triangles. Ensuite, l'ensemble des triangles et des N_k-étoiles-triangles résultantes sont mis à jour. Le processus est ainsi répété jusqu'à ce qu'il ne reste plus de triangles non associés à une N_k-étoile-triangles.

La décomposition proposée génère un nombre réduit d'étoiles-triangles ts par rapport au nombre de sommets $\|TS\| \ll \|V\|$. Les étoiles-triangles résultantes sont disjointes (Définition 5) et couvrent une zone locale plus large que les étoiles classiques. Cette décomposition est également paramétrable en fonction du degré de voisinage. En effet, plus le degré de voisinage est élevé, plus le nombre d'étoiles-triangles est petit ($\|N_{k+1}\text{-}TS\| \leq \|N_k\text{-}TS\|$) tout en couvrant un voisinage plus large ($\|T(N_{k+1}\text{-}TS)\| \geq \|T(N_k\text{-}TS)\|$). De plus, la décomposition proposée est unique.

Algorithm 1 Décomposition en N_k-étoiles-triangles.

Entrées : Un graphe G_{tr} et le degré de voisinage N_k.
Sorties : Un ensemble de N_k-étoiles-triangles (N_k-TS).
Début
  Appliquer un ordre total strict **descendant** sur les triangles
  N_k-$TS = \varnothing$;
  Tant que $(T(G_{tr}) \neq \varnothing)$ **faire**
    $t_i = T(G_{tr})[0]$
    $T(N_k\text{-}ts_i) = t_i \cup N_k$-**voisins**$(t_i)$;
    $N_k\text{-}TS = N_k\text{-}TS \cup N_k\text{-}ts_i$;
    $T(G_{tr}) = T(G_{tr}) - T(N_k\text{-}ts_i)$;
  fin Tant que
  retourner N_k-TS;
Fin

Exemple 4. La Table 2 montre les correspondances résultantes en termes d'étoiles-triangles, en utilisant notre approche TSM avec différents degrés de voisinage, entre deux poses de quatre objets appartenant à la base de données TOSCA.

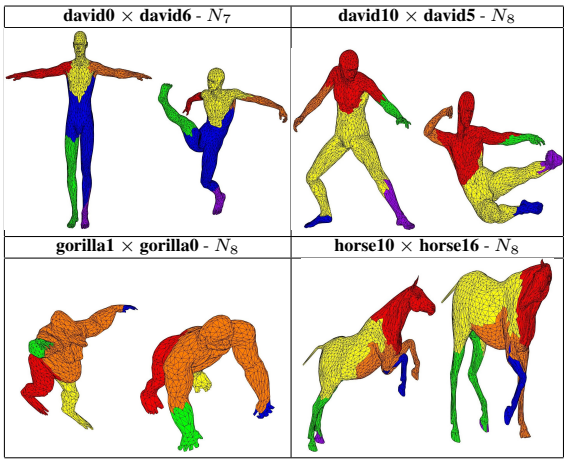


TABLE 2 – La correspondance, en utilisant notre approche TSM avec différents degrés de voisinage, entre deux poses de quatre objets appartenant à la base de données TOSCA.

4 Une nouvelle distance pour les tesselations de triangles : description de l'algorithme

Dans cette section, nous proposons une nouvelle distance entre les étoiles-triangles de deux tessellations triangulaires pour trouver leur appariement.

4.1 Distance d'édition entre étoiles-triangles

Nous introduisons d'abord la distance d'édition de graphes entre étoiles-triangles. La mesure de dissimilarité proposée est définie pour être appliquée aux objets déformables. En conséquence, l'ensemble des descripteurs doivent être invariants aux déformations les plus courantes. La mesure de dissimilarité est basée sur les paramètres suivants : la surface AG et le périmètre PG de l'étoile-triangles, la surface A et le périmètre P des triangles, les poids W des arêtes et les degrés Deg de sommets. Formellement une étoile-triangles est représentée comme suit : $\{AG(ts),\ PG(ts),\ \{A(t_i),\ P(t_i),\ W(t_{i,\ j=1\ldots3}),\ Deg(t_{i,\ j=1\ldots3})\}_{i=1}^{i=\|T(ts)\|}\}$. La mesure de dissimilarité d entre deux étoiles-triangles ts_i et ts_j est définie par :

$$d(ts_i, ts_j) = \sum_{k=1}^{k=6} dsim_k\ (ts_i, ts_j) \qquad (1)$$

La mesure dissimilarité d est normalisée ($0 \leq d \leq 1$) et nécessite la définition de six fonctions auxiliaires $dsim_k$. Ces fonctions sont définies comme suit :

$$dsim_k\ (ts_i, ts_j) =$$

$$\begin{cases} \alpha_1\ \frac{|AG(ts_i)\ -\ AG(ts_j)|}{AG_{MAX}} & \text{si } k=1 \\ \alpha_2\ \frac{|PG(ts_i)\ -\ PG(ts_j)|}{PG_{MAX}} & \text{si } k=2 \\ \alpha_3\ \frac{\sum_{l=1}^{l=\Gamma} |\ A(T(ts_i)_l)\ -\ A(T(ts_j)_l)\ |}{A_{MAX}\ \ \Gamma} & \text{si } k=3 \\ \alpha_4\ \frac{\sum_{l=1}^{l=\Gamma} |\ P(t_{i,l})\ -\ P(t_{j,l})\ |}{P_{MAX}\ \ \Gamma} & \text{si } k=4 \\ \alpha_5\ \frac{\sum_{l=1}^{l=\Gamma} \sum_{k=1}^{k=3} |\ W_{i,l,k}\ -\ W_{j,l,k}\ |}{3\ \ W_{MAX}\ \ \Gamma} & \text{si } k=5 \\ \alpha_6\ \frac{\sum_{l=1}^{l=\Gamma} \sum_{k=1}^{k=3} |\ Deg_{i,l,k}\ -\ Deg_{j,l,k}\ |}{3\ \ Deg_{MAX}\ \ \Gamma} & \text{si } k=6 \end{cases} \quad (2)$$

Telle que $\sum_{k=1}^{k=6} \alpha_k = 1$.
$dsim_1(ts_i, ts_j)$ et $dsim_2(ts_i, ts_j)$ comparent respectivement la surface AG et le périmètre PG de deux étoiles-triangles ts_i et ts_j. $dsim_3(ts_i, ts_j)$ et $dsim_4(ts_i, ts_j)$ comparent respectivement la surface A et le périmètre P de leurs triangles respectifs. Tandis que $dsim_5(ts_i, ts_j)$ et $dsim_6(ts_i, ts_j)$ comparent respectivement les poids W associés à leurs arêtes respectives et les degré Deg des sommets correspondants. Les symboles associés à la mesure de dissimilarité sont définis dans la Table 1.

4.2 Distance d'édition entre deux tessellations triangulaires

Le calcul de la distance entre deux tessellations triangulaires représentés par des étoiles-triangles constitue la dernière étape de notre algorithme. Nous appelons cette mesure de dissimilarité TSM. Cette mesure détermine le meilleur appariement possible entre deux ensembles d'étoiles-triangles. La dissimilarité entre deux ensembles d'étoiles-triangles est définie comme suit :

Définition 6 (TSM) Soient g_{Tr1} et g_{Tr2} deux tessellations triangulaires, TS_1 et TS_2 leurs ensembles d'étoiles-triangles correspondants, M l'ensemble de tous les appariements possibles entre TS_1 et TS_2, et $m \in M$ la fonction de correspondance. La mesure de dissimilarité $TSM(TS_1, TS_2)$ (dissimilarité normalisée) est définie comme suit :

$$TSM(TS_1,\ TS_2) =$$

$$\frac{\min_{m \in M} \sum_{ts_i\ \in\ TS_1,\ m(ts_i)\ \in\ TS_2} d(ts_i,\ m(ts_i))}{\max(\|TS_1\|,\ \|TS_2\|)} \quad (3)$$

Le calcul de $TSM(TS_1, TS_2)$ est équivalent à la résolution du problème d'affectation, qui est l'un des problèmes fondamentaux d'optimisation combinatoire qui vise à trouver le minimum/maximum coût de correspondance dans un graphe bipartite pondéré. Pour résoudre ce problème d'affectation, nous définissons une matrice D $n \times n$, tel que n est donné par $n = \max(\|TS_1\|,\ \|TS_2\|)$. Chaque élément $D_{i,j}$ de la matrice représente la mesure de dissimilarité $d(ts_i,\ ts_j)$ (Eq. 1) entre une étoile-triangles ts_i dans TS_1 et une étoile-triangles ts_j dans TS_2. Dans le cas de $\|TS_1\| \neq \|TS_2\|$, le plus petit ensemble d'étoiles-triangles est complété par $(max(||TS_1||, ||TS_2||) - min(||TS_1||,\ ||TS_2||))$ étoiles-triangles vides ε. La distance entre une étoile-triangles vide ε et une étoile-triangles ts est calculée par Eq. 1 et correspond au coût de l'ajout de ts à l'ensemble d'étoiles-triangles le plus petit (ou en supprimant ts du l'ensemble d'étoiles-triangles le plus grand).

Nous appliquons l'algorithme hongrois [23] sur la matrice D pour trouver la meilleure affectation avec une complexité de $\mathcal{O}(n^3)$. L'évaluation de la distance entre deux graphes est résumée dans l'Algorithme 2.

La classification est ensuite effectuée en utilisant des techniques d'apprentissage supervisées. Notre approche définit un espace de métriques décrivant les différents objets, en utilisant des techniques de plongement et de noyau de graphes : Chaque objet (l'ensemble de ses étoiles-triangles) TS_i est projeté sur un espace vectoriel, où il est représenté par un vecteur de distances $TSM(TS_i, TS_{j=1\ldots n})$, entre TS_i et l'ensemble des autres objets $TS_{j=1\ldots n}$.

Algorithm 2 Distance entre graphes en utilisant TSM.

1: **Entrées** : Deux graphes g_1 et g_2.
2: **Sorties** : La distance entre g_1 et g_2.
3: **Début**
4: Décomposition de g_1 et g_2 en TS_1 et TS_2, (Algo. 1).
5: Construire une matrice de distance D.
6: **Pour chaque** $ts_i \in TS_1$ et $ts_j \in TS_2$ faire
7: $D_{i,j} = d(ts_i,\ ts_j)$ (Eq. 1);
8: **Fin Pour chaque**
9: Résoudre (Eq. 3), algorithme Hongrois [23] sur la matrice D.
10: Retourner la distance $TSM(TS_1,\ TS_2)$;
11: **Fin**

5 Expérimentation

Afin d'évaluer l'approche proposée, nous avons entrepris une série d'expérimentations au cours desquelles nous avons comparé notre approche à des algorithmes de comparaison des objets issus de l'état de l'art, selon différents critères d'évaluation, sur la base de données TOSCA [15].

5.1 Complexité temporelle

Comme mentionné précédemment, l'algorithme hongrois [23] est utilisé pour trouver la meilleure affectation avec une complexité de $\mathcal{O}(n^3)$, où n est le nombre maximal de composants dans les deux graphes. Dans le pire des cas, la complexité est de l'ordre de $\mathcal{O}(0.037\ \ n^3)$. Cependant, le nombre d'étoiles-triangles dépend de la structure du graphe sous-jacent ainsi que du degré de voisinage N_k. En effet, le nombre d'étoiles-triangles diminue lorsque le degré de voisinage N_k augmente. La complexité temporelle, pour la base de données TOSCA [15], est de l'ordre de $\mathcal{O}(\alpha\ \ [\frac{n}{\log(n)}]^3)$, où $\alpha \in [1.80 * 10^{-7},\ 0.74]$ pour $N_{k=1\ldots6}$: $\mathcal{O}(0.74\ \ [\frac{n}{\log(n)}]^3)$. Le degré de voisinage $N_{k=1}$ a la complexité la plus élevée, tandis que le degré de voisinage $N_{k=6}$ a la complexité la plus basse.

5.2 Résultats expérimentaux

Dans cette section, nous comparons et discutons nos résultats avec ceux obtenus avec la base de données TOSCA [15]. La distance proposée TSM est paramétrable via les paramètres α_k, qui déterminent les poids attribués aux différentes mesures de similarité. La valeur par défaut de ces paramètres est la suivante : $\alpha_k = 1/6, \forall\, k$. Comme prouvé dans [1], la distance TSM est pseudo-métrique. Afin de classer les différents objets appartenant aux bases de données, la métrique est apprise avec des techniques d'apprentissage automatique. Ceci est en contraste avec l'approche standard dans laquelle un descripteur invariant est associé à chaque objet et la classification est effectuée avec des techniques d'apprentissage automatique par apprentissage supervisé. Dans notre approche, la distance TSM définit un espace de métrique décrivant les différents objets. Par conséquent, pour effectuer la classification, la métrique doit être apprise à l'aide d'un processus d'apprentissage supervisé. Cinq classificateurs ont été évalués afin de déterminer leur pertinence pour l'apprentissage de métrique, à savoir : le classificateur naïf de Bayes, le classificateur des forêts aléatoires, l'approche des arbres boostés par gradient, le classificateur à vaste marge (SVM) et la régression logistique. Ces classificateurs ont été décrits en détail dans la littérature [24, 25]. Nous nous limiterons donc à une brève description. Dans ce qui suit, les descripteurs font référence à la métrique. Le classificateur naïf de Bayes (NB) [24, 25] est un classificateur probabiliste basé sur le théorème de Bayes. Il suppose que les descripteurs sont générés indépendamment de la classe et utilise le théorème de Bayes pour prédire la classe. Le classificateur des forêts aléatoires (RF) [24, 25] utilise un ensemble d'arbres de décision pour prédire la classe. Chaque arbre de décision a été entrainé sur un sous-ensemble aléatoire de l'ensemble d'apprentissage, et utilise uniquement un sous-ensemble aléatoire des descripteurs. Le classificateur GBT (gradient boosted tree) [24, 25] prédit les étiquettes en entraînant de manière itérative une séquence d'arbres de décision sur des données d'apprentissage et en les combinant. Le classificateur à vaste marge (SVM) [24, 25] sépare les données d'apprentissage en deux classes à l'aide d'un hyperplan à marges maximales. Le problème de la classification multi-classes est réduit à un ensemble de problèmes de classification binaire. Enfin, le classificateur par régression logistique (LR) [24], connu aussi sous le nom de classificateur d'entropie maximum, modélise les probabilités de classe avec des fonctions logistiques de combinaisons linéaires des descripteurs.

Pour chaque classificateur, un petit ensemble de validation a été généré automatiquement afin d'évaluer les paramètres du classificateur. Le paramètre unique du classificateur naïf de Bayes (NB) était le paramètre de lissage, qui se situait généralement autour de 0.2. Pour le classificateur des forêts aléatoires (RF), quatre paramètres étaient requis, à savoir : fraction caractéristique, nombre de feuilles, nombre d'arbres et distribution de lissage, qui ont été définis sur $1/2\sqrt{37}$, 2, 100 et 0.5 respectivement. Cet algorithme a été implémenté en utilisant la bibliothèque DAAL (Intel Data Analytics Accelerations Library) [26]. Le classificateur GBT (gradient boosted tree) requiert plusieurs paramètres y compris la méthode de renforcement. L'algorithme est basé sur le gradient, le nombre maximum de tours d'apprentissage (50), le nombre de feuilles (13), le taux d'apprentissage (0.1), le nombre maximum de taches (255), le nombre de taches (20), la profondeur maximale (6), la taille de la feuille (15), la fraction caractéristique (1) et la fraction d'ensachage (1), entre autres. Pour le classificateur SVM, les paramètres suivants ont été déterminés : type de noyau (fonction de base radiale), paramètre de mise à l'échelle gamma (0.00725065), paramètre de marge souple (3), paramètre de biais (1), stratégie multi-classes (un contre un) et la taille du cache de noyau (100). Enfin, le classificateur de régression logistique (LR) a été optimisé au moyen de l'algorithme à mémoire limitée de Broyden-Fletcher-Goldfarb-Shanno (LBFGS) avec un terme de régularisation quadratique (0.001). Quatre mesures ont été utilisées pour évaluer les performances de notre système, à savoir : exactitude, précision, rappel et F-mesure. Ces mesures ont été choisies en raison de leurs performances et afin de comparer notre approche à celles utilisées dans les bases de données de référence. Comme ces métriques sont bien connues et largement utilisées, nous renvoyons les lecteurs à [27, 28] pour plus de détails. Dans certains cas, nous avons également ajouté la matrice de confusion et la courbe de taux de précision-rejet afin d'affiner davantage les performances du système ainsi que sa sensibilité à la probabilité de seuil de détection. Ici, le taux de rejet fait référence à la probabilité de classification d'un résultat donné : si cette probabilité est inférieure à un certain seuil, le résultat du processus de classification est considéré comme indéterminé et, par conséquent, non utilisé dans l'évaluation des métriques de performance. Les bases de données considérés étant relativement petites, elles ne conviennent pas vraiment pour la validation croisée. Par conséquent, pour évaluer la métrique, nous avons utilisé une technique de bootstrapping ou d'ensachage. L'ensachage suppose que l'ensemble de données est représentatif de sa distribution réelle. Les ensembles d'apprentissage et de validation sont échantillonnés de manière uniforme et aléatoire avec remplacement à partir du jeu de données d'origine afin d'entrainer les classificateurs. Le processus a été répété dix fois. exactitude, précision, rappel et F-mesure correspondent à la moyenne de ces dix itérations.

Dans la mesure du possible, les classificateurs ont été implémentés sur le GPU. Les calculs ont été effectués sur une station de travail équipée de deux processeurs Xeon dotés de 40 coeurs, de 64 Go de RAM et d'un GPU NVIDIA Quadro GP-100 doté de 3584 coeurs CUDA et de 16 Go de mémoire.

Résultats expérimentaux sur la base de données TOSCA. La distance TSM entre chaque paire d'objets a été évaluée pour les six premiers degrés de voisinage. Ces

distances forment la métrique qui doit être apprise. Pour chaque degré de voisinage et pour chaque classificateur : exactitude, précision, rappel, F-mesure, matrice de confusion et la courbe de taux de précision-rejet ont été déterminées. Seuls les résultats associés aux trois meilleurs classificateurs sont rapportés dans la Table 3 étant ces résultats sont nettement meilleurs que les autres.
Pour le deuxième degré de voisinage, les meilleurs résultats ont été obtenus avec le classificateur de régression logistique (LR), tandis que, pour le premier et les quatre derniers degrés de voisinage, les meilleurs résultats ont été obtenus avec le classificateur des forêts aléatoires (RF) ou le classificateur GBT (gradient boosted tree). Pour les meilleurs classificateurs, l'exactitude la plus faible était de 76.35%, tandis que l'exactitude la plus élevée était de 88.51%. Les deux ont été obtenus avec le premier ordre de voisinage en utilisant le classificateur GBT (gradient boosted tree).

N_k	exactitude			Précision			Rappel			F-measure		
	GBT	RF	LR	GBT	RF	LR	GBT	RF	LR	GBT	RF	LR
1	**88.51**	83.78	85.81	**91.80**	85.12	88.34	**89.64**	80.41	87.41	**0.90**	0.80	0.86
2	71.62	79.05	**79.73**	72.31	**81.70**	81.60	74.40	80.41	**82.04**	0.72	0.80	**0.81**
3	**81.76**	79.73	77.27	**85.39**	79.87	77.41	**81.47**	78.50	77.52	**0.82**	0.78	0.76
4	75.00	**76.35**	61.49	77.35	**78.98**	68.07	75.40	**76.50**	64.26	0.74	**0.76**	0.62
5	**77.70**	72.30	62.21	**81.04**	74.29	66.06	**76.75**	71.10	68.20	**0.76**	0.70	0.68
6	**80.41**	79.05	62.21	**83.34**	80.18	66.06	**79.94**	78.96	68.20	**80.41**	0.78	0.68

TABLE 3 – exactitude, précision, rappel, F-mesures obtenues avec les classificateurs GBT (gradient boosted tree), les forêts aléatoires (RF) et la régression logistique (LR) pour la base de données TOSCA.

La courbe de taux de précision-rejet et la matrice de confusion sont indiquées pour le premier degré de voisinage dans les figures 1 et 2 respectivement. Cette courbe de taux de précision-rejet indique clairement qu'une exactitude de 95% peut être facilement obtenue simplement en imposant un taux de rejet de 0.1. La matrice de confusion (Figure 2) illustre les performances de notre approche.

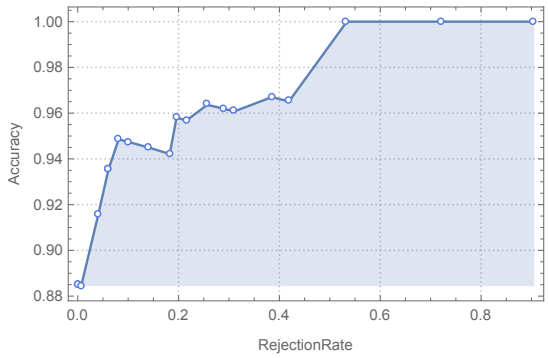


FIGURE 1 – La courbe de taux de précision-rejet pour $N_{k=1}$ avec le classificateur GBT (gradient boosted tree).

Nous avons également comparé notre méthode TSM à quatre autres algorithmes de l'état de l'art relatifs aux bases

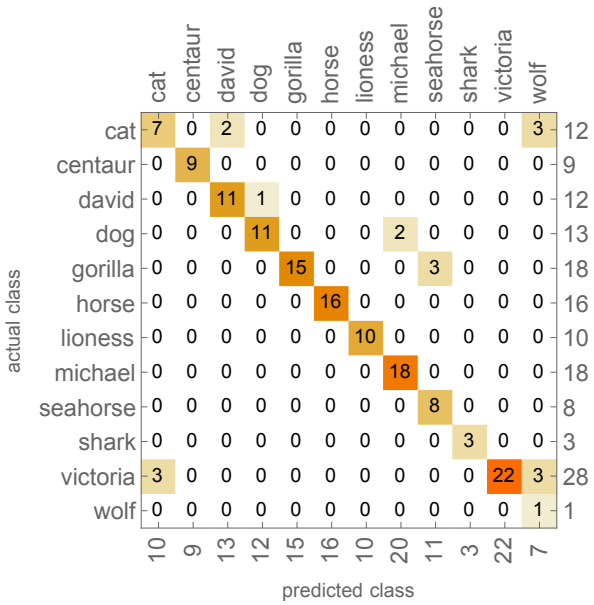


FIGURE 2 – La matrice de confusion pour $N_{k=1}$ avec le classificateur GBT (gradient boosted tree).

de doonées SHREC, à savoir CAM, GeodesicD2, DSR et RSH. La Table 4 compare l'exactitude obtenue par ces approches à l'exactitude obtenue par notre méthode pour le premier degré de voisinage avec le classificateur GBT (gradient boosted tree). Notre approche surpasse nettement les autres avec une exactitude de 91,80% contre un maximum de 30% pour les quatre autres approches.

Méthode	Précision (%)	Rappel (%)
TSM, $N_{k=1}$, GBT	**91.80**	**89.64**
CAM	30	89.64
Geodesic2D	26	89.64
DSR	24	89.64
RSH	21	89.64

TABLE 4 – Comparaison de notre méthode TSM ($N_{k=1}$) en termes de exactitude avec les approches CAM, GeodesicD2, DSR et RSH pour la base de données TOSCA.

Nous avons également comparé notre approche pour tous les degrés de voisinages avec les méthodes TD, ShapeDNA et SRCP-TD en termes de F-mesure. Les résultats sont rapportés dans la Table 5. Une fois de plus, notre approche surpasse les autres, quel que soit l'ordre du voisinage considéré.

Résultats expérimentaux pour la base de données TOSCA face au re-maillage. Nous avons étudié la robustesse de notre approche TSM face à la réduction du maillage et le ré-maillage. Ceci est d'une importance primordiale car notre méthode est basée sur le maillage ou le graphe associé aux objets 3D. La base de données TOSCA

Method	N_k	Classificateur	F-measure
TSM	1	GBT	**0.90**
TSM	2	LR	**0.81**
TSM	3	GBT	**0.82**
TSM	4	RF	**0.76**
TSM	5	GBT	**0.76**
TSM	6	GBT	**0.80**
TD	N/A	N/A	0.67
Shape-DNA	N/A	N/A	0.45
SRCP-TD	N/A	N/A	0.44

TABLE 5 – F-mesure de notre approche TSM du meilleur classificateur associé à un ordre de voisinage donné comparée à les F-mesure obtenues avec TD, Shape-DNA et SRCP-TD pour la base de données TOSCA.

[15] a été utilisée pour l'évaluation. Le nombre de triangles a été réduit de 10% et 20% afin de générer les bases de données TOSCA_90 et TOSCA_80 respectivement. La réduction triangulaire a été réalisée avec une décimation d'effondrement d'arête quadrique [29]. Les résultats que nous avons évalués en termes de exactitude, de précision, de rappel et de F-measure pour les bases de données TOSCA_90 et TOSCA_80 sont rapportés dans la Table 6.

N_k	Classificateur		exactitude (%)		Précision (%)		Rappel (%)		F-measure	
	T80	**T90**	**T80**	**T90**	**T80**	**T90**	**T80**	**T90**	**T80**	**T90**
1	RF	RF	84.45	84.46	86.97	87.44	84.29	85.44	0.84	0.85
2	GBT	GBT	71.62	77.70	75.20	78.00	72.50	78.26	71.49	0.78
3	RF	GBT	71.62	73.65	73.50	76.85	74.09	75.29	0.73	0.75
4	RF	RF	79.73	75.00	81.97	78.32	80.95	77.08	0.80	0.76
5	GBT	RF	77.70	76.35	80.51	77.34	78.77	77.82	0.79	0.76
6	GBT	RF	74.32	73.65	80.82	74.74	75.59	73.06	0.76	0.73

TABLE 6 – exactitude, précision, rappel, F-mesures de TSM obtenues avec les classificateurs GBT (gradient boosted tree) et forêts aléatoires (RF) pour $N_{k=1...6}$ pour les bases de données TOSCA_80 (T80) et TOSCA_90 (T90) Databases.

La réduction et le ré-maillage ont une incidence directe sur notre approche puisque celle-ci est une approche à base de graphes. Néanmoins, TSM est basé sur une approximation de la distance d'édition de graphes tolérante aux bruits et à la distorsion. De plus, notre approche utilise un ensemble de descripteurs invariants face aux déformations géométriques les plus courantes. Par conséquent, notre approche TSM est robuste face à la réduction et le ré-maillage. En dépit du fait que la réduction triangulaire soit relativement importante (10% à 20%), l'algorithme affiche une résistance étonnamment élevée à la réduction du maillage : l'exactitude atteignait 84.5% pour le premier degré de voisinage pour 10% et 20% de réduction du maillage, contre 88.51% sans réduction du maillage. Ces résultats ont tous été obtenus avec le classificateur des forêts aléatoires (RF). Notre expérimentation avec la bases de données TOSCA et TOSCA avec maillage réduit, comme décrit précédemment, montre les performances et la robustesse de notre approche.

En effet, nous avons obtenu d'excellents résultats en termes de exactitude, de précision, de rappel et de F-mesure pour la base de données TOSCA. Notre complexité temporelle prévue a été systématiquement confirmée par nos expérimentations. Notre méthode est plus performante que CAM, GeodesicD2, DSR et RSH en termes de exactitude, de précision et de rappel pour la base de données TOSCA. Les mêmes remarques s'appliquent lorsque notre approche est comparée à d'autres méthodes en termes de F-mesure.

6 Conclusions

Dans cet article, nous avons présenté un nouvel algorithme d'appariement de graphes pour résoudre le problème de la comparaison des objets 3D déformables représentés par des graphes (tessellations triangulaires). L'approche proposée est basée sur une nouvelle décomposition de tessellations triangulaires en étoiles-triangles. Les étoiles-triangles obtenues sont utilisées pour déterminer la distance entre les tessellations en utilisant l'algorithme hongrois. L'algorithme proposé garantit un nombre minimum d'étoiles-triangles disjointes, offre une meilleur dissimilarité en couvrant un voisinage plus large en étoiles-triangles et utilise un ensemble de descripteurs qui sont invariants face aux déformations les plus courantes. L'approche proposée est basée sur une approximation de la distance d'édition de graphes, qui est tolérante aux bruit et à la distorsion, ce qui rend notre technique particulièrement adaptée à la comparaison des objets déformables. La classification est effectuée en utilisant des techniques d'apprentissage automatique supervisées. Notre approche définit un espace de métrique en utilisant des techniques de plongement et de noyaux de graphes. Nous avons prouvé que la distance proposée TSM est pseudo-métrique. Nos résultats expérimentaux, obtenus à partir de différentes bases de données de référence pour les objets 3D déformables, confirment les performances et l'exactitude de notre algorithme. Dans un proche avenir, nous prévoyons d'enrichir la description des étoiles-triangles tout en réduisant la complexité temporelle et en améliorant les performances de l'algorithme d'appariement de graphes proposé. Nous prévoyons également de combiner notre approche avec des techniques d'apprentissage en profondeur.

Références

[1] Kamel Madi, Eric Paquet, and Hamamache Kheddouci. New graph distance for deformable 3d objects recognition based on triangle-stars decomposition. *Pattern Recognition*, 90 :297–307, 2019.

[2] Horst Bunke, Andreas Münger, and Xiaoyi Jiang. Combinatorial search versus genetic algorithms : A case study based on the generalized median graph problem. *Pattern Recognition Letters*, 20(11-13) :1271–1277, 1999.

[3] Kamel Madi, Hamida Seba, Hamamache Kheddouci, and Olivier Barge. A graph-based approach for kite

recognition. *Pattern Recognition Letters*, 87 :186–194, 2017.

[4] Horst Bunke and Kim Shearer. A graph distance metric based on the maximal common subgraph. *Pattern Recognition Letters*, 19(3-4) :255–259, 1998.

[5] Donatello Conte, Pasquale Foggia, Carlo Sansone, and Mario Vento. Thirty years of graph matching in pattern recognition. *IJPRAI*, 18(3) :265–298, 2004.

[6] Kaspar Riesen and Horst Bunke. Approximate graph edit distance computation by means of bipartite graph matching. *Image Vision Comput.*, 27(7) :950–959, 2009.

[7] Johan W. H. Tangelder and Remco C. Veltkamp. A survey of content based 3d shape retrieval methods. *Multimedia Tools Appl.*, 39(3) :441–471, 2008.

[8] Jin Xie, Yi Fang, Fan Zhu, and Edward Wong. Deepshape : Deep learned shape descriptor for 3d shape matching and retrieval. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1275–1283, 2015.

[9] Skeleton graph matching vs. maximum weight cliques aorta registration techniques. *Computerized Medical Imaging and Graphics*, 46, Part 2 :142 – 152, 2015. Information Technologies in Biomedicine.

[10] Vincent Barra and Silvia Biasotti. 3d shape retrieval using kernels on extended reeb graphs. *Pattern Recognition*, 46(11) :2985–2999, 2013.

[11] Yanir Kleiman, Oliver van Kaick, Olga Sorkine-Hornung, and Daniel Cohen-Or. SHED : shape edit distance for fine-grained shape similarity. *ACM Trans. Graph.*, 34(6) :235, 2015.

[12] Sicheng Zhao, Hongxun Yao, Yanhao Zhang, Yasi Wang, and Shaohui Liu. View-based 3d object retrieval via multi-modal graph learning. *Signal Processing*, 112 :110–118, 2015.

[13] Hermilo Sánchez-Cruz and Ernesto Bribiesca. A method of optimum transformation of 3d objects used as a measure of shape dissimilarity. *Image and Vision Computing*, 21(12) :1027–1036, 2003.

[14] Johan WH Tangelder and Remco C Veltkamp. Polyhedral model retrieval using weighted point sets. *International journal of image and graphics*, 3(01) :209–229, 2003.

[15] Alexander M. Bronstein, Michael M. Bronstein, and Ron Kimmel. Calculus of nonrigid surfaces for geometry and texture manipulation. *IEEE Trans. Vis. Comput. Graph.*, 13(5) :902–913, 2007.

[16] Hedi Tabia, Mohamed Daoudi, Jean-Philippe Vandeborre, and Olivier Colot. A new 3d-matching method of nonrigid and partially similar models using curve analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(4) :852–858, 2011.

[17] Robert Osada, Thomas A. Funkhouser, Bernard Chazelle, and David P. Dobkin. Shape distributions. *ACM Trans. Graph.*, 21(4) :807–832, 2002.

[18] Dejan Vranic. *3D Model Retrieval*. PhD thesis, University of of Leipzig, may 2004.

[19] Dietmar Saupe and Dejan V. Vranic. 3d model retrieval with spherical harmonics and moments. In *Pattern Recognition, 23rd DAGM-Symposium, Munich, Germany, September 12-14, 2001, Proceedings*, pages 392–397, 2001.

[20] Yi Fang, Mengtian Sun, and Karthik Ramani. Temperature distribution descriptor for robust 3d shape retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR Workshops 2011, Colorado Springs, CO, USA, 20-25 June, 2011*, pages 9–16, 2011.

[21] Martin Reuter, Franz-Erich Wolter, and Niklas Peinecke. Laplace-beltrami spectra as 'shape-dna' of surfaces and solids. *Computer-Aided Design*, 38(4) :342–366, 2006.

[22] Mostafa Abdelrahman, Moumen T. El-Melegy, and Aly A. Farag. Heat kernels for non-rigid shape retrieval : Sparse representation and efficient classification. In *Ninth Conference on Computer and Robot Vision, CRV 2012, Toronto, Ontario, Canada, May 28-30, 2012*, pages 153–160, 2012.

[23] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2) :83–97, 1955.

[24] Kevin P. Murphy. *Machine learning - a probabilistic perspective*. Adaptive computation and machine learning series. MIT Press, 2012.

[25] Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. *Deep Learning*. Adaptive computation and machine learning. MIT Press, 2016.

[26] James Reinders. Intel data analytics acceleration library.

[27] Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8) :861–874, 2006.

[28] David Martin Ward Powers. Evaluation : from precision, recall and f-factor to roc, informedness, markedness and correlation. *School of Informatics and Engineering Technical Reports*, (SIE-07-001), 2007.

[29] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1997, Los Angeles, CA, USA, August 3-8, 1997*, pages 209–216, 1997.

LUDII - Le Système Ludémique de General Game Playing

Éric Piette
Dennis J.N.J. Soemers
Matthew Stephenson
Chiara Sironi
Mark Winands
Cameron Browne

Department of Data Science and Knowledge Engineering (DKE)
Maastricht University

{eric.piette, dennis.soemers, matthew.stephenson, c.sironi, m.winands, cameron.browne}
@maastrichtuniversity.nl

Résumé

*Bien que les systèmes actuels de General Game Playing (*GGP*) facilitent la recherche en Intelligence Artificielle (*IA*) autour des jeux, ils sont souvent trop spécialisés et fournissent une capacité de calcul trop faible. Cet article décrit une première version du système ludémique de* GGP *dénommé* LUDII *qui apporte un outil efficace à la recherche en* IA *tout autant qu'aux concepteurs de jeux ou aux historiens mais aussi dans bien d'autres domaines.* LUDII *définit les jeux comme des arbres de ludèmes, correspondant à des concepts et mécanismes de jeu de haut niveau et facilement interprétables. Nous établissons les bases de* LUDII *en analysant ses principaux avantages : généralité, extensibilité, compréhensibilité et efficacité. Expérimentalement,* LUDII *surpasse l'un des plus performants raisonneurs décrit avec le General Game Description Language (*GDL*), basé sur un réseau de propositions (propnet), pour tous les jeux disponibles sur le serveur* GGP.

Mots Clef

General Game Playing, *Game AI*, Représentation des Connaissances, Apprentissage Automatique.

Abstract

*While current General Game Playing (*GGP*) systems facilitate useful research in Artificial Intelligence (*AI*) for game-playing, they are often somewhat specialized and computationally inefficient. In this paper, we describe an initial version of a "ludemic" general game system called* LUDII*, which has the potential to provide an efficient tool for* AI *researchers as well as game designers or historians and practitioners in related fields.* LUDII *defines games as structures of ludemes, i.e. high-level, easily understandable game concepts. We establish the foundations of* LUDII *by outlining its main benefits : generality, extensibility, understandability and efficiency. Experimentally,* LUDII *outperforms one of the most efficient Game Description Language (*GDL*) reasoners, based on a propositional network, for all available games in the* GGP *repository.*

Keywords

General Game Playing, *Game AI*, Knowledge Representation, Machine Learning.

1 Introduction

L'objectif du *General Game Playing* (GGP) est de développer des programmes-joueurs capables de jouer à une grande variété de jeux [22]. De nombreux systèmes utilisés pour modéliser des jeux, communément appelés *General Game Systems*, existent actuellement pour différents types de jeux incluant : les jeux déterministes à information complète [11], les jeux combinatoires [2], les puzzles [28], les jeux de stratégies [17], les jeux de cartes [10] et les jeux-vidéos [25].

Depuis 2005, le *General Game Systems* GGP-BASE [1] utilisant le *Game Description Language* (GDL) [16] afin de représenter les jeux est devenu le standard pour la recherche académique autour du GGP. GDL est un ensemble de clauses logiques de premier ordre décrivant les jeux à l'aide d'instructions simples. Bien que ce dernier soit dédié aux jeux déterministes avec information complète, il possède une extension dénommée GDL-II [26] incluant les jeux avec information incomplète et une seconde dénommée GDL-III [31] développée récemment intégrant les jeux épistémiques.

1.1 GDL

La capacité générique induite par GDL propose un défi algorithmique de haut niveau et a conduit à de nombreuses contributions importantes [1], particulièrement autour de *Monte Carlo Tree Search* (MCTS) [8, 9] tout en apportant différents algorithmes originaux comme la combinaison de la programmation par contraintes, MCTS et la détection de symétries [13]. Malheureusement, les aspects clés définissant un jeu (tels que les plateaux, les pièces d'un jeu, les jeux de cartes, etc) ou même les opérateurs arithmétiques doivent être redéfinies explicitement de zéro pour chaque nouvelle description de jeu.

1. https://github.com/ggp-org/ggp-base

De ce fait, modéliser et/ou débugger un jeu peut être chronophage tout en étant difficilement déchiffrable une personne n'étant pas familière avec la programmation logique. L'équipement et les règles sont fortement interconnectés à tel point que changer n'importe quel aspect du jeu nécessiterait une réécriture importante du code. Par exemple, changer la taille du plateau de 3×3 à 4×4 dans la description GDL du *Tic-Tac-Toe* demande la modification et l'ajout de nombreuses lignes de code.

De plus, GDL est actuellement limité en terme d'applicabilité en dehors de la recherche en IA pour les jeux. En effet, ce langage est verbeux et difficile à interpréter tout en n'intégrant pas les concepts clés relatifs aux jeux que tout concepteur utilise typiquement. De même, traiter informatiquement de telles descriptions est coûteux (en temps et en mémoire) du fait qu'il requiert d'utiliser un interpréteur logique rendant le langage difficilement portable pour tout autre application externe.

De nombreux jeux plus "complexes" peuvent être difficiles et demander un temps important pour être modéliser (ex : *Go*), ou sont rendus injouables suite à un important coût en temps de calcul (ex : *Échecs*). Au sein des différents dépots de jeux/instances GDL [27], seuls quelques jeux sont ajoutés chaque année suite à ces défauts.

1.2 Le Digital Ludeme Project

Le *Digital Ludeme Project* (DLP)[2] est un project de recherche sur cinq ans, récemment lancé à l'université de Maastricht, avec pour objectif de modéliser l'ensemble des jeux traditionnels dans une seule base de données numérique et permettant de jouer à ces derniers. Cette base sera utilisée afin de découvrir les liens intrinsèques entre les jeux et leurs composants dans le but de développer un modèle décrivant leur évolution tout au long de l'histoire humaine et de retracer leur propagation à travers les cultures du monde entier. Ce projet établira un nouveau champs de recherche dénommée *l'archéoludogie numérique* [6].

L'une des principales missions du DLP est de modéliser les 1.000 jeux traditionnels les plus influents au cours de l'histoire, où chacun d'entre eux peuvent avoir de multiples interprétations et nécessitant le test de plusieurs centaines de variantes. Ce n'est pas seulement un défi mathématiques et informatique mais aussi un défi logistique requérant un nouveau type de *General Game System*. Le DLP traite l'ensemble des jeux anciens de stratégies incluant principalement des jeux de plateaux, des jeux de cartes, des jeux de dés, des jeux d'empilement, des jeux de tuiles, etc, pouvant impliquer des éléments non déterministes ou de l'information incomplète. Notons que le DLP exclut tout jeu de dextérité, jeu social, sport ou jeu vidéo, etc.

Dans ce papier, nous introduisons formellement une première version de LUDII[3], le premier système Ludémique GGP complet capable de modéliser et de jouer (par un humain ou une IA) à l'ensemble des jeux de stratégies traditionnels. La notion de ludème est introduite en Section 2 avant de décrire l'approche ludémique que nous avons implémentée en Section 3. En Section 4, le système LUDII est détaillé avant de mettre en avant ses différentes aptitudes lui permettant de répondre au DLP en Section 5. L'efficacité sous-jacente du système LUDII en terme de raisonnement est démontré expérimentalement en Section 6 en le comparant à l'un des des meilleurs raisonneurs utilisant le système GGP-BASE [29].

2. *Digital Ludeme Project* : `http://ludeme.eu`
3. LUDII est nommé ainsi suite à son prédécesseur LUDI [2]

2 Ludèmes

La décomposition de jeux en ludèmes [21], i.e. unités conceptuelles d'informations relatives au jeu, nous permet de distinguer sa forme (ses règles et son équipement) de sa fonction (le contexte). Cette séparation fournit une analogie claire au génotype/phénotype qui rend possible l'analyse phylogénétique, où les ludèmes constituent les blocs d'ADN définissant chaque jeu.

Ce modèle ludémique de jeux a été démontré avec succès dans des travaux précédents impliquant de nouveaux jeux de plateau générés à partir de jeux existants [3]. Un important bénéfice de l'approche ludémique est d'encapsuler les concepts clés des jeux et de les décrire par des termes significatifs. Ceci permet la description automatique d'ensemble de règles de jeux, la comparaison entre différents jeux et potentiellement l'explication automatique des stratégies apprises par une IA en terme humainement compréhensible. De récents travaux ont montré comment ce modèle pouvait être amélioré pour une plus grande généralité et extensibilité en permettant à tout ludème modélisable d'être défini par une approche grammaticale de classe, qui dérive le langage de description de jeu directement de la hiérarchie de classes sous-jacente de la bibliothèque du code source correspondant [5].

Cette approche assure à une seule IA d'être capable de modéliser, jouer et analyser tout jeu traditionnel de stratégie comme une architecture de ludèmes. Mais aussi fournissant un mécanisme pour identifier les correspondances mathématiques entre les jeux en établissant des liens probabilistes entre eux en tant que patrimoine génétique concret.

3 L'approche Ludèmique

L'approche ludèmique utilisée pour modéliser les jeux est maintenant décrite.

3.1 Syntaxe

Definition 1 *Un état de jeu* LUDII s *encode le joueur devant jouer dans* s *(formulé par* $mover(s)$*) et cinq vecteurs contenant respectivement une donnée pour chaque localisation :* $what$*,* who*,* $count$*,* $state$*, et* $hidden$*. Une description plus précise de ses localisations et des données spécifiques fournies par ces vecteurs est donnée après la Définition 3.*

Definition 2 *Une fonction successeur* LUDII *est donnée par*

$$\mathcal{T} : (S \setminus S_{ter}, \mathcal{A}) \mapsto S,$$

où S est l'ensemble de tous les états de jeu LUDII*, S_{ter} l'ensemble des état terminaux, et $\mathcal{A}$ l'ensemble de toutes les listes d'actions possibles.*

Étant donné un état courant $s \in S \setminus S_{ter}$ et une liste d'actions $A = [a_i] \in \mathcal{A}$, $\mathcal{T}$ retourne un état successeur $s' \in S$. Intuitivement, une liste complète des actions A peut être interprétée comme un unique mouvement sélectionné par un joueur, pouvant avoir de multiples conséquences sur un état de jeu (chacune implémentée par une action primitive différente).

Definition 3 *Un jeu* LUDII *est représenté par un 3-tuple de ludèmes $\mathcal{G} = \langle Mode, Equipment, Rules \rangle$ où :*

- $Mode = \{p_0, p_1, \ldots, p_k\}$ *est un ensemble fini de $k+1$ joueurs, où $k \geq 1$. Les éléments aléatoires (tels que lancer un dé, lancer une pièce, distribuer des cartes, etc) sont joués par p_0, décrivant la nature/l'environnement. Le premier joueur à réaliser une action dans tout jeu est p_1, et le mover fait référence au joueur.*
- $Equipment = \langle C^t, C^p \rangle$ *où :*
  - *C^t désigne une liste de conteneurs (plateaux, main du joueur, paquets de cartes, etc). Chaque conteneur $c^t = \langle V, E \rangle$, où $c^t \in C^t$, est un graphe possédant des sommets V et des arêtes E. Chaque sommet $v_i \in V$ correspond à un emplacement jouable (ex : une case aux échecs ou encore une intersection au Go), alors que chaque arête $e_i \in E$ représente deux emplacements adjacents.*
  - *C^p dénote une liste de composants (pièces, cartes, tuiles, dés, etc) dont certaines peuvent être placées sur les emplacements des conteneurs dans C^t. Nous convenons que le composant $c_0^p \in C^p$ est placé sur l'ensemble des emplacements vides.*
- $Rules$ *définit les différentes règles d'un jeu, incluant :*
  - $Start = [a_0, a_1, \ldots, a_k]$ *représentant une liste d'actions de départ. Les actions de début de jeu sont appliquées séquentiellement sur un état "vide" (état où c_0 est présent sur chaque emplacement de tout conteneur) pour modéliser l'état initial s_0.*
  - $Play : S \mapsto \mathcal{P}(\mathcal{A})$*, où $\mathcal{P}(\mathcal{A})$ décrit l'ensemble des parties de l'ensemble $\mathcal{A}$ de toutes les listes possibles d'actions légales. C'est une fonction qui selon un état $s \in S$ retourne un ensemble de listes d'actions.*
  - $End = (Cond_0(s), \vec{\mathbb{S}}_0) \cup \ldots \cup (Cond_e(s), \vec{\mathbb{S}}_e)$ *désigne un ensemble de conditions $Cond_i(s)$ sous lesquelles un état donné s est considéré terminal. Chaque condition de terminaison $Cond_i(s)$ mène à un ensemble de scores $\vec{\mathbb{S}}_i$.*

LUDII propose quelques vecteurs pré-définis afin de simplifier leurs écritures : `Win`, `Loss`, `Draw`, `Tie`, et `Abort`. De plus, si le *mover* n'a pas d'actions légales alors il se trouve (temporairement) dans une impasse et il doit réaliser l'action spéciale `pass` sauf si les règles *End* dictent ce cas autrement. Les états dans lesquels tous les joueurs sont forcés de réaliser l'action `pass` lors du dernier tour complet de jeu sont abandonnés en tant que match nul (*Draw*).

Nous spécifions les *localisations* $loc = \langle c, v_i, l_i \rangle$ par leur conteneur $c = \langle V, E \rangle$, un sommet $v_i \in V$ et un niveau $l_i \geq 0$. Chaque localisation spécifie un emplacement particulier dans un conteneur spécifique à un certain niveau, bien que dans la plus part des jeux $l_i = 0$ excepté pour les jeux d'empilement (*Stacking games*) où possiblement plusieurs niveaux peuvent être utilisés. Pour chacune de ses localisations, un état de jeu s encode plusieurs données tel que décrit dans la Définition 1. L'indice d'un composant localisé en loc dans s est donné par $what(s, loc)$, le propriétaire (indice du joueur) par $who(s, loc)$, le nombre de composants par $count(s, loc)$, l'état interne d'un composant (coté, direction, promotion actuelle, etc) par $state(s, loc)$. Si l'état d'une localisation loc est une information cachée pour un joueur en particulier p_i alors cela est indiqué par $hidden(s, loc, p_i)$.

3.2 Exemple de jeu LUDII

Suite à la Définition 3, LUDII fournit divers ludèmes correspondant à de simples opérations pouvant être utilisées pour définir les joueurs, l'équipement et les règles. Par exemple, `(line 3)` est un ludème booléen retournant `Vrai` si dans l'état courant une ligne de trois pièces se trouve dans un conteneur et `(empty)` est un ludème retournant une liste contenant l'ensemble des emplacements libre d'un conteneur.
Dans la Figure 1, une description complète du jeu *Tic-Tac-Toe* est donnée en utilisant une grammaire de type EBNF générée par LUDII. Le ludème `mode` décrit le mode de jeu ; dans l'exemple il s'agit d'un jeu dont les tours alternent entre deux joueurs nommés `P1` et `P2`. Le premier sous-ensemble de ludème `equipment` décrit le plateau principale comme un carré de taille 3×3 et le second sous-ensemble liste les composants du jeu : un disque nommé "O" pour le joueur 1 et une croix nommée "X" pour le second. Chaque tour, le *mover* place sa pièce sur un emplacement libre, ceci est indiqué par `(to Mover (empty))`. La condition pour gagner pour le *mover* est de créer une ligne de trois de ses pièces. A noter qu'ici *Tic-Tac-Toe* ne requiert pas de règles $Start$.

```
(game "Tic-Tac-Toe"
   (mode {(player "P1")(player "P2")})
   (equipment
      {(board "Board" (square 3))}
      {(disc "O" 1) (cross "X" 2))}
   )
   (rules
      (play (to Mover (empty)))
      (end (line 3)(result Mover win))
   )
)
```

FIGURE 1: Le jeu Tic-Tac-Toe décrit avec LUDII.

Si le plateau est plein sans qu'aucun joueur ne remporte la partie, alors le jeu finira par un match nul (*Draw*) suite à l'action obligatoire *Pass* appliquée par les deux joueurs. Notons que l'utilisation judicieuse de ce paramètre par défaut permet une description plus succincte des jeux.

4 Le Système LUDII

La prochaine section introduit le système LUDII lui-même, décrivant l'approche grammaticale de classe et le noyau du système.

4.1 L'approche Grammaticale de Classe

LUDII est un système GGP complet [4] utilisant une approche grammaticale de classe dans laquelle le langage de description de jeu est automatiquement généré par les constructeurs de chaque classe du code source LUDII [5]. Les descriptions de jeux exprimées dans la grammaire sont automatiquement instanciées dans la bibliothèque de code correspondant afin d'être compilées, donnant ainsi une garantie 1:1 de correspondance entre le source code et la grammaire.

Schaul *et al.* [24] souligne que *"any programming language constitutes a game description language, as would a universal Turing machine"*. LUDII utilise ainsi efficacement son langage de programmation (Java) comme son langage de description. De ce fait, théoriquement, il peut supporter toute règle, équipement ou comportement pouvant être implémentés en Java, mais l'ensemble des détails de programmation sont cachés à l'utilisateur, qui ne voit uniquement qu'une grammaire simplifiée lui indiquant le code devant être appelé.

4.2 Le Noyau de LUDII

Le coeur de LUDII est une bibliothèque de ludèmes implémentée en Java 8, composée de nombreuses classes, chacune implémentant un ludème spécifique. Un jeu LUDII $\mathcal{G}$ définissant tous les ludèmes adéquates pour le modéliser (joueurs, équipement règles) est stocké dans un objet `Game`. Dans le contexte du GGP, permettre l'affichage de tout jeu automatiquement est crucial afin de faciliter la compréhension des stratégies appliquées par les IA. À cette fin, tous les équipements dans le système LUDII implémentent l'interface `Drawable`, permettant à chacun d'entre eux d'être illustrer par une image *bitmap* adaptable à toute résolution, afin de représenter tout état du plateau et de ses composants. Les conteneurs C^t peuvent ainsi dessiner leurs composants actuels aux positions, orientations, états appropriés. Un object `View` fournit le mécanisme permettant d'afficher l'état courant du jeu à l'écran et un object `Controller` permet de mettre à jour l'état du jeu suite aux entrées de l'utilisateur tel que les clics de la souris. De ce fait, tous les jeux disponibles dans le système peuvent être joués à la fois par tout humain et/ou par une IA.

En tant qu'exemple, la Figure 2 représente un jeu à deux joueurs $\mathcal{G}$ avec $C^t = \{c_0^t\}$, où c_0^t est un conteneur hexagonal composé de cellules hexagonales. $C^p = \langle c_0^p, c_1^p, c_2^p \rangle$, où c_0^p et le composant "vide", c_1^p est le disque blanc pour le joueur p_1 et c_2^p illustre le disque noir du joueur p_2. Le

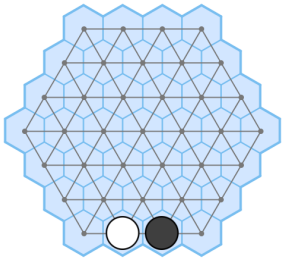


FIGURE 2: Un conteneur hexagonal fragmenté par des hexagones et le dual du graphe.

système possède une modélisation par un graphe pour visualiser le plateau, les sommets, les arêtes et les côtés de ce dernier sont dépeints en bleu. Le *dual* de ce graphe, qui est le graphe donné par c_0^t est dépeint en gris.

Le graphe du jeu peut lui-même être modifié dans certains "jeux de graphes" (ex : *Dots & Boxes*) dans lequel les mouvements d'un joueur implique des modifications du graphe lui-même (ex : Ajouter/Supprimer des arêtes ou des sommets). Afin d'optimiser le raisonnement/calcul, plusieurs données sont pré-générées comme les coins du plateau, les sommets extérieurs, les sommets le long du haut du plateau, etc. dans la classe `Graph` et les sommets voisins pour chaque direction, etc dans la classe `Vertex`.

Les vecteurs de données $what$, who, etc. d'un état s sont implémentées par une collection d'objets `ContainerState`. Différentes représentations de l'état sont implémentées dans le but de minimiser l'empreinte mémoire et d'optimiser le temps nécessaire pour accéder aux données nécessaires au raisonnement dans tout jeu :

- Pièces identiques pour chaque joueur (ex : *Hex*).
- Pièces distinctes par joueur (ex : *Échecs*).
- Pièces avec état local (ex : *Reversi*).
- Compteur de pièces par localisation (ex : *Mancala*).
- Jeux d'empilement (ex : *Laska*).
- Sans plateau fixe (ex : *Dominoes*).
- Information cachée (ex : *Stratego*, *jeux de cartes*).

LUDII sélectionne automatiquement la représentation appropriée de l'état à partir des règles afin de créer l'objet `Game`, assurant la représentation qui convient le mieux.

Les états des conteneurs sont définis dans une classe `BitSet` modifiée sur mesuse, dénommée `ChunkSet`, qui condense les informations induites par l'état de jeu dans un espace mémoire minimum en fonction de la description de chaque jeu. Par exemple, si un jeu n'inclue pas plus d'équipement qu'un plateau et des pièces identiques de N couleurs, alors l'état de jeu sera décrit par un `ChunkSet` subdivisé en bloc (*chunk*) de B *bits* par cellule du plateau, où B est la plus faible puissance de 2 fournissant assez de *bits* pour représenter chaque état possible par cellule (incluant l'état 0 pour les cellules vides) [4]

4. La taille des blocs est établie par la puissance de 2 la plus basse afin d'éviter les problèmes de chevauchement issues de valeurs `long` consécutives.

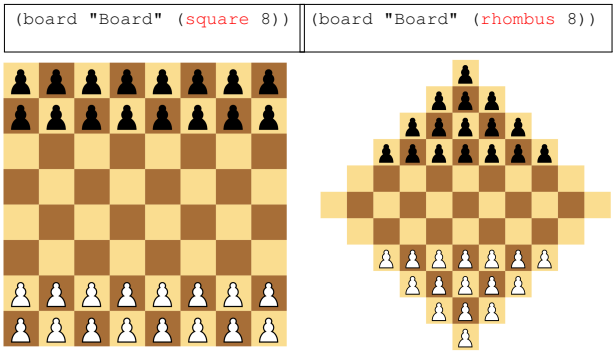


FIGURE 3: Le jeu *Breakthrough* sur un plateau carré (gauche) et sur un plateau losange (droite).

En utilisant l'état de jeu et les différents ludèmes décrivant les règles du jeu, le système calcule les mouvements légaux pour tout état. L'arbre de ludèmes est évalué en retournant la liste d'objets `Action` correspondant au *mover*. Chaque objet `Action` décrit une ou plusieurs actions atomiques qui doivent être appliquées à l'état du jeu pour exécuter le mouvement. Les actions incluent typiquement d'ajouter ou supprimer des composants des conteneurs ou changer le compte des composants par exemple.

Definition 4 *Un trial τ est une suite d'états s_i et une liste d'actions A_i :*

$$\langle s_0, A_1, s_1, \ldots, s_{f-1}, A_f, s_f \rangle$$

tel que $f \geq 0$, et pour tout $i \in \{1, \ldots f\}$,

- *la liste d'actions réalisée A_i est légale pour $mover(s_{i-1})$*
- *les états sont mis à jour : $s_i = \mathcal{T}(s_{i-1}, A_i)$*
- *seul s_f peut être terminal : $\{s_0, \ldots, s_{f-1}\} \cap S_{ter} = \emptyset$*

τ est donné par l'objet `Trial`, fournissant un enregistrement complet du jeu joué du début à la fin incluant les mouvements réalisés.

Dans tout jeu, le raisonnement d'une IA peut être parallélisé en utilisant des *trials* séparés par *thread*. Toutes les données d'un objet `Game` sont constantes et de ce fait peuvent être partagées entre plusieurs *threads*. Un *thread* se référera à un objet `Trial` pour réaliser tout *playout* à partir de n'importe quel état. Dans le système, chaque objet IA décrit l'implémentation d'une IA choisie par le joueur, incluant les limites de temps/budgets de calcul, les indices caractéristiques (*features*) pour biaiser un *playout* [7], etc.

5 Les aptitudes et propriétés clés

LUDII est conçu et implémenté essentiellement pour fournir des réponses aux questions soulevées par le DLP mais sera également une plate-forme pour la recherche autour des jeux dans différents domaines incluant IA, conception, histoire et éducation. LUDII fournit de nombreux avantages par rapport aux système GGP existants, comme suit :

Simplicité : La simplicité fait référence à la facilité avec laquelle les descriptions de jeux peuvent être créées et modifiées. Elle peut être estimée par le nombre de symboles (*tokens*) requis pour définir un jeu. Décrire un jeu avec l'approche ludémique est typiquement beaucoup plus simple que tout approche basée sur la logique (ex : LUDII demande seulement 29 symboles pour *Tic-Tac-Toe* et 298 pour les *Échecs* alors que GDL demande 381 et 4.932 symboles respectivement). De plus, la modélisation ludèmique permet aussi de modifier un jeu facilement pour expérimenter différentes tailles, géométries ou règles. Par exemple, modifier la taille et la forme d'un plateau (ex : Figure 3) peut être accompli en modifiant un seul paramètre, tandis que la même modification avec GDL requiert de nombreuses lignes de code à ajouter ou à modifier.

Clarté : La clarté mesure la capacité de chaque description de jeu d'être comprise pour les non spécialistes. La description de jeu par l'approche logique que propose GDL est souvent difficile à interpréter pour un humain. Dans LUDII, les classes Java définissant chaque ludème sont qualifiées par des noms significatifs décrivant clairement les rôles et concepts qu'ils impliquent. Ceci est particulièrement utile pour les jeux dans lesquels des concepts mathématiques (géométries, algèbres, arithmétique, etc.) sont encapsulés au sein des ludèmes les composants.

Généralité : La généralité se reporte à la quantité de jeux pouvant être couverte par le système. Comme LUDII utilise l'approche grammaticale de classes pour décrire les ludèmes, il peut théoriquement supporter tout jeu pouvant être programmé en Java, la version initiale de LUDII décrit dans ce papier inclut déjà de nombreux jeux différents en dehors de ceux pouvant être implémentés par GDL. La version finale de LUDII sera encore plus générale et facilitera l'ajout de nombreux types de jeux additionnels.

Extensibilité. L'extensibilité décrit la capacité d'ajouter de nouvelles fonctionnalités au système. La version initiale de LUDII propose 68 jeux différents (et 179 variantes) utilisant 259 différents ludèmes, qui testent les différentes options et capacités du système. Étendre LUDII revient simplement à ajouter de nouvelles classes à la bibliothèque de ludèmes, qui seront ensuite automatiquement incorporées à la grammaire, rendant ainsi l'extensibilité très libre. Étendre GDL implique des modifications significatives du langage et du coeur du système correspondant.

Évolutivité : Ceci représente la capacité d'utiliser des descriptions de jeux afin de produire des "enfants" qui ressemblent à leurs "parents". Les descriptions de jeux GDL tendent à impliquer des chaînes d'opérations logiques qui doivent être élaborées avec le plus grand soin. L'application de mutations et croisements aléatoires entre descriptions de jeux GDL est très peu susceptible de produire des résultats corrects (e.i. jouables) et encore moins d'améliorer les parents. Réciproquement, l'approche ludèmique est idéale aux approches évolutives telles que la programmation génétique [15] et a déjà fait ses preuves dans l'évolution/génération de nouveaux jeux de haute qualité [2].

Application Culturelle : Outre ses avantages pour le GGP, LUDII a également plusieurs applications en tant qu'outil pour le nouveau domaine de l'archéoludologie [6]. Le système LUDII sera relié à un serveur et à une base de données contenant les informations culturelles et historiques pertinentes sur les jeux. Ces informations fourniront non seulement un contexte réel supplémentaire, mais permettra également de reconstituer des ensembles de règles viables et historiquement authentiques pour tout jeu contenant certaines informations manquantes, pour développer un "arbre généalogique" des jeux traditionnels et de cartographier la propagation des jeux, à travers l'histoire.

Universalité : Bien que LUDII supporte un large éventail de jeux, y compris des jeux non déterministes à information cachée, nous ne pouvons pas prouver l'universalité de l'ensemble de la grammaire dans le cadre de cet article. Toutefois, nous démontrons que LUDII est universel pour les jeux finis et déterministes à information complète :

Théorème 1 LUDII *est universel pour la classe de jeux finis et déterministes à information complète.*

Dans LUDII, cette classe de jeux ne requiert pas de nature (p_0) et d'informations cachées ($hidden(s, loc, p_i)$ retourne faux pour tout état s, dans toute localisation loc, et pour tout joueur p_i). La preuve est structurée similairement à la preuve d'universalité pour GDL et GDL-II [30]. Basée sur la définition d'un jeu en forme extensive [23], nous formalisons les jeux déterministes et prouvons que LUDII peut définir un arbre de jeux fini arbitrairement. Par conséquent, tous les jeux qui peuvent être modélisés en GDL peuvent aussi être décrit dans LUDII.

5.1 Preuve du Théorème 1

Similairement à Kowalski *et al.* [2019], nous formalisons un jeu fini et déterminisite à k joueurs avec information complète par un tuple (k, T, ι, υ), où :

- $k \in \mathbb{N}$ indique le nombre de joueurs.
- T est un arbre fini avec :
  - Les noeuds S (faisant références aux états du jeu).
  - Un état initial $s_0 \in S$ (la racine de T).
  - Les états terminaux $S_{ter} \subseteq S$ (les feuilles de T).
  - Une fonction prédécesseur $f : (S \setminus \{s_0\}) \mapsto S$, tel que $f(s)$ désigne le parent de s dans T.
- $\iota : (S \setminus S_{ter}) \mapsto \{0, \ldots, k\}$ indiquant quel joueur à le contrôle dans un état donné.
- $\upsilon : S_{ter} \mapsto \mathbb{R}^k$, tel que $\upsilon(s)$ désigne le vecteur de gains pour k joueurs pour état terminal $s \in S_{ter}$.

Cela équivaut à la formalisation des jeux en forme extensives à k joueurs par [23], excluant tous les éléments non déterministes ou à information incomplète.

Nous prouvons que, étant donné tout jeu déterministe fini avec information complète tel que défini ci-dessus, un jeu LUDII peut être construit de telle sorte qu'il existe un mappage un à un entre les états et les transitions d'états entre le jeu original et le jeu LUDII. L'intuition de la preuve est de construire un jeu LUDII où le plateau de jeu est représenté par un graphe avec une structure identique à l'arbre complet de jeu T. Le jeu LUDII se joue en déplaçant un seul jeton, placé sur la racine à l'état initial, le long du graphe jusqu'à atteindre l'une des feuilles. Pour tout état z du jeu d'origine, il existe un état correspondant s dans le jeu LUDII tel que le jeton se trouve sur le sommet correspondant à la position z dans T. Notez qu'énumérer explicitement l'arborescence complète du jeu en tant que graphe n'est probablement pas la représentation la plus optimale pour la plupart des jeux, mais cela montre que LUDII est capable de représenter tous ces jeux.

Definition 5 *Soit $\mathcal{D} = (k, T, \iota, \upsilon)$ le jeu fini déterministe à information complète à k joueurs tel que formalisé ci-dessus. Nous définissons un jeu* LUDII *$\mathcal{G}(\mathcal{D}) = \langle Mode, Equipment, Rules \rangle$, où $Rules = \langle Start, Play, End \rangle$, tel que :*

- *$Mode = \langle p_0, p_1, \ldots, p_k \rangle$, où p_i pour $i \geq 1$ correspond aux différents k joueurs. La joueur environnement p_0 restera inutilisé dans les jeux déterministes.*
- *$Equipment = \langle \{c_0^t\}, \{c_0^p, c_1^p\} \rangle$. Le conteneur $c_0^t = \langle V, E \rangle$ est un graphe possédant une structure identique à l'arbre T du jeu original $\mathcal{D}$. Étant donné que la structure c_0^t étant identique à celle de T, nous pouvons uniquement identifier un sommet $v(z)$ pour chaque état $z \in T$ du jeu original. Pour tout sommet de ce type – à l'exception de $v(s_0)$ – nous pouvons également identifier uniquement un sommet "parent" adjacent $p(v(z))$, tel que $p(v(z)) = v(f(z))$; le parent d'un sommet correspond au prédécesseur de l'état correspondant dans l'arbre d'origine T.*
- *Les règles $Start$ sont données par une liste contenant uniquement une action. Cette action crée l'état initial du jeu en plaçant le jeton c_1^p sur le site $v(s_0)$ de c_0^t qui correspond à la racine de T.*
- *Soit s tout état non terminal du jeu* LUDII *tel qu'il est exactement un site $v(z)$ pour chaque $what(s, \langle c_0^t, v(z), 0 \rangle) = c_1^p$. Soit z l'état dans le jeu original qui correspond au site $v(z)$. Soit $g(z)$ les enfants de z dans T. Étant donné s, nous définissons $Play(s)$ pour retourner un ensemble $\{A_i\}$ de listes d'actions A_i, avec une liste d'actions pour chaque noeud enfant $z' \in g(z)$. Chacune de ces listes contient deux actions primitives; une qui prend le jeton c_1^p du site $v(z)$ (le replaçant par le jeton "vide" c_0^p), et une seconde action qui place ce jeton c_1^p sur le site $v(z')$ de c_0^t qui correspond à l'enfant $z' \in T$.*
- *Les règles End sont données par $End = \{(Cond_i(\cdot), \vec{\mathbb{S}}_i)\}$. Pour tout état terminal $z_i \in S_{ter}$, soit $v(z_i)$ le site dans le graphe c_0^t qui correspond à la position de z_i dans T. Nous ajoutons un tuple $(Cond_i(\cdot), \vec{\mathbb{S}}_i)$ à End tel que $Cond_i(s)$ retourne vrai si et seulement si $what(s, loc) = c_1^p$ pour $loc = \langle c_0^t, v, 0 \rangle$, et $\vec{\mathbb{S}}_i = \upsilon(z_i)$. Intuitivement, nous utilisons une condition de fin séparée pour chaque état terminal*

possible $z_i \in S_{ter}$ dans le jeu original $\mathcal{D}$, qui vérifie sa véracité spécifiquement pour cet état en s'assurant que le jeton c_1^p est placé sur le sommet $v(z_i)$.

— *Soit s l'état* LUDII *non terminal, tel qu'il existe un site $v(z)$ pour chaque $what(s, \langle c_0^t, v(z), 0\rangle) = c_1^p$. Soit z l'état original du jeu qui correspond au site $v(z)$. Alors nous définissons $mover(s) = \iota(z)$.*

Lemme 1 *Soit $\mathcal{G}(\mathcal{D})$ le jeu* LUDII *construit par la Définition 5. Chaque état de jeu s qui peut être atteint par une suite d'actions légales dans un tel jeu a exactement un sommet $v \in c_0^t$ tel que $what(s, \langle c_0^t, v, 0\rangle) = c_1^p$, et $what(s, \langle c_0^t, u, 0\rangle) = c_0^p$ pour tous les autres sommets $u \neq v$.*

Intuitivement, ce lemme dit que chaque état de jeu accessible par une suite d'actions légales a le jeton c_1^p situé sur exactement un sommet et que tous les autres sommets sont toujours vides (indiqué par c_0^p).

Preuve 1 *Soit s_0 l'état initial du jeu. Les règles $Start$ sont définies en plaçant un jeton c_1^p sur $v(z_0)$, où z_0 est l'état initial du jeu d'origine $\mathcal{D}$, signifiant que le lemme est valide pour s_0.*
Soit s l'état non terminal pour lequel le lemme est validé. Alors les hypothèses de la Définition 5 pour une définition adéquate de $Play(s)$ sont satisfaites, ce qui signifie que $\{A_i\} = Play(s)$ est un ensemble non vide de listes d'actions, dont l'une doit être sélectionnée par le $mover(s)$. Chaque A_i est définie pour déplacer le jeton c_1^p d'un sommet où il est actuellement, pour le placer sur un nouveau sommet. Cela signifie que le lemme est également valable pour tout état successeur $\mathcal{T}(s, A_i)$, ce qui prouve que le lemme est valide par induction pour tout état.

Nous sommes maintenant prêt à prouver le Théorème 1 :

Preuve 2 *Soit $\mathcal{D}$ un jeu en forme extensive, avec un arbre T. Soit $\mathcal{G}(\mathcal{D})$ un jeu* LUDII *construit par la Définition 5. Nous démontrons que pour toute traversée arbitraire à travers T, de s_0 à tout état terminal $z_{ter} \in S_{ter}$, il existe un trial τ équivalent, tel que définit dans la Définition 4, dans $\mathcal{G}(\mathcal{D})$. Par trial "équivalent", nous voulons dire que la séquence d'états traversés est de même taille, que l'ordre dans lequel les joueurs sont en contrôle est égale et que les vecteurs de gains sont les mêmes.*

Soit $z_0, z_1, \ldots, z_f$ un playout arbitraire de $\mathcal{D}$, tel que z_0 est l'état initial, et $z_f \in S_{ter}$. Par construction, l'état initial s_0 de $\mathcal{G}(\mathcal{D})$ a le jeton c_1^p placé sur le sommet $v(z_0)$ correspondant à la racine de T. Cela signifie que nous avons une correspondance un à un de z_0 à s_0, où $what(s_0, \langle c_0^t, v(z_0), 0\rangle) = c_1^p$.

Soient z_i les états non terminaux dans la séquence $z_0, z_1, \ldots, z_{f-1}$, tels que nous ayons une correspondance unique z_i sur un état LUDII *s_i où $what(s_i, \langle c_0^t, v(z_i), 0\rangle) = c_1^p$. Le Lemme 1 garantit que les hypothèses requises pour la Définition 5 pour bien définir $Play(s_i)$ sont satisfaites. De plus, nous savons que $\iota(z_i) = mover(s_i)$, ce qui signifie que le même joueur a le contrôle. La définition de $Play(s_i)$ assure qu'il y a exactement une liste d'actions légales A_i tel que $\mathcal{T}(s_i, A_i) = s_{i+1}$, où $what(s_{i+1}, \langle c_0^t, v(z_{i+1}), 0\rangle) = c_1^p$ (Notons que z_{i+1} doit être le successeur de z_i dans T). Nous choisissons s_{i+1} pour correspondre uniquement à z_{i+1}.*

Par induction, ceci termine la correspondance unique entre les séquences d'états $z_0, z_1, \ldots, z_f$ et $s_0, s_1, \ldots, s_f$, spécifiant uniquement la liste d'actions A_i qui doit être sélectionnée en cours de route, et assure que z_i correspond toujours à l'état s_i tel que le jeton c_1^p est placé sur $v(z_i)$. Cette dernière observation assure que l'une des conditions End dans $\mathcal{G}(\mathcal{D})$ est déclenchée pour s_f, et que le bon vecteur de gain $\vec{\mathbb{S}} = v(z_f)$ est sélectionné.

6 Expériences

Le système LUDII, comme la plupart des autre systèmes GGP, utilise MCTS comme méthode de base pour les IA, car cette méthode s'est avérée être une approche supérieure pour les jeux en général en l'absence de connaissances spécifiques à un domaine [9]. Les simulations MCTS (e.i. *playouts*) nécessitent des moteurs de raisonnement rapides pour obtenir le nombre souhaité de simulations. Par conséquent, nous utilisons des simulations *flat Monte Carlo* (i.e. *trials* τ où $s_f \in S_{ter}$) comme métrique pour comparer l'efficacité de LUDII sur les autres systèmes GGP.

6.1 Schéma expérimental

Dans la comparaison suivante, nous comparons GGP-BASE et LUDII en nous basant sur le nombre de *playouts* obtenu par seconde. Pour GGP-BASE, nous utilisons les descriptions des jeux les plus rapides du dépôt [27] et testons l'un des raisonneurs GGP-BASE les plus efficaces basé sur les réseaux de propositions (*propnets*) [29].

Les *Propnets* accélère le processus de raisonnement par rapport aux raisonneurs personnalisés ou basés sur le *Prolog* en traduisant les règles GDL en un graphe dirigé ressemblant à un circuit logique, dont les noeuds correspondent à des portes logiques ou à des propositions GDL qui représentent l'état, les mouvements des joueurs et d'autres aspects du jeu. Les informations sur l'état actuel peuvent être calculées en définissant la valeur de vérité des propositions qui correspondent à l'état et en propageant ces valeurs à travers le graphique. Définir et propager les valeurs de vérité des propositions qui correspondent aux actions des joueurs permet de calculer l'état suivant.

Toutes les expérimentations sont réalisées sur un seul coeur d'un Intel(R) XEON(R) CPU E5-2680 v2 à 2.80 GHz avec 2GB RAM, passant 10 minutes par test.

6.2 Résultats

Les résultats de nos expérimentations sur une sélection de jeux GDL disponibles sont présentés dans le Tableau 1. La partie haute du tableau est dédiée aux jeux à un joueur et la partie basse aux jeux multi-joueurs. La colonne la plus à droite illustre le facteur d'accélération obtenu par LUDII sur GGP-BASE.

Jeu	LUDII	GDL	Taux
Jeux à un joueur			
8 Puzzle	26.958	4.113	6,55
8 Queens	963.443	1.946	495,10
16 Queens	785.973	522	575,85
Futoshiki (4×4)	294.772	23.797	12,39
Futoshiki (5×5)	273.348	11.494	23,78
Futoshiki (6×6)	182.245	5.838	31,56
Knight's Tour (8×8)	91.156	75.000	1,21
Lights Out (5×5)	32.889	11.799	2,79
Nonogram (5×5)	279.242	59.044	4,73
Nonogram (10×10)	45.817	6.883	6,65
Peg Solitaire	7.713	3,.72	2,43
Sudoku	88.565	635	139,47
Towers of Hanoi 3	264.345	35.847	7,37
Jeux Multi-joueurs			
Amazons (10×10)	2.933	185	15,85
Breakthrough (6×6)	19.022	3.923	4,85
Breakthrough (8×8)	6.632	1.123	5,91
Chess	1.075	0,06	17,916,67
Connect 4 (6×7)	124.349	13.664	9,10
Connect 4 (12×9)	99.697	9.856	10,12
English Draughts	1.914	872	2,19
Gomoku (15×15)	2.514	927	2,71
Hex (9×9)	21.244	195	108,94
Knightthrough (8×8)	5.085	1.869	2,72
Reversi (8×8)	2.085	203	10,27
Skirmish (8×8)	2.278	124	18,37
Tic-Tac-Toe (3×3)	641.445	85.319	7,52
Tic-Tac-Toe (5×5)	197.972	11.453	17,29
Tron (10×10)	494.999	121.989	4,06
Wolf and Sheep (8×8)	22.305	5.532	4,03

TABLE 1: Le nombre moyen de *playouts* par seconde pour des jeux disponibles dans le dépôt GGP Tiltyard.

Jeu	LUDII	Jeu	LUDII
Alquerque	796	Mu-Torere	7.719
Ashtapada	12.755	Nine men's morris	5.121
Connect 6 (19×19)	21.873	Oware	3.275
Dara	5.408	Ploy	819
Fanorona	1.906	Tant Fant	5.549
Hnefatafl (11×11)	326	Three men's morris	100.174
MineSweeper (8×8)	68.233	Yavalath	189.335

TABLE 2: Le nombre moyen de *playouts* par seconde pour des jeux indisponibles en GDL.

Le Tableau 2 met en évidente nos résultats sur une sélection de jeux, incluant plusieurs jeux historiques qui n'ont pas d'équivalence en GDL. Le fait qu'aucun système GGP ou langage de description existant ne prenne en charge l'ensemble des jeux requis pour le DLP a été une motivation majeure pour le développement de LUDII.

6.3 Discussion

LUDII surpasse GGP-BASE en terme d'efficacité pour l'ensemble des jeux testés. L'amélioration de performance pour les jeux à un joueur (puzzles) s'accroit avec la taille des puzzles (ex : *Futoshiki* de 12.39 à 31.56). Pour quelques uns proposant des descriptions GDL optimisées (ex : *Knight's Tour*) LUDII accomplit des performances similaires, mais pour d'autres (ex : *N Queens* ou *Sudoku*) LUDII obtient un nombre de *playouts* significativement plus important, de 100 à pratiquement 600 fois plus rapide.

LUDII est au moins 2 fois plus rapide pour tous les jeux multi-joueurs testés. Pour des jeux simples, la taille du plateau est étroitement liée à l'amélioration de la vitesse ; LUDII est près de 7 fois plus rapide que le *Tic-Tac-Toe* 3×3 mais plus de 17 fois plus rapide pour une version plus grande de 5×5. Pour des jeux plus complexes, tels que *Amazons* et *Skirmish*, LUDII est une fois encore plus efficace que GDL (plus de 15 fois rapide dans ces cas).

La plus grande différence de vitesse concerne les *Échecs*, avec un taux d'amélioration de près de 18.000. La description GDL pour les *Échecs* ne peut être traduite en un réseau de proposition suite à une taille très importante dépassant la mémoire disponible, c'est pourquoi nous utilisons un prouveur GDL disponible dans le GGP-BASE pour comparaison. Le GGP-BASE prouveur est généralement plus lent pour les jeux complexes comparés au *propnet*, expliquant le faible nombre de *playouts* obtenu $(0, 06)$.

Le DLP nécessite la capacité de modéliser un large éventail de jeux de stratégies traditionnels. Dans le Tableau 2, LUDII démontre sa capacité à raisonner sur une variété de différents jeux : jeux de courses (ex : *Ashtapada*), jeux à information incomplète (ex : jeux de cartes ou *MineSweeper*), jeux Mancala (ex : *Oware*), jeux avec un plateau imposant (ex : *Connect-6*), etc. Cependant, pour *Hnefatal*, le plus connu des jeux de *Tafl*, la version actuelle de LUDII obtient un faible nombre de *playouts* par seconde. Ceci est probablement dû au fait que les *playouts* peuvent être long, souvent supérieur à 1.000 mouvements, car il est très peu probable que la victoire de chaque joueur soit atteinte en jouant aléatoirement. Des travaux sont en cours pour résoudre ce problème dans la version suivante de LUDII.

Kowalski *et al.*[14] ont récemment proposé un nouveau langage le *Regular BoardGames* (RBG) basé sur les langages réguliers, qui fournit une meilleure expressivité, efficacité et clarté que GDL. Nos premières recherches autour de ce nouveau langage révèlent qu'il est concis mais limité aux jeux de société déterministes dont la géométrie peut être décrite au format ASCII. Il semble également être moins efficace que l'approche grammaticale de classes ; ex : Pour le jeu de *Hex* 9×9, RBG obtient 3.425 *playouts* par seconde mais LUDII en obtient 21.244. De futurs travaux seront conduis afin d'établir une comparaison plus détaillée entre ces deux approches.

7 Conclusion

Le système ludèmique de GGP LUDII surpasse GGP-BASE, l'actuel standard pour la recherche académique en IA autour de GGP, en termes de raisonnement. Il apporte aussi de nombreuses aptitudes inégalées par aucun autre système en terme de simplicité, de clarté, de généralité, extensibilité, évolutivité et il est réalisé de sorte d'être applicable à d'autres champs de recherche que l'IA.

Les bénéfices de cette nouvelle approche GGP présentent plusieurs opportunités pour de futurs travaux. Par exemple, la découverte de nouvelles *features* par l'apprentissage par renforcement pourra être visualisé automatiquement afin de révéler des nouvelles stratégies efficaces ou

fournir des descriptions de stratégies humainement compréhensibles basées sur les ludèmes. Améliorer MCTS en le polarisant par ces *features* est un travail en cours.

Références

[1] Yngvi Björnsson et Stephan Schiffel, General Game Playing, *Handbook of Digital Games and Entertainment Technologies*, pp. 1–23, 2016.

[2] Cameron B. Browne, *Automatic generation and evaluation of recombination games*, Queensland University of Technology, 2009.

[3] Cameron B. Browne, *Evolutionary Game Design*, Springer, 2011.

[4] Cameron Browne, Julian Togelius et Nathan Sturtevant, Guest Editorial : General Games, *IEEE Transactions on Computational Intelligence and AI in Games*, Vol. 6, pp. 317–319, 2014.

[5] Cameron B. Browne, A Class Grammar for General Games, *Advances in Computer Games*, Vol. 10068, pp. 167–182, 2016.

[6] Cameron B. Browne, Back to the Past : Ancient Games as a New AI Frontier, *AAAI*, 2017.

[7] Cameron Browne, Dennis J.N.J. Soemers et Eric Piette, Strategic Features for General Games, *Proceedings of the 2nd Workshop on Knowledge Extraction from Games co-located with 33rd AAAI Conference on Artificial Intelligence*, pp. 70–75, 2019.

[8] Hilmar Finnsson et Yngvi Björnsson, Simulation-Based Approach to General Game Playing, *The Twenty-Third AAAI Conference on Artificial Intelligence*, pp. 259–264, 2008.

[9] Hilmar Finnsson and Yngvi Björnsson, Learning Simulation Control in General Game-Playing Agents, *The Twenty-Fourth AAAI Conference on Artificial Intelligence*, pp. 954–959, 2010.

[10] Jose M. Font, Tobias Mahlmann, Daniel Manrique et Julian Togelius, Applications of Evolutionary Computation, *16th European Conference, EvoApplication Proceedings*, Vol. 7835 LNCS, pp. 254–263, 2013.

[11] Michael R. Genesereth, Nathaniel Love et Barney Pell, General Game Playing : Overview of the AAAI Competition, *AI Magazine*, Vol. 26, pp. 62-72, 2005.

[12] Michael R. Genesereth et Yngvi Björnsson, The International General Game Playing Competition, *AI Magazine*, Vol. 34, pp. 107-111, 2013.

[13] Frédéric Koriche, Sylvain Lagrue, Éric Piette et Sébastien Tabary, Constraint-Based Symmetry Detection in General Game Playing, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pp. 280-287, 2017.

[14] Jakub Kowalski, Mika Maksymilian, Jakub Sutowicz et Marek Szykula, Regular Boardgames, *The Thirty-Third AAAI Conference on Artificial Intelligence*, 2019.

[15] John Koza, *Genetic Programming*, MIT Press, 1992.

[16] Nathaniel Love, Timothy Hinrichs, David Haley, Eric Schkufza et Michael Genesereth, *General Game Playing : Game Description Language Specification*, Stanford University, 2008.

[17] Tobias Mahlmann, Julian Togelius et, Georgios N. Yannakakis, Modelling and evaluation of complex scenarios with the Strategy Game Description Language, *IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 174–181, 2011.

[18] Jeff Mallet et Mark Lefter, Zillons of Games : Unlimited Board Games & Puzzles, `hhttps://www.zillions-of-games.com`, 1998.

[19] Joe Marks et Vincent Hom, Automatic Design of Balanced Board Games, *Proceedings of the Third Artificial Intelligence and Interactive Digital Entertainment Conference* pp. 25–30, 2007.

[20] Harold J. R. Murray, *A History of Board-Games Other Than Chess.*, Clarendon Press, 1952.

[21] David Parlett, What's a Ludeme ?, *Game Puzzle Design*, Vol. 2, pp. 83–86, 2016.

[22] Jacques Pitrat, Realization of a general game-playing program, *IFIP Congress*, Vol. 2, pp. 1570–1574, 1968.

[23] Eric Rasmusen, *Games and Information : An Introduction to Game Theory, 4th ed.*, B. Blackwell, 2007.

[24] Tom Schaul, Julian Togelius et Jürgen Schmidhuber, Measuring Intelligence through Games, *CoRR*, Vol. abs/1109.1314, 2011.

[25] Tom Schaul, An Extensible Description Language for Video Games, *IEEE Transactions on Computational Intelligence and AI in Games*, Vol. 6, pp. 325–331, 2014.

[26] Stephan Schiffel et Michael Thielscher, Representing and Reasoning About the Rules of General Games With Imperfect Information, *Journal of Artificial Intelligence Research*, Vol. 49, pp. 171–206, 2014.

[27] Sam Schreiber, Games-base repository, `http://games.ggp.org/base/`, 2016.

[28] Mohammad Shaker, Mhd Hasan Sarhan, Ola Al Naameh, Noor Shaker et Julian Togelius, Automatic generation and analysis of physics-based puzzle games, *IEEE Conference on Computational Inteligence in Games (CIG)*, pp. 1–8, 2013.

[29] Chiara F. Sironi et Mark H.M. Winands, Optimizing Propositional Networks, *Computer Games. Springer* pp. 133–151, 2017.

[30] Michael Thielscher, The general game playing description language is universal, *Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence, IJCAI-11*, pp. 1107–1112, 2011.

[31] Michael Thielscher, GDL-III : A Description Language for Epistemic General Game Playing, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pp. 1276–1282, 2017.

Analyse formelle de concepts incertains pour l'analyse d'un questionnaire d'évaluation des enseignements

G. Petiot[1]

[1] UR CERES, Institut Catholique de Toulouse, 31 rue de la Fonderie, 31068, Toulouse

guillaume.petiot@ict-toulouse.fr

Résumé

L'Analyse Formelle de Concepts est une méthode d'analyse des données. Plusieurs travaux ont démontré qu'il est possible de compléter l'opérateur de Galois de l'AFC par de nouveaux opérateurs issus de la théorie des possibilités. Il est aussi possible de modéliser les incertitudes avant de les propager lors du calcul des concepts formels. Nous allons nous focaliser sur ce dernier point dans le cadre de cette étude. Nous proposons une expérimentation de l'AFC pour l'extraction de connaissances concernant un questionnaire d'évaluation des enseignements d'un cours de professionnalisation en licence. Certaines questions sont des questions ouvertes dont les réponses sont formulées librement. Afin de pouvoir les exploiter dans l'analyse du questionnaire nous réalisons une classification des réponses. Lors de la classification, des incertitudes peuvent apparaître. La méthode que nous proposons permet de prendre en compte ces incertitudes et de calculer une mesure de nécessité pour chaque concept formel. Ensuite, nous utilisons des requêtes pour extraire les concepts formels incertains et nous visualisons le résultat en utilisant un outil de visualisation.

Mots Clef

Analyse formelle de concepts, théorie des possibilités, traitement du langage naturel, réseaux de neurones, fouille de données.

Abstract

Formal Concept Analysis is an approach of data analysis. Several studies have demonstrated that it is possible to complete the Galois operator of the FCA by new operators from possibility theory. Thus, it is possible to model uncertainties before propagating them during the calculation of formal concepts. In this study, we will focus our interest on this last point. We propose experimentation of FCA for the knowledge extraction concerning a satisfaction questionnaire for a course of professionalisation in bachelor. Some of the questions are open questions, which require free answers. In order to be able to use them in the analysis of the questionnaire, we carry out a classification of the answers. During the classification, uncertainties may appear. The proposed method allows us to take into account these uncertainties and to compute a necessity measure for each formal concept. Then, we will use queries to extract uncertain formal concepts and we visualize the result by using a visualization tool.

Keywords

Formal concept analysis, possibility theory, natural language processing, neural networks, data mining.

1 Introduction

La réalisation d'un questionnaire d'évaluation des enseignements à la fin d'un cours à l'université permet de mieux connaître les étudiants et leurs attentes. Elle permet aussi d'évaluer la qualité de l'approche pédagogique, des ressources, des évaluations,... De nombreuses universités en proposent déjà en utilisant par exemple des outils comme Moodle. L'évaluation des enseignements contribue à l'amélioration continue des formations. Il est assez simple de réaliser un questionnaire, par contre son exploitation peut demander des traitements complexes bien connus en fouille de données. Il est aussi possible de proposer des réponses libres afin que les étudiants puissent s'exprimer librement. On peut par la suite réaliser un traitement du langage naturel ou une fouille de texte pour extraire une catégorisation des réponses. Il est ensuite nécessaire d'utiliser une approche permettant d'extraire une synthèse des réponses données pour en faciliter l'exploitation. Les statistiques descriptives et les tests statistiques sont souvent utilisés. Cependant, les tests statistiques peuvent concerner des hypothèses parfois mal connues. De plus l'interprétation que l'on peut faire des statistiques peut être difficile si l'on n'est pas expert dans ce domaine. L'AFC est une alternative aux approches statistiques [3]. Cette méthode connue en analyse des données a pour objectif l'extraction d'un ensemble ordonné de concepts formels. Un concept formel étant composé d'un ensemble d'objets et d'un ensemble d'attributs partagés par ces objets. L'analyse des concepts formels est beaucoup plus facile et intuitive que des statistiques et ne présente pas de perte d'informations. Plusieurs travaux existent concernant des applications de l'AFC au

domaine de l'éducation [1, 11, 15] ou l'analyse des réseaux sociaux [21]. Cependant, on retrouve rarement une prise en compte de l'incertitude dans les traitements. Pourtant, cela est possible, comme le montrent certains travaux de recherche qui proposent une généralisation de l'AFC en utilisant la théorie des possibilités [7, 8, 9, 6, 19, 17]. Nous proposons dans cet article une expérimentation de l'AFC qui prend en compte les incertitudes. Notre application vise à analyser les réponses d'un questionnaire de satisfaction proposé à des étudiants de licence pour un cours de PPP (Projet Personnel Professionnel). Les questions fermées ne présentent pas d'incertitude car les réponses possibles sont connues par avance. Nous avons aussi des questions ouvertes où l'étudiant peut librement s'exprimer sur ce qu'il a apprécié dans le cours. Pour analyser les réponses de ces questions un traitement du langage naturel est nécessaire. Nous proposons de réaliser une classification des réponses. Les méthodes de classification les plus utilisées sont décrites dans [13], on retrouve la méthode SVM, les classifieurs bayésiens naïfs, les k plus proches voisins, les arbres de décision, les réseaux de neurones,... Dans notre application, nous avons choisi d'utiliser un réseau de neurones. Le processus de classification génère des incertitudes sur l'appartenance d'une réponse à une classe. Ces incertitudes doivent être prises en compte dans l'AFC. L'approche que nous proposons peut se généraliser à tous les questionnaires qui comportent des réponses incertaines. Cela permet de prendre en compte l'ensemble des informations, des plus certaines au moins certaines, pour le calcul des concepts formels. On peut ensuite calculer une mesure de certitude pour chaque concept formel. Cet article est organisé de la façon suivante. Nous allons dans une première partie présenter la théorie des possibilités, ensuite nous allons présenter l'AFC et rappeler comment généraliser l'AFC en utilisant la théorie des possibilités. Dans la partie suivante nous allons décrire notre application et comment nous avons réalisé le traitement des questions ouvertes. Enfin, nous proposerons quelques résultats illustrant notre approche.

2 La théorie des possibilités

La théorie des possibilités a été développé par L. A. Zadeh [25] dans le prolongement de la théorie des ensembles flous. Cette théorie permet de modéliser à la fois les imprécisions associées aux connaissances mais aussi leur incertitude. Elle propose aussi une représentation de l'ignorance. Dans [10] on définit une distribution de possibilité π comme une représentation d'un état de connaissance. Par exemple si Ω est l'univers et π_x une distribution de possibilité d'une variable x définie de Ω dans $[0, 1]$, alors si $\pi_x(u) = 0$ alors $x = u$ est impossible. Si $\pi_x(u) = 1$ alors $x = u$ est possible. Par la suite, on appellera Π la mesure de possibilité, *N* la mesure de nécessité, Δ la mesure de possibilité garantie, et ∇ la mesure de nécessité potentielle. Ces mesures sont définies sur l'ensemble des parties de Ω noté $P(\Omega)$ dans $[0, 1]$:

$$\forall A \in P(\Omega), \Pi(A) = \sup_{x \in A} \pi(x). \quad (1)$$

$$\forall A \in P(\Omega), N(A) = 1 - \Pi(\neg A) = \inf_{x \notin A} 1 - \pi(x). \quad (2)$$

$$\forall A \in P(\Omega), \Delta(A) = \inf_{x \in A} \pi(x). \quad (3)$$

$$\forall A \in P(\Omega), \nabla(A) = 1 - \inf_{x \notin A} \pi(x). \quad (4)$$

La théorie des possibilités n'est pas additive mais maxitive :

$$\forall A, B \in P(\Omega), \Pi(A \cup B) = \max(\Pi(A), \Pi(B)). \quad (5)$$

Ces différentes mesures permettent de proposer d'autres opérateurs dans le cadre de l'AFC nécessaire pour l'analyse complète d'une situation. La mesure de possibilité garantie correspond à l'opérateur de Galois utilisé dans l'AFC.

3 L'analyse formelle de concepts

L'analyse formelle de concepts est une méthode d'analyse des données proposée par R. Wille [23] qui consiste à étudier les concepts en utilisant les treillis. L'intention et l'extension permettent de définir un concept. On retrouve ces notions aussi en philosophie. L'intention est tout simplement la définition du concept et l'extension l'ensemble des éléments sur lequel il s'applique. Mathématiquement un contexte est un triplet $(O, P, \Re)$ où $O = \{o_1, ..., o_n\}$ est l'ensemble des objets, $P = \{p_1, ..., p_m\}$ l'ensemble des propriétés possibles et $\Re$ une relation telle que $\Re \subseteq O \times P$. En fait si $(o, p) \in \Re$ alors l'objet o a la propriété p. Généralement on représente cela en utilisant une table dont les lignes sont les objets et les colonnes les propriétés, la relation quant à elle est représentée soit par un 0 si $(o, p) \notin \Re$ ou par un 1 si $(o, p) \in \Re$ et correspond à la valeur de la table. On peut définir une valuation $\vartheta(o, p)$ qui retourne la valeur de la table pour un objet o et une propriété p. Un concept formel de $(O, P, \Re)$ est un couple (X, Y) tel que $X \in O$ et $Y \in P$ tel que Y n'est en fait que l'ensemble des propriétés partagées par l'ensemble des objets de X. On peut le noter $X^{\uparrow} = Y$ ou $Y^{\downarrow} = X$. Par exemple pour le contexte formel ci-dessous, $(\{o_2, o_3, o_5\}, \{p_2, p_3\})$ et $(\{o_4, o_5\}, \{p_1, p_2\})$ sont deux concepts formels.

Objet	p_1	p_2	p_3
o_1	0	1	0
o_2	0	1	1
o_3	0	1	1
o_4	1	1	0
o_5	1	1	1

TABLE 1 – Exemple de contexte formel.

L'ensemble de tous les concepts formels de $(O, P, \Re)$ est noté $\beta(U, V, \Re) = \{(X, Y) | X^{\uparrow} = Y, Y^{\downarrow} = X\}$. Soit un ordre partiel $\leq$ tel que pour $(X_1, Y_1), (X_2, Y_2) \in$

$\beta(U, V, \Re)$ alors $(X_1, Y_1) \leq (X_2, Y_2)$ si $X_1 \subseteq X_2$ ou $Y_2 \subseteq Y_1$. On peut alors construire le treillis de concepts. Un treillis de concepts peut se visualiser en utilisant un diagramme de Hasse. Nous pouvons visualiser le treillis de concepts formels en utilisant l'outil ConExp (*http ://sourceforge.net/projects/conexp*) :

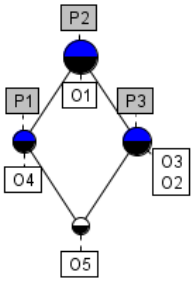


FIGURE 1 – Treillis de concepts formels.

On peut remarquer que la notation est réduite car on utilise l'héritage des propriétés et des objets. Lorsque des propriétés sont multivaluées, il est nécessaire de réaliser une transformation du contexte pour se ramener à un contexte formel binaire. Prenons l'exemple suivant :

Objet	Note	Qualité
o_1	5	faible
o_2	15	bonne
o_3	15	moyenne
o_4	12	faible
o_5	18	moyenne

TABLE 2 – Exemple de contexte multivalué.

Nous pouvons voir que la note est dans l'intervalle $[0, 20]$. On peut de ce fait proposer par exemple une catégorisation des valeurs selon trois classes. La première correspond à la classe faible pour des valeurs dans $[0, 7]$, la seconde est moyenne pour des valeurs dans $[8, 14]$, enfin la dernière est forte pour des valeurs dans $[15, 20]$. On retrouve ci-dessous un exemple de transformation en contexte formel binaire :

Objet	N_{faible}	$N_{moyenne}$	N_{forte}	Q_{faible}	$Q_{moyenne}$	Q_{forte}
o_1	1	0	0	1	0	0
o_2	0	0	1	0	0	1
o_3	0	0	1	0	1	0
o_4	0	1	0	1	0	0
o_5	0	0	1	0	1	0

TABLE 3 – Transformation du contexte multivalué en contexte binaire.

Jusqu'à présent les valeurs des propriétés étaient certaines cependant si ce n'est pas le cas on peut proposer d'utiliser la théorie des possibilités [25] pour prendre en compte les incertitudes comme le proposent les auteurs dans [6]. Ainsi, on peut avoir la distribution de possibilité $\pi_{o_p}(u)$ définie pour $u \in \Omega$ qui est la possibilité que la propriété p de l'objet o soit u. Il faut bien sûr que cette distribution de possibilité soit normalisée. Les auteurs proposent aussi d'étendre l'AFC en définissant quatre opérateurs qui s'inspirent de la théorie des possibilités que l'on peut rappeler brièvement. Si $\Re$ est une relation (ou un contexte) et si $R(o) = \{p \in P | (o, p) \in \Re\}$ et $R^t(p) = \{o \in O | (o, p) \in \Re\}$ alors pour S, un sous-ensemble de O, on a les opérateurs suivants :

— $(S)^{\Pi} = \{p \in P | R^t(p) \cap S \neq \oslash\}$
— $(S)^{N} = \{p \in P | R^t(p) \subseteq S\}$
— $(S)^{\Delta} = \{p \in P | R^t(p) \supseteq S\}$
— $(S)^{\nabla} = \{p \in P | R^t(p) \cup S \neq O\}$

Ci-dessous nous proposons un exemple d'application de ces opérateurs pour différents ensembles d'objets de la Table 1 :

S	$(S)^{\Pi}$	$(S)^{N}$	$(S)^{\Delta}$	$(S)^{\nabla}$
(o_1, o_2, o_3)	(p_2, p_3)	$(\oslash)$	(p_2)	$(\oslash)$
(o_2)	(p_2, p_3)	$(\oslash)$	(p_2, p_3)	(p_1, p_3)
(o_4, o_5)	(p_1, p_2, p_3)	(p_1)	(p_1, p_2)	(p_1, p_3)
(o_1, o_2, o_3, o_4)	(p_1, p_2, p_3)	$(\oslash)$	(p_2)	$(\oslash)$

TABLE 4 – Exemple d'application des opérateurs.

$(S)^{\Pi}$ est l'ensemble des propriétés possédées par au moins un objet de S. $(S)^{N}$ est l'ensemble des propriétés possédées seulement par les objets de S. $(S)^{\Delta}$ est l'ensemble des propriétés partagées par tous les objets de S. $(S)^{\nabla}$ est l'ensemble des propriétés qui ne sont pas satisfaites pour au moins un objet de $\bar{S}$. Le comportement de l'opérateur $(.)^{\Delta}$ est celui que l'on retrouve en général dans l'analyse formelle de concepts. Par la suite, nous allons utiliser cet opérateur. Dans l'article [6] les auteurs rappellent que ces opérateurs ont déjà été proposés sans toutefois faire référence à la théorie des possibilités. Ces opérateurs peuvent être représentés sous la forme d'un cube des oppositions pour l'AFC (*AffIrmo nEgO*) :

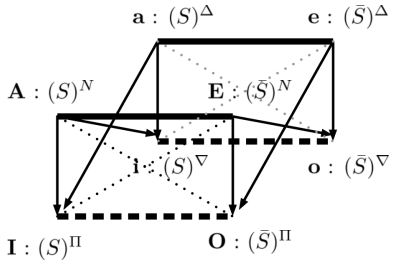


FIGURE 2 – Cube des oppositions.

Ces opérateurs sont définis pour un ensemble d'objets S. On peut facilement proposer des opérateurs équivalents pour des ensembles de propriétés notés $(.)^{-1\Pi}$, $(.)^{-1N}$, $(.)^{-1\Delta}$, $(.)^{-1\nabla}$ comme dans [6]. La prise en compte de l'imprécision pour les propriétés peut se faire en utilisant les ensembles flous comme dans [2]. Il y a par contre moins de travaux concernant la prise en compte de l'incertitude et donc de l'ignorance qui peut être partielle ou totale. La certitude est associée à la mesure de nécessité dans la théorie des possibilités. Cette mesure peut être intégrée à la table et

être propagée lors du calcul des concepts formels. Ainsi, si l'on dispose comme dans [9] d'une paire de mesure de nécessité $(\alpha(o,p), \beta(o,p))$ avec $\alpha(o,p) = N((o,p) \in \Re)$ et $\beta(o,p) = N((o,p) \notin \Re)$, on peut en déduire pour un contexte donné un ensemble de concepts formels. Il faut satisfaire la propriété $min(\alpha(o,p), \beta(o,p)) = 0$ de la théorie des possibilités. Les paires $(1,0)$ et $(0,1)$ représentent le fait qu'un objet a une propriété ou pas. Si $1 > max(\alpha(o,p), \beta(o,p)) > 0$, l'ignorance est partielle, et si l'on a $(0,0)$, l'ignorance est totale. On peut définir un contexte formel incertain comme ci-dessous :

$$\Re' = \{(\alpha(o,p), \beta(o,p)) | o \in O, p \in P\} \tag{6}$$

Pour cette première expérimentation nous allons transformer le contexte incertain en remplaçant les valeurs $(\alpha(o,p), 0)$ par 1 et $(0, \beta(o,p))$ par 0. On remplace les valeurs incertaines par des valeurs certaines. Par exemple, $(0.2, 0)$ peut être transformé en 1 et $(0, 0.6)$ en 0. On obtient ainsi un nouveau contexte formel pour lequel on peut facilement extraire les concepts formels. Une fois les concepts formels extraits, il est possible de calculer la mesure de nécessité d'un concept formel $C = (X, Y)$ donc sa certitude en utilisant la formule suivante :

$$N(C) = \min_{o \in X, p \in Y} N((o,p) \in \Re) \tag{7}$$

Nous présentons ci-dessous un exemple de contexte formel incertain pour illustrer ce calcul de certitude :

Objets	p_1	p_2	p_3
o_1	(0,1)	(1,0)	(0.2,0)
o_2	(0,0.5)	(1,0)	(1,0)
o_3	(0.5,0)	(1,0)	(0,0.9)
o_4	(1,0)	(1,0)	(0.8,0)
o_5	(1,0)	(1,0)	(1,0)

TABLE 5 – Exemple de contexte formel incertain.

En transformant ce contexte comme décrit précédemment nous obtenons :

Objets	p_1	p_2	p_3
o_1	0	1	1
o_2	0	1	1
o_3	1	1	0
o_4	1	1	1
o_5	1	1	1

TABLE 6 – Transformation du contexte formel incertain en contexte formel binaire.

Dans cet exemple on peut voir que $(\{o_1, o_2, o_4, o_5\}, \{p_2, p_3\})$, $(\{o_3, o_4, o_5\}, \{p_1, p_2\})$, $(\{o_4, o_5\}, \{p_1, p_2, p_3\})$ et $(\{o_1, o_2, o_3, o_4, o_5\}, \{p_2\})$ sont les concepts formels de ce contexte formel. On obtient pour ces concepts formels les résultats suivants pour le calcul de la certitude :

Concept formel	Certitude
$(\{o_1, o_2, o_4, o_5\}, \{p_2, p_3\})$	0.2
$(\{o_3, o_4, o_5\}, \{p_1, p_2\})$	0.5
$(\{o_4, o_5\}, \{p_1, p_2, p_3\})$	0.8
$(\{o_1, o_2, o_3, o_4, o_5\}, \{p_2\})$	1

TABLE 7 – Calcul de la certitude des concepts formels.

De nombreux algorithmes existent concernant le calcul des concepts formels [16]. Nous avons choisi pour notre expérimentation d'implémenter l'algorithme de Ganter *Next Closure* [12] qui fait partie des algorithmes les plus connus. Cet algorithme permet de trouver toutes les intentions (il peut aussi permettre de trouver toutes les extensions). Nous l'avons adapté au calcul des concepts formels incertains. Pour cela, nous proposons de modifier la fonction R pour un seuil de certitude μ qui devient $R_\mu(o) = \{p \in P | N((o,p) \in \Re) > \mu\}$ et $R^t_\mu(p) = \{o \in O | N((o,p) \in \Re) > \mu\}$. Ce qui donne deux nouveaux opérateurs que nous allons utiliser $(.)^{\Delta_\mu}$ et $(.)^{-1\Delta_\mu}$:

$$(S)^{\Delta_\mu} = \{p \in P | R^t_\mu(p) \supseteq S\} \tag{8}$$

$$(S)^{-1\Delta_\mu} = \{o \in O | R_\mu(o) \supseteq S\} \tag{9}$$

On définit aussi l'opérateur $\oplus_\mu$ de la fermeture tel que :

$$X \oplus_\mu i = ((X \cap \{p_1, ..., p_{i-1}\}) \cup \{p_i\})^{-1\Delta_\mu \Delta_\mu} \tag{10}$$

Enfin, nous définissons aussi l'opérateur de comparaison $<_i$ (ordre lexicographique). Si $X \in P$ et $Y \in P$ alors $X <_i Y$ si :

$$\begin{cases} p_i \in Y - X \text{ et} \\ X \cap \{p_1, ..., p_{i-1}\} = Y \cap \{p_1, ..., p_{i-1}\} \end{cases} \tag{11}$$

De plus on a $X < Y$ s'il existe un i pour lequel la relation $X <_i Y$ est vérifiée. L'algorithme de calcul des concepts formels incertains est le suivant :

Algorithme 1 : NextClosure incertain

Entrée : Un contexte incertain R; Le seuil de certitude μ
Sortie : L'ensemble des intentions noté I

1 **début**
2 $V = \oslash^{-1\Delta_\mu \Delta_\mu}$
$Sauver(V)$
tant que $V \neq P$ **faire**
3 **pour** $i \leftarrow |P|$ **à** 1 **faire**
4 $V^+ = V \oplus_\mu i$
si $V <_i V^+$ on sort de la boucle.
5 $Sauver(V^+)$
$V \longleftarrow V^+$

L'ensemble des concepts formels est noté $\beta(U, V, \Re) = \{(X^{-1\Delta}, X)|X \in I\}$. Si on utilise un opérateur de comparaison $\leq$ entre les concepts formels alors on a un treillis de concept.

4 Traitement du langage naturel

Dans le questionnaire de notre expérimentation nous allons traiter une question ouverte dont les réponses peuvent être données librement contrairement aux réponses des questions fermées qui sont limitées à une liste de propositions. Nous avons toutefois une idée des réponses possibles pour cette question ouverte car elle porte sur les parties du cours que les étudiants préfèrent. De ce fait, nous pouvons réaliser une classification des réponses à partir d'une description textuelle de la classe et d'échantillons issus des réponses. Nous avons choisi de réaliser la classification en utilisant un réseau de neurones. La première étape consiste à utiliser une méthode de fouille de texte pour extraire les corpus des descriptions des classes, des échantillons et des autres réponses. Ensuite, nous réalisons un traitement sur ces corpus pour changer la casse, éliminer les caractères indésirables, la ponctuation, les chiffres, et les mots non voulus car ne donnant aucune information sur la réponse. Ci-dessous, nous présentons le processus de traitement des corpus :

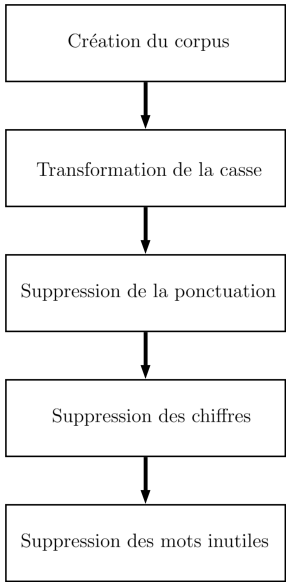


FIGURE 3 – Processus de traitement des corpus.

Ensuite, nous calculons des Matrices Documents-Termes (DTM en anglais pour *Document-Term Matrix*) pour les réponses et les descriptions des classes. La classification utilisera pour l'apprentissage la matrice MDT de la description des classes et des échantillons. La classification se fera en utilisant le dictionnaire des mots utilisés pour l'apprentissage. Un problème important dans le traitement du langage naturel est que lorsque l'on traite des réponses libres dans un sondage il y a très souvent des fautes de syntaxes, et d'orthographe. L'idée que nous avons eue est d'utiliser une mesure de similarité entre les mots lors du calcul de la matrice MDT et lors de la classification. Plusieurs mesures existent pour comparer deux mots [4, 14], les plus connues sont des distances. On peut citer par exemple la distance de Levenshtein, la distance de Jaccard, la distance de Damerau-Levenshtein, la distance de Hamming, la distance de la plus longue sous chaîne commune, la distance de Smith-Waterman ou la distance de Jaro-Winkler [24]. Pour que ces distances deviennent des mesures de similarité normalisées il faut réaliser une transformation afin qu'elles vérifient la définition suivante :
Une mesure de similarité normalisée entre deux mots m_1 et m_2 est une fonction notée λ de $\Omega \times \Omega \rightarrow [0, 1]$ où Ω est l'ensemble des mots possibles. Cette fonction doit respecter les propriétés suivantes :

- $\lambda(m_1, m_2) \in [0, 1]$
- $\lambda(m_1, m_2) = \lambda(m_2, m_1)$
- $\forall m_2, \lambda(m_1, m_1) \geq \lambda(m_1, m_2)$
- $\lambda(m_1, m_1) = 1$

La distance de Jaro-Winkler vérifie cette définition, de plus elle donne de bons résultats comme le décrit l'auteur de [4] et ne nécessite pas de transformation car elle renvoie des résultats dans $[0, 1]$. Cette distance se calcule en utilisant la distance de Jaro entre les mots m_1 et m_2 :

$$d_J(m_1, m_2) = \frac{1}{3}\left(\frac{\chi}{|m_1|} + \frac{\chi}{|m_2|} + \frac{\chi - \tau}{\chi}\right) \quad (12)$$

Avec $|m_i|$ la longueur du mot i, χ le nombre de caractères correspondant (que l'on retrouve dans les deux mots à une distance inférieure à $\left\lfloor \frac{\max(|m_1|,|m_2|)}{2} \right\rfloor - 1$), τ le nombre de transposition (les caractères inversés). La distance de Jaro-Winkler est la suivante :

$$d_{JW}(m_1, m_2) = d_J(m_1, m_2) + \alpha\beta(1 - d_J(m_1, m_2)) \quad (13)$$

Avec α la longueur du préfixe commun aux deux mots avec un maximum de 4 caractères et β un coefficient que l'on prend en général égal à 0.1.
Prenons l'exemple suivant qui concerne l'écriture du mot *intelligence* et comparons la distance de Jaro-Winkler pour différents mots :

Mot	Mesure de Jaro-Winkler
Intelligence	1.0
Inteligence	0.95
Inelligence	0.89
Intelijence	0.92
Hasard	0

TABLE 8 – Comparaison de mots proches de *intelligence* à l'aide de la mesure de Jaro-Winkler.

On peut remarquer que lorsque les mots ressemblent au mot *intelligence* alors la mesure renvoie une valeur proche de 1. Plus le mot est différent, plus la mesure de ressemblance sera proche de 0. Si la mesure n'est pas assez proche

de 1 alors on peut considérer que les mots sont différents. Donc si d_{JW}(mot n°1, mot n°2)$< \eta$ alors mot n°1 $\neq$ mot n°2. Dans les études existantes [20, 5, 18] η est souvent pris entre 0.7 et 0.9. Nous choisissons de prendre $\eta = 0.85$. Avec cette valeur de η le mot *hasard* présent dans le tableau ci-dessus ne serait pas considéré comme proche du mot *intelligence*. Ensuite, nous construisons la matrice MDT. Par exemple, pour notre application nous avons obtenu la matrice MDT suivante :

Etudiants \ Mots	intelligences	gardner	questionnaire	proust	cv	europass	blason	projet	professionnel	personnalise	...
Etudiant n°1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
Etudiant n°2	1.0	0.96	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
Etudiant n°3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.93	...
Etudiant n°4	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
Etudiant n°5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	...
...	...	...	...	...	...	...	...	...	...	...	...
Etudiant n°144	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.92	...

TABLE 9 – Exemple de matrice MDT pour les réponses des étudiants.

L'étape suivante est la classification des réponses par un réseau de neurones. Ce réseau prend en entrée les poids de la matrice MDT des réponses en ne considérant que les mots présents dans la matrice MDT des échantillons. Ensuite on propage ces valeurs dans le réseau de neurones. En sortie nous avons un degré d'appartenance pour chaque classe. Ci-dessous nous présentons le réseau de neurones :

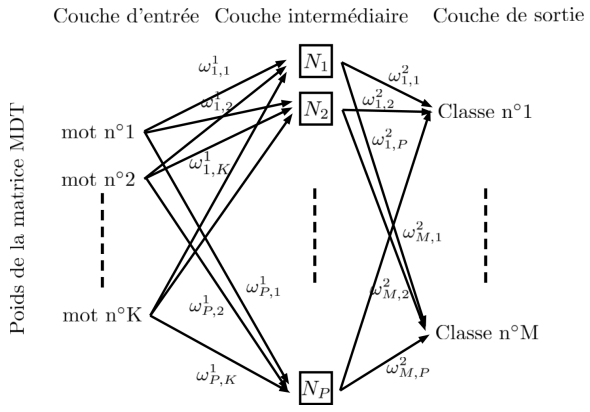


FIGURE 4 – Le réseau de neurones.

L'apprentissage du réseau de neurones est fait en utilisant une rétro-propagation du gradient. Pour cela, nous construisons des groupes d'échantillons qui comprennent des échantillons de chaque classe. Le résultat de la classification peut être vu comme une mesure de possibilité après renormalisation pour chaque classe. Nous pouvons aussi calculer une mesure de nécessité.

5 Expérimentation

L'expérimentation que nous proposons concerne l'analyse d'un questionnaire d'évaluation d'un cours de professionnalisation en licence. 144 étudiants ont répondu au questionnaire. Dans ce questionnaire, il y avait des questions fermées et des questions ouvertes. Pour simplifier le traitement nous prenons en compte uniquement une question ouverte. Celle qui nous semble la plus importante et qui porte sur les parties abordées en cours que les étudiants ont le plus appréciés. Notre objectif est de réaliser une classification des réponses en utilisant le réseau de neurone décrit dans la partie précédente. Nous avons décidé d'associer une classe par partie et de décrire chaque partie du cours avec une phrase. Au final, nous avons 8 classes. Nous avons aussi sélectionné 3 échantillons pour chacune des 8 classes en plus de la description de la classe pour constituer des groupes d'échantillons. Ensuite, nous avons calculé les matrices MDT pour les groupes d'échantillons. Nous en déduisons le nombre de mots en entrée du réseau de neurone $N = 45$, le nombre de classe $M = 8$, et nous choisissons le nombre de neurones pour la couche intermédiaire $P = 34$ (75% de 45 comme décrit dans [22]).
Ensuite, nous avons réalisé l'apprentissage des poids du réseau de neurones avant de faire la classification. Afin de vérifier notre approche nous avons calculé la matrice de confusion en considérant l'ensemble des réponses. Pour cela, nous avons associé pour chaque réponse une classe. La matrice de confusion représente suivant les lignes, les classes des réponses, et suivant les colonnes les classes prédites.

Réelle \ Prédite	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
C_1	**35**	0	0	0	0	0	0	0
C_2	0	**29**	0	0	0	0	0	0
C_3	0	0	**25**	0	0	0	0	0
C_4	0	0	0	**20**	2	0	0	0
C_5	0	0	0	0	**11**	0	0	0
C_6	0	0	0	0	1	**7**	0	0
C_7	0	0	0	0	0	0	**4**	0
C_8	0	0	0	0	2	0	0	**8**

TABLE 10 – Matrice de confusion.

Nous présentons ci-dessous le calcul pour chaque classe de la précision, du rappel et du score F1 :

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	**Moyenne**
Précision	1	1	1	1	0.68	1	1	1	**0.96**
Rappel	1	1	1	0.90	1	0.87	1	0.8	**0.95**
Score F1	1	1	1	0.95	0.81	0.93	1	0.88	**0.95**

TABLE 11 – Calcul de la précision, du rappel et du score F1.

Les erreurs de classification proviennent pour la plupart d'une confusion avec la classe C_5 bien que l'ensemble des réponses identifiées comme appartenant à cette classe soient bien classées. Cette confusion est principalement

due à des formulations ambiguës lors des réponses qui utilisent des mots proches de ceux rencontrés dans les échantillons de la classe C_5.
A partir du résultat de la classification nous calculons pour toutes les réponses une mesure de possibilité et de nécessité pour chaque classe. Parmi les résultats, nous avons observé des problèmes de quasi équipossibilité (4 réponses sur 144) dus à des réponses qui portent sur plusieurs classes. Deux solutions sont possibles pour ces réponses. Soit nous ne les prenons pas en compte dans l'analyse formelle de concepts. Soit nous tentons de les prendre en compte en partant de l'hypothèse que si la mesure de possibilité de plusieurs classes est proches de 1 pour une réponse alors la réponse peut être associée à plusieurs classes. Dans ce cas nous acceptons une erreur si l'équipossibilité représente un cas d'ignorance entre le choix de deux classes. Compte tenu du faible nombre de cas nous choisissons de ne pas prendre en compte ces réponses dans l'analyse formelle de concept (première solution). Cela revient à prendre μ différent de 0 dans notre algorithme de calcul des concepts formels. En prenant $\mu = 0.11$ nous ignorons les réponses anormales dans notre expérimentation.
Il y a en plus 32 autres questions fermées dont les réponses sont multivaluées. Nous avons transformé les données initiales qui sont multivaluées et réalisé un traitement des réponses libres en utilisant l'approche proposée dans la partie précédente. Le résultat est un contexte formel incertain dont les colonnes sont les réponses possibles (propriétés de l'AFC) et les lignes correspondent aux étudiants (objets de l'AFC). L'intégration du traitement de la question ouverte consiste à insérer autant de colonnes que de classes possibles. Les valeurs de la table pour ces colonnes sont une paire de mesures de nécessité. Ce qui donne au final une table de 160 colonnes et 144 lignes. Le contexte formel incertain obtenu est le suivant :

Etudiants	Classe n°1	Classe n°2	Classe n°3	Classe n°4	Classe n°5	...
Etudiant n°1	(0,1)	(0,1)	(0,1)	(0,1)	(0.5,0)	...
Etudiant n°2	(0.99,0)	(0,1)	(0,1)	(0,1)	(0,1)	...
Etudiant n°3	(0,1)	(0,1)	(0,1)	(0,1)	(0.99,0)	...
Etudiant n°4	(0.1,0)	(0,1)	(0,1)	(0,1)	(0,1)	...
Etudiant n°5	(0,1)	(0,1)	(0,1)	(0,1)	(0.91,0)	...
...	...	...	...	...	...	...
Etudiant n°144	(0,1)	(0,1)	(0,1)	(0,1)	(0.99, 0)	...

TABLE 12 – Contexte formel incertain obtenu.

Si nous recherchons les concepts formels qui représentent le plus les réponses des étudiants, nous pouvons définir un score. Si $\beta(U, V, \Re)$ est l'ensemble des concepts formels et (X, Y) un concept formel avec X son extension et Y son intention, alors le score peut-être le suivant :

$$S = \frac{|X| + |Y|}{\max\limits_{(u,v)\in\beta(U,V,\Re)} |u| + |v|} \tag{14}$$

Compte tenu du nombre important de concepts formels, il est nécessaire de pouvoir filtrer les concepts formels pour ne visualiser que ceux qui comportent des informations importantes pour l'utilisateur. Pour cela nous avons réalisé deux traitements sur les concepts formels. Le premier est l'utilisation d'une requête pour extraire les concepts formels qui correspondent à ce que l'on recherche. Le second est la visualisation du résultat. Le résultat de la requête sera représenté par un graphe directionnel. L'orientation des arêtes étant définie par la relation d'ordre partiel entre les concepts formels. La taille de chaque noeud sera proportionnelle au score du concept formel. L'incertitude des concepts sera représentée par un dégradé de couleur (Rouge, Jaune, Vert) allant du rouge pour les moins certains au vert pour les certains. Plusieurs outils de visualisation des données existent. Nous avons choisi d'utiliser l'outil Gephi qui est gratuit et qui propose un certain nombre de fonctionnalités. Les résultats des requêtes sont générés dans des fichiers Excel pour faciliter l'import dans Gephi. Nous proposons pour illustrer ces traitements un exemple de requête :

Q=SELECT c **FROM** $\beta(U, V, \Re)$
WHERE
($c.C_1$ **OR** $c.C_2$ **OR** $c.C_3$ **OR** $c.C_4$ **OR**
$c.C_5$ **OR** $c.C_6$ **OR** $c.C_7$ **OR** $c.C_8$)
AND
$Score(c) \geq 0.1$
AND
$Card(c.X) \geq 15$

Avec $c = (X, Y)$ un concept formel de $\beta(U, V, \Re)$, $Card$ le nombre de propriétés ou d'objets du concept. Enfin, $c.C_i$ est vrai si la propriété C_i qui représente la classe n°i est présente dans Y sinon la valeur retournée est fausse. Le circuit logique qui permet l'évaluation de la requête Q pour chaque concept formel est le suivant :

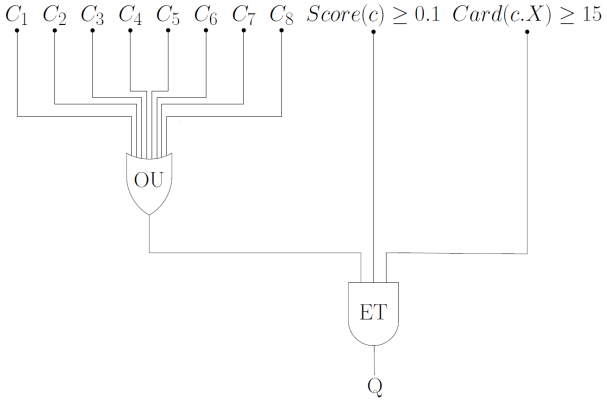


FIGURE 5 – Le circuit logique associé à la requête Q.

Le résultat de la requête est donc une évaluation de ce circuit logique pour chaque concept formel. Compte tenu des opérateurs booléens utilisés, le résultat de chaque évaluation est soit vrai si le concept formel est compatible avec la

requête soit faux dans le cas contraire. Par exemple, pour notre expérimentation nous obtenons le résultat suivant :

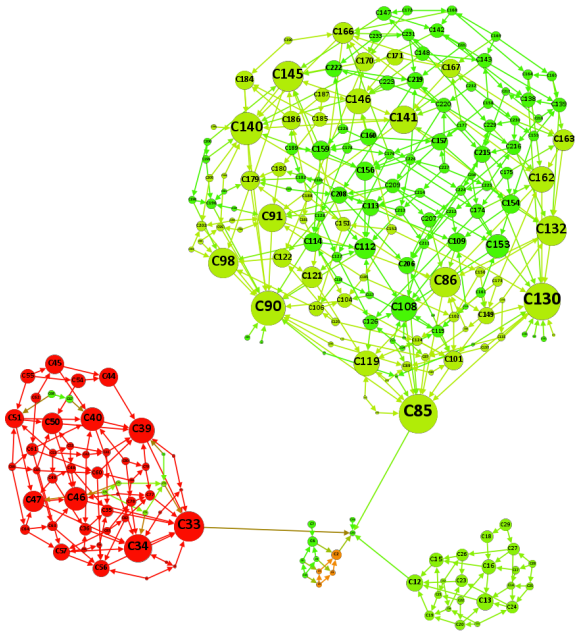


FIGURE 6 – Exemple de résultat d'une requête.

Dans cette figure, on peut voir l'incertitude des concepts formels mais aussi les concepts formels qui sont associés à de nombreuses réponses d'étudiants. Prenons l'exemple du concept formel C85 qui contient dans son intention la propriété associée à la classe n°1 qui représente la partie du cours concernant la théorie des intelligences multiples de H. Gardner et dans son extension 33 étudiants. Ce concept formel semble plus important que les autres car le noeud est d'un rayon supérieur. On peut voir qu'il présente une bonne certitude. La connaissance que nous pouvons extraire de ce concept formel est que de nombreux étudiants ont apprécié la partie du cours portant sur la théorie des intelligences multiples de H. Gardner. Nous pouvons généraliser cette approche d'évaluation des requêtes par des circuits logiques en utilisant des réseaux possibilistes en particulier des portes logiques incertaines (ET, OU, ...). Les réseaux possibilistes permettent de prendre en compte l'imprécision des connaissances que l'on souhaite prendre en compte dans les requêtes, ils permettent aussi de propager l'incertitude et d'obtenir un score de pertinence pour chaque concept formel qui serait la possibilité que le concept formel corresponde à ce que l'on recherche.

6 Conclusion

L'analyse formelle de concepts permet une analyse des données pour en extraire des connaissances. Cependant, il peut s'avérer que certaines données soient incertaines. La théorie des possibilités permet dans ce cas de prendre en compte les incertitudes et de les propager lors du calcul des concepts formels. Nous avons proposé d'utiliser des requêtes pour extraire les concepts formels. Ces requêtes peuvent porter sur les objets ou les propriétés mais aussi sur d'autres informations comme la cardinalité ou un score. Nous avons proposé d'utiliser un outil de visualisation pour mettre en évidence les concepts les plus intéressants en faisant ressortir leur incertitude et le score de chaque concept formel. En termes de perspective, nous souhaitons évaluer l'intérêt pour l'extraction de connaissances des différents opérateurs issus de la théorie des possibilités qui complètent l'opérateur de Galois de l'analyse formelle de concepts. Pour l'instant nous avons pris en compte que des critères booléens dans les requêtes qui peuvent être représentées par un circuit logique. Nous comptons expérimenter l'utilisation des portes logiques incertaines afin d'élaborer des requêtes traduisant des connaissances imprécises et incertaines. Nous comptons proposer un score de pertinence pour l'indexation des concepts formels. Il sera ainsi possible de classer les concepts formels pour faciliter la lecture des résultats.

Références

[1] M. A. Bedek, S. Kopeinik, B. Prünster, D. Albert. Applying the Formal Concept Analysis to Introduce Guidance in an Inquiry-Based Learning Environment. *IEEE 15th International Conference on Advanced Learning Technologies*, pp. 285-289, july, 2015.

[2] R. Bělohlávek. Concept lattices and order in fuzzy logic. *Annals of Pure and Applied Logic*, Vol. 128, Issues 1-3, pp. 277-298, 2004.

[3] R. Bělohlávek, V. Sklenar, J. Zacpal, E. Sigmund. Evaluation of questionnaires by means of formal concept analysis. *In : J. Diatta, P. Eklund, M. Liquiere (Eds.) : CLA 2007, Int. Conference on Concept Lattices and Their Applications*, Montpellier, France, pp. 100-111, October 24-26, 2007.

[4] P. Christen. A Comparison of Personal Name Matching : Techniques and Practical Issues. *Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops, ICDMW '06*, IEEE Computer Society, Washington, DC, USA, pp. 290-294, 2006.

[5] W. W. Cohen, P. Ravikumar, S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. *Proceedings of the 2003 International Conference on Information Integration on the Web*, pp. 73-78, 2003.

[6] D. Dubois, F. Dupin de Saint-Cyr, H. Prade. A Possibility-Theoretic View of Formal Concept Analysis. *Fundamenta Informaticae*, IOS Press, Vol. 75, Amsterdam, The Netherlands, pp. 195-213, 2007.

[7] D. Dubois, H. Prade. Possibility theory and formal concept analysis in information systems. *IFSA-EUSFLAT*, pp. 1021-1026, 2009.

[8] D. Dubois, H. Prade. Possibility theory and formal concept analysis : Characterizing independent sub-

contexts. *Fuzzy Sets and Systems*, Vol. 196, pp. 4-16, 2012.

[9] D. Dubois, H. Prade. Formal Concept Analysis from the Standpoint of Possibility Theory. *In : J. Baixeries, C. Sacarea, M. Ojeda-Aciego (eds) Formal Concept Analysis. ICFCA 2015*. Lecture Notes in Computer Science, Vol 9113. Springer, Cham, pp. 21-38, 2015.

[10] D. Dubois, H. Prade. *Possibility theory : An Approach to Computerized Processing of Uncertainty*, New York, Plenum Press, 1988.

[11] B. Fernandez-Manjon, A. Fernandez-Valmayor. Building Educational Tools Based on Formal Concept Analysis. *Journal of Education and Information Technologies*, Vol. 3, no. 3, pp. 187-201, 1 décembre, 1998.

[12] B. Ganter. Algorithmen zur Formalen Begriffsanalyse. *In B. Ganter, R. Wille, K. Wolf, (eds.), Beitrage zur Begriffsanalyse, Wissenschaftsverlag, Mannheim*, pp. 241-255, 1987.

[13] A. Hotho, A. Nürnberger, G. Paass. (2005). A Brief Survey of Text Mining. *LDV Forum - GLDV Journal for Computational Linguistics and Language Technology*. Vol. 20, pp 19-62, 2005.

[14] M. A. Jaro. Advances in record linking methodology as applied to the 1985 census of Tampa Florida. *Journal of the American Statistical Society*, Vol. 84, no. 406, pp. 414-420, 1989.

[15] M. D. Kickmeier-Rust, M. Bedek, D. Albert. Theory-based Learning Analytics : Using Formal Concept Analysis for Intelligent Student Modelling. *In H. R. Arabnia, D. de la Fuente, R. Dziegiel & E. B. Kozerenko (Eds.), Proceedings of the 2016 International Conference on Artificial Intelligence*, Las Vegas, Nevada, USA, pp. 97-100, 25-28 July 2016.

[16] S. O. Kuznetsov, S. A. Obiedkov. Comparing performance of algorithms for generating concept lattices. *J. Experimental & Theoretical Artificial Intelligence*, Vol. 14, pp. 189-216, 2003.

[17] L. Miclet, H. Prade, D. Guennec. Looking for Analogical Proportions in a Formal Concept Analysis Setting. *Conference on Concept Lattices and Their Applications*, Nancy, France, pp. 295-307, octobre, 2011.

[18] F. Mastjik, C. Varol, A. Varol. Comparison of Pattern Matching Techniques on Identification of Same Family Malware.*International Journal of Information Security Science (IJISS)*, Vol. 4, Issue 3, pp. 104-111, September, 2015.

[19] E. Navarro, H. Prade, B. Gaume. Clustering Sets of Objects Using Concepts-Objects Bipartite Graphs. *In : E. Hüllermeier, S. Link, T. Fober, B. Seeger (eds) Scalable Uncertainty Management. SUM 2012*. Lecture Notes in Computer Science, vol 7520. Springer, Berlin, Heidelberg, pp. 420-432, 2012.

[20] M. del Pilar Angeles, A. Espino-Gamez. Comparison of methods Hamming Distance, Jaro, and Monge-Elkan. *DBKDA 2015, The Seventh International Conference on Advances in Databases, Knowledge, and Data Applications*, pp 63-69, 2015.

[21] Václav Snášel, Zdenek Horák, Ajith Abraham. Understanding Social Networks Using Formal Concept Analysis. *WI-IAT '08 Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, Vol. 03, pp. 390-393, December 09-12, 2008.

[22] V. Venugopal, W. Baets. Neural Networks and Statistical Techniques in Marketing Research : A Conceptual Comparison. *Marketing Intelligence & Planning*, Vol.12, pp. 30-38, 1994.

[23] R. Wille. Restructuring lattice theory : an approach based on hierarchies of concepts. *I. Rival, (ed.) Ordered Sets. Reidel, Dordrecht-Boston*, pp. 445-470, 1982.

[24] W. E. Winkler. The state of record linkage and current research problems. *Statistical Research Division, U.S. Bureau of the Census*, 1999.

[25] L. A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, Vol. 1, pp. 3-28, 1978.

Apprendre à jouer aux jeux à deux joueurs à information parfaite sans connaissance

Quentin Cohen-Solal
CRIL, Univ. Artois et CNRS
62300 Lens, France
cohen-solal@cril.fr

Résumé

Dans cet article, plusieurs techniques pour l'apprentissage par renforcement de fonctions d'évaluation d'états de jeu sont proposées. La première consiste à apprendre les valeurs de l'arbre de jeu au lieu de se restreindre à la valeur de la racine. La seconde consiste à remplacer le gain classique d'un jeu ($+1$ / -1) par une heuristique favorisant les victoires rapides et les défaites lentes. La troisième permet de corriger certaines fonctions d'évaluation en tenant compte de la résolution des états. La quatrième est une nouvelle distribution de sélection d'actions. Enfin, la cinquième est une modification du minimax à profondeur non bornée étendant les meilleures séquences d'actions jusqu'aux états terminaux. En outre, nous proposons une autre variante du minimax non borné, qui joue l'action la plus sure au lieu de jouer la meilleure action. Les expériences menées suggèrent que cela améliore le niveau de jeux lors des confrontations. Enfin, nous appliquons ces différentes techniques pour concevoir un programme-joueur au jeu de Hex (taille 11*) atteignant le niveau de Mohex 2.0 à la suite d'un apprentissage par renforcement contre soi-même sans utilisation de connaissance.*

Mots Clef

Décision séquentielle, Jeux, Planification, Apprentissage, Renforcement, Minimax non borné.

Abstract

In this paper, several techniques for learning game states evaluation functions by reinforcement are proposed. The first is to learn the values of the game tree instead of restricting oneself to the value of the root. The second is to replace the classic gain of a game ($+1$ / -1) with a heuristic favoring quick wins and slow defeats. The third corrects some evaluation functions taking into account the resolution of states. The fourth is a new action selection distribution. Finally, the fifth is a modification of the minimax with unbounded depth extending the best sequences of actions to the terminal states. In addition, we propose another variant of the unbounded minimax, which plays the safest action instead of playing the best action. The experiments conducted suggest that this improves the level of play during confrontations. Finally, we apply these different techniques to design a program-player to the Hex game (size 11) reaching the level of Mohex 2.0 with reinforcement learning from self-play without knowledge.

Keywords

Sequential Decision, Games, Planning, Learning, Reinforcement, Unbound Minimax.

1 Introduction

Une des tâches les plus difficiles en intelligence artificielle est la *prise de décision séquentielle* [18], dont les applications inclues la robotique et les jeux. Concernant les jeux, les succès sont nombreux. La machine dépasse l'homme pour plusieurs jeux, tels que le backgammon, les dames, les échecs et le go [31]. Une classe importante de jeux constitue les jeux à deux joueurs à *information parfaite*, c'est-à-dire les jeux où les joueurs jouent à tour de rôle, sans hasard ni information cachée. Il reste encore de nombreux défis pour ces jeux. Par exemple, pour le *jeu de Hex*, l'homme reste toujours supérieur à la machine. C'est également le cas du *general game playing* [31] (même restreint aux jeux à information parfaite) : l'homme est toujours supérieur à la machine sur un jeu inconnu (lorsque l'homme et la machine ont un temps d'apprentissage relativement court pour maîtriser les règles du jeu). Dans cet article, nous nous concentrons sur les jeux à deux joueurs à information parfaite et à somme nulle, bien que la plupart des contributions de cet article devraient pouvoir s'appliquer ou s'adapter aisément à un cadre plus général.

Les premières approches utilisées pour concevoir un programme capable de jouer à un jeu se basent sur un algorithme de recherche dans les arbres de jeu, tel que le *minimax*, combiné à une fonction d'évaluation des états de jeu, conçue à partir de connaissances expertes. L'avénement de cette technique est le programme Deep Blue [31] au jeu d'échecs. Cependant, le succès de Deep Blue tient en grande partie à la puissance brute de sa machine, lui permettant d'analyser deux cents millions états de jeu par seconde. De plus, cette approche est limitée par le fait de devoir concevoir une fonction d'évaluation manuellement (au moins partiellement). Cette conception est une tâche très complexe, qui doit, en plus, être réalisée pour chaque

jeu. Plusieurs travaux se sont donc focalisés sur l'apprentissage automatique de fonctions d'évaluation [19]. L'un des premiers succès de l'apprentissage de fonctions d'évaluation est sur le jeu Backgammon [31]. Cependant, pour de nombreux jeux, comme le Hex ou le Go, les approches basées sur le minimax, avec ou sans apprentissage automatique, n'ont pas permis de dépasser l'homme. Deux causes ont été identifiées [2]. Premièrement, le très grand nombre d'actions possibles à chaque état de jeu empêche d'effectuer une recherche exhaustive à une profondeur significative (le cours de la partie ne peut être anticipé qu'à un petit nombre de tours à l'avance). Deuxièmement, pour ces jeux, aucune fonction d'évaluation suffisamment performante n'a pu être identifiée. Une approche alternative permettant de résoudre ces deux problèmes a été proposée, donnant notamment de bons résultats au Hex et au Go, nommée recherche arborescente Monte Carlo et notée MCTS (pour Monte Carlo Tree Search) [5]. Cet algorithme explore l'arbre de jeu non uniformément, ce qui constitue une solution au problème du très grand nombre d'actions. De plus, il évalue les états de jeu à partir de statistiques de victoire provenant d'un grand nombre de simulations aléatoires de fin de partie. Il n'a ainsi pas besoin de fonction d'évaluation. Cela n'était cependant pas suffisant pour dépasser le niveau des joueurs humains. Plusieurs variantes de la recherche arborescente Monte Carlo ont alors été proposées, utilisant notamment des connaissances pour orienter l'exploration de l'arbre de jeu et/ou les simulations aléatoires de fin de partie [5]. Les récentes améliorations de la recherche arborescente Monte Carlo ont porté sur l'apprentissage automatique des connaissances du MCTS et leurs utilisations. Ces connaissances ont d'abord été générées par *apprentissage supervisé* [7, 8, 9, 6] puis par apprentissage supervisé suivi d'un *apprentissage par renforcement* [25], et enfin par apprentissage par renforcement seulement [27, 26, 1]. Cela a permis d'atteindre et de dépasser le niveau de champion du monde au jeu de Go avec les dernières versions du programme Alphago [25, 27]. En particulier, Alphago zero [27], qui n'utilise que l'apprentissage par renforcement, n'a donc eut besoin d'aucune connaissance initiale, humaine, pour atteindre son niveau de jeu. Ce dernier succès a toutefois nécessité 29 millions de parties et plus de 6 milliards évaluations d'états de jeu, ce qui est très largement supérieur à ce qu'il a fallu au champion humain du Go pour atteindre son niveau de jeu. Cette approche a également été appliquée aux échecs [26]. Le programme résultant a battu le meilleur programme des échecs, qui était toujours basé sur le minimax.

On peut dès lors se demander si les approches de type minimax sont totalement dépassées ou si les succès spectaculaires des récents programmes tiennent plus de l'apprentissage par renforcement que de la recherche arborescente Monte Carlo. En particulier, on peut se demander si l'apprentissage par renforcement permettrait à une approche de type minimax de rivaliser avec la recherche arborescente Monte Carlo sur les jeux où celle-ci domine le minimax jusqu'à présent, tels que le Go ou le Hex.

Dans cet article, nous nous focalisons par conséquent sur l'apprentissage par renforcement dans le cadre du minimax. Nous proposons de nouvelles techniques pour l'apprentissage par renforcement de fonctions d'évaluation. Nous les appliquons pour concevoir un nouveau programme-joueur au jeu de Hex (sans utiliser d'autres connaissances que les règles du jeu). Nous comparons en particulier ce programme-joueur à Mohex 2.0 [13], le champion au Hex (taille 11) lors des Olympiades informatiques de 2013 à 2017 [11], qui est également le programme-joueur le plus fort publiquement disponible.

Dans la section 2, nous présentons succinctement les algorithmes de jeux et en particulier le minimax à profondeur non bornée sur lequel nous basons plusieurs de nos expériences. Nous présentons également l'apprentissage par renforcement dans les jeux, le jeu de Hex et l'état de l'art des programmes-joueurs sur ce jeu. Dans la section suivante, nous proposons différentes techniques ayant pour objectif d'améliorer l'apprentissage. Ensuite, nous exposons dans la section 4 les expériences réalisées utilisant ces techniques. Enfin, dans la dernière section, nous concluons et exposons les différentes perspectives de recherche.

2 Contexte et travaux connexes

Dans cette section, nous présentons succinctement les algorithmes de recherche dans les arbres de jeux, l'apprentissage par renforcement dans le contexte des jeux ainsi que leurs applications dans le cadre du jeu de Hex (pour plus de détails sur les algorithmes de jeux, voir [31]).

Les jeux peuvent être représentés de manière arborescente par leur *arbre de jeu* (un nœud correspond à un état de jeu et les fils d'un nœud sont les états atteignables par une action). À partir de cette représentation, on peut déterminer l'action à jouer en utilisant un algorithme de recherche dans l'arbre de jeu. Afin de gagner, chaque joueur cherche à maximiser son score (i.e. la valeur de l'état du jeu pour ce joueur à la fin de la partie). Comme nous nous plaçons dans le cadre des jeux à deux joueurs à somme nulle, maximiser le score d'un joueur revient à minimiser le score de son adversaire (le score d'un joueur est la négation du score de son adversaire).

2.1 Recherche dans les arbres de jeu

L'algorithme central est le *minimax* qui détermine récursivement la valeur d'un nœud à partir de la valeur de ses fils et des fonctions $\min$ et $\max$, jusqu'à une profondeur limite de récursion. Avec cet algorithme, l'arbre de jeu est exploré uniformément. Une implémentation plus performante du minimax utilise l'*élagage alpha-bêta* [31] qui permet de ne pas explorer les sections de l'arbre de jeux qui sont moins intéressantes étant données les valeurs des nœuds déjà rencontrés et les propriétés du $\min$ et du $\max$. De nombreuses variantes et améliorations du minimax ont été proposées [21]. Certaines de ces variantes effectuent une recherche à profondeur non bornée (c'est-à-dire que la profondeur de

leur recherche n'est pas fixée) [30]. Contrairement au minimax avec ou sans élagage alpha-bêta, l'exploration de ces algorithmes est non uniforme. L'un de ces algorithmes est la *recherche minimax en meilleur d'abord* [16]. Pour éviter toute confusion avec certaines approches en meilleur d'abord à profondeur fixée, nous appelons cet algorithme le *minimax en meilleur d'abord non borné*, ou plus succinctement UBFM (Unbound Best-First Minimax). UBFM étend itérativement l'arbre de jeu en ajoutant les fils de l'une des feuilles de l'arbre de jeu ayant la même valeur que celle de la racine (valeur minimax). Ces feuilles sont les états obtenus après avoir joué une des meilleures séquences d'actions possibles étant donnée la connaissance actuelle partielle de l'arbre de jeu. Ainsi, cet algorithme étend itérativement les meilleures séquences d'actions *a priori*. Ces meilleures séquences changent généralement à chaque extension. Cela permet d'explorer non uniformément l'arbre de jeu en se focalisant sur les actions les plus intéressantes *a priori* sans pour autant n'explorer qu'une seule séquence d'actions. Dans cet article, nous utilisons la version anytime d'UBFM [16], c'est-à-dire que nous laissons un temps de réflexion fixé à UBFM pour déterminer l'action à jouer. Nous utilisons en plus les tables de transposition [21] avec UBFM, ce qui permet de ne pas avoir à construire explicitement l'arbre de jeu et de fusionner les nœuds correspondant au même état.

2.2 Apprentissage de fonctions d'évaluation

L'apprentissage par renforcement de fonctions d'évaluation peut être effectué par différentes techniques [19, 27, 1, 32]. L'idée générale de l'apprentissage par renforcement de fonctions d'évaluation d'états est d'utiliser un algorithme de recherche dans les arbres de jeu et une fonction d'évaluation adaptative f (par exemple un réseau de neurones) pour jouer une séquence de parties (par exemple contre soi-même, ce qui est le cas dans cet article). Chaque partie va générer des couples (e, v) où e est un état et v la valeur de e calculée par l'algorithme de recherche choisi utilisant la fonction d'évaluation f. L'apprentissage consiste alors à modifier f de façon à ce que pour tous les couples (e, v) générés $f(e)$ se rapproche de v suffisamment pour en constituer une bonne approximation. Au lieu de restreindre l'apprentissage aux données générées lors de la dernière partie, les données des dernières parties peuvent être gardées en mémoire et utilisées lors de l'apprentissage (cette technique est appelée *experience replay* [22]). Cependant, dans cet article, les données utilisées lors d'un apprentissage sont celles générées lors de la dernière partie.

Remarque 1. Les fonctions d'évaluation adaptatives ne servent qu'à évaluer les états non terminaux puisque l'on connait la vraie valeur des états terminaux.

2.3 Distribution de sélection d'actions

Un des problèmes lié à l'apprentissage par renforcement est le *dilemme exploration-exploitation* [19]. Celui-ci consiste à choisir entre explorer de nouveaux états pour apprendre de nouvelles connaissances et exploiter les connaissances acquises. De nombreuses techniques ont été proposées pour gérer ce dilemme [20]. Cependant la plupart de ces techniques ne passent pas à l'échelle car leur application nécessite de mémoriser tous les états rencontrés. Pour cette raison, dans le cadre des jeux à grand nombre d'états, les travaux se basent sur une exploration probabiliste [32, 27, 19]. Avec cette approche, exploiter, c'est jouer la meilleure action et explorer, c'est jouer au hasard uniformément. Une distribution de probabilité paramétrique est alors utilisée pour associer à chaque action sa probabilité d'être jouée. Le paramètre associé à la distribution correspond au taux d'exploration compris entre 0 et 1, que nous notons ϵ (le taux d'exploitation est donc $1 - \epsilon$, que nous notons ϵ'). Celui est souvent fixé expérimentalement. Un *recuit simulé* [15] peut cependant être appliqué afin de ne pas avoir besoin de choisir une valeur pour ce paramètre. Dans ce cas, au début de l'apprentissage par renforcement, le paramètre vaut 1 (on ne fait qu'explorer). Celui-ci diminue au fur et à mesure jusqu'à atteindre 0 à la fin de l'apprentissage. La distribution de sélection d'actions la plus simple est ϵ-*greedy* [32] (de paramètre ϵ). Avec cette distribution, l'action est choisie uniformément avec probabilité ϵ et la meilleure action est choisie avec probabilité $1 - \epsilon$. La distribution ϵ-greedy a le désavantage de ne pas différencier les actions (hormis la meilleure action) en termes de probabilités. Une autre distribution est souvent utilisée, corrigeant cette inconvénient. Il s'agit de la fonction *softmax*, également appelée *distribution de Boltzmann* [19].

2.4 Jeu de Hex

Le jeu de Hex est un jeu de stratégie combinatoire pour deux joueurs. Il se joue sur un plateau hexagonal $n \times n$ vide. On dira qu'un plateau $n \times n$ est de taille n. Le plateau peut être de taille quelconque, bien que les tailles classiques soient 11, 13 et 19. A son tour, chaque joueur pose une pièce de sa couleur sur une case vide (chaque pièce est identique). Le but du jeu est d'être le premier à relier les deux bords opposés du plateau correspondant à sa couleur. La figure 1 illustre une fin de partie. Bien que ces règles soient simplistes, les tactiques et stratégies au jeu de Hex sont complexes. Le nombre d'états et le nombre d'actions par état sont très importants, similaires au jeu de Go. Le nombre d'états est par exemple supérieur à celui des échecs, au moins dès le plateau de taille 11 (table 6 de [29]). Quelle que soit la taille du plateau, le premier joueur a une stratégie gagnante [3] qui est inconnue, sauf pour les plateaux de taille inférieure ou égale à 9 [23]. En fait, résoudre un état particulier est PSPACE-complet [24, 4].

2.5 Programmes-joueurs au Hex

De nombreux programmes-joueurs ont été développés au Hex. Par exemple, Mohex 1.0 [13] est un programme basé sur la recherche arborescente Monte Carlo. Il utilise en plus de nombreuses techniques dédiées au Hex, basées sur des résultats théoriques spécifiques au jeu de Hex. Il est en particulier capable de déterminer une stratégie gagnante pour

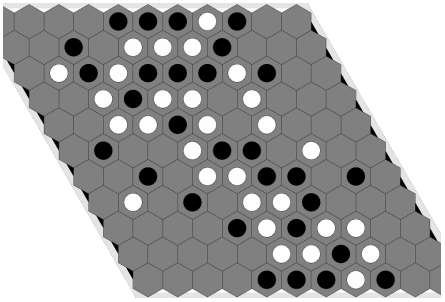


FIGURE 1 – Fin de partie au Hex, taille 11 (blanc gagne)

certains états et d'élaguer à chaque état de nombreuses actions qu'il sait être *inférieures*. Il utilise également des connaissances ad-hoc pour biaiser les simulations de la recherche arborescente Monte Carlo.

Mohex 2.0 [13] est une amélioration de Mohex 1.0 qui utilise des connaissances apprises par apprentissage supervisé pour orienter à la fois l'exploration de l'arbre et les simulations de la recherche arborescente Monte Carlo. Cet apprentissage supervisé a permis d'apprendre les corrélations de victoire de motifs du plateau.

D'autres travaux se sont ensuite focalisés sur la prédiction des meilleures actions, par apprentissage supervisé d'une base de données de parties, à l'aide d'un réseau de neurones. Le réseau de neurones est utilisé pour apprendre une *politique*, c'est-à-dire une distribution de probabilité a priori sur les actions à jouer. Ces probabilités a priori sont utilisées pour guider l'exploration de la recherche arborescente Monte Carlo. Il y a d'abord Mohex-CNN [8] qui est une amélioration de Mohex 2.0 utilisant un réseau de neurones convolutionnel [17]. Une nouvelle version de Mohex a ensuite été proposée : Mohex-3HNN [9]. Contrairement à Mohex-CNN, il est basé sur un réseau de neurones résiduel [12]. Il calcule, en plus de la politique, une valeur pour les états *et* les actions. La valeur des états remplace l'évaluation des états à base de simulations de la recherche arborescente Monte Carlo. L'ajout d'une valeur pour les actions permet à Mohex-HNN de diminuer le nombre d'appel du réseau de neurones, améliorant ainsi les performances.

Des programmes apprenant la fonction d'évaluation par renforcement ont également été conçus. Ces programmes sont NeuroHex [32], EZO-CNN [28] et ExIt [1]. Ils apprennent en jouant contre eux-même. Contrairement aux deux autres, NeuroHex effectue au préalable un apprentissage supervisé afin de diminuer le temps d'apprentissage par renforcement (en apprenant une heuristique commune du jeu de hex). NeuroHex démarre en plus ses parties d'un état provenant d'une base de données de parties. EZO-CNN, lui, utilise durant l'apprentissage des connaissances lui permettant de jouer (et donc d'apprendre) une stratégie gagnante dans certains états. ExIt, quand à lui, apprend une politique en plus de la valeur des états de jeu. Il est le seul programme a avoir appris à jouer au Hex sans utiliser de connaissances. Ce résultat est cependant limité aux plateaux de taille 9. Un comparatif des caractéristiques principales de ces différents programmes est présenté dans la table 1.

3 Techniques proposées

Nous proposons maintenant plusieurs techniques visant à améliorer et/ou accélérer l'apprentissage de fonctions d'évaluation d'états de jeu. Nous terminons cette section en proposant une légère modification d'UBFM que nous appelons UBFM_s afin d'améliorer le niveau de jeu.

3.1 Apprentissage de l'arbre de jeu

Avec les différents travaux de la littérature sur l'apprentissage par renforcement de fonctions d'évaluation, les états e des couples (e, v) utilisés lors de l'apprentissage sont les différents états du jeu obtenus au cours de la partie (il ne s'agit pas des états générés lors de la recherche dans l'arbre de jeu). Chaque valeur v est déterminée par l'algorithme de recherche utilisé. Nous appelons cette technique le *root learning*. Cette approche perd ainsi une partie des informations contenues dans l'arbre de jeu générées lors de la recherche de l'action à jouer. Plus précisément, les valeurs des états de l'arbre de jeu sont perdues puisqu'elles ne sont pas utilisées lors de l'apprentissage de la fonction d'évaluation. Nous proposons donc d'apprendre l'intégralité de l'arbre partiel de jeu (à l'exception des feuilles non terminales puisque l'on a déjà $f(e) = v$) au lieu de se restreindre à apprendre la valeur de la racine. Nous appelons cette technique le *tree learning*.

Le tree learning peut sembler moins intéressant que le root learning puisqu'il ajoute à l'apprentissage des valeurs de précision inférieure à celle de la racine (puisque moins d'états sont générés pour les déterminer). Il est de plus possible qu'un certain nombre d'états de l'arbre de jeu n'aient jamais été explorés depuis le début de l'apprentissage par renforcement. Des valeurs abberantes seraient alors apprises continuellement. Ces valeurs supplémentaires ont un gros risque de biaiser significativement l'apprentissage par renforcement. Il y a donc un dilemme entre le gain en information qui peut être apporté par l'apprentissage de ces valeurs supplémentaires et le risque de biaiser l'apprentissage dû à l'imprécision de ces valeurs.

Afin de réduire le biais dû à l'apprentissage de ces valeurs, nous proposons une technique supplémentaire que nous appelons *coefficiented tree learning*. Cette technique affecte à chaque état un coefficient c indiquant à quel point sa valeur est précise. Les valeurs avec un coefficient plus grand doivent alors être préférentiellement mieux apprises que celles avec un faible coefficient. Pour implémenter cette idée, lors des expériences de cet article, le coefficient est calculé par la formule suivante : $c = 1 + \lfloor \log_2(n+1) \rfloor$ avec n le nombre de fois que la valeur de cet état a été mise à jour au cours de la recherche (n est donc le nombre de nœuds descendants du nœud de cet état). Pour tenir compte des coefficients lors de l'apprentissage, nous dupliquons c fois chaque couple (e, v) où c est le coefficient de ce

Programmes	Tailles	Recherche	Apprentissage	Réseau	Utilisation
Mohex-CNN	13	MCTS	supervisé	convolutionnel	politique
Mohex-3HNN	13	MCTS	supervisé	résiduel	politique, état, action
NeuroHex	13	aucun	supervisé et renforcement	convolutionnel	état
EZO-CNN	7, 9, 11	Minimax	renforcement	convolutionnel	état
ExIt	9	MCTS	renforcement	convolutionnel	politique et état

TABLE 1 – Comparatif des caractéristiques principales des derniers programmes au Hex. Ces caractéristiques sont respectivement les tailles de plateau sur lesquelles ils peuvent jouer, l'algorithme de recherche dans les arbres utilisé, le type d'apprentissage effectué, le type de réseau de neurones utilisé et l'utilisation du réseau de neurones qui en est fait, c'est-à-dire s'il est utilisé pour approximer les valeurs des états, des actions et/ou de la politique.

couple.

Exemple 2. Suite à une partie, nous obtenons les couples suivant $(e_1, 1)$, $(e_2, -2)$ et $(e_3, 5)$. Les états e_1, e_2 et e_3 ont respectivement 0, 1 et 3 états descendants dans l'arbre de jeu. Le coefficient de e_1 (resp. e_2 ; resp. e_3) est donc 1 (resp. 2 ; resp. 3). Les données utilisées lors de l'apprentissage de la fonction d'évaluation adaptative f sont les couples suivants $(e_1, 1)$, $(e_2, -2)$, $(e_2, -2)$, $(e_3, 5)$, $(e_3, 5)$, $(e_3, 5)$.

Notons qu'il y a une autre approche dans la littérature, utilisée par alphago zéro [27]. Avec celle-ci, les états des couples (e, v) sont toujours les états obtenus au cours de la partie, mais leur valeur est le résultat de la partie (la valeur de l'état terminal de la partie). Nous appelons cette technique le *terminal learning*. Elle a l'avantage de ne pas apprendre des valeurs approximées et donc de ne pas approximer des approximations. Cependant, il y a toujours un biais, puisque la valeur terminale n'est en général pas la vrai valeur de chaque état de la partie.

3.2 Heuristique de la profondeur

Dans cette section, nous proposons une fonction d'évaluation des états terminaux alternative à la fonction d'évaluation classique des états terminaux, qui retourne 1 si le joueur gagne et -1 si son adversaire gagne [32, 27, 9]. Nous l'appelons *heuristique de la profondeur* et nous la notons $\mathrm{p_t}$. Elle donne une meilleure valeur aux états gagnants proches du début de partie qu'aux états gagnants loin du début de partie. Elle donne également une meilleur valeur aux états perdants loin du début de partie qu'aux états perdants proches du début de partie. Avec cette fonction d'évaluation, on cherche à gagner le plus tôt possible et perdre le plus tard possible. En apprenant par renforcement avec cette fonction d'évaluation terminale, on préférera ainsi un état que l'on pense gagnant et proche de la fin de partie à un autre état que l'on pense gagnant et loin de la fin de partie. On peut espérer que cela améliore le niveau de jeu. Un état que l'on pense gagnant et proche de la fin de partie a moins de chances d'être un état perdant qu'un état que l'on pense gagnant et loin de la fin de partie. En effet, plus un état est proche de la fin de partie, plus il y a de chances que sa valeur soit précise. De plus, avec une partie longue, un joueur en difficulté aura plus d'occasions de reprendre le dessus. Inversement, un état que l'on pense perdant et loin de la fin de partie sera préféré à un état que l'on pense perdant et proche de la fin de la partie, puisque le premier état a hypothétiquement une valeur moins précise et a donc *a priori* plus de chances d'être en réalité un état gagnant.
Nous proposons comme réalisation de l'heuristique de la profondeur la fonction $\mathrm{p_t}$ suivante. La fonction d'évaluation $\mathrm{p_t}$ retourne la valeur l si le joueur gagne et la valeur $-l$ si son adversaire gagne, avec l le nombre maximum d'actions réalisables au cours d'une partie plus 1 moins le nombre d'actions jouées depuis le début de la partie. Pour le Hex, l est le nombre de cases vides du plateau plus 1.

3.3 Complétion de fonctions d'évaluation

Nous introduisons dans cette section la complétion C d'une fonction d'évaluation d'états de jeu f. La complétion a pour objectif d'améliorer la fonction d'évaluation en tenant compte de la résolution des états. Un état est dit *résolu* si l'algorithme de recherche a permis de déterminer la vraie valeur de cet état. Sans la complétion, on peut avoir un état résolu et gagnant de valeur plus faible qu'un état non résolu que l'on pense gagnant mais qui est peut-être perdant. L'objectif de la complétion est donc de toujours choisir une action menant à un état résolu gagnant et de ne jamais choisir une action menant à un état résolu perdant lorsque l'on a le choix (lors de l'exploration de l'arbre ou de la décision de l'action à jouer). La fonction de complétion peut se définir par $C(f(e)) = (r(e), f(e))$ avec e un état et r la fonction suivante qui retourne 1 si l'état est résolu et si le joueur gagne, -1 si l'état est résolu et si son adversaire gagne et 0 si l'état n'est pas résolu. On utilisera alors l'ordre lexicographique pour comparer les états.

Remarque 3. Bien que la complétion remplace la fonction d'évaluation lors de l'exploration et lors du choix de l'action à jouer, les couples utilisés lors de l'apprentissage restent les couples de la forme $(e, f(e'))$.

Il n'y a pas besoin de compléter une fonction d'évaluation d'états non terminaux si elle est à valeurs dans $]a, b[$ (avec $a < b$) et si les états terminaux sont évalués par a s'ils sont perdants et par b s'ils sont gagnants.

3.4 Descente : générer de meilleures données

Nous introduisons dans cette section une modification d'UBFM, que nous nommons *descente*, visant à être uti-

lisée lors de l'apprentissage par renforcement afin de générer des couples (e, v) différents mais supposément de meilleure qualité. L'idée de *descente* est de combiner UBFM avec des simulations déterministes de fin de partie apportant des valeurs intéressantes du point de vue de l'apprentissage. L'algorithme *descente* (algorithme 1) sélectionne récursivement le meilleur fils du nœud actuel, qui devient le nouveau nœud actuel. Il ajoute les fils du nœud actuel s'ils ne sont pas dans l'arbre. Il effectue cette récursion partant de la racine (l'état actuel du jeu) jusqu'à atteindre un nœud terminal (une fin de partie). Il met alors à jour la valeur des nœuds sélectionnés (valeur minimax). L'algorithme *descente* recommence cette opération récursive repartant de la racine tant qu'il reste du temps de recherche. *Descente* est quasiment identique à UBFM. La seule différence est que *descente* effectue une iteration jusqu'à atteindre un état terminal alors qu'UBFM effectue cette itération jusqu'à atteindre une feuille de l'arbre (UBFM arrête donc l'itération beaucoup plus tôt). Autrement dit, lors d'une itération, UBFM étend juste une des feuilles de l'arbre de jeu alors que *descente* étend récursivement le meilleur des fils en partant de cette feuille jusqu'à atteindre une fin de partie. L'algorithme *descente* a l'avantage d'UBFM, c'est-à-dire d'effectuer une recherche plus longue permettant de déterminer une meilleure action à jouer. Grâce au tree learning, il a également l'avantage d'une recherche minimax à profondeur 1, c'est-à-dire de faire remonter les valeurs des nœuds terminaux vers les autres nœuds plus rapidement. En outre, les états ainsi générés sont plus proches des états terminaux. Leurs valeurs sont donc de meilleures approximations.

3.5 Distribution ordinale d'actions

Dans cette section, nous proposons une distribution de probabilité alternative à la fonction softmax et à ϵ-greedy (voir la section 2.3). Cette distribution ne dépend pas de la valeur des états. Elle dépend par contre de l'ordre de leurs valeurs. Sa formule est la suivante :

$$P(f_i) = \left(\epsilon' + \frac{1-\epsilon'}{n-i}\right) \cdot \left(1 - \sum_{j=0}^{j<i} P(f_j)\right)$$

avec n le nombre de fils de la racine, $i \in \{0, \ldots, n-1\}$, f_i le $i^{ème}$ meilleur fils de la racine, $P(f_i)$ la probabilité de jouer l'action menant au fils f_i et ϵ' le paramètre d'exploitation ($\epsilon' = 1 - \epsilon$).

Nous proposons en plus d'utiliser la règle suivante lors de la sélection de l'action à jouer, pour réduire la durée des parties et donc *a priori* la durée de l'apprentissage : toujours jouer une action menant à un état résolu gagnant s'il existe et ne jamais jouer une action menant à un état résolu perdant si cela est possible. Ainsi, si parmi les actions disponibles on sait qu'une des actions est gagnante, on la joue. S'il n'y en a pas, on joue aléatoirement suivant la loi choisie parmi les actions ne menant pas à un état résolu perdant (si les états ne sont pas tous perdants).

3.6 UBFM_s : jouer la sécurité

Nous proposons maintenant une légère modification d'UBFM, notée UBFM_s, qui vise à fournir un jeu plus sûr. L'action qu'UBFM choisit de jouer est celle qui mène à l'état de meilleure valeur. Dans certains cas, la meilleure action (*a priori*) peut mener à un état qui n'a pas été *suffisamment* visité (comme une feuille non terminale). Choisir cette action est donc une décision risquée. Nous proposons, pour éviter ce problème, une décision différente qui vise à jouer l'action la plus sure, à l'instar du MCTS (max child selection [5]). Si aucune action ne mène à un état résolu gagnant, l'action choisie par UBFM_s est celle

Fonction `descente(`e`)`
- **if** `terminal(`e`)` **then**
  - retourner $f(e)$
- **else**
  - **if** $e \notin E$ **then**
    - $E \leftarrow E \cup \{e\}$
    - **foreach** $a \in$ `actions(`e`)` **do**
      - $v(e, a) \leftarrow f(a(e))$
  - $a_b \leftarrow$ `meilleure_action(`e`)`
  - $v(e, a_b) \leftarrow$ `descente(`$a_b(e)$`)`
  - $a_b \leftarrow$ `meilleure_action(`e`)`
  - retourner $v(e, a_b)$

Fonction `meilleure_action(`e`)`
- **if** `premier_joueur(`e`)` **then**
  - retourner $\underset{a \in \texttt{actions}(e)}{\arg\max} \; v(e, a)$
- **else**
  - retourner $\underset{a \in \texttt{actions}(e)}{\arg\min} \; v(e, a)$

Fonction `apprentissage_descente()`
- $E \leftarrow \{\}$
- $e \leftarrow$ état initial
- **while** ¬`terminal(`e`)` **do**
  - $t =$ `time()`
  - **while** `time()` $- t < \tau$ **do** `descente(`e`)`
  - $a_b \leftarrow$ `selection_action(`e`)`
  - $e \leftarrow a_b(e)$
- $D \leftarrow \{\}$
- **foreach** $e \in E$ **do**
  - $a_b \leftarrow$ `meilleure_action(`e`)`
  - $D \leftarrow D \cup \{(e, v(e, a_b))\}$
- adapter f par rapport à D

Algorithme 1 : Apprentissage d'une partie avec *descente* et *tree learning* ($a(e)$: état obtenu après avoir joué l'action a dans l'état e ; $v(e, a)$: valeur obtenue après avoir joué a dans e ; f est du point de vue du premier joueur ; E : clefs de la table de transposition ; τ : temps de réflexion par action ; selection_action : meilleure_action() ou une autre distribution d'actions).

qui a été le plus de fois sélectionnée (depuis l'état actuel du jeu) au cours de l'exploration de l'arbre. En cas d'égalité, UBFM_s départage en choisissant celle qui mène à l'état de meilleure valeur. Cette décision est plus sure du fait que le nombre de fois qu'une action est sélectionnée correspond au nombre de fois que cette action est plus intéressante que les autres (la recherche est effectuée en meilleur d'abord).

Exemple 4. Le joueur en cours a le choix entre deux actions a_1 et a_2. L'action a_1 mène à un état de valeur 5 et a été sélectionnée 7 fois (depuis l'état courant et depuis le début de la partie). L'action a_2 mène à un état de valeur 2 et a été sélectionnée 30 fois. UBFM choisit l'action a_1 alors que UBFM_s choisit l'action a_2.

4 Expériences

Nous décrivons maintenant les différentes expériences réalisées. Elles se basent sur les techniques présentées dans la section précédente. Nous commençons par décrire les détails de l'apprentissage. Ensuite, nous évaluons le résultat d'un apprentissage en temps long utilisant nos techniques contre la version 1.0 et la version 2.0 de Mohex. Enfin, nous comparons expérimentalement différents algorithmes pour jouer, tels que UBFM_s et la complétion.

4.1 Détails techniques

Après chaque partie, afin d'améliorer la variabilité des données (exactes) utilisées par l'apprentissage, les états terminaux e des données récoltées $D = \{(e, v)\}$ sont retirés de D. À la place, pour un état sur deux e (différent) restant dans D, une simulation aléatoire de fin de partie est effectuée et l'état terminal obtenu e_t et sa valeur $\mathrm{p_t}(e_t)$ sont ajoutés dans D. Les données de D sont ensuite augmentées, le jeu présentant une symétrie par rapport au plateau (rotation de $180°$). L'ajout du symétrique de chaque état double ainsi le nombre de données : $D = \{(e, v)\} \cup \{(s_{180°}(e), v)\}$. La dernière étape avant la phase d'apprentissage est la répartition aléatoire des données de D en sous-ensembles disjoints D_i (de taille 1024). La méthode d'optimisation utilisée pour l'apprentissage est *Adam* [14]. Nous l'utilisons pour minimiser itérativement la valeur $\sum_{(e,v)\in D_i} (f(e) - v)^2$ pour chaque D_i. Adam est appliqué une fois par D_i (une mise à jour par D_i). Les données sont ensuite effacées et une nouvelle partie débute.
La fonction d'évaluation adaptative utilisée est un réseau de neurones convolutif [17] ayant trois couches de convolution suivi d'une couche cachée entièrement connectée. Pour chaque couche convolutive, le noyau est de taille 3×3 et le nombre de calque est 150. Le nombre de neurones de la couche entièrement connectée est 81. La marge de chaque couche est à zéro. Après chaque couche sauf la dernière, la fonction d'activation ReLU [10] est utilisée. La couche de sortie contient un neurone. Il n'y a pas de fonction d'activation pour la sortie. L'entrée du réseau de neurones est un plateau de jeu étendu d'une ligne en haut, en bas, à droite et à gauche (à la manière de [32, 1]). Plus précisément, chacune de ces lignes est remplie entièrement des pièces du joueur du bord où elle se trouve. Cette extension revient simplement à représenter explicitement les bords du plateau et leur appartenance.
Le taux d'exploitation de la distribution de sélection d'actions évolue linéairement au cours du temps : $\epsilon' = \frac{t}{T}$ avec t le temps écoulé depuis le début de l'apprentissage par renforcement et T la durée totale de cet apprentissage.
Lors de l'apprentissage (en jouant contre soi-même), les deux joueurs utilisent la même fonction d'évaluation et la même table de transposition, qui est vidée après chaque partie (les états résolus sont cependant conservés).

4.2 Résultats contre Mohex 1.0 et Mohex 2.0

Un long apprentissage contre soi-même pour les plateaux de taille 11 a été effectué avec l'algorithme descente utilisant l'heuristique de la profondeur, la complétion, le coefficiented tree learning ainsi que la distribution et la règle associée de la section 3.5. La fonction d'évaluation a été pré-initialisée en apprenant les valeurs d'états terminaux aléatoires (au nombre de $8.836.000$). L'apprentissage par renforcement de la fonction d'évaluation a duré 58.674 parties, à raison d'une seconde de réflexion par action.
UBFM_s doté de la fonction d'évaluation générée par cet apprentissage atteint le score de 60% de victoire (84% en premier joueur ; 37% en second joueur) contre Mohex 1.0 à la suite de 300 parties en premier joueur et 300 autres parties en second joueur, avec une seconde de réflexion par action. Son score contre Mohex 2.0 est de 53% victoire (81% en premier joueur ; 25% en second joueur) à la suite de 1000 parties en premier joueur et 1000 autres parties en second joueur, avec une seconde et demi par action.

4.3 Comparaison d'algorithmes de jeu

Nous comparons maintenant les taux de victoire de différents algorithmes pour jouer en les évaluant contre Mohex 2.0. Chacun d'eux se base sur la fonction d'évaluation des plateaux de taille 11 générée au cours de l'expérience précédente. Nous comparons UBFM_s sans complétion et les algorithmes suivants avec complétion : UBFM, UBFM_s, alpha-bêta à profondeur 1, alpha-bêta à profondeur 2 ainsi qu'une modification d'UBFM_s, notée UBFM_s + UCT. Cette variante effectue une exploration en meilleur d'abord tenant compte du nombre de visites des nœuds à la manière de la version la plus populaire du MCTS, UCT [5]. Pour cela, le fils sélectionné est le meilleur selon la somme de la valeur d'UBFM_s (normalisée pour être dans $[0, 1]$) et du terme d'exploration suivant : $c \cdot \sqrt{\frac{\log n}{N}}$ avec n le nombre de visites du fils considéré, N le nombre de visites du nœud père et c une constante d'exploration (en théorie $\sqrt{2}$).
Les taux de victoire contre Mohex 2.0 sont décrits dans la table 2. C'est UBFM_s qui obtient le meilleur taux de victoire. L'utilisation de la complétion apporte un très léger gain de l'ordre de 2%. Jouer l'action la plus sure a apporté un gain de 6%. UBFM_s est supérieur de 24% à l'alpha-bêta à profondeur 1. Il est supérieur de 18% à l'alpha-bêta à profondeur 2, bien que ce dernier ait un temps de réflexion

Algorithmes	1er	2ème	moyenne
UBFM_s	81%	25%	53%
UBFM_s sans complétion	77%	25%	51%
UBFM	61%	34%	47%
Alpha-bêta à profondeur 2	56%	13%	35%
Alpha-bêta à profondeur 1	37%	20%	29%
UBFM_s + UCT ($c = 2 \cdot \sqrt{2}$)	55%	12%	33%
UBFM_s + UCT ($c = \sqrt{2}$)	55%	15%	35%
UBFM_s + UCT ($c = \frac{\sqrt{2}}{2}$)	50%	12%	31%
UBFM_s + UCT ($c = \frac{\sqrt{2}}{10}$)	56%	15%	36%

TABLE 2 – Statistiques de victoire contre Mohex 2.0 de différents algorithmes de recherche pour 600 parties en premier joueur et 600 autres parties en second joueurs. UBFM, UBFM_s, chaque UBFM_s + UCT et Mohex 2.0 ont une seconde et demi par action. Alpha-bêta à profondeur 2 prend en moyenne environ deux secondes par action.

par action $1/4$ de temps supérieur en moyenne. Enfin, ajouter de l'exploration à UBFM_s a diminué le taux de victoire d'au moins 17% (pour chaque valeur c choisie).
Cette expérience suggère qu'une fois que l'apprentissage par renforcement est terminé, il est plus avantageux d'utiliser tout son temps à l'exploitation, en réfléchissant en meilleur d'abord. Cette stratégie amène à se focaliser sur des zones spécifiques et profondes de l'arbre de jeux (pour les parties de l'espace d'états suffisamment explorés lors de l'apprentissage). Au moment de l'évaluation, en général, il ne faut plus explorer. Il faut encore moins faire une recherche exhaustive en largeur. C'est à l'inverse plutôt le moment de valider que la meilleure action *a priori* est bien la meilleure action. Notons qu'UBFM_s (ou UBFM) est malgré tout capable de faire une recherche plus ou moins en largeur lorsqu'il rencontre des états ayant une grande variabilité de valeurs d'une action à l'autre. Cela arrive dans les parties de l'espace d'états qu'il n'a pas suffisamment explorés au cours de l'apprentissage. En résumé, jouer en meilleur d'abord permet d'effectuer un élagage très important, éventuellement temporaire, permettant ainsi d'obtenir un coup d'avance sur son adversaire. Il est toutefois possible que pour des temps de réflexion beaucoup plus long, ajouter de l'exploration donne un avantage.

5 Conclusion

Nous avons proposé plusieurs nouvelles techniques pour l'apprentissage par renforcement de fonctions d'évaluation. La première technique consiste à apprendre les valeurs de l'arbre de jeu au lieu de se restreindre à la valeur de la racine. Pour limiter le biais induit sur l'apprentissage par l'ajout de ces valeurs, nous avons proposé d'associer un coefficient à chaque état, dépendant du nombre de visites. Nous avons également proposé de remplacer le gain classique d'un jeu $(+1/-1)$ par l'heuristique de la profondeur. Cela permet de tenir compte de la durée de la partie afin de favoriser les victoires rapides et les défaites lentes. Nous suggérons l'utilisation de la complétion pour améliorer le comportement de certaines fonctions d'évaluation, comme celles basées sur l'heuristique de la profondeur. Enfin, nous avons proposé l'algorithme descente qui explore à la manière du minimax à profondeur non bornée. Contrairement à ce dernier, descente explore itérativement les séquences de meilleures actions *jusqu'aux états terminaux*. Son objectif est d'améliorer la qualité des données utilisées lors de l'apprentissage, tout en gardant les avantages du minimax à profondeur non bornée.
En outre, nous avons proposé une autre variante du minimax à profondeur non bornée, qui joue l'action la plus sure au lieu de jouer la meilleure action. Nos expériences suggèrent que cela améliore le niveau de jeux lors des confrontations. Nos expériences suggèrent également qu'une fonction d'évaluation apprise par renforcement donne un meilleur niveau de jeux lorsqu'elle est utilisée par un algorithme de recherche en meilleur d'abord.
Enfin, l'utilisation des techniques proposées a permis de concevoir, pour la première fois, un programme-joueur au jeu de Hex (taille 11) atteignant le niveau de Mohex 2.0 par un apprentissage par renforcement contre soi-même sans utilisation de connaissance (ni de techniques Monte Carlo).
Une étude comparant la vitesse d'apprentissage en fonction des techniques utilisées est en cours. Les résultats préliminaires suggèrent d'une part que l'utilisation de l'heuristique de la profondeur améliore l'apprentissage. Ils suggèrent d'autre part que l'apprentissage avec descente est meilleur qu'avec UBFM.
Les perspectives de recherche incluent l'application de nos contributions au jeu de Go et au General Game Playing avec information parfaite dans un premier temps. Elles incluent également, par conséquent, l'adaptation de nos contributions aux contextes des informations cachées, des jeux stochastiques et des jeux multi-joueurs.

Remerciements

Je remercie le GREYC de m'avoir donné accès à son serveur de calcul, ce qui m'a permis d'effectuer l'expérience de la section 4.2.

Références

[1] Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search. In *Advances in Neural Information Processing Systems*, pages 5360–5370, 2017.

[2] Hendrik Baier and Mark HM Winands. Mcts-minimax hybrids with state evaluations. *Journal of Artificial Intelligence Research*, 62 :193–231, 2018.

[3] Elwyn R Berlekamp, John H Conway, and Richard K Guy. *Winning Ways for Your Mathematical Plays, Volume 3*. AK Peters/CRC Press, 2018.

[4] Édouard Bonnet, Florian Jamain, and Abdallah Saffidine. On the complexity of connection games. *Theoretical Computer Science*, 644 :2–28, 2016.

[5] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *Transactions on Computational Intelligence and AI in games*, 4(1) :1–43, 2012.

[6] Tristan Cazenave. Residual networks for computer go. *Transactions on Games*, 10(1) :107–110, 2018.

[7] Christopher Clark and Amos Storkey. Training deep convolutional neural networks to play go. In *International Conference on Machine Learning*, pages 1766–1774, 2015.

[8] Chao Gao, Ryan B Hayward, and Martin Müller. Move prediction using deep convolutional neural networks in hex. *Transactions on Games*, 2017.

[9] Chao Gao, Martin Müller, and Ryan Hayward. Three-head neural network architecture for monte carlo tree search. In *International Joint Conference on Artificial Intelligence*, pages 3762–3768, 2018.

[10] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *The fourteenth international conference on artificial intelligence and statistics*, pages 315–323, 2011.

[11] Ryan Hayward and Noah Weninger. Hex 2017 : Mohex wins the 11× 11 and 13× 13 tournaments. *ICGA Journal*, 39(3-4) :222–227, 2017.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Shih-Chieh Huang, Broderick Arneson, Ryan B Hayward, Martin Müller, and Jakub Pawlewicz. Mohex 2.0 : a pattern-based mcts hex player. In *International Conference on Computers and Games*, pages 60–71. Springer, 2013.

[14] Diederik P Kingma and Jimmy Ba. Adam : A method for stochastic optimization. *arXiv preprint arXiv :1412.6980*, 2014.

[15] Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. Optimization by simulated annealing. *Science*, 220(4598) :671–680, 1983.

[16] Richard E Korf and David Maxwell Chickering. Best-first minimax search. *Artificial intelligence*, 84(1-2) :299–337, 1996.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[18] Michael Lederman Littman. *Algorithms for sequential decision making*. PhD thesis, 1996.

[19] Jacek Mandziuk. *Knowledge-free and learning-based methods in intelligent game playing*, volume 276. Springer, 2010.

[20] Joseph Mellor. *Decision Making Using Thompson Sampling*. PhD thesis, 2014.

[21] Ian Millington and John Funge. *Artificial intelligence for games*. CRC Press, 2009.

[22] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540) :529, 2015.

[23] Jakub Pawlewicz and Ryan B Hayward. Scalable parallel dfpn search. In *International Conference on Computers and Games*, pages 138–150. Springer, 2013.

[24] Stefan Reisch. Hex ist pspace-vollständig. *Acta Informatica*, 15(2) :167–191, 1981.

[25] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587) :484, 2016.

[26] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv :1712.01815*, 2017.

[27] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676) :354, 2017.

[28] Kei Takada, Hiroyuki Iizuka, and Masahito Yamamoto. Reinforcement learning for creating evaluation function using convolutional neural network in hex. In *2017 Conference on Technologies and Applications of Artificial Intelligence*, pages 196–201. IEEE, 2017.

[29] H Jaap Van Den Herik, Jos WHM Uiterwijk, and Jack Van Rijswijck. Games solved : Now and in the future. *Artificial Intelligence*, 134(1-2) :277–311, 2002.

[30] H Jaap Van Den Herik and Mark HM Winands. Proof-number search and its variants. In *Oppositional Concepts in Computational Intelligence*, pages 91–118. Springer, 2008.

[31] Georgios N Yannakakis and Julian Togelius. *Artificial Intelligence and Games*. Springer, 2018.

[32] Kenny Young, Gautham Vasan, and Ryan Hayward. Neurohex : A deep q-learning hex agent. In *Computer Games*, pages 3–18. Springer, 2016.

[blue sky] A la recherche d'une planification plus humaine

Kevin Colombier[12] Axel Buendia[12]

[1] CEDRIC-CNAM
[2] SpirOps

prenom.nom@spirops.com

Résumé

La planification est la recherche d'un plan d'actions afin d'atteindre un objectif. La planification classique est basée sur un ensemble d'hypothèses qui permet une résolution optimale de problèmes complexes. Ils sont notamment composés d'un grand nombre d'instances, ce qui implique un élargissement du graphe de recherche. Cependant, elle est moins efficace dans des environnements plus proches du monde réel. Nous pensons qu'il est préférable d'avoir une planification plus humaine, avec une résolution satisfaisante de problèmes dans des environnements dynamiques. Il en résulte un besoin d'utiliser des connaissances générales de sens commun et donc d'avoir une mémoire plus expressive. Ces travaux permettraient d'améliorer la planification dans des contextes du monde réel comme dans la robotique ou la simulation.

Mots Clef

Planification humaine, planification dynamique, monde réel, sens commun.

Abstract

Planning is the task of searching an action plan to achieve a goal. Classical planning is based on a set of assumptions which makes it possible to solve optimally some complex problems. They are composed of a huge number of instances, and that implies a larger search graph. However, in real-world environment this approach is less efficient. We think a human planning would be preferable for solving problems in dynamic environments in a satisfying way. It results in a need of a common-sense knowledge and thus a more expressive memory. This work aims at improving planning in real-world contexts like robotics or simulations.

Keywords

Human planning, dynamic planning, real world, common sense.

1 Introduction

Un agent cognitif est un agent intelligent, c'est-à-dire qu'il est capable de percevoir son environnement et d'agir pour le modifier. A l'opposé d'un agent réactif, qui ne se base que sur ses perceptions pour prendre des décisions, l'agent cognitif est capable de stocker de nouvelles connaissances dans une mémoire et de faire des inférences à partir de celle-ci afin de raisonner. L'agent est communément composé de deux systèmes de décisions : le *système 1* qui reflète l'intuition et le *système 2* qui exprime la réflexion ([10]). Pour interagir avec son environnement et en fonction de la rapidité à laquelle il doit réagir, il peut prendre des décisions de manière réflexe avec le système 1 ou bien intentionnelle avec le système 2. La première est une réponse à un stimulus sans objectif précis. La deuxième demande une réflexion plus longue et vise à l'accomplissement d'objectifs, ce qui implique la capacité de pouvoir se représenter des états futurs possibles et de les juger. En effet, pour accomplir des *objectifs*, des états hypothétiques que l'agent souhaite atteindre, il doit planifier, c'est-à-dire trouver un *plan*, une suite d'actions, qui lui permettra de passer de l'état actuel à l'état souhaité de l'environnement.

La planification est un des domaines fondateurs de l'intelligence artificielle (IA). Initialement, le but de l'IA est de fabriquer une machine capable de penser comme un humain. Or, la planification est un des mécanismes de base du raisonnement humain. De nombreux travaux de recherche ont été réalisés et ont défini le cadre classique de la planification. Par la suite, un langage commun a été créé, le *PDDL* [4], permettant la création d'un ensemble de problèmes classiques utilisés pour se comparer. Dans ce cadre, des algorithmes efficaces ont été développés et sont devenus la norme de l'état de l'art [6]. Ils ont comme objectif de résoudre des problèmes de manière *optimale* avec des objectifs fixés à l'avance. Ces planificateurs permettent de résoudre des problèmes insolubles pour un humain en temps raisonnable. Ils sont très performants pour un nombre restreint et bien défini de problèmes. Le cadre de la planification classique est très circonscrit, il définit un ensemble d'hypothèses simplificatrices qui limitent principalement l'environnement du problème [6]. L'humain, en comparaison, planifie des tâches plus simples, mais dans un monde plus complexe.

Nous souhaitons mettre en avant le besoin d'un nouveau type de planification, non pas *optimale* mais *humaine*. Pour

nous, la planification humaine est une planification *satisfaisante* dans un environnement *dynamique*, *inconnu* et *ouvert*. Autrement dit, nous voulons nous placer dans un environnement en constante modification, par sa physique ou par les actions d'autres acteurs. De plus nous souhaitons que l'agent n'ait qu'une connaissance limitée de son environnement -à ce qu'il peut percevoir- et qu'il ait conscience qu'il ne pourra jamais avoir toutes les connaissances sur celui-ci. Dans cet article, nous présenterons d'abord les différences majeures que nous voyons entre la planification classique et une planification plus humaine puis nous montrerons les apports et les intérêts qu'une telle planification peut avoir avant de conclure. Dans la suite de l'article excepté la conclusion, le "nous" designe les humains en général et non pas les auteurs, qui sont désignés comme tels.

2 Différence entre la planification humaine et la planification classique

2.1 Hypothèses de base de la planification classique

La planification classique se base sur un ensemble d'hypothèses : elle doit être faite dans un environnement parfaitement observable, déterministe, fini, statique, discret par rapport au temps, aux actions et aux objectifs et où il n'y a pas d'autres agents qui agissent [6]. De plus, elle est effectuée en hors ligne, c'est-à-dire que l'intégralité de la planification est réalisée avant l'exécution. Ces hypothèses permettent de simplifier la recherche et, par conséquent, de travailler sur des problèmes larges et complexes en cherchant les solutions optimales. Cependant, ces hypothèses sont très fortes, en particulier lorsqu'il s'agit de travailler dans un environnement proche du monde réel. Notre environnement de tous les jours dépasse fortement toutes ces limitations : il est dynamique, inconnu, ouvert [1, 21].

Il est vrai que des travaux ont cherché à dépasser ces limites, en supprimant quelques hypothèses dans leur cadre d'étude. Cela a permis de créer différents frameworks de planification (Fig.1) comme la planification temporelle, qui ajoute une durée aux actions, ou la planification incertaine, où l'environnement est partiellement observable et les conséquences des actions non déterministes. Une solution commune pour arriver à résoudre ces problématiques est de trouver un moyen de traduire le problème et de revenir dans une planification classique. Il suffit alors d'utiliser le résultat issu de la planification classique, qui s'obtient assez rapidement, dans la nouvelle problématique. Il n'y a, à la connaissance des auteurs, aucun travail qui a cherché à contredire l'ensemble de ces hypothèses. Il est vrai que celles-ci ont été posées car la recherche du plan solution est très complexe, et donc fixer certaines contraintes permet aux planificateurs de le trouver dans un temps raisonnable. Cependant ces contraintes portent sur l'environnement qui entoure l'agent, limitant de ce fait le cadre d'usage de la planification. Les auteurs pensent qu'il peut être intéressant de s'affranchir de l'ensemble de ces limites sur l'environnement en particulier pour des problématiques du monde réel. En contrepartie, nous ne visons pas une planification optimale en hors ligne mais au contraire une planification satisfaisante en ligne dans un environnement complexe.

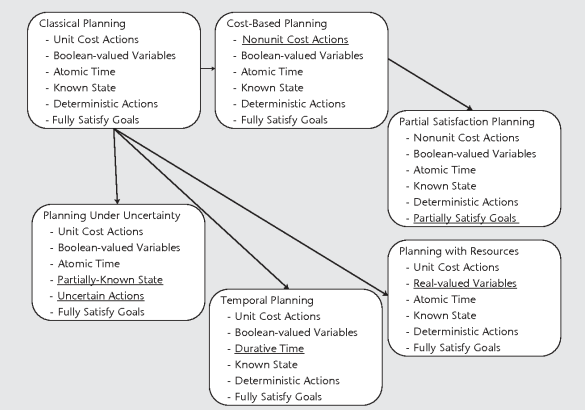


FIGURE 1 – Différent framework de planification [2]

2.2 Une multitude d'objectifs

Classiquement, la planification est souvent synonyme de résolution de problèmes. Elle est définie comme la recherche d'une suite d'actions permettant d'atteindre un objectif donné [6]. Cette recherche se fait dans un graphe de recherche où les noeuds sont les états de l'environnement et les liens sont des actions. Cette définition souligne que la planification fonctionne pour un problème précis sur un objectif précis. Cela signifie que sortie du cadre défini dans le problème, la planification ne pourra pas être faite. L'objectif est donné au préalable de la planification, par un utilisateur ou un système externe. Lorsque le plan permettant d'atteindre l'objectif est trouvé, la planification s'arrête. Le problème est résolu. On passe à un nouvel objectif et un nouveau problème.

Au contraire les humains ont toujours un grand nombre d'objectifs en tête. Certains sont des objectifs à très long terme (être riche, être sportif, maigrir), d'autres plus proches (manger, lire, aller au cinéma). Certains objectifs arrivent à la suite d'une sensation (la vue d'un objet convoité, le parfum d'un gâteau au four, le son de l'eau qui coule), d'autres à la suite d'un besoin de base pour la survie ou d'envie du corps (faim, soif, fatigue) [14]. Le choix du plan suivi dépend de l'ensemble de ces objectifs.

Un objectif comprend un ensemble de sous-objectifs. Il représente l'état final que l'agent souhaite atteindre. Quand on parle d'état, on parle en fait d'un état global qui incarne la situation du monde à un instant donné. Celui-ci se décompose en un ensemble d'états atomiques. Par exemple un état final pourrait être : le gâteau est chaud, il est posé sur une assiette, l'assiette est dans la cuisine, le four est éteint ... Pour l'état final, chacun de ces états atomiques est

un sous-objectif. Le but de l'agent est d'atteindre cet état global précis comprenant tous les sous-objectifs. La planification classique cherche à maximiser le nombre de sous-objectifs atteints. Si un sous-objectif n'est pas atteint à la fin de celle-ci, on dit qu'elle a échoué.

En général en planification, les objectifs sont donnés en entrée de l'algorithme et ne sont pas modifiés durant tout le processus. Cela signifie que l'agent n'a pas de contrôle sur ses objectifs, il ne doit que les suivre et les achever. Pourtant les objectifs évoluent au cours du temps en fonction des observations du monde. L'étude des objectifs, appelée le raisonnement sur les objectifs [22], s'intéresse à la génération de nouveaux objectifs et à l'adaptation des objectifs existants à la vue de nouveaux besoins. Les objectifs possèdent, tout comme les actions, une hiérarchie. Lorsque l'on boit de la limonade, l'objectif peut tout aussi bien être "étancher ma soif", ou "équilibrer mon niveau de saccharose" ou encore "survivre" [16].

Dans la littérature, pour sélectionner l'objectif le plus intéressant par rapport à une multitude d'objectifs, il faut assigner à chacun un score qui dépend, entre autres, de trois paramètres : l'*envie* que l'objectif cherche à résoudre, le *coût* estimé que cela engendrera, et la *probabilité d'accomplissement* [8]. Pour les deux derniers, il faut avoir une estimation du plan à effectuer. Il faut donc une première planification, peut-être simplifiée, permettant de calculer un score pour chacun des objectifs afin de les comparer. Mais ceci implique souvent de repasser plusieurs fois dans des parties du graphe de recherche déjà explorées par de précédents objectifs. Les auteurs pensent qu'il est préférable d'effectuer une recherche en largeur dès le départ et de déterminer au fur et à mesure les scores de chaque objectif, plutôt que de planifier pour chaque objectif.

2.3 Planification dynamique

Un environnement dynamique est sans cesse modifié par les agents qui agissent dedans et par les règles physiques de l'environnement lui-même. Un agent peut connaître les règles de l'environnement, par une connaissance externe ou bien par apprentissage, mais il est difficile de prévoir précisément le comportement d'autres acteurs autonomes présents dans l'environnement. Cela apporte dans notre plan un grand nombre d'inconnues. Plus on planifie dans le futur, plus il y a de chance que cela ne se passe pas comme prévu. Cela signifie que plus on se place dans le futur, moins il est important de prévoir tout dans le moindre détail, puisqu'il est plus probable qu'il y ait des différences entre ce qu'on imagine du monde et ce qu'il sera réellement. Dans une planification dynamique et en ligne, il ne sert à rien de chercher à avoir une solution optimale pour chaque planification. Il vaut mieux trouver un bon compromis entre le temps passé à planifier et la qualité de la solution trouvée. Un agent rationnel devrait essayer d'avoir une décision "suffisfaisante", c'est-à-dire pas encore satisfaisante mais suffisante [17]. Bien sûr, en continuant à chercher, il pourrait trouver un meilleur plan. Cependant ce serait au prix d'un coût mental qui n'est pas si intéressant au vue des modifications continuelles du monde.

La capacité mentale humaine n'est pas infinie. Nous ne pouvons pas imaginer toutes les situations possibles indéfiniment, et prévoir tous les champs possibles. Un exemple classique est la tour d'Hanoi. Le but est de déplacer des disques de différentes tailles d'un point à un autre en se servant d'un point intermédiaire. L'unique contrainte est qu'un disque de plus grande taille ne peut pas être au dessus d'un disque de plus petite taille. Pour des niveaux simples, avec trois disques par exemple, nous pouvons arriver à imaginer mentalement la solution complète. A partir d'un certain niveau, si l'on ne connait pas la solution mathématique et logique de ce problème, cela devient presque impossible. Cependant, lorsque nous sommes confrontés à ce genre de problématique, nous planifions et agissons tout de même. Nous n'avons pas forcément le plan complet en tête mais nous en avons une idée plus ou moins floue.

Une bonne représentation du monde est essentielle pour la planification car plus elle sera précise et juste, plus la planification pourra trouver un bon plan rapidement. Mais dans des environnements complexes, notre connaissance du monde ne représentera jamais parfaitement tous les détails et les lois du monde. Une représentation est très souvent incomplète ou même incorrecte. Il peut lui manquer des mécanismes physiques du monde ou bien y avoir des erreurs dans les conséquences d'une action. Lorsque l'agent planifie, il faut qu'il soit conscient qu'il y aura probablement des écarts dus à sa représentation et il doit pouvoir s'adapter. Par ailleurs, nous n'arrivons pas toujours à achever nos actions comme nous les avons prévues. Certaines actions délicates et difficiles à réaliser peuvent demander une certaine habilité ou certaines compétences. Cette incertitude est en général représentée par des états et des conséquences non déterministes [3].

Il faut prendre en compte cette incertitude dans notre planification et adapter notre comportement à la dynamicité du monde. Lorsque nous exécutons notre plan, nous supervisons l'état du monde par rapport à nos attentes. Cette étape s'appelle en général le *contrôle*. Elle est classiquement décorrélée de la planification en elle-même [9, 23]. Les auteurs pensent que, dans une planification humaine, l'exécution est un autre aspect important de la planification et qu'il faut le prendre en compte dans un tout. D'autres auteurs [5, 18] estiment également qu'il est important de ne pas négliger le point de vue global de l'acteur. La planification prévoit un état du monde après l'exécution de chaque action, et c'est au contrôle de vérifier que les actions se sont bien passées comme prévu. En cas d'échec, il faut replanifier c'est-à-dire retrouver un nouveau plan pour achever notre objectif, en prenant en compte le nouvel état du monde.

2.4 Apprentissage pour la représentation

Nous sommes capables aussi bien d'adapter nos plans face aux imprévus, que d'apprendre de nos erreurs. Quel que

soit le résultat de l'exécution d'un plan, nous pouvons apprendre de lui et améliorer les connaissances que nous avons sur l'environnement. Sa réussite signifie que notre représentation est correcte vis-à-vis du monde. Son échec signifie qu'il manque des informations ou que des connaissances sont fausses dans notre représentation. Dans le premier cas, nous pouvons alors stocker le plan correct afin qu'il puisse être réutilisé dans des situations similaires. Dans le deuxième cas, une analyse de la situation peut déterminer les causes possibles de ces erreurs, ce qui va permettre d'améliorer notre représentation.

La réutilisation et l'adaptation de plans ont déjà été étudiées en particulier dans la planification par cas [7] et dans la planification hiérarchique [15]. Dans le premier framework, l'idée est, à partir d'un ensemble de *cas*, c'est-à-dire de couples problème/plan, de pouvoir trouver un plan solution à un nouveau problème. Pour ce faire il faut d'abord trouver le problème le plus similaire à celui cherché puis adapter le plan au nouveau problème. Enfin le nouveau couple problème/plan est stocké pour pouvoir être réutilisé à son tour. Dans le deuxième framework, l'objectif est de décomposer un ensemble de *tâches buts* en des *tâches primitives*. Pour ce faire, un ensemble de *méthodes*, qui correspondent à des plans, décrit la décomposition d'une tâche en un ensemble de tâches. Dans ces deux frameworks, la réutilisation de plans n'a pas la même signification. Pour la planification par cas, les plans sont des ensembles de faits, c'est-à-dire d'actions agissant concrétement sur le monde. Pour la planification hiérarchique, les plans sont donnés à un niveau plus général et ont besoin d'être instanciés avec des objets concrets du monde.

Les auteurs pensent que la majorité des planifications que nous effectuons réutilise et adapte des plans que nous avons déjà faits ou observés. Or ces deux frameworks souffrent de certaines problématiques vis-à-vis d'une planification humaine. Au sujet de la planification par cas, il nous est parfois difficile de nous rappeler les actions et les plans que nous avons effectués la veille, il est très peu probable que nous stockons l'intégralité de nos plans pour les réutiliser. Concernant la planification hiérarchique, les méthodes doivent être données en entrée de l'algorithme, en général par un expert du domaine. Or, les données que nous recevons du monde sont en majeure partie des faits. Les auteurs estiment qu'un framework commun, prenant en compte certaines idées de chacun des frameworks, pourrait résoudre ces problématiques. Entre la planification par cas, où il y a trop de faits pour que nous puissions tous les stocker, et la planification hiérarchique, où les méthodes sont données par un expert extérieur et ne sont pas apprises, il manque une phase de *généralisation*. L'idée serait de concaténer des faits similaires et pouvoir créer des méthodes à partir d'eux.

Par exemple, si un agent veut manger une pomme précise, il trouve alors le plan : la laver, l'éplucher, puis la découper, pour enfin la manger. Une autre fois, il voit un autre agent manger une poire précise, avec un plan semblable mais concernant cette fois la poire. Au bout d'un certains nombres de fois, il pourrait généraliser l'ensemble de ces plans : pour manger un fruit, il faut le laver, l'éplucher, le découper, et enfin le manger. La prochaine fois qu'il voudra manger un fruit, il pourra directement utiliser ce plan général. Cette généralisation nécessite, d'une part, un calcul de similarité pour trouver les plans qui sont similaires les uns aux autres et, d'autre part, un opérateur de généralisation qui, à partir d'un ensemble de plans, trouve un plan plus général, qui regroupe l'ensemble des informations contenus dans les autres plans. Cela implique un besoin d'une représentation qui inclue des connaissances générales qui dépassent la planification. Dans l'exemple ci-dessus, il est nécessaire de savoir qu'une pomme et qu'une poire sont tous les deux des fruits.

2.5 Représentation des connaissances générales de sens commun

Dans la planification classique, l'acteur a accès à toutes les instances disponibles du monde. C'est-à-dire qu'il ne connait que celles-là et il suppose qu'il n'y en a pas d'autres. Il a toutes les connaissances à leurs propos. Il doit trouver un plan précis avec ces instances : chaque variable de chaque action du plan doit être reliée à une instance. Une des conséquences est qu'il ne peut pas y avoir de disparitions, d'apparitions ou de modifications d'instances [12]. Une représentation intelligente en utilisant des prédicats malins peut suffire à contourner ces problématiques. Cependant il peut arriver des cas où ces techniques ne fonctionnent pas en particulier lorsque l'on se trouve face à une action qui peut créer de nouvelles instances indéfiniment. Si l'on imagine une action qui fait une copie d'un objet sans autre condition que l'existence de l'objet, le PDDL classique ne permet pas facilement de représenter chacune de ces nouvelles instances.

Lorsque nous planifions, il est commun de ne pas préciser quelles instances nous utilisons. En général, nous n'avons pas toutes les instances en tête ou nous laissons ce choix au contrôle. Lorsque l'on prévoit de boire de l'eau, on ne choisit pas quel verre on veut prendre, on sait qu'il y en a dans la cuisine. La décision n'est pas importante pour le plan. Elle peut être faite à n'importe quel moment. Elle peut cependant modifier mon plan : si aucun verre n'est propre, il faut en nettoyer un avant de boire. Cela nécessite de se servir de ses connaissances générales et de ses croyances pour planifier. Par ailleurs, on peut aussi se servir d'autres objets similaires au verre. Si aucun verre n'est propre et que l'agent ne veut pas en nettoyer, il peut utiliser une tasse ou un bol qui sont d'autres contenants. Il faut pouvoir mesurer la similarité entre ces objets et définir les fonctions nécessaires, ici contenir de l'eau, pour déterminer quels objets peuvent être échangés. Lorsque l'on planifie, on peut alors utiliser des objets génériques qui seront choisis au contrôle comme "un objet pouvant contenir de l'eau" [1].

Avoir des connaissances générales de sens commun, comme le fait qu'il y ait des verres dans la cuisine, ou

qu'une tasse soit similaire à un verre, peut être utile lors d'une planification. Le PDDL ne permet pas de décrire ce type d'information car les prédicats ne peuvent être appliqués qu'aux instances du monde. Il faut donc spécifier, pour chaque verre, qu'il se trouve dans la cuisine. L'objectif classique de la planification est l'optimisation de la recherche, ce qui a incité à une représentation simple mais efficace. Cependant, si on veut aller plus loin et planifier comme un humain, il nous faut utiliser toutes les connaissances dont nous disposons. Certains ont déjà exprimé ce besoin et utilisé une mémoire externe, plus expressive, pour représenter le sens commun [13]. Pour cela ils traduisent la partie utile de leur représentation en PDDL pour faire la planification. Ils sont conscients que cette étape peut leur faire perdre de l'expressivité et, du coup, trouver un plan moins adapté à la situation, mais cela ne les dérange pas pour leurs cas d'utilisation. Les auteurs pensent que l'utilisation d'une mémoire externe est primordiale pour utiliser les connaissances générales de l'agent. Cependant, ils pensent qu'une perte d'informations par une traduction n'est pas souhaitable et ils favorisent une planification directement dans la mémoire externe, qu'elle puisse utiliser directement toutes ses connaissances.

2.6 Taxonomie d'actions

Dans la planification classique, une action est décrite par ses préconditions et ses conséquences, qui décrivent l'état que doit avoir le monde pour que l'action puisse être exécutée et l'état dans lequel il sera après l'exécution. Cependant, il n'y a aucun lien entre les différentes actions. Pourtant l'existence et le besoin de différents liens entre les actions ont été mis en évidence plusieurs fois [9, 11, 21]. En particulier pour deux liens principaux : une action peut être décrite comme une décomposition ou bien comme une spécification.

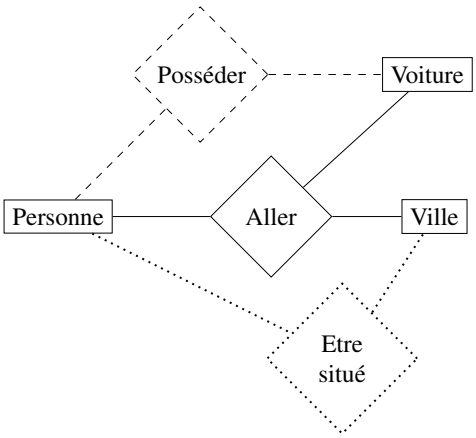
La décomposition décrit un ensemble d'actions par une seule action, la *macro-action*. Celle-ci peut être vue comme un plan condensé. Par exemple lorsque l'on cuisine un gâteau, en fait on prépare une pâte, puis on l'insère dans un moule et enfin on le fait chauffer au four. On peut décomposer chacune des actions une nouvelle fois, par exemple pour préparer la pâte il faut insérer les différents ingrédients puis les mélanger et ainsi de suite. Il peut y avoir plusieurs décompositions possibles pour une macro-action. L'utilisation de la décomposition pour de la planification n'est pas nouvelle et est à l'origine d'un framework, la planification hiérarchique, qui effectue des recherches dans des graphes de tâches [15]. La spécification décrit une action comme une autre action mais avec des spécificités. Par exemple voler et marcher sont toutes les deux des spécificités de se déplacer, la première dans les airs avec des ailes, la deuxième sur terre avec des jambes. Ces deux liens créent deux taxonomies d'actions.

3 Intérêt de la planification humaine

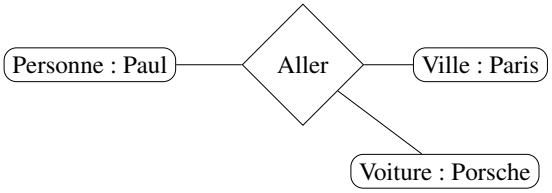
La planification classique vise des problématiques optimales et hors ligne dans des environnements sans autre acteur que l'agent. A l'inverse, la planification humaine vise des problèmes dynamiques dans des environnements complexes et où l'optimalité n'est pas le but. Cela concerne principalement deux domaines : la robotique et la simulation. Une planification humaine pourrait améliorer la planification des robots dans des environnements dynamiques et inconnus. Améliorer leurs comportements pourrait permettre de les faire paraître plus intelligents et qu'ils soient plus acceptés. La planification humaine pourrait aussi permettre de simuler des comportements humains, comme par exemple de foules, ou de voitures, pour mieux comprendre et pouvoir analyser des environnements ou des réponses à des évènements. Une planification humaine nous rapprocherait du comportement humain et cela pourrait diminuer la distance entre la simulation et la réalité.
Un autre intérêt majeur d'essayer de se rapprocher du comportement humain est dans des problématiques où l'agent et l'homme jouent un rôle important, en particulier dans des contextes de coopération. Planifier comme un humain pourrait permettre une meilleure compréhension dans les deux sens, de l'homme sur les actions de l'agent et, inversement, de l'agent sur les actions de l'homme. Ce rapprochement pourrait permettre plus de souplesse et donc une meilleure réussite dans certains problèmes où l'homme est un élément important de la représentation.
Enfin un autre intérêt, cette fois un peu plus mineur, est qu'une planification humaine peut permettre de diminuer l'effort cognitif. En effet, une planification "suffisfaisante" [19] permet de gagner du temps décisionnel par rapport à une planification parfaite [17]. Moins de temps de calcul décisionnel veut dire probablement un besoin moins grand en puissance de calcul et donc une limitation en coût énergétique nécessaire pour la planification. Cela permettrait de s'imaginer à long terme des agents planificateurs sur des plateformes moins puissantes.

4 Discussion et conclusion

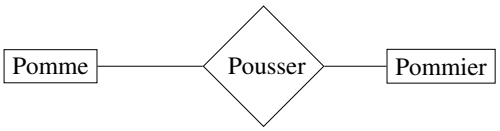
Nous avons présenté dans cet article ce que nous entendons par une planification humaine et l'avons comparé avec la planification classique : c'est la recherche continue d'un plan d'actions, répondant au mieux à une multitude d'objectifs pouvant évoluer dans le temps, dans un environnement dynamique, ouvert et inconnu. Nous avons précisé le besoin d'une planification au sens global qui comprend le contrôle. Nous avons montré le besoin d'une mémoire capable de représenter aussi bien les données classiques de la planification, comme les actions et les objectifs, mais aussi des connaissances plus générales, de sens commun. Nous pensons qu'il est important que la planification prenne en compte ces connaissances dans sa recherche pour avoir des comportements plus intelligents. Bien que le sujet puisse paraître à priori ambitieux, il est, à nos yeux, important pour des domaines comme la robotique et la simulation.



(a) Modèle d'action. Elle est composée d'une relation principale (en trait plein), d'une précondition (en tiret) et d'une conséquence (en pointillé). "Pour qu'une personne aille en ville en voiture il faut qu'il possède cette voiture et, après exécution, la personne se trouvera dans la ville"



(b) Exemple de fait "Paul va à Paris avec sa Porsche."



(c) Exemple de connaissances de sens commun "Les pommes poussent sur des pommiers"

FIGURE 2 – Extraits de graphe conceptuel. Les rectangles représentent des concepts, les losanges des relations et les arrondis des individus.

Il demande l'utilisation conjointe d'un ensemble de techniques issus de différents domaines de recherche dont la représentation des connaissances pour la mémoire, la planification pour la recherche et la science cognitive pour l'inspiration humaine.

Nous avons, pour notre part, envisagé de répondre à cette problématique en utilisant une représentation de type graphe conceptuel [20]. Ce sont des graphes composés de noeuds concepts, de noeuds relations et d'individus. Les noeuds relations sont à mettre en rapport avec les actions, les noeuds concepts sont les classes génériques des objets et les individus sont les instances du monde (Fig. 2). Ils permettent de stocker tout type d'information, aussi bien des faits (Fig. 2b), par des relations entre les individus, que des connaissances plus générales comme les actions

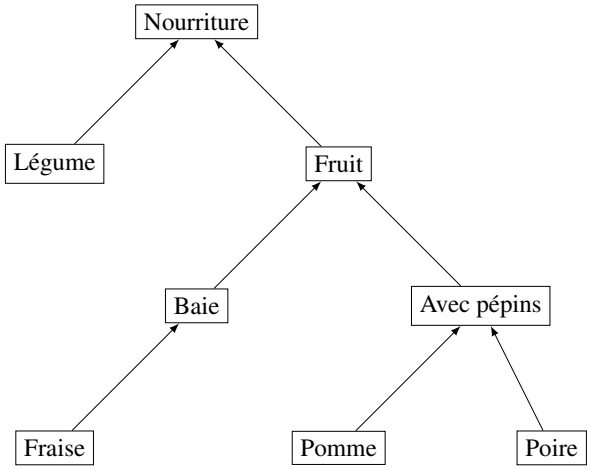


FIGURE 3 – Exemple de taxonomie.

(Fig. 2a). L'intérêt de cette représentation est qu'elle peut aussi représenter des connaissances de sens commun (Fig. 2c). Nous pensons utiliser cette spécifité notamment dans la validation des préconditions d'une action. Par exemple, si l'agent a faim et qu'il voit un pommier, il peut en déduire qu'il y a des pommes dessus et en cueillir une pour la manger. Ce type de mémoire peut représenter les taxonomies, aussi bien pour les concepts que pour les relations (Fig. 3). Une représentation graphique permet, entre autre, de pouvoir mesurer des distances entre noeuds, en particulier cela permet de trouver des alternatives proches d'un concept et donc de pouvoir échanger des objets. Par exemple dans la taxonomie, la pomme et la poire sont assez proches sémantiquement.

Concernant la planification, elle doit se faire sur plusieurs niveaux de hiérarchie : de spécification et de décomposition. L'intérêt de la spécification est que, si une action ne peut pas être exécutée à cause d'une précondition, l'ensemble de ses spécifications ne peut pas être exécuté. Nous commençons notre recherche par les actions les plus générales puis nous spécifions les actions valides. Cela permet de concentrer la recherche en abandonnant le plus tôt possible les actions qui ne pourront pas être exécutées. L'intérêt de la décomposition est de regrouper des actions sous forme de plan. En supposant que les décompositions sont valides si la macro-action est valide, en utiliser une permet de prendre un raccourci dans la recherche. La décomposition peut se faire dans une deuxième phase après la recherche.

Nous avons comme perspective de travailler sur l'aspect multi-objectifs en prenant l'idée d'une recherche en largeur mais un minimum guidé par l'ensemble des objectifs. Nous allons explorer des techniques de recherche en faisceau ou de calculs d'utilités par rapport à une multitude d'objectifs. Une autre piste de réflexion est l'utilisation des graphes conceptuels pour la généralisation des faits. L'idée est de créer un nouveau noeud relation à partir d'un en-

semble de faits qui généralise l'ensemble des informations des faits. L'objectif est de pouvoir apprendre de nouvelles macro-actions en observant des évènements du monde.

Références

[1] Iman Awaad, Gerhard K Kraetzschmar, and Joachim Hertzberg. Challenges in finding ways to get the job done. In *Planning and Robotics (PlanRob) Workshop at 24th International Conference on Automated Planning and Scheduling*, 2014.

[2] Daniel Bryce and Subbarao Kambhampati. A tutorial on planning graph based reachability heuristics. *AI Magazine*, 28(1) :47–47, 2007.

[3] Alessandro Cimatti, Marco Pistore, Marco Roveri, and Paolo Traverso. Weak, strong, and strong cyclic planning via symbolic model checking. *Artificial Intelligence*, 147(1-2) :35–84, 2003.

[4] Malik Ghallab, Craig Knoblock, David Wilkins, Anthony Barrett, Dave Christianson, Marc Friedman, Chung Kwok, Keith Golden, Scott Penberthy, David Smith, Ying Sun, and Daniel Weld. Pddl - the planning domain definition language. 08 1998.

[5] Malik Ghallab, Dana Nau, and Paolo Traverso. The actors view of automated planning and acting : A position paper. *Artificial Intelligence*, 208 :1–17, 2014.

[6] Nau D. Traverso P. Ghallab, M. *Automated Planning : theory and practice*. Elsevier, 2004.

[7] Kristian J Hammond. *Case-based planning : Viewing planning as a memory task*. Elsevier, 2012.

[8] Nick Hawes. A survey of motivation frameworks for intelligent systems. *Artificial Intelligence*, 175(5-6) :1020–1036, 2011.

[9] Barbara Hayes-Roth and Frederick Hayes-Roth. A cognitive model of planning. *Cognitive science*, 3(4) :275–310, 1979.

[10] Daniel Kahneman and Shane Frederick. Representativeness revisited : Attribute substitution in intuitive judgment. *Heuristics and biases : The psychology of intuitive judgment*, 49 :81, 2002.

[11] Christel Kemke and Erin Walker. Planning with action abstraction and plan decomposition hierarchies. In *2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 447–451. IEEE, 2006.

[12] LJ Manso, P Bustos, R Alami, G Milliez, and P Núnez. Planning human-robot interaction tasks using graph models. In *Proceedings of International Workshop on Recognition and Action for Scene Understanding (REACTS 2015)*, pages 15–27, 2015.

[13] Fiona McNeill, Alan Bundy, and Chris Walton. Planning from rich ontologies through translation between representations. In *Workshop on the Role of Ontologies in Planning and Scheduling*, pages 13–21. Citeseer, 2005.

[14] Giovanni Mirabella. Should i stay or should i go ? conceptual underpinnings of goal-directed actions. *Frontiers in systems neuroscience*, 8 :206, 2014.

[15] Dana S Nau, Tsz-Chiu Au, Okhtay Ilghami, Ugur Kuter, J William Murdock, Dan Wu, and Fusun Yaman. Shop2 : An htn planning system. *Journal of artificial intelligence research*, 20 :379–404, 2003.

[16] Giovanni Pezzulo and Cristiano Castelfranchi. Thinking as the control of imagination : a conceptual framework for goal-directed systems. *Psychological Research PRPF*, 73(4) :559–577, 2009.

[17] Martha E Pollack. The uses of plans. *Artificial Intelligence*, 57(1) :43–68, 1992.

[18] Martha E Pollack and John F Horty. There's more to life than making plans : plan management in dynamic, multiagent environments. *AI Magazine*, 20(4) :71, 1999.

[19] Helmut Simon et al. *Models of a man : Essays in memory of Herbert A. Simon*. MIT Press, 2004.

[20] John F Sowa. Conceptual graphs for a data base interface. *IBM Journal of Research and Development*, 20(4) :336–357, 1976.

[21] Lee A Spector. Supervenience in dynamic-world planning. Technical report, MARYLAND UNIV COLLEGE PARK SYSTEMS RESEARCH CENTER, 1992.

[22] Swaroop Vattam et al. Breadth of approaches to goal reasoning : A research survey. In *Naval Research Lab Washington DC*, 2013.

[23] David E Wilkins et al. A call for knowledge-based planning. *AI magazine*, 22(1) :99, 2001.

LSTM Path-Maker : a new LSTM-based strategy for Multiagent Patrolling *
LSTM Path-Maker : une nouvelle stratégie pour la patrouille multiagent basée sur l'architecture LSTM

Mehdi Othmani-Guibourg[1,2] Amal El Fallah-Seghrouchni[2] Jean-Loup Farges[1]

[1] ONERA, 31000, Toulouse, France
[2] Sorbonne Université - Faculté des Sciences
CNRS, UMR 7606, LIP6,
F-75005, Paris, France

{prénom}.{nom}@onera.fr
{prénom}.{nom}@lip6.fr

Résumé

Depuis plus d'une décennie, la tâche de la patrouille multiagent a attiré l'attention de la communauté multiagent de manière croissante, en raison de son grand nombre d'applications potentielles. Cependant, les algorithmes basés sur des méthodes d'apprentissage profond pour traiter cette tâche sont à ce jour peu développés. Dans cet article, nous proposons d'intégrer un réseau de neurone récurrent à une stratégie de patrouille multiagent. Ce faisant, nous avons proposé un modèle formel de stratégie d'agent basée sur l'architecture LSTM, que nous avons nommé LSTM-Path-Maker. Le réseau LSTM est entraîné sur des traces de simulation d'une stratégie coordonnée et centralisée, puis embarqué dans chaque agent en vue de patrouiller efficacement sans communication. Enfin, cette nouvelle stratégie basée sur l'architecture LSTM est évaluée en simulation et comparée d'une part à une stratégie coordonnée et d'autre part à une stratégie réactive. Les résultats préliminaires indiquent que la stratégie proposée est meilleure que la stratégie réactive.

Mots Clef

Systèmes multiagents, Réseaux de neurones artificiels, Réseaux récurrents à mémoire court et long terme

Abstract

For over a decade, the multi-agent patrol task has received a growing attention from the multi-agent community due to its wide range of potential applications. However, the existing patrolling-specific algorithms based on deep learning algorithms are still in preliminary stages. In this paper, we propose to integrate a recurrent neural network as part of a multi-agent patrolling strategy. Hence we proposed a formal model of an LSTM-based agent strategy named LSTM Path Maker. The LSTM network is trained over simulation traces of a coordinated strategy, then embedded on each agent of the new strategy to patrol efficiently without communicating. Finally this new LSTM-based strategy is evaluated in simulation and compared with two representative strategies : a coordinated one and a reactive one. Preliminary results indicate that the proposed strategy is better than the reactive.

Keywords

Multiagent Systems (MAS), Artificial Neural Networks (ANN), Long Short-Term Memory (LSTM)

1 Introduction

The generic task of patrolling is by nature conveniently well-suited for being shared in space and time by several agents. There is a wide variety of tasks that may be formulated as particular multi-agent patrolling (MAP) problem. As a concrete example, the task consisting in monitoring an area by a swarm of drones faces with the problem of coordinating them to patrol that area. Area monitoring is useful as part of crises, for example in order to detect a start of fire in a forest, but also to provide an alert, either to save people or to detect the presence of intruders as part of a complex humanitarian mission in a conflict area.

A fully-fledged feature of patrolling and other complex systems is the difficulty to derive analytical results from their system-wide equations. Thereby it appears that the only method enabling to predict their behaviour is to simulate the local interactions of their components : this is exactly the main purpose of agent-based simulation. Thus, the quality of a patrolling strategy is evaluated in simulation by using different measures, each one measuring a specific property of distributions of visits generated by strate-

* Paper presented at the 52nd Hawaii International Conference on System Sciences (HICSS52 2019), titre, résumé et mots-clés en français ajoutés.

gies. Informally, it is consensual that a good strategy is one that minimises the time lag between two passages on the same place and for all places.

For over fifteen years different types of agent strategy for the (MAP) were proposed : centralised [1], emergent [1], idleness-based [1], heuristic (idleness and distance) with pathfinding [2], hamiltonian-cycle-based [3], TSP-heuristic-based [4], reinforcement-learning-based [5] and even auctions-based [6] strategies. In this context, Almeida et al. [2] defined two main types of agent : reactive agents that act only according to their perception, and cognitive agents that can pursue a goal.

Until now, as part of the cooperative multi-agent learning, few works concentrate on the problematic of using Artificial Neural Networks (ANNs) for the multi-agent patrolling. For example, a few of these studied non-hierarchical neural network-based methods for planning a complete coverage patrolling path where each neuron encodes a specific region of the space, such as Guo et al. [7] or even a cooperative multi-agent learning where each robot is endowed with a neural network directly connected to nodes of others robots' internal neural network whose weights of connections are evolved, with D'Ambrosio et al. [8]. However, none tackles the advantages that may be afforded by deep artificial neural networks in order to outperform the previous strategies. *This paper thereupon proposes the use of the ANN architecture Long Short-Term Memory (LSTM) for the multi-agent patrolling problem. The Recurrent Neural Networks (RNN), as machines to learn temporal series, are well adapted to this problem to the extent that they can be viewed as a temporal decision problem. In this way, a new strategy based on the LSTM architecture is introduced where the ANN is used as a path generation device by non-communicating agents to navigate as optimally as possible through the graph. To that end each neural network architecture is trained offline over data generated for this purpose, then embedded in the agents which will use it to select the next node to visit with respect to the previous ones. Finally, the performances are evaluated according to the usual evaluation criteria used until now in this field of study.*

The Section 2 presents the background on the multi-agent patrolling and the LSTM networks useful to understand proposed developments as well as the previous works using the ANNs for the MAP. Then, Section 3 introduces LSTM Path-Maker (LPM), the new strategy for the multi-agent patrolling based on the LSTM architecture. In Section 4 the learning results are analysed and and the new strategies evaluated. Finally, Section 5 draws some conclusions, shows certain boundaries for this new strategy and indicates directions for further works.

2 Background

This section presents the background on multi-agent patrolling and the LSTM architecture.

2.1 Multi-agent patrolling

Formal definition. The MAP model consists formally of a society of agents noted $\mathbf{A}$, able to move in an environment with the same mobility parameters, and a graph noted $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ as an abstraction representing a discretisation of the area to patrol. Here, $card(\mathbf{V}) = N$ and $\mathbf{V} = \{1, .., N\}$ is the set of nodes identified by their indexes and standing for the places to visit. $\mathbf{E}$, which is included in the set of 2-element subsets of $\mathbf{V}$, is the set of edges of $\mathbf{G}$ accounting for the paths between the places. With each edge $\{v, w\}$ corresponds a *transit time* $c_{v,w}$ representing the travel time of the edge $\{v, w\}$. At the beginning of an instance of a patrol task, agents are positioned on nodes of $\mathbf{G}$. To each node is associated a dynamic variable named *idleness*, indicating the time elapsed since it has not been visited by any agent[4]. The idleness of a node v at time t, noted $i_t(v)$, is defined as being the amount of time elapsed since that node has received the visit of an agent. The idleness of all nodes at the beginning of the patrolling task is set to 0. Finally, each time an agent arrives at a node v, it shall decide, among the edges including v, namely all the edges $\{v, w\}$, the next edge to travel.

Strategies. A *strategy* of agent is an information processing method, or algorithm, allowing each agent to take a decision each time it arrives at a node. In the MAP, whatever the strategy considered, each agent intends actions based on its appropriated perceptions from the environment and its knowledge about idlenesses of nodes. Among the wide family of strategies, two are relevant for this work as representative strategies : *Conscientious Reactive* (CR) and *Heuristic Pathfinder Cognitive Coordinated* (HPCC).
The algorithm of CR is to select the next node to visit as the one with the highest idleness in its neighbourhood. There is no communication between agents : idlenesses are estimated by each agent on the basis of its own path. CR can be thought of as a good representative and thereby a comparison strategy for the reactive and decentralised ones. Note that for CR as well as for any decentralised strategy, the considered idlenesses are estimated by each agent from its own visits to nodes. This estimation is an overestimation of the actual value of idlenesses because visits of other agents are neglected.
For HPCC, there is a perfect communication between agents : idlenesses are estimated by a coordinator on the basis of all paths of agents. The decision process includes two steps :

- selection of a target node that is not necessary in the neighbourhood,
- computation of a path between the current position of the agent and the target node previously selected.

The selection of the target node takes into account not only the normalised idleness, but also the normalised *time to go* of a candidate goal node from the agent's current position. The time to go between two nodes of $\mathbf{V}$ corresponds here to the shortest path between these two nodes. Idleness and time-to-go are normalised by scaling from 0 to 1. A zero

normalised value is attributed to a the maximum idleness - encouraging the agent to traverse nodes with high idleness, since the objective function will be minimized - whereas a value equal to 1 is attributed to the minimum idleness. Intermediary values are calculated by means of proportions as shown in **(1)** :

$$\begin{aligned} &\text{If } min_{v\in\mathbf{V}}\{i_t(v)\} \neq max_{v\in\mathbf{V}}\{i_t(v)\}, \forall v_0 \in \mathbf{V} \\ &\bar{i}_t(v_0) = \frac{max_{v\in\mathbf{V}}\{i_t(v)\} - i_t(v_0)}{max_{v\in\mathbf{V}}\{i_t(v)\} - min_{v\in\mathbf{V}}\{i_t(v)\}} \end{aligned} \tag{1}$$

where $i_t(v)$ and $\bar{i}_t(v)$ are the global and normalised idleness, respectively.
Normalised time to go is calculated similarly. For that purpose, at the minimum time to go is attributed a zero normalised value - encouraging the agent to traverse edges with short distance, since the objective function will be minimized - whereas at the maximum time to go is attributed a value equal to 1. Intermediary values are calculated by means of proportions as shown in **(2)** :

$$\begin{aligned} &\forall d(v_0, v) \text{ a time-to-go from } v_0 \text{ to } v, \\ &\bar{d}(v_0, v) = \frac{d(v_0, v) - min\{d\}}{max\{d\} - min\{d\}} \end{aligned} \tag{2}$$

where $max\{d\}$ and $min\{d\}$ are the maximum and the minimum time-to-go respectively, over all the $v, w \in \mathbf{V} : v \neq w$.
Finally, for an agent at the position v_0 at time t, the values associated to nodes are given by **(3)** :

$$\begin{aligned} &\forall v \in \mathbf{V}, val_{r_H}(v, t) = \\ &\qquad r_H \times \bar{i}_t(v) + (1 - r_H) \times \bar{d}(v_0, v) \end{aligned} \tag{3}$$

where the weighting factor $r_H \in [0, 1]$ must be chosen by the strategy designer. Minimising the node values according to that expression i.e. selecting the nodes with the minimum value, allows agents to visit nearby nodes with higher idleness first and foremost. Moreover, there is a mechanism forbidding the coordinator to select nodes that are currently assigned to other agents.
The path computation takes into account the idleness of the nodes between the current location and the goal to compute the best path leading there. For that, it weights the edges as shown in **(4)** :

$$\begin{aligned} &\forall e \in \mathbf{E} : e = \{v, w\}, \\ &\qquad c_{r_P}(e) = r_P \times \bar{i}_t(w) + (1 - r_P) \times \bar{c}_{v,w} \end{aligned} \tag{4}$$

where the weighting factor $r_P \in [0, 1]$ must be chosen by the strategy designer. In that case, it is the normalised transit time $\bar{c}_{v,w}$ of edge and not the normalised time-to-go $\bar{d}(i, j)$ of path that is used to value edges :

$$\forall \{v, w\} \in \mathbf{E}, \bar{c}_{v,w} = \frac{c_{v,w} - min\{c\}}{max\{c\} - min\{c\}} \tag{5}$$

where $max\{c\}$ and $min\{c\}$ are the maximum and the minimum transit times, respectively.
Minimising the edge weights according to that expression allows agents as well to visit nearby nodes with higher idleness first and foremost.
HPCC as a communicating, fully-informed, coordinated, and thereby centralised strategy is one of the best online - namely without pre-calculation of paths - strategy. It can be then regarded as a representative and thereupon a comparison strategy for the coordinated and centralised ones. Note also that HPCC uses actual idlenesses because visits of nodes by all agents are analysed by the coordinator in a centralised manner.

Evaluation criteria. Sampaio et al. [9] introduced evaluation criteria, relevant to establish aggregation measures not based on idleness but on the intuitive concept of *interval between visits* to the node. In this class of evaluation criteria, the size of intervals between visits at each node is calculated by registering the value of idleness just before each visit by an agent. All intervals for all nodes are used to make an aggregated calculation. The two interval-based evaluation criteria we selected are the *Mean Interval* (MI) and the *Quadratic Mean Interval* (QMI), the mean and the root mean square respectively, on all intervals between visits of a mission execution. QMI as quadratic mean, takes better into account the difference of time interval between the nodes and thus, measures the tendency of nodes to be equitably visited through a simulation run.
In order to better evaluate the contribution of each agent when the population size varies, these evaluation criteria are normalised by multiplying values by the number of agents.

2.2 LSTMs

Recurrent Neural Networks (RNNs) are neural networks that process an input sequence one element at a time, maintaining in their hidden units - neurons in the *hidden layers* - a *state vector* called *hidden state*, containing information about the history of the sequence's past elements. Each output of the hidden units h_t, depends upon the hidden state h_{t-1}. This hidden state can be viewed as a *memory*. Indeed, adding memory to a neural network allows to process information of the sequence itself : the sequential information is preserved in the hidden state that enable to find correlations between events separated by several time steps. This memory is contained in the *hidden layers* which have a *feedback loop*, and therefore they constitute *recurrent layers*.
LSTM are a special kind of RNN introduced and designed to take into account long-term dependencies. They have the same general chain structure as the RNNs except that the repeating module has a different structure as shown in **Fig. 1**. In the first place, as stated by Hochreiter et al. [10], an LSTM network was a RNN with one input layer, one fully self-connected hidden layer containing purpose-built *memory cells*, *gate units*, and an output layer. This memory unit corresponds to a neuron with a recurrent self-

connection. Thereby a cell referred originally to an object with a single scalar output. The activations of those neurons within the memory units constitute the *state* noted c_t, sometimes called *cell state*, of the LSTM network.

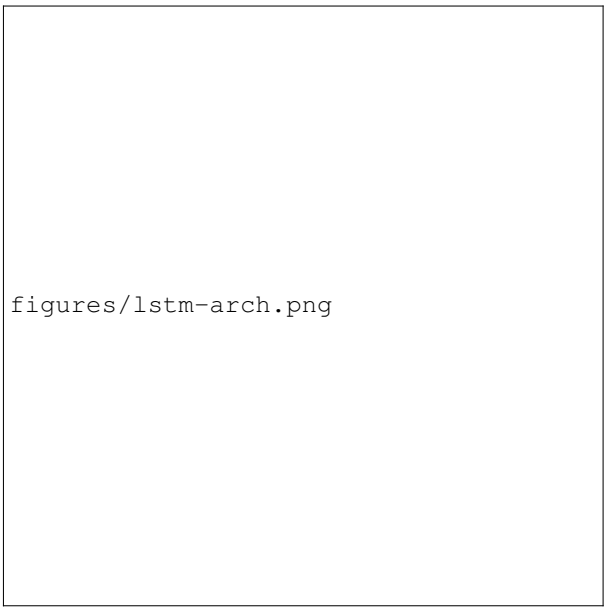


FIGURE 1 – Layered LSTM unit : the core composant of the LSTM architecture.

As stated by Graves et al.[11] an *LSTM layer* consists of a set of recurrently connected blocks, known as *memory blocks*, which in turn consists of cells. One cell, as a neuron, outputs one scalar. Originally, each memory block has contained one or more layered recurrently connected neurons called memory cells and sharing the same three multiplicative units : i_t the *input gate*, o_t the *output gate* and f_t the *forget gates*, i.e. all the cells of a memory block are connected to the same gate units. The gate units provide continuous analogies of write, read and reset operations for the cells. A memory block of size 1 is then a simple memory cell[10] connected to $tanh$ activations. These blocks, can be thought of as a differentiable version of the memory chips in a digital computer. In doing so, it follows the network can only interact with the cells via the gates. Besides, the memory block and the gates form the *LSTM unit* as shown in the **Fig. 1**, which corresponds to a repeating module. The state is thereupon, the memory accumulated by the LSTM through time by using its forget, input and output gates. However, unlike the base RNN model in which it cover the same concept, the cell state c_t must here not be confused with the hidden state h_t, the former being the cell output while and the latter the output of the hidden layers.
Also, it should be emphasised that the hidden state, respectively the cell state, noted h_t, respectively c_t, of an LSTM network, must be distinguished from the hidden state, respectively the cell state, of the layer l (for a multi-layer LSTM) noted h_t^l, respectively c_t^l.

For some years and hitherto, most implemented LSTM architectures contain only one cell in their LSTM units. The LSTM units of a same layer can thereupon be "layered" into only one LSTM unit where for all t a time step, i_t, f_t, o_t and c_t, the input gate, forget gate, output gate and cell activation turn into vectors with the same size as the hidden vector h_t ; hence the element-wise multiplication $*$. In that context, an LSTM layer can be viewed as a vectorial LSTM unit and thereby the vectorial cell and gates compose a layer. It follows that defining the size of a layer's cell defines that of its memory cell block and that of its hidden state in cascade.
The hidden state output from an LSTM layer l is then computed from the following composite function :

$$i_t^l = \sigma(W_{xi}^l\, x_t^l + W_{hi}^l\, h_{t-1}^l + b_i^l) \tag{6}$$
$$f_t^l = \sigma(W_{xf}^l\, x_t^l + W_{hf}^l\, h_{t-1}^l + b_f^l) \tag{7}$$
$$o_t^l = \sigma(W_{xo}^l\, x_t^l + W_{ho}^l\, h_{t-1}^l + b_o^l) \tag{8}$$
$$c_t^l = f_t^l * c_{t-1}^l + i_t^l * \tanh(W_{xc}^l\, x_t^l + W_{hc}^l\, h_{t-1}^l + b_c^l) \tag{9}$$
$$h_t^l = o_t^l * tanh(c_t^l) \tag{10}$$

The parameters of an LSTM layer that must be learned for a layer l are thereby :
— $W_{xi}^l, W_{hi}^l, W_{xf}^l, W_{hf}^l, W_{xo}^l, W_{ho}^l, W_{xc}^l$ and W_{hc}^l
— b_i^l, b_f^l, b_o^l and b_c^l
The structure corresponding to several memory blocks in a layer l can be derived from its more general architecture by setting to 0 the elements of W_{hi}^l, W_{hf}^l, W_{ho}^l which are not block-diagonal.
Deep LSTMs combine the multiple levels of representation that have proved so effective in deep networks with the flexible use of long-range context that empowers RNNs. The architecture of the deep LSTMs is the same as that presented previously apart from the fact that there are several LSTM layers.

2.3 Related works

Few works addressed the problematic of the use of ANNs in the context of the MAP. Among related works, Guo et al. [7] studied the use of ANN-based methods for planning a complete coverage patrolling path. In that work, the area to patrol is discretised into fixed radius disks that can be thought of as nodes to visit. Then, each neuron, as a state variable, represents a region activated negatively or positively in function of either the presence of obstacle, or the absence of a visit by the patrol, respectively. Finally, the activities of all the neurons compose a dynamic landscape such that the non-visited regions globally attract the robot in the entire space, and the obstacle locally repel the robot to avoid collisions. Even though being an original and interesting approach, the type of neural network used in this work is not relevant to our problematic laid down. Indeed, the latter consists of learning temporal sequences

which corresponds here to paths in the graph. Also, in that work only one neural network was used concomitantly by all agents, while in our current framework agents do not communicate, and instead, act in a decentralised way.

D'Ambrosio et al. [8] developed a new communication scheme they called the *hive brain*, as part of the cooperative multi-agent learning. In this scheme, each robot is endowed with a neural network directly connected to nodes of others robots' internal ANN, whose weights of connections are evolved. As stated by the authors, this technique is drawn from an interesting physical phenomenon called *odd sympathy* [12], which is the tendency of pendulum clocks to synchronise when mounted near each other due to a small amount of physical information transferred between the pendulums. Thereby they elaborated the hive brain in an analogical way where the robots learn to synchronise by training their respective ANN in a robot simulator; the training is performed here using an evolutionary algorithm. In our perspective, this work presents the same problem than the foregoing, namely the implicit use of communications in the simulator between agents to feed the *brain* of one agent from another one. However, it inspired our new strategy of agent to the extent that each agent embeds its individual ANN. Also, although in our work agents are currently embedded with the same trained ANN whose the parameters remain unchanged during a mission execution, in a not too distant future it will bee valuable to draw from this work to synchronise the agents' ANNs' state and thereby improve our new strategy's performances.

Finally, the works of Sales et al. [13] is also related to our problematic to some extent. Indeed, they developed an autonomous patrolling system composed of four intelligent robots that can freely move through an indoor environment and detect intruders. The robots are endowed with a localisation/navigation system composed of an ANN used in combination with a Finite State Machine (FSM), whose the states correspond to the key features of the environment. The FSM associates a sequence of actions to execute with a sequence of states. The ANNs process the sensors data to identify and classify the FSM states (current and transitions), and to determine the actions to perform. After being trained offline to identify the key features of the environment such as corridors, intersections and turns, they are fed into data obtained from robots' sensors, then they output the FSM states. From this work we retained the method to train the ANNs offline in order to set them for a specific mission leading agents to navigate as efficient as possible without communicating. Lastly, in this work, each robot calculates the shortest path by using A* to reach the intruder's position when detected taking into account its teammates, while in ours, the network is used to select the next node in the neighbourhood with respect, on the one hand, to the previous ones, and on the other hand, to what was learned during the offline training stage.

3 An LSTM-based strategy

This section presents our contribution, that is LPM, a new LSTM-network-based decentralised agent strategy, which learns to navigate the nodes composing the area to patrol, from series of histories collected upon numerous simulation executions of a fully-informed and coordinated strategy : the HPCC strategy. The first assumption has been that if agents learn in average to behave in the same way as the coordinator, which is fully-informed, then they may approach performances reached by the coordinated strategy. The main goal of this work is to use LSTM networks to perform that.

3.1 Formal definition

The LPM strategy is an ANN-based strategy : the decision-making process is carried out by means of an LSTM network which outputs the next node from the current node provided as input of the network. This strategy can be thought of as a reactive strategy using an artefact for guidance through the area to patrol, such as a compass, which takes implicitly into account the idleness of nodes and the agents' positions. In our context, the temporal series representing the successive visited nodes by an agent is called a *path*. Any vertex of a path, has for subsequent vertex one of its neighbours.
For a given scenario, the LSTM network temporally learns the next node to visit v_{t+1} from a *model strategy*, according to the previous ones $v_t, v_{t-1}, ..., v_0$ in the path and that for all paths : each path, as a temporal series accounting for the path of an agent over the graph, is fed into the network node after node. It follows that, with defining f as the decision procedure of the model strategy - and thereby the strategy itself -, the LSTM network of the scenario that approximates f can be defined as follows :

Let $I_t = \{i_t(v_0), i_t(v_1), ..., i_t(v_N)\}$ being the set of shared idlenesses at the time t and v_a the node from which a next node to visit, noted $\bar{v}$, must be selected as a decision process, by an agent a. Then, the next node to visit $\bar{v}$ will be selected from the procedure f, the *requesting node* at the time t $v_t \in \mathbf{V}$ which corresponds to the node visited by an agent a at the time $t \in \mathbf{T}$, and the set of shared idlenesses I_t such as :

$$\forall t \in \mathbf{T}, \bar{v} = f(v_t, I_t) \tag{11}$$

Thus, with considering $\tilde{f}$ the vertex-purpose-LSTM-network-based decision procedure as a function approximator of f, and v_t we have :

$$\forall t \in \mathbf{T}, v_{t+1} = \tilde{f}(v_t, ..., v_1) \tag{12}$$

This equation pertains to the formal definition of an LSTM network : each output depends upon the previous outputs.
Let N the number of nodes in the graph. With the aim of feeding the LSTM network with the most appropriate and relevant information about the nodes, each node has been

encoded as a N-dimensional one-hot vector : for the vertex v_i, all the coordinates of the vector will be set to 0 except the i-th coordinate which will be set to 1. The output of the network thereupon is an N-dimensional vector whose the values are in $[0, 1]$; these values can be regarded as probabilities. Thus, to ensure that all values are positive and their sum equal to one, the output layer of the networks is a softmax layer. The node represented by the maximum output vector's coordinate should be selected as the next one to visit.

Let (L, H) the *profile of parameters* of an LSTM architecture so that L and H stand for the number of layers and the number of hidden units (or memory cells) per layer respectively, of a given LSTM architecture. Formally, by defining $b : \mathbf{V} \rightarrow \mathbf{V_{bin}}$ as being the function mapping all the indices of nodes into their one-hot representation, the proposed architecture can then be described with :

$$x_t^1 = b(v_t) \tag{13}$$

$$\text{If } L > 1, \forall l \in \{1, ..., L-1\}\ x_t^{l+1} = h_t^l \tag{14}$$

$$L_{net} = softmax(W \cdot h_t^L) \tag{15}$$

where $dim(h) = H$ and W is a $card(V) \times H$-matrix of parameters.

Finally, upon training stage's completion, each LPM agent will be endowed with the same parametrised LSTM network.

3.2 Network training

The training of the LSTM network is performed from logged paths of any high-performance strategy f. Generally, the high-performance strategies make use of communications and centralised decision-making process. The purpose here, is then to approach the performances of these strategies without communicating and thereupon *distributing* and *decentralising* the decision-making process. Indeed, for example in the context of a drones' reconnaissance mission or even silent bots penetrating a network, communications may be impossible or discouraged. Such a strategy to learn will be called the *model strategy* or simply the *model* if that does not lead to confusion.

For each *scenario* $\{f, \mathbf{G}, N_a\}$, also called *simulation configuration* or simply *configuration*, with N_a the number of agents, whether it does not cause any confusion, the LSTM network is first pre-trained with the purpose of learning the topology i.e. the structure of the graph representing the area. Thereafter, the network is trained over all the paths retrieved from the executions of configuration for $\{f, \mathbf{G}, N_a\}$, so that it learns to output with the highest probability the next node to visit in the path. The process described here can be thought of as performing sequence modelling where the sequence is a path of nodes ; here the sequence modelling corresponds to a *path generation*.

As aforementioned in the **Subsection 3.1**, the network's output layer is a softmax layer. It can be interpreted as a probability distribution. Thus path generation aims at learning a probability distribution over paths by minimising the cross-entropy of a model given a set of N training sequences of length T :

$$\min_{\boldsymbol{\theta}} -\sum_{n=1}^{N}\sum_{t=1}^{T} log\ p(v_t^n | v_1^n, ..., v_{t-1}^n; \boldsymbol{\theta}) \tag{16}$$

where $\boldsymbol{\theta}$ is the set of the model's parameters, whose the dimension is $dim(H) = 4(2L - 1)H^2 + (4L + 5Card(V))H$, and p is the predicted probability for the current element of the observed sequence (v_t^n).

3.3 Decision

Generally, despite of the pre-training stage, the network may output the highest probability for a node that is not in the neighbouring of the one given in input. In doing so, the decision shall be made only among the output probabilities standing for the neighbour nodes. It follows that each time the one-hot vector of the current vertex is presented to the network, the decision procedure $\tilde{f}$ concerning the next node to visit consist of selecting the next node among the neighbours of the current vertex with the maximum probability. This can be mathematically rewritten as bellow :

Let :

- $\mathbf{V_{bin}} \subset \{0, 1\}^{card(V)}$ be the set of nodes formatted into one-hot vectors,
- $L_{net} : \{0, 1\}^{card(\mathbf{V})} \rightarrow [0, 1]^{card(\mathbf{V})}$ the function represented by the LSTM network used here,
- $Ng : [0, 1]^{card(\mathbf{V})} \times \mathbf{V} \rightarrow [0, 1]^{card(\mathbf{V})}$, the function setting to zero the values of the coordinates not corresponding to the neighbours of a given node's one-hot vector.

Then,

$$\begin{aligned} &\forall t \in \mathbf{T}, \forall v_t \in \mathbf{V_{bin}}, \\ &\quad v_{t+1} = argmax(\ Ng(L_{net}(b(v_t)), v_t)\ ) \end{aligned} \tag{17}$$

It then follows that :

$$\begin{aligned} &\forall t \in \mathbf{T} :\ v_t \in \mathbf{V}, \\ &\quad \tilde{f}(v_t, ..., v_1) = argmax(\ Ng(\ L_{net}(b(v_t)), v_t\ )\ ) \end{aligned} \tag{18}$$

with $\tilde{f}$ depending upon $v_1, ..., v_t$ due to their relevant features being stored in the memory of L_{net}.

An alternative way to use the output of the LSTM would be to compute a path leading to the node output by the network and to provide the first node of this path as v_{t+1}. This procedure was omitted because the network learns to predict only neighbour nodes.

Finally, the first experiments showed that using LSTM network as it stands, tends to lead agents to converge indefinitely towards a small set of nodes, leaving thereupon others nodes non-visited until the end of the execution. In doing so, the decision procedure was slightly improved : henceforth, with the aim to make the system more robust, the

next vertex to visit from the current one is randomly selected according to the distribution of probability output by the LSTM network, normalised over the neighbourhood of the current vertex using the Bayes' theorem over the distribution of neighbours. This new procedure enables therefore to add a little randomness in the decision process when selecting the next node in the neighbourhood, leading to increase the robustness of the system, and thereby to avoid agents to visit only a restricted set of nodes. This new resulting strategy was called *Random-Next-Neighbour-LSTM-Path-Maker*, abbreviated *RLPM*.

4 Experiments and results

LPM was tested and compared to HPCC and CR. This section presents the results pertaining to.

4.1 Scenarios

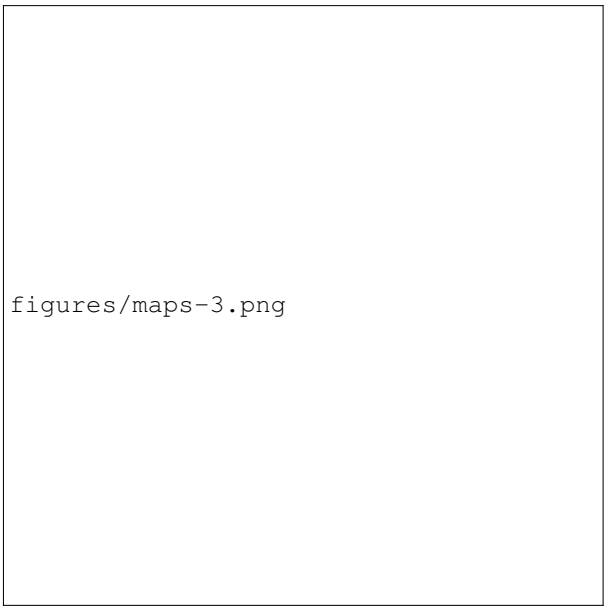


FIGURE 2 – Graphs used for assessment.

In order to align with previous works carried out in this field [14] and pursue the experiments in a comparative way, three different graphs were selected to evaluate the strategies CR and HPCC as a benchmark for the MAP : *Islands*, *Grid* and *A*, as shown in the **Figure 2**. Considering the different structures of these graphs, each one can be thought of as the representative of a class of graphs, hence the choice.

For each graph we tested in simulation the strategies CR, HPCC with a value of 0.2 for r_H and r_P, RLPM was trained from HPCC's simulation with the same value for r as well, then tested. To that end we used *Pytrol*, a new Python MAP-purpose simulator that we specially designed for this purpose, while a MAP-specific training program was coded using the deep learning library *PyTorch* to train the ANNs. These tests were performed over population sizes of 1, 5, 10, 15 and 25 agents and for each size we selected 100 random starts, also called executions. For each start, each strategy was tested over 3000 time steps. For any topology, each execution in simulation on an high-performance server takes approximately between 20 seconds and 130 seconds for 1 and 25 agents, respectively. This time complies with real life applications involving drones equipped with more basic computers and patrolling an area. Considering that each move takes exactly one period in the proposed model, after excluding the moves upon edges, an agent visits in average 600 nodes during one execution of 3000 time steps. In doing so, the paths used to train the LSTM networks have approximately a length of 600 nodes.

4.2 Training results

For each scenario, we trained seven architectures with profiles of parameters, defined as (L, H) in **Section 3.1** : $(1, 1)$, $(2, 2)$, $(4, 10)$, $(1, 50)$, $(2, 50)$, $(3, 50)$, and $(50, 2)$, with an end-to-end training i.e. a non truncated back-propagation through time. For any architecture we trained over 10000 epochs one LSTM network for each simulation configuration in two stages using the PyTorch library. First, the network is pre-trained over 2×10^6 epochs to capture as far as possible the structure of the topology, with 2-length series which stand for the edges of the graph. Then, the network is trained over the paths of agents with parameters initialised with the values learnt during the pre-training stage, over 10000 epochs. The **Figure 3** shows the initial and final values of the cost, that is the cross-entropy, during the validation stage for each architecture, averaged over the maps and numbers of agents. Here the initial cost corresponds to the validation cost after the first epoch. This figure shows that the better networks are $(2, 50)$, $(1, 50)$ and $(4, 10)$. Interestingly, the initial and final cost for the architecture $(50, 2)$ are almost identical and it has the worst cost with a value of 3.87. This result tends to show that the network's parameters converged very quickly, that is in 1 epoch. The number of parameters for $(1, 1)$, $(2, 2)$ and $(50, 2)$ are 258, 564 and 2484 respectively. It seems that those numbers are too low for a satisfactory approximation of the sequences. At the opposite, $(3, 50)$ has 63100 parameters. It is likely that this number is too large to avoid overfitting and a relatively bad performance in term of validation cost. Indeed, for one agent the size of the training data is approximately 50000, that is lower than the number of parameters of the $(3, 50)$ network.

Considering the bad final validation costs of $(50, 2)$, $(1, 1)$ and $(2, 2)$, of 3.87, 3.03, and 2.08 respectively, we tested and evaluated the four LSTM architectures : $(4, 10)$, $(1, 50)$, $(2, 50)$, $(3, 50)$. Thus, each architecture has given rise to four variants of RLPM named RLPM-L-H.

4.3 Performance results

To evaluate their performances, the RLPMs were tested and compared with CR, the reactive strategy, and HPCC, the cognitive one wherefrom they were trained. We used normalised *MI* and *QMI* as evaluation criteria, also referred to

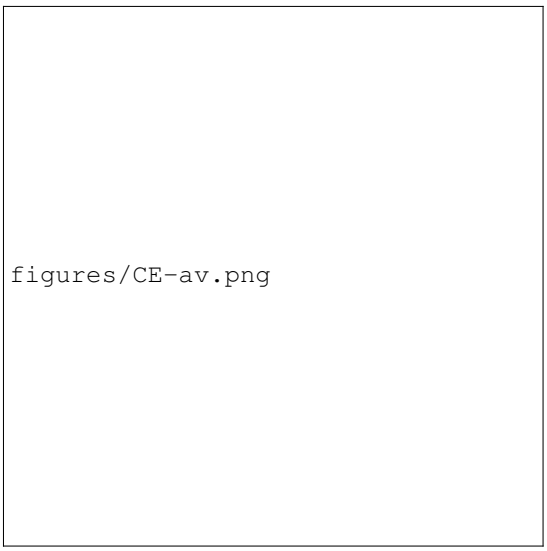


FIGURE 3 – Costs averaged over the maps and numbers of agents for each architecture.

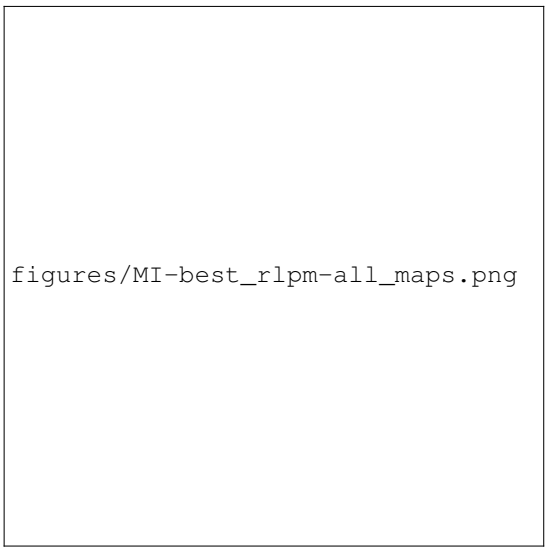


FIGURE 4 – Normalised MI of the evaluated strategies in y-axis for the three maps and the population sizes of agents in x-axis.

as metrics.

Fig. 4 shows all the results from our experiments for the normalised MI. For the sake of clarity, we show the RLPM version with the best value over this criterion , i.e. with the lowest value of MI, and that for each configuration. The best corresponding version for each configuration is denoted upon the graph. Not surprisingly HPCC always outperformed all the other strategies (CR, and the RLPMs) on all the maps and for all the population sizes of agents, except for the map A where RLPM-3-50 barely outperforms HPCC for 1 agent with a MI of 210 against 212 for HPCC. For all the maps the RLPMs overwhelmingly outperform the reactive strategy CR. For all the sizes of agent societies, the RLPMs are very close from HPCC, especially for 1 agent where their difference over MI ranges from -2, as previously stated for the map A where RLPM-3-50 is even better than HPCC, to 11 for the map $Grid$ with a value of 251 for RLPM-1-50 against 240 for HPCC. Besides, the evolution of RLPMs' performances over MI with respect to the number of agents fit rather well the HPCC's ones with an average difference of MI over all the population sizes for the three maps of 15.

For the maps Islands and A the architecture $(2, 50)$ is always the best. However, for the map Grid the architecture $(1, 50)$ is the best for 1, 5, 10 and 25 agents, but for 15 agents it is $(4, 10)$. Further analyses showed that the architecture $(1, 50)$ for 15 agents is only worse than $(4, 10)$ of 1 time step, 297 against 296 for $(4, 10)$. RLPM-1-50 is thus globally the best strategy for this map. Also, for the same map the average difference of performances over the population sizes between the best architectures previously enumerated and the architecture $(2, 50)$ is only of 2 time steps. This leads to consider RLPM-2-50 as being globally the best RLPM strategy for MI.

The **Fig. 5** shows the results for the normalised QMI. As for MI, for the sake of clarity, it is only showed the RLPM version with the best value, i.e. with the lowest value of QMI, that for each configuration. As well, the best corresponding version for each configuration is denoted upon the graph. For the map Islands, the QMI of the best RLPM is worse than HPCC and CR for all the numbers of agents, except for 25 agents where CR is worse of 28. Also, it must be pointed out that for 1 agent CR is a little better than HPCC, 290 against 320. This result can be explained by the topology of the map in combination with the HPCC's decision-making rule regarding the next node to visit : HPCC takes into account the distance from the agent's current node while CR chooses its next node to visit as the one having the greatest idleness in its neighbour. The best architectures are $(2, 50)$ for 1 agent, $(1, 50)$ for 5, 15 and 25 agents and $(4, 10)$ for 10 agents. In average, over the whole population sizes and the three maps, for the map Islands the best RLPMs are worse than HPCC of 256 periods with a significant difference of 533 for 15 agents. For the map A, the RLPMs are always better than CR but worse than HPCC, and except for 25 agents where RLPM-3-50 is the best RLPM strategy, RLPM-1-50 is always the best one. However, RLPM-2-50 turns out to be the best strategy for QMI when averaging over the population sizes with a value of 481 periods. Also, in average the best RLPMs are worse than HPCC of 152 periods. Lastly, for the map *Grid*, the RLPMs are worse than HPCC, but better than CR except for 1 agent where CR is better than the RLPMs of 32 periods. The best architectures are $(2, 50)$ for 1 agent and $(1, 50)$ for 5, 10, 15 and 25 agents. As well as for the A map, in average over the population sizes, RLPM-2-50 is the best strategy and the best RLPMs are worse than HPCC

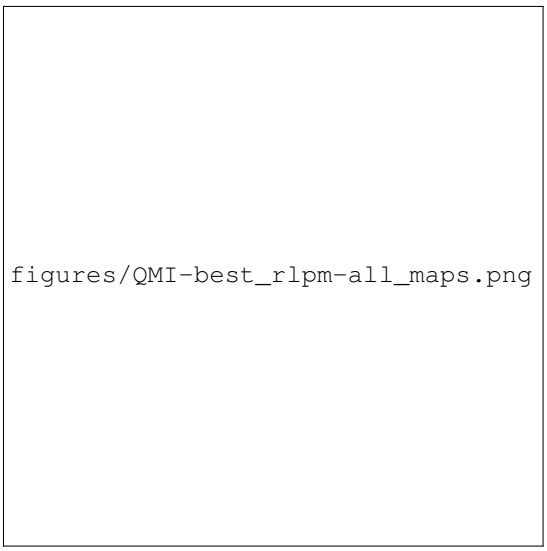


FIGURE 5 – Normalised QMI of the evaluated strategies in y-axis for the three maps and the population sizes of agents in x-axis.

of 105 periods.
Finally, the architecture $(2, 50)$ tends to be the best RLPM strategy for MI, except for the map Grid where $(1, 50)$ is slightly better, while for QMI, $(1, 50)$ is irremediably and globally the best strategy.

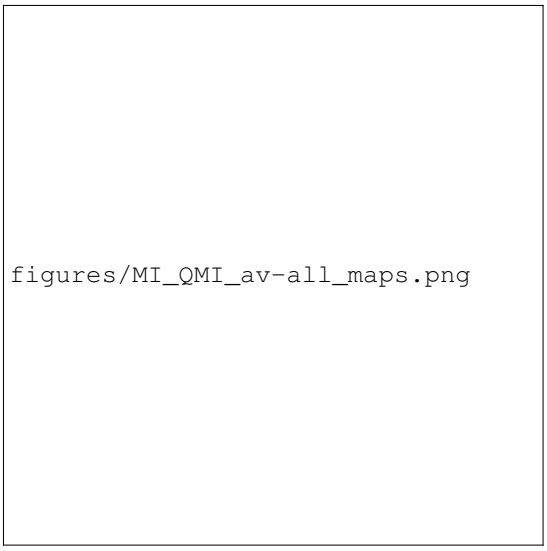


FIGURE 6 – Criterion space of MI and QMI.

The **Fig. 6** represents the criterion space for MI and QMI with the results of RLPM strategies averaged over the different numbers of agents. Here the different RLPM strategies and thus the LSTM architectures makes up the decision space. For the map Islands, it exists two Pareto optimal solutions in the decision space : the architectures $(2, 50)$ and $(1, 50)$ where the former is the best for MI with a value of 211, while the latter is the best for QMI with a value of 629. For the map A, both are also the only Pareto optimal solutions where the former is still the best for MI with a value of 234, while the latter is still the best for QMI with a value of 480. Finally for the map Grid, the architecture $(2, 50)$ is the only Pareto optimal solution with a value of $(290, 517)$. This analysis thereby tends to confirm our preliminary and foregoing assumption regarding the architecture $(2, 50)$ and $(1, 50)$ as globally the best ones pertaining to our problematic, where the former tends to be better for MI and the latter better for QMI.
As well, the architecture $(3, 50)$ is the worst strategy for the QMI criterion. QMI as quadratic mean, takes better into account the difference of time interval between the nodes and thus measures the tendency of nodes to be equitably visited through a run. Indeed, it penalises strategies that leave nodes unvisited (or which produces wide intervals between visits) during the simulation run[9]. Therefore, it provides an additional precision upon the distribution of visits over the nodes : one node with wide intervals have a little impact upon MI while it has upon QMI. Similarly, the architecture $(4, 10)$ is most of the time the worst strategy for MI. The performances over QMI of these strategies tends to show that they visit perpetually the same little set of nodes, whereupon the visits are poorly distributed over the nodes. It is likely that $(4, 10)$ presents a too small number of parameters to learn the behaviour of the HPCC startegy and, conversely, $(3, 50)$ a too large number of parameters to avoid over-fitting.

5 Conclusion and perspectives

In this paper we proposed and evaluated a new strategy for the multi-agent patrolling problem, based on the LSTM network architecture. To that end, we reminded the model underlying the multi-agent patrolling problem as well as the LSTM architecture. Then, we formally defined the new proposed LSTM-based strategy, wherefore the LSTM network was trained from the traces of a high-performance strategy. Seven architectures of LSTM were analysed in this work. Finally, we developed a new fully-fledged simulator in Python, specially designed for the multi-agent patrolling ; this simulator, that we named *Pytrol*, allowed to gather data to learn, test and evaluate the new strategies which were confronted to the reactive and cognitive standard strategies.
The evaluation demonstrated that RLPM-2-50 and RLPM-1-50, the strategies set from the LSTM architectures with 2 layers and 50 neurons, and 1 layers and 50 neurons respectively, are globally the best. RLPM-2-50 is the best upon MI - a central tendency measure - while RLPM-1-50 is the best upon QMI - measure that tends to emphasise the node with long times without visits.
These first experiments show good results as far as for each topology the proper architecture is selected. It has been showed that in an extreme situation where communications are prohibited, a learning strategy based on the

LSTM architecture can perform missions in a context of crisis with good performances, even better than the reactive and decentralised representative CR. The latter result show thereupon that a supervised-learning-based strategy with directed randomness is better than a reactive one and close to HPCC the cognitive representative for the criterion MI, although RLPM does not communicate, given that CR and HPCC are good representatives for the reactive and cognitive strategies respectively. Moreover, CR and RLPM are decentralised strategies, by design. However, RLPM was obtained by adding randomness in the decision procedure, otherwise the system being too much rigid tends to lead agents to converge indefinitely towards a small set of nodes. This entails that the learning system resting upon the LSTM architecture used here is not adaptive. A preliminary avenue to explore would be to use a new cost function to optimise, instead of the cross-entropy, to train the models in a different way, what could improve QMI by increasing the variability of the learned distribution. Also, in order to exploit the potential of the LSTM networks for the generation of paths in the multi-agent patrolling, new deeper and more complex architecture will be implemented and evaluated in the future, as well as other ANN architectures to improve the distribution of agents over the nodes and thereby QMI, but also the performances more generally. In fact, results presented here are based on LSTM predictor, but the method proposed here is generic and can be applied using any type of temporal series predictor.

Markov Decision Process (MDP), Decentralised Markov Decision Process (DEC-MDP) or Decentralised Partially Observable Markov Decision Process (DEC-POMDP) models could have been considered to model MAP. However, in a centralised perspective, that is with centralised strategies, the general MAP model used here is equivalent to a MDP model where the state corresponds to both positions of agents and global idlenesses. In that, it would be a determinist MDP. For decentralised strategies, in the considered model each agent has an overestimation of idlenesses, but there is not any probabilist distribution over them. This model can then not be regarded as a DEC-POMDP. The extension of this work to any MDP model is thereby limited by what has been stated before.

Finally, in order to bring RLPM in real life, several steps remain to be performed. First, simulation tests using a graph constructed from geographical data of an area to be patrolled shall be conducted. Then validation in field tests with actual drones will be possible.

References

[1] A. Machado, G. Ramalho, J.-D. Zucker, and A. Drogoul, "Multi-agent patrolling : An empirical analysis of alternative architectures," in *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pp. 155–170, Springer, 2002.

[2] A. Almeida, P. Castro, T. Menezes, and G. Ramalho, "Combining idleness and distance to design heuristic agents for the patrolling task," in *II Brazilian Workshop in Games and Digital Entertainment*, pp. 33–40, 2003.

[3] Y. Elmaliach, N. Agmon, and G. A. Kaminka, "Multi-robot area patrol under frequency constraints," *Annals of Mathematics and Artificial Intelligence*, vol. 57, no. 3-4, pp. 293–320, 2009.

[4] Y. Chevaleyre, "Theoretical analysis of the multi-agent patrolling problem," in *Intelligent Agent Technology, 2004.(IAT 2004). Proceedings. IEEE/WIC/ACM International Conference on*, pp. 302–308, IEEE, 2004.

[5] H. Santana, G. Ramalho, V. Corruble, and B. Ratitch, "Multi-agent patrolling with reinforcement learning," in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pp. 1122–1129, IEEE Computer Society, 2004.

[6] T. Menezes, P. Tedesco, and G. Ramalho, "Negotiator agents for the patrolling task," in *Advances in Artificial Intelligence-IBERAMIA-SBIA 2006*, pp. 48–57, Springer, 2006.

[7] Y. Guo, L. E. Parker, and R. Madhavan, "Collaborative robots for infrastructure security applications," in *Mobile robots : the evolutionary approach*, pp. 185–200, Springer, 2007.

[8] D. B. D'Ambrosio, S. Goodell, J. Lehman, S. Risi, and K. O. Stanley, "Multirobot behavior synchronization through direct neural network communication," in *International Conference on Intelligent Robotics and Applications*, pp. 603–614, Springer, 2012.

[9] G. Sampaio, G. Ramalho, and P. Tedesco, "A technique inspired by the law of gravitation for the timed multi-agent patrolling," in *22nd IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, vol. 1, pp. 113–120, 2010.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[11] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm networks," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 4, pp. 2047–2052, IEEE, 2005.

[12] M. Bennett, M. F. Schatz, H. Rockwood, and K. Wiesenfeld, "Huygens's clocks," *Proceedings of the Royal Society of London. Series A : Mathematical, Physical and Engineering Sciences*, vol. 458, no. 2019, pp. 563–579, 2002.

[13] D. O. Sales, D. Feitosa, F. S. Osório, and D. F. Wolf, "Multi-agent autonomous patrolling system using ann and fsm control," in *2012 Second Brazilian Conference on Critical Embedded Systems*, pp. 48–53, IEEE, 2012.

[14] M. Othmani-Guibourg, A. El Fallah-Seghrouchni, J.-L. Farges, and M. Potop-Butucaru, "Multi-agent patrolling in dynamic environments," in *(ICA), 2017 IEEE International Conference on Agents*, pp. 72–77, IEEE, 2017.

Interprétabilité et explicabilité pour l'apprentissage machine : entre modèles descriptifs, modèles prédictifs et modèles causaux. Une nécessaire clarification épistémologique.

C. Denis[1] F. Varenne[2]

[1] Sorbonne Université, CNRS, Laboratoire d'Informatique de Paris 6, LIP6, F-75005 Paris
[2] Université de Rouen, UFR LSH, Rue Lavoisier, 76821 Mont-Saint-Aignan, ERIAC & IHPST

christophe.denis@lip6.fr franck.varenne@univ-rouen.fr

Résumé

Le déficit d'explicabilité des techniques d'apprentissage machine (AM) pose des problèmes opérationnels, juridiques et éthiques. Un des principaux objectifs de notre projet est de fournir des explications éthiques des sorties générées par une application fondée sur de l'AM, considérée comme une boîte noire. La première étape de ce projet, présentée dans cet article, consiste à montrer que la validation de ces boîtes noires diffère épistémologiquement de celle mise en place dans le cadre d'une modélisation mathématique et causale d'un phénomène physique. La différence majeure est qu'une méthode d'AM ne prétend pas représenter une causalité entre les paramètres d'entrées, qui peuvent être de plus de haute dimensionnalité, et ceux de sortie. Nous montrons dans cet article l'intérêt de mettre en œuvre les distinctions épistémologiques entre les différentes fonctions épistémiques d'un modèle, d'une part, et entre la fonction épistémique et l'usage d'un modèle, d'autre part. Enfin, la dernière partie de cet article présente nos travaux en cours sur l'évaluation d'une explication, qui peut être plus persuasive qu'informative, ce qui peut ainsi causer des problèmes d'ordre éthique.

Mots Clés

IA, apprentissage machine, interprétabilité, explicabilité, causalité, modèles descriptifs, modèles prédictifs, modèles causaux, épistémologie

Abstract

The lack of validation and of explainability of some Machine Learning (ML) models involves operational, legal and ethical issues. One of the main objectives of our research project is to provide ethical explanations of the outputs produced by a ML based application, considered as a black box. The first step of this project, presented in this article, is to underline the epistemic differences between the validation of an ML model and of a causal mathematical model. ML is based on statistical correlations between input - which could have a high dimension - and output parameters without building causality links between them, unlike most mathematical models in science and engineering. This absence of causality is the major drawback of ML, making difficult the validation and the explanation of some ML methods, generally the most efficient ones. Our scientific contribution is to highlight, in this context, the epistemic distinctions between the different functions of a model, on the one hand and between the function and the use of a model, required to build explanation. Our current work in the evaluation of the quality of an explanation, which could be more persuasive than informative and consequently generates ethical problems, is reported in the last part of the article.

Keywords

AI, machine learning, interpretability, explainability, causality, descriptive modeling, predictive modeling, causal modeling, epistemology

1 Contexte et motivations

Depuis 2010, l'Intelligence Artificielle (IA) numérique ou connexionniste fondée sur de l'apprentissage machine (AM) produit des résultats impressionnants, principalement dans les domaines de la reconnaissance de forme, du traitement naturel du langage et de la perception, succédant à la domination de l'IA symbolique centrée sur le raisonnement logique. Ces succès ont entraîné un engouement médiatique : on parle souvent d'un *renouveau de l'IA*, voire de l'apparition d'une nouvelle forme d'IA. Il s'agit plutôt de la fin de l'hibernation de l'IA connexionniste, dont l'origine remonte à la critique du perceptron par Minsky en 1969. Elle s'explique par plusieurs phénomènes concomitants : augmentation de la puissance de calcul, production et traitement performant d'un volume de plus en plus important de données et algorithmes efficaces de pondération des réseaux de neurones profonds. Une analyse pertinente du rapport entre IA symbolique et IA connexionniste a été proposée dans [3] : *« alors que les concepteurs des machines symboliques cherchaient à insérer dans le calculateur et le monde et l'horizon, la réussite actuelle des machines connexionnistes tient au fait que de façon presque opposée, ceux qui les fabriquent vident le calculateur pour que le monde se donne à lui-même son propre horizon »*. L'IA bouleverse des pans entiers de la société humaine comme l'économie, le salariat, la justice et la médecine.

Les résultats, souvent spectaculaires, de l'AM suscitent à la fois de forts espoirs, des craintes légitimes, notamment en termes d'éthique et de transformation du travail [4], et véhiculent un certain nombre de fantasmes [6]. Le déficit de

transparence de ces méthodes d'apprentissage, notamment signalé dans le rapport [25], permet de le considérer, pour certains domaines, comme un colosse au pied d'argile. L'industrialisation de démonstrateurs développés dans des laboratoires n'est que peu souvent au rendez-vous, comme le souligne [8]. L'acceptabilité opérationnelle de telles applications est largement conditionnée par la capacité des ingénieurs et décideurs à comprendre le sens et les propriétés des résultats produits par ces outils. On se heurte au manque de compréhension de leurs mécanismes de décision ou d'aide à la décision. De plus, la délégation décisionnelle croissante proposée par les outils d'IA rivalise avec des règles métier éprouvées, en nombre limité, constituant parfois des systèmes experts certifiés. L'apprentissage machine est comparé dans [20] à une forme d'alchimie et le problème de sa transparence est considéré comme un défi scientifique majeur dans [25].

La première difficulté rencontrée est la polysémie du mot transparence. La transparence d'un algorithme peut faire référence à deux types de propriétés selon [19] : extrinsèques, comme par exemple la loyauté et l'équité, ou intrinsèques, comme l'interprétabilité et l'explicabilité. En nous installant en amont du choix conceptuel de [15] selon qui *"interpretability is the degree to which a human can understand the cause of a decision"* ([15], p.10), nous entendrons ici par interprétabilité pour un sujet humain la capacité d'une représentation à se voir composée d'éléments (signes, figures concepts, données, etc.) qui ont un sens pour le sujet en question. L'explicabilité dénotera ici soit l'explicabilité de l'algorithme de l'AM, soit celle des sorties de l'AM, c'est-à-dire la capacité de déploiement et d'explicitation de cet algorithme ou de ses sorties en séries d'étapes reliées entre elles par ce qu'un être humain peut interpréter sensément comme des causes ou des raisons. Nous nous intéressons plus précisément dans cet article au problème de l'explicabilité des sorties produites par l'AM. De telles sorties peuvent être en particulier des suggestions de décisions, de prédictions ou d'actions [19]. Il est à noter que le déficit d'explicabilité se retrouve au-delà de l'AM : par exemple dans le cadre d'une modélisation mathématique d'un phénomène physique mal connu (incertitudes épistémiques) ou d'un système expert possédant un nombre important de règles.

Notre projet de recherche consiste à construire des explications de méthodes d'AM que nous considérons ici comme une boîte noire. Dans ce contexte plus précis, nous définissons l'explicabilité comme la capacité à fournir pour un ensemble d'utilisateurs une explication des résultats obtenus par l'AM adaptée à leurs connaissances scientifiques et métier. Les évaluations humaines d'une explication introduisent un biais cognitif envers les raisonnements les plus simples [9]. Il est donc nécessaire sur le plan éthique d'évaluer le meilleur compromis entre explication intelligible et explication persuasive. La première étape de ce projet, celle qui est principalement présentée dans cet article, consiste à montrer que le processus d'explicabilité de cette boîte noire diffère épistémologiquement de celui mis en place dans le cadre de la modélisation mathématique et causale d'un phénomène physique. La différence majeure est qu'une méthode d'AM ne prétend pas représenter une causalité entre les paramètres d'entrées et ceux de sortie, cela, malgré le recours aux termes trompeurs, issus de la théorie statistique, de variables dites « explicatives ». Nous soutenons que c'est en grande partie cette absence de représentation d'une causalité qui est à l'origine des trois points de fragilité de l'apprentissage machine déjà signalés et étudiés dans la littérature :

1. l'interprétabilité du processus computationnel - ou de ses éléments - n'assure pas à elle seule son explicabilité;
2. la conception d'un modèle d'AM nécessite un prétraitement et une sélection des données. Quand ces données ne sont pas reliées à un scénario causal explicite, cela a pour effet de masquer les choix ontologiques qui accompagnent inévitablement les choix de formats, de données ou de leurs prétraitements ;
3. enfin, l'explication, quand elle est possible, n'est pas pour autant assurée d'être dépourvue de biais et peut se révéler être un dispositif servant davantage un usage rhétorique qu'une fonction épistémique objective, à savoir un usage pour la persuasion et la mise en confiance des utilisateurs [9]. Nous montrerons ici l'intérêt de mettre en œuvre la distinction épistémologique entre fonction et usage d'un modèle [23], [24] ;

2 Contribution et organisation de l'article

Notre contribution s'organise de la manière suivante :

— la section 3 présente tout d'abord la différence épistémique, en termes d'usage général des données, entre une prédiction d'un phénomène fondée sur sa modélisation mathématique (modèle hypothético-déductif) et une prédiction fondée sur un apprentissage machine (modèle inductif). Nous rappelons qu'une application fondée sur de l'AM n'utilise pas les données de la même manière que l'approche hypothético-déductive. Les deux ne se nourrissent donc pas des données de manière équivalente ou indifférente : des contraintes ontologiques spécifiques en termes de format et de traitement de données interviennent déjà lors des choix de conception.
— la section 4 caractérise ensuite les différentes fonctions de connaissance ou fonctions épistémiques d'un modèle. Nous nous interrogeons sur les critères qui permettent de décider si et quand un processus de modélisation relève d'une explication causale. Nous rappellerons succinctement les liens que la philosophie contemporaine des sciences voit entre explication causale et mécanisme. Nous y montrons comment cette classification peut être adaptée aux applications fondées sur l'AM.
— la section 5 marque les similitudes et les différences entre l'explication causale *par* un modèle d'AM et l'explication causale *d'un* modèle d'AM. Cela nous permettra de distinguer plus clairement trois

grands types d'explicabilité dans ce contexte : tournée vers le système cible, tournée vers le concepteur, tournée vers l'utilisateur. Comme suite à ces distinctions, les rapports entre différents types d'interprétabilité et d'explicabilité pourront être élucidés.

— la section 6 introduit la différence entre fonction (épistémique) et usage (pratique ou rhétorique) d'un modèle. Couramment, la demande d'explicabilité mélange ces deux notions. Nous montrerons qu'il n'est certes pas possible de séparer complètement dans les faits fonction et usage (comme il n'est pas possible de séparer complètement l'aspect *ethos* - confiance - et l'aspect *logos* - rationnel - d'un discours persuasif), mais qu'il peut être nécessaire de savoir les séparer conceptuellement, c'est-à-dire de savoir exprimer la différence entre les types de savoirs qui les fondent pour se rendre capable de distinguer et de clarifier les sources de fragilité de l'AM.

La synthèse de cet article et la suite de notre projet de recherche sont présentées en dernière partie.

3 Conception d'une application fondée sur de l'AM

Nous présentons tout d'abord le rôle différent joué par les données dans une prédiction fondée tantôt sur de l'AM, tantôt sur une formulation mathématique explicite du phénomène à prédire. Bien que les données jouent un rôle majeur dans la conception d'une application procédant par AM, des contraintes ontologiques en termes de format et de traitement, engendrées elles-mêmes par des choix de conception, influencent le comportement et l'explication de cette application.

3.1 Rôle des données dans le cadre d'une machine hypothético-déductive et d'une machine inductive

Prenons l'exemple d'un fluide dont on souhaite estimer la vitesse d'écoulement dans un canal. Deux options sont possibles pour obtenir la prédiction de la vitesse :

— utiliser *une machine hypothético-déductive* : l'écoulement est modélisé à l'aide d'équations mathématiques, par exemple en utilisant les équations de Navier-Stokes. La discrétisation des ces équations sur ordinateur conduit à un programme. Les données servent seulement à instancier un problème déjà résolu par le programme sur la machine hypothético-déductive. Dans une démarche de validation et de quantification d'incertitudes, d'autres données, essentiellement des mesures, sont utilisées pour caler et pour valider le programme [17].

— utiliser une *machine inductive* : le programme utilisé pour prédire la vitesse de l'écoulement peut résulter d'un AM prenant en entrée une base de données. Ici, les données servent à la fois à construire et à valider le programme. Cependant, la finalité d'un modèle inductif n'est pas uniquement la prédiction mais aussi l'explication du phénomène. Actuellement, cela n'est pas toujours le cas puisque certaines techniques d'AM, généralement les plus performantes au niveau statistique, souffrent d'un déficit de capacité à expliquer leur système cible comme aussi d'un déficit d'explicabilité.

3.2 Etapes de conception d'une application d'AM

Nous nous plaçons sans perte de généralité dans le cadre de l'AM supervisé. La conception d'un modèle d'apprentissage n'est pas un processus automatique. Elle ne se résume pas au choix de la méthode d'apprentissage (régression linéaire, arbre de décision, réseau de neurones, etc.). La figure 1 présente les principales étapes intervenant lors de la conception d'une application d'AM. En particulier, le prétraitement, le choix et la combinaison des variables dites explicatives sont la source de choix ontologiques résultant de ces manipulations. Il est à juste titre indiqué dans [6] « *qu'il y a une connaissance implicite cachée derrière la formulation des données que l'on utilise, autrement dit, des dogmes que l'on entre, par-devers soi, dans les machines* ».

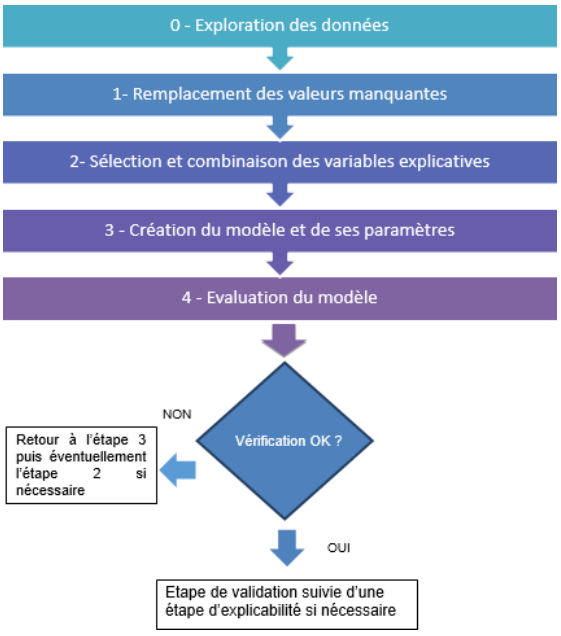


FIGURE 1 – Principales étapes intervenant lors de la conception d'une application d'AM

La conception comprend une phase de vérification et de validation [5] :

— *étape de vérification* : s'assurer que le modèle d'AM permet d'obtenir la précision statistique souhaitée et possède de bonnes capacités de généralisation ;

— *étape de validation* : justifier le choix du modèle d'AM en particulier lors de la sélection ou de la combinaison de variables explicatives.

4 Fonctions principales d'un modèle : réduction de données, description, prédiction, explication

4.1 Sur la fin programmée des modèles face au déluge des données

L'argumentation portant sur la fin des modèles et des hypothèses théoriques exprimée dans [1] a entraîné des débats sur le rôle de la démarche scientifique dans un monde marqué par une production effrénée de données, d'une part, et par une augmentation continue de la puissance de calcul, d'autre part. Le traitement petaflopique, bientôt exaflopique voire quantique d'un déluge de données rendrait obsolète la modélisation ou la théorie scientifique : *"Petabytes allow us to say: 'Correlation is enough.' We can stop looking for models. We can analyze the data without hypotheses about what it might show. We can throw the numbers into the biggest computing clusters the world has ever seen and let statistical algorithms find patterns where science cannot.* [1]*"*. Indéniablement, le traitement massif de données permet d'obtenir des avancées remarquables dans de nombreux domaines, notamment en santé. Un enjeu par exemple est la prévention des interactions médicamenteuses dangereuses pour des patients atteints de maladies complexes ou concomitantes. Les essais cliniques, en nombre relativement restreint, ne permettent pas de prédire toutes les interactions en raison d'une combinatoire élevée. L'utilisation d'une technique d'AM supervisé recourant à un nombre important de données pharmacogénomiques et de populations de patients a permis d'améliorer considérablement la prévention des effets indésirables en polypharmacie [26].

L'argumentation de [1], également reprise par d'autres auteurs, s'inscrit dans le courant empiriste de la pensée scientifique moderne. En 1620, Francis Bacon argumente dans [2] que la démarche scientifique ne doit pas être fondée prioritairement sur des hypothèses (modèle déductif dominant la science depuis Aristote) mais sur des données expérimentales (modèle inductif). Or, l'explicabilité de la prédiction reste malgré tout une fonctionnalité importante du modèle inductif. Sinon, la performance statistique de la prédiction sans explicabilité conduit à une forme de sophisme pragmatique : est jugée abusivement réaliste ou conforme au réel une représentation qui permet seulement - pragmatiquement - de le prédire [22]. À bien y regarder, le traitement performant statistique d'un important volume de données n'est pas l'annonce de la fin des modèles mais l'annonce d'une utilisation accrue d'autres types de modèles et pour lesquels un travail épistémologique affiné s'impose plus que jamais.

4.2 Caractérisation d'un modèle et des fonctions d'un modèle

Dans son caractère le plus général, un modèle peut être défini comme un objet médiateur auquel on adresse une question au sujet d'un objet cible qu'on ne peut interroger directement : « *pour un observateur B, un objet A* est un modèle d'un objet A dans la mesure où B peut utiliser A* pour répondre à des questions qui l'intéressent au sujet de A* » [14].

La prédiction et l'explication d'un phénomène sont des fonctions de connaissance parmi d'autres. Parmi, les nombreuses fonctions de connaissance que peut faciliter la médiation d'un modèle, il est possible d'en identifier et d'en classer une vingtaine [23], [24]. Les plus fréquentes sont l'analyse ou la réduction de données, la description, la prédiction et l'explication ([23], [24] : fonctions 5, 6, 7 et 8). La figure 2 évoque l'utilisation d'un modèle mathématique causal simulé numériquement pour expliquer et prédire un phénomène physique.

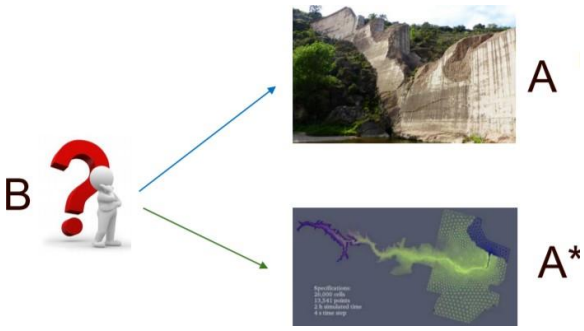


FIGURE 2 – Cas d'un modèle à la fois explicatif et prédictif

Un modèle d'analyse de données ne décrit pas encore des structures propres au système cible mais seulement des structures entre des signaux auxquels il donne lieu. Les modèles de réduction de données structurent, élaguent ou classent les données sans permettre encore de description ni d'interprétation de ces classifications en termes d'ontologies interprétables et valant pour le système cible de manière signifiante. Ils servent de représentation intermédiaire de la seule structure informationnelle des données du système cible, mais pas directement de la structure des propriétés intrinsèques du système cible ni des relations mutuelles entre ces propriétés : ils traitent les données comme des *signaux* non comme des *signes*. Un signal indique, qualifie ou quantifie une interaction. Un signe désigne, qualifie ou quantifie une propriété. Un signal est le résultat de la détection ou de la mesure par capteur physique d'un phénomène d'interaction entre l'objet cible et son environnement physique (qui est au minimum son cadre spatial, temporel ou spatio-temporel). Dans l'approche « signal », les propriétés physiques intrinsèques de l'objet cible sont certes supposées mais leur nature peut demeurer largement inconnue, alors que l'approche « signe » entend rendre compte d'une propriété du système cible et de sa valeur, en recourant à cet autre type de médiateurs que sont les instruments de mesure. Les modèles de réduction de données sont donc faiblement prescriptifs ontologiquement. Ils préparent l'utilisation d'autres modèles : les modèles à fonction de description ou d'explication du système cible.

Tout en dépassant déjà la considération superficielle de la seule structurelle informationnelle des données, les modèles

descriptifs et prédictifs sont toutefois nommés encore « phénoménologiques » ([23], [24] ; fonction 7) : ils facilitent la reproduction ou production de structures de données dont l'apparence (la phénoménologie) est jugée fidèle à certaines structures des propriétés observables du système cible. Ils réalisent cette production par des moyens intelligibles, déductifs ou calculatoires. Un modèle descriptif structure des données qui, séparément, ont déjà un sens minimal, c'est-à-dire qui sont interprétables en termes de propriétés au regard de la connaissance minimale que l'on a, par ailleurs, du système cible. En revanche, la structure que le modèle descriptif propose pour ces propriétés (leur relation mutuelle représentée dans le modèle) peut être complètement phénoménologique, c'est-à-dire ne rien signifier de robuste, ne renvoyer à rien de réel, *i.e.* ne pas se fonder elle-même sur une propriété profonde de structure du système cible.

Un modèle prédictif est un cas particulier de modèle descriptif. Il décrit le système à travers deux types minimaux de données qui le représentent (le décrivent) partiellement sans pour autant encore l'expliquer : les données prédictives - qui servent dans l'algorithme ou le modèle - et les donnés comportementales ou prédictives qui servent à évaluer la qualité de la prédiction, donc la qualité du modèle. Un modèle descriptif peut en effet être statique ou dynamique. Une dynamique au sens large (*i.e.* permettant de distinguer des conditions initiales et un état final, ou des variables d'entrée et des variables de sortie, ou encore des variables prédictives et des variables prédites) peut être reproduite de manière elle aussi purement descriptive, *i.e.* sans qu'une séquence temporelle soit représentée de manière ontologiquement significative (explicative par exemple) pour les séquences d'états du système cible. Quand cette dynamique permet non seulement de décrire correctement le comportement observable du système dans les cas connus d'entrées/sorties mais aussi d'interpoler ou d'extrapoler correctement une description de son comportement observable à partir de données qui n'ont pas été utilisées pour calibrer le modèle (données nouvelles, période de temps non encore testée), le modèle descriptif (à AM, de régression, de classification, etc.) se trouve être également ce qu'on appelle un modèle prédictif.

Il y a deux grands types de modèles prédictifs : à régression au sens large (dès lors qu'ils servent à prédire des variables quantitatives), et de classification, ces derniers servant à prédire une variable qualitative ou, plus largement, à estimer la probabilité d'un événement. Dans les modèles prédictifs de classification, il est utile de distinguer les modèles discriminatifs qui ne font pas d'hypothèse *a priori* sur la distribution (régression logistique, perceptron, SVM), et les modèles génératifs se fondant sur l'hypothèse de l'existence d'une forme paramétrique précise pour une distribution sous-jacente [21]. Ces derniers permettent l'adoption d'une approche bayésienne qui, si les *priors* sont acceptés et se révèlent féconds à l'usage, peuvent nous mener à l'idée que le modèle est explicable. Il n'en reste pas moins que le fondement de l'explicabilité de tels modèles reste épistémique car n'entendant pas reposer sur une hypothèse ontologique de lois de la nature ou de causalité.

En philosophie des sciences contemporaine, il n'existe pas de consensus sur la différence précise entre expliquer et comprendre. Une grande partie des auteurs s'accorde cependant sur le fait d'associer l'explication à la causalité et la compréhension à l'unification d'une diversité de phénomènes sous un principe unique ([16] p. 18). En se fondant sur cette idée toujours discutable mais également fréquemment acceptée qu'une explication réfère à une causalité (nous n'entendrons pas ici causalité au sens où l'entend la pratique de l'inférence causale, même si la sophistication récente de sa stratégie de formalisation des propositions contrefactuelles au moyen d'une approche structurelle se révèle pragmatiquement efficace : [18]), on peut dire qu'un modèle mathématique ou algorithmique est explicatif d'un objet cible lorsque :

1. il est au moins partiellement prédictif pour ce système ;
2. il offre une représentation interprétable, c'est-à-dire signifiante et accessible à un esprit humain non aidé, des éléments dont il est composé et des processus élémentaires d'interaction qu'il met en œuvre ;
3. ces éléments et processus élémentaires sont supposés eux-mêmes représenter plus ou moins iconiquement des éléments et des processus d'interaction causale (ou mécanismes) intervenant réellement et majoritairement dans le système cible lui- même.

Ainsi, la modélisation mathématique causale d'un phénomène physique repose sur un ensemble de causes X reconnues par les physiciens pour être réellement et majoritairement à l'origine du phénomène physique Y. Plus précisément, les équations mathématiques dérivent d'un modèle mécaniste théorique obtenu à partir des lois théoriques hypothétiques de la physique. Le modèle mécaniste théorique est alors un ensemble de relations structurelles, causales, entre des variables X décrivant le lien entre Y et X. C'est le cas de la grande majorité des modèles théoriques utilisées pour représenter les phénomènes. Le phénomène Y n'est certes pas entièrement explicable par l'ensemble des causes X puisque, d'une part, il existe des incertitudes (aléatoires ou épistémiques) sur l'évaluation des causes X et que, d'autre part, certaines causes peuvent être inconnues dans l'état actuel de la connaissance scientifique. La procédure de V&V d'un code de calcul fondé sur une modélisation physique d'un phénomène a pour vocation de répondre successivement aux deux questions suivantes : 1) étape de vérification : est-ce que le programme informatique résout correctement les équations mathématiques choisies ? 2) étape de validation : est-ce que l'on a choisi les bonnes équations et les bons paramètres d'entrée ? On voit ici que la validation dépend étroitement de la fonction épistémique attendue du modèle. Un modèle explicatif ne devra pas être seulement validé dans sa capacité à reproduire certains comportements du système cible. Il faudra aussi évaluer sa capacité à représenter pas à pas, de manière correcte, *i.e.* approximativement réaliste, non seulement les états successifs du système cible mais aussi chaque étape de calcul, chaque opération du processus lui-même.

5 Explication causale, interprétabilité et explicabilité en apprentissage machine

Il y a plusieurs types d'explication qui peuvent intervenir dans l'évaluation d'un modèle. On doit distinguer d'abord l'explication du système cible par le modèle de l'explication du modèle lui-même et de son fonctionnement. C'est cette seconde explication qui est l'enjeu de cet article. Mais il y a des liens possibles entre les deux. D'abord, il peut exister un mécanisme causal réellement existant et affectant le système cible. Ce mécanisme peut reposer sur des lois connues, en être la déduction, le résultat calculatoire : par exemple une interaction locale entre deux astres repose sur les lois de Newton, une interaction entre deux atomes en chimie reposent sur l'équation de Schrödinger, etc. On peut alors modéliser le système cible en modélisant de manière fidèle la causalité même affectant ce système cible. On le fait en représentant de manière iconique (*i.e.* au moins termes à termes) les principaux éléments en interaction et leurs principales interactions : par là, le modèle est fidèle au moins à l'individuation des éléments naturels réels comme une planète, un atome, même s'il y a une part d'idéalisation dans leur représentation individuelle comme celle qui consiste à supposer que la masse de la planète est entièrement située sur le point géométrique centre de gravité de la planète. Dans ce type de modèle, la première forme d'explication intervient au sens d'abord où c'est un modèle expliquant le système cible : il est explicatif (voir plus haut). C'est-à-dire qu'en même temps qu'il effectue ses computations, il explicite, il rend visible pas à pas le processus qui affecte le système cible de manière assez fidèle et suffisamment réaliste au vu des connaissances que nous avons par ailleurs de ses éléments, de leurs propriétés, des lois de la nature qui les affectent et des mécanismes d'interaction que ces lois déterminent (loi de la physique, de la chimie, voire de la biologie). Mais, de manière similaire, dans le cas d'un système expert modélisant une décision médicale, par exemple, les bases de données et les règles de raisonnement sensées et appliquées pas à pas dans le modèle expliquent le processus même de la décision. Notons que, dans le cas de la décision humaine motivée, on peut considérer qu'une raison joue le même rôle qu'une cause dans un système physique soumis à des lois physiques.

Dans tous ces cas favorables, une des conséquences est que le modèle est non seulement explicatif mais aussi explicable. Cela veut dire que le processus de computation suivi par le modèle implémenté dans le programme est également interprétable et explicable en lui-même. Il est interprétable car l'ontologie du modèle (ses représentations) renvoie à des ensembles d'entités et de propriétés reconnues comme existant réellement dans le système cible auquel on a accès par ailleurs sous une forme interprétable. Dans ce cas de modèle explicable, on utilise donc la connaissance préalable que l'on a 1) de la structuration réelle du système cible (l'ontologie qu'on lui reconnaît), 2) du fait que le modèle utilise cette structuration et n'utilise qu'elle dans ses processus, 3) du fait que les processus du modèle sont également supposés réalistes, 4) du fait que ce dépliement processuel pas à pas converge mathématiquement (théorème de convergence) vers les résultats, pour, au final, décider que le modèle non seulement explique son système cible mais qu'il est également interprétable et explicable en lui-même.

Dans ce cas favorable d'un modèle expliquant son système cible (explication *par* le modèle), c'est par l'effet d'une transitivité de la représentation de l'ontologie, des structures et des processus, que l'on peut également conclure à une explicabilité du modèle lui-même (explication *du* modèle). Cette explicabilité du modèle est ici assurée et légitimée par notre connaissance des lois qui affectent réellement le système cible. On peut prendre l'exemple d'une simulation numérique en mécanique des structures ou en mécanique des fluides. Quand un tel modèle est validé, même s'il est traité numériquement, il reste à la fois explicatif, interprétable et explicable. Pour un modèle de décision experte à base de règles motivées et une à une significatives au regard de règles métier, l'explicabilité du modèle est également assurée du fait de la représentation de cette causalité généralisée (raisonnement symbolique significatif) dans le modèle lui-même.

À première vue, on pourrait se dire qu'on peut adapter ce processus de validation aux modèles d'AM puisqu'on peut considérer que leur nature est également mathématique et numérique. En effet, il existe bien des équations mathématiques pour concevoir un réseau de neurones comme par exemple l'algorithme de rétropropagation du gradient servant à déterminer les pondérations. De ce point de vue, une validation de l'algorithme, c'est-à-dire la preuve qu'il assure bien la fonction de prédiction désirée, entraînerait une confiance sur le calcul des poids du réseau et *de facto* sur le réseau de neurones. Mais, dans le cas de l'AM, à la différence des cas précédents, l'explicabilité du modèle n'est pas aussi facile à assurer car elle ne peut pas être directement héritée du fait que le modèle serait explicatif. L'explicabilité du modèle que l'on recherche doit être fondée autrement pour deux raisons. Premièrement, comme pour un modèle d'analyse de données standard, en AM, le modèle contrôlant les relations entrées/sorties n'entend pas représenter, même de manière seulement stylisée, un scénario causal d'interaction pas à pas opérant sous l'effet de lois ou de règles motivées. Le modèle se fonde sur l'analyse de corrélations entre les paramètres d'entrée. Or, une corrélation statistique entre deux paramètres ne signifie pas qu'il existe une causalité entre eux. Deuxièmement, la situation de l'AM est pire encore que celle des modèles classiques d'analyse de données : car le modélisateur ne cherche même pas à ce que les conditions minimales d'exercice d'un hypothétique modèle explicatif soient réunies. L'ontologie sous-jacente aux données et à leur structure peut en effet être complètement inconnue ou fictionnelle. On ne connaît pas d'ontologie robuste et objective du domaine qui soit explicitement prescrite et sur laquelle pourrait éventuellement s'exercer un ensemble de mécanismes déterminés par des lois. Ainsi, on ne peut pas s'appuyer d'emblée sur une reconnaissance préalable, ne serait-ce que descriptive, de la structure interne des données (car les données sont dites mal structurées ou bien leur structure significative - s'il en est une - nous est inaccessible). Enfin, quand bien même une structure serait

perceptible dans les données, une technique comme les RN par exemple met en œuvre des modèles non linéaires reliant les valeurs prédictives et les valeurs prédites. Les valeurs prédictives interagissent fortement : donc on ne peut plus parler de simples corrélations. Dans le cas d'un modèle non linéaire à arbres de décision, les étapes élémentaires restent certes interprétables une à une, mais le processus d'ensemble n'est pas pour autant aisément ni sensément résumable : il n'est pas compréhensible.

En analyse des données classique, toutefois, le modèle d'analyse repose sur des hypothèses globales et minimales - qu'on peut dire métaphysiques - de symétries temporelles ou spatiales liées à l'environnement de captation des données. C'est ce genre d'hypothèse qui autorise l'approche par traitement de signal, très fréquente en ingénierie : analyse linéaire, analyse de Fourier, transformée en Z, analyses non paramétriques, etc. Mais ces hypothèses métaphysiques minimalistes ne sont même pas toujours possibles en AM. Le rapprochement récent entre l'analyse par ondelettes et les réseaux de neurones convolutionnels [12] ne fait que confirmer ce soupçon qu'un RN quelconque (non convolutionnel) est en général plus neutre encore et moins-disant d'un point de vue métaphysique et causal que les approches par analyse de données paramétriques ou non paramétriques. Ainsi, les modèles à AM ne peuvent pas hériter directement leur interprétabilité et leur explicabilité du caractère réaliste et causal des interactions qu'ils modélisent dans leur calcul. Car, ils sont a priori dépourvus d'un tel ancrage réaliste et causaliste. Ce défaut fragilise les pratiques de vérification, de validation, mais aussi de diffusion et d'appropriation par les utilisateurs, d'où la demande d'interprétabilité et d'explicabilité de ces modèles. Au vu des distinctions faites précédemment, remarquons que la demande d'interprétabilité d'un modèle d'AM revient finalement à demander la construction d'un modèle descriptif de ce modèle d'AM. La demande d'explicabilité d'un modèle d'AM, quant à elle, revient à demander d'en construire un modèle explicatif.

Remarquons enfin que dans la demande d'explicabilité, il y entre souvent en même temps la demande de compréhension du modèle. On recherche alors des grands principes unificateurs permettant de penser et représenter de manière unitaire le fonctionnement global, la logique globale, du fonctionnement du modèle. La légitimation et l'acceptabilité du modèle va souvent de pair avec sa compréhensibilité. On cherche alors à construire un modèle de compréhension du modèle d'AM (fonction 9 des modèles selon [23], [24]). Plus encore que l'interprétabilité, la compréhensibilité est très sensible aux compétences de la personne à laquelle elle s'adresse. Aristote avait souligné que la rhétorique existe pour mettre en confiance et persuader les personnes qui ne peuvent suivre de manière attentive de longues chaînes de raisonnement : une explication pas à pas, même si elle est disponible et même si elle est causale, ne leur suffit pas ; il faut alors que les experts pratiquent la rhétorique et leur fournissent une représentation prenant la forme d'un grand mouvement simple et uniforme de pensée supportable par un esprit humain non aidé. Pour satisfaire les demandes d'interprétabilité et d'explicabilité d'un modèle d'AM, on cherche ainsi souvent à en construire un modèle second qui recourt à une rematématisation, une factorisation ou tout autre simplification de ses multiples couches humainement inextricables de représentations, d'une part, de computations, d'autre part [7]. Par là, on cherche à simplifier et remodéliser de manière humainement maniable (voir fonction 9 dans [23]) un comportement de modèle sinon inextricable. Cette simplification est donc une modélisation de second degré, un modèle de modèle d'AM. Cette modélisation peut ensuite elle-même s'accompagner de différents usages. Elle peut en effet être recherchée pour une vérification, une validation ou servir encore à un dispositif explicite de persuasion - plus ou moins biaisé - à destination des utilisateurs.

6 Fonction épistémique et usage du modèle en apprentissage machine

Comme présenté en section 3, la conception comprend des choix techniques le plus souvent décidés empiriquement et qui déterminent eux-mêmes des choix ontologiques. L'objectif de l'étape de validation consiste à justifier les choix effectués lors de la conception du modèle d'apprentissage, y compris au regard de la fonction (pour l'AM, le plus souvent, une fonction de prédiction). C'est là ce que nous appelions le troisième point de fragilité de l'AM : la conception d'un modèle d'apprentissage nécessite toujours un formatage, une sélection puis un prétraitement des données. Comme ces données ne sont pas reliées d'entrée de jeu à une ontologie explicite ni à un scénario causal explicite (voir sections précédentes) mais que l'on peut continuer à dire qu'on en propose une sorte d'interprétabilité puis une sorte d'explicabilité pour le processus qui les traite ensuite, cette interprétabilité et cette explicabilité proposées peuvent avoir pour effet de masquer davantage encore les choix de format et de représentation qui structurent implicitement les données initiales et leurs prétraitements. Le problème peut venir de la confusion suivante : ce n'est pas parce qu'on a réussi à modéliser de manière explicative ou compréhensive un modèle par ailleurs purement prédictif que l'on a rendu ce modèle explicatif. On a pu expliquer le fonctionnement interne de ce modèle prédictif mais pas le fonctionnement du système cible initial. On n'a pas non plus davantage confirmé le caractère réaliste de l'ontologie de ce modèle prédictif. Ainsi, les structures implicites de données peuvent être à l'œuvre sans que l'on sache dans quelle mesure ni à quel niveau c'est le cas, même quand le modèle remplit son office. Le succès d'un modèle de prédiction peut éventuellement être une indication que les éléments qu'ils postulent explicitement pour effectuer ses calculs (par exemple : des neurones très simplifiés) reflètent finalement quelque chose qui serait réellement à l'œuvre dans le système cible : c'est ainsi ce qui fonde l'opinion biomimétiste de Yann Le Cun. Mais le théorème d'universalité concernant les RN peut aussi nous engager plutôt à penser qu'il s'agit simplement d'une autre forme, parmi d'autres, d'automate universel de calcul, simplement plus commode à utiliser en pratique pour certaines formes de données et de questions qu'on leur adresse. Un usage normatif et prescriptif des modèles de prédiction en AM peut ainsi s'exercer non seulement du fait du caractère prescriptif du choix de modélisation lui-même, pour peu

qu'il s'accompagne d'une interprétabilité jugée acceptable parce que suffisamment performante et compréhensible par les utilisateurs, mais aussi du fait que cette interprétabilité peut masquer la non explicitation des choix ontologiques demeurés latents dans les données d'apprentissage.

7 Évaluation de la qualité d'une explication

Il existe une taxonomie de méthodes pour produire des explications sur les résultats produits par de l'AM [7]. La forme de l'explication doit être évaluée et choisie pour minimiser les biais cognitifs de l'utilisateur, comme indiqué dans [13] : *"The motivations and benefits of different types of transparency can vary significantly depending on context, and objective criteria are difficult to identify."*

L'explication doit en particulier permettre :

- pour un développeur, de comprendre le fonctionnement de l'application afin de la déboguer ou de l'améliorer ;
- pour un utilisateur, de comprendre le périmètre d'utilisation et les hypothèses sous-jacentes donnant des clés de lecture des résultats obtenus ;
- pour un expert, de statuer sur un audit lors d'un incident.

Il existe en outre un problème éthique du fait de la possible dérive consistant à produire des explications davantage persuasives que transparentes. Nous avons commencé un travail de recherche visant à rendre possible la mesure de la qualité d'une explication. Au vu de nos analyses préalables, il nous apparaît désormais clairement que cette mesure devra se faire en fonction de l'usage et du destinataire de l'explication. Pour cela, nous nous inspirerons notamment de certains travaux de psychologie cognitive [11]. Un protocole d'évaluation qualitative d'explications d'une application basée sur de l'AM sera alors défini et testé sur un panel d'utilisateurs.

8 Conclusion et perspectives

Le déficit d'explicabilité des techniques d'apprentissage machine profond pose des problèmes d'ordre opérationnel, juridique et éthique. Notre projet de recherche a pour objectif de fournir et évaluer des explications de méthodes d'apprentissage machine considérées comme des boîte noires. La première étape de ce projet, celle qui est présentée dans cet article, consiste à montrer que la validation de cette boîte noire diffère épistémologiquement de celle mise en place dans le cadre de la modélisation mathématique et causale d'un phénomène physique. La différence majeure est qu'une méthode d'apprentissage machine ne prétend pas représenter une causalité entre les paramètres d'entrées et ceux de sortie, cela, malgré le recours aux termes trompeurs, issus de la théorie statistique, de variables « explicatives ». Nous soutenons que c'est en grande partie cette absence de représentation d'une causalité qui est à l'origine des trois points de fragilité de l'apprentissage machine déjà signalés et étudiés dans la littérature. Cette première analyse nous a conduits à distinguer plusieurs fonctions pour les modèles : analyse de données, description, prédiction, explication. Nous avons distingué également deux approches des données : en termes de signaux ou en termes de signes. Dans une approche purement « signal », le modèle prend pour objet d'étude et de traitement le seul niveau de la structure informationnelle du système cible. Dans les approches « signe », les modèles s'engagent sur le lien entre les données et une certaine ontologie plus ou moins réaliste du système cible, réalisme souvent fondé sur des théories scientifiques et sur des hypothèses métaphysiques associées de symétrie et d'invariance. Selon que cet engagement réaliste en reste aux entités et aux propriétés ou qu'il en passe ensuite aux structures voire aux liens causaux, on a affaire à des modèles descriptifs, prédictifs ou explicatifs.

Remarquons en passant que [22], souvent cité dans ces débats tournant autour de prédire et expliquer, n'est justement pas si clair à ce sujet. Lorsqu'il soutient une conception métaphysique continuiste et causaliste, qu'il s'oppose ensuite aux approches discrétisées et statistiques supposées par principe ne pouvoir que décrire, il écrit en effet : *« il n'y a de science [explicative] que dans la mesure où l'on plonge le réel dans un virtuel contrôlé. Et c'est par l'extension du réel dans un virtuel plus grand que l'on étudie ensuite les contraintes qui définissent la propagation du réel au sens de ce virtuel »* ([22], p. 122). Cependant, il n'est justement pas certain que l'approche topologique générale du réel qu'il propose, avec cette métaphore de la plongée dans un espace topologique différentiel à la fois général et supposé par là causalement contraignant, soit en réalité elle-même autre chose qu'une « approche signal » qui veut se faire passer pour une « approche signe » : dans quelle mesure, en effet, une telle plongée est-elle assurée d'être davantage qu'une simple insertion dans un cadre spatio-temporel, cadre lui aussi surimposé, avec ses choix ontologiques et ses biais donc, pour servir à une approche signal ? Mais cela reste ici un débat seulement connexe à notre contribution. En tous les cas, la prise de conscience de la réelle diversité des modèles alternatifs aux modèles exclusivement théorico-explicatifs (diversité également et significativement sous-estimée par [1]) permet justement de poser à nouveaux frais ce genre de questions et d'y répondre plus précisément au regard du contexte technique et de la fonction épistémique du modèle chaque fois recherchée.

Dans cet article, nous avons ensuite caractérisé la recherche d'interprétabilité et d'explicabilité des modèles en termes de modélisation de modèle. Nous avons enfin montré que les modèles explicatifs sont d'emblée explicables, mais que les modèles prédictifs à AM ne le sont pas directement le plus souvent, bien qu'ils puissent être expliqués secondairement par d'autres modèles : ces derniers rendent les modèles d'AM explicables, sans pour autant - ni toujours ni directement - légitimer les ontologies mobilisées par ces modèles ni pouvoir montrer qu'ils expliquent leur système cible. Les usages que l'on fait des modèles interprétant ou expliquant les modèles à AM, aussi impressionnants soient-ils, ne doivent donc pas faire oublier les fragilités persistantes des modèles qu'ils modélisent.

Cet article a présenté ainsi la première étape de notre projet de recherche dont l'objectif final est de définir un cadre épistémique pour l'explicabilité et la validation d'applications d'AM. Pour cela, un ensemble de techniques d'explicabilité, notamment répertoriées dans [7], sera

évalué dans ce cadre, ainsi que la prise en compte de l'incertitude de modèle d'AM telle qu'effectué par exemple dans [10] en utilisant des réseaux de neurones bayésiens.

Références

[1] C. Anderson, The End of Theory: The Data Deluge Makes the Scientific Method Obsolete, *Wired: Science*, 2008.

[2] F. Bacon, *Novum Organum*, 1620.

[3] D. Cardon, J-P. Cointet, et A. Mazières, La revanche des neurones. L'invention des machines inductives et la controverse de l'intelligence artificielle, *Réseaux*, 2018.

[4] A. Cassili, *En attendant les robots. Enquête sur le travail du clic*, Le Seuil, 2019.

[5] C. Denis, Interprétabilité et validation d'applications métiers basées sur de l'IA statistique, *Journée Ethique et IA, PFIA*, 2018.

[6] J-G. Ganascia, *Le Mythe de la Singularité. Faut-il craindre l'intelligence artificielle ?*, Le Seuil, 2017.

[7] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter and L. Kagal, Explaining Explanations: An Approach to Evaluating Interpretability of Machine Learning, *CoRR, http ://arxiv.org/abs/1806.00069*, 2018.

[8] J-M. Ghidaglia, N. Vayatis, Comment faire sortir l'intelligence artificielle des labos ?, *Les Echos*, 2019.

[9] B. Herman, The Promise and Peril of Human Evaluation for Model Interpretability, *Thirsty-first Conference on Neural Information Processing Systems*, 2017.

[10] A. Kendall, Y. Gal, What Uncertainties Do We Need in Bayesian Deep Learning for Computer, *Thirsty-first Conference on Neural Information Processing Systems*, 2017.

[11] T. Lombrozo, *Explanation and Abductive Inference*, The Oxford Handbook of Thinking and Reasoning, 2012.

[12] S. Mallat, Understanding deep convolutional networks. *Phil. Trans. R. Soc*, 2016.

[13] T. Miller, Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 2019.

[14] M. Minsky, Matter, Mind and Models, *Proc. of the International Federation of Information Processing Congress*, 1965

[15] C. Molnar, *Interpretable Machine Learning*, 2018.

[16] M. Morrison M., *Reconstructing Reality: Models, Mathematics, and Simulations*, Oxford University Press, 2015.

[17] W. L. Oberkampf, Christopher J. Roy, *Verification and Validation in Scientific Computing*, Cambridge, 2015.

[18] J. Pearl, *Models, Reasoning, and Inference*, 2009.

[19] M. Pegny, M. I. Ibnouhsein, Quelle transparence pour les algorithmes d'apprentissage machine ?, *Revue d'Intelligence Artificielle*, 2019.

[20] A. Rahimi, Machine Learning has become alchemy, *Thirsty-first Conference on Neural Information Processing Systems*, 2017.

[21] S. Shalev-Schwartz, S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*, 2014.

[22] R. Thom, *Prédire n'est pas expliquer*, Flammarion 1991.

[23] F. Varenne, Modèles et simulations dans l'enquête scientifique : variétés traditionnelles et mutations contemporaines, *Modéliser & Simuler. Épistémologies et pratiques de la modélisation et de la simulation*, Tome I, F. Varenne, M. Silberstein (dir.), Matériologiques, 2013.

[24] F. Varenne, *From Models to Simulations*, Routledge, 2018.

[25] C. Villani, M. Schoenauer, Y. Bonnet, C. Berthet, A. C. Cornut, F. Levin, B. Rondepierre, Donner un sens à l'intelligence artificielle *Mission Villani sur l'intelligence artificielle*, 2018.

[26] M. Zitnik, M. Agrawal, J. Leskovec, Modeling polypharmacy side effects with graph convolutional net- works, *Bioinformatic*, 2018.

Utilité des informations et agents cognitifs

L. Cholvy[1] C. da Costa Pereira[2]

[1] ONERA, Toulouse
[2] Université Côte d'Azur, CNRS, I3S, UMR 7271, France

Résumé

Dans cet article, nous nous intéressons à la modélisation de l'utilité des informations. Plus précisément, nous visons à caractériser ce qu'est, pour un agent cognitif qui a des croyances et des buts, une information utile ou à quel degré une information lui est utile. Nous présentons trois modèles différents et nous prenons la Recherche d'Information comme domaine applicatif en comparant les mesures existantes avec une mesure d'utilité définie ici.

Mots Clef

Utilité, agent cognitif, buts, croyances.

Abstract

This paper focuses on modelling information usefulness. More precisely, it aims at characterizing how useful a piece of information is for a cognitive agent which has some beliefs and goals. The paper presents three different approaches. We take Information Retrieval as a particular applicative domain and we compare some existing measures with the usefulness measure introduced here.

Keywords

Usefulness, cognitive agent, beliefs, goals.

1 Introduction

Usefulness is a ubiquitous notion. For instance, in Data Mining, evaluating the interest of the extracted knowledge is necessary [FSP+14]; in Natural Language Processing, identifying useful terminology [ZPM18] is a prerequisite to any analysis. In Social Science, studying how people achieve effective conversational communication in common social situations is needed. There, Grice [Gri75], introduced the maxim of quantity which emphasizes the fact that a speaker contribution must be as informative as required for the current purposes of the exchange, but not more informative. In Database domain, taking the goals and the preferences of the user who asks a query is necessary for generating cooperative answers [Min98]. In Information Retrieval (IR), the aim is to take into account a query expressed by a user and provide documents which best suit the user need i.e., which are the most useful ones. Initially, the *topical relevance* approach considered that relevant documents are those whose topics best match the topics of the user query [HS13]. This led to the *aboutness* measure. Then, other dimensions have been considered : *coverage*, which measures how strongly the user interests are included in a document [PBV07]; *appropriateness*, which measures how seemly a document is with respect to the user interests [dCPDP09]; and *novelty*, which measures how novel is the document with respect to what the system has already proposed to the user [CKC+08]. However, the user who asks a query is a cognitive agent [RG91, dCMLVC98] : he/she has some goals to achieve and he/she has some beliefs about the world. Moreover, these beliefs are generally incomplete and the user asks queries to the system in order to get new information which will help him/her achieve his/her goals.

In the present work, we consider a general framework in which there are two cognitive agents : one is the user who has some beliefs and some goals modelled as propositional formulas; the second is the system. This latter has some beliefs about the user's beliefs and goals. Its goal is to provide the user with information which is the most useful for him/her to achieve his/her own goals. This framework is general enough to model the paradigm of cooperative exchanges with a system (a speaker, a database, the search engine) who answers the query expressed by the user (the listener, the database user, the web user...) in which the system has to provide the most useful information to the user. Defining the concept of information usefulness in such a context is the main aim of this paper. More precisely, we take the system point of view and try to characterize how useful a piece of information can be for the user.

The paper is organised as follows. In Section 2 we give some preliminaries and state our working hypotheses. In Sections 3, 4, 5 we propose three different definitions of information usefulness, respectively called binary, ordinal and numerical. In Section 6, we consider the particular case of Information Retrieval and compares some measures defined there with ours. Some concluding remarks are given in Section 7.

2 Preliminaries

We consider a propositional language L of which a subset, L_G, is the language used to represent the goals. We consider an agent a with a goal set G_a, which is a finite set of positive literals from L_G. For example, *finish the state of the art of my article, prepare for Monday's class*. Moreover, agent a has a belief base B_a composed of two subsets

B_a^m and B_a^g. B_a^m is the set of formulas from $L \setminus L_G$ which represents a's beliefs. For example, *I know modal logic* and *I know the Python language*. B_a^g contains as many formulas $l_G^1 \wedge \ldots \wedge l_G^{m_G} \rightarrow G$, where each l_G^i is a positive literal of $L \setminus L_G$, as there are $G \in G_a$. Such formulas represent the beliefs of a about what is needed to achieve its goals. For example, *to finish the state of the art (G) I need knowledge about modal logic (p) and BDI agents (q)* (i.e., $p \wedge q \rightarrow G$). The conjunction $l_G^1 \wedge .. \wedge l_G^{m_G}$ is called *premise* of G and it is noted $premise(G)$. Notice that, according to the previous assumptions, we consider that the agent knows how to achieve its goals (in G_a)—the agent knows which are the pieces of information it needs to achieve its goals. This amounts to discarding the goals the agent does not know how to achieve.

Definition 1 *Let C and C' be two conjunctions of literals. C is included in C', noted $C \subseteq C'$, iff all the literals of C are literals of C'. C is equal to C', noted $C = C'$, iff the literals of C are exactly the same as the literals of C'.*

Definition 2 *Let S and S' be two sets of conjunctions of positive literals. $S \preceq^1 S'$ iff (i) $|S| \leq |S'|$ and (ii) if $|S| = |S'|$ then there is a bijection $f : S \rightarrow S'$ such that : $\forall \psi \in S \ \ \psi \subseteq f(\psi)$. $S \prec^1 S'$ iff $S \preceq^1 S'$ and $S' \not\preceq^1 S$.*

Definition 3 *Let S and S' be two sets of conjunctions of positive literals. $S \preceq^2 S'$ iff (i) $|S| \leq |S'|$ and (ii) if $|S| = |S'|$ then there is a bijection $f : S \rightarrow S'$ such that : $\forall \psi \in S \ \ |\psi| \leq |f(\psi)|$. $S \prec^2 S'$ iff $S \preceq^2 S'$ and $S' \not\preceq^2 S$.*

Thus $S \preceq^1 S'$ (resp., $S \preceq^2 S'$) iff S does not have more elements than S'; if S and S' have the same number of elements, then the conjunctions in S are included in the conjunctions of S' (resp., are shorter than those of S'). Notice that $\preceq^1$ is a preorder but it is not total. Some sets of conjunctions are incomparable, such as $\{p, q \wedge r\} \not\preceq^1 \{r, s\}$ and $\{r, s\} \not\preceq^1 \{p, q \wedge r\}$. $\preceq^2$ is a total preorder.

Definition 4 (Missing Information) *Let a be an agent with its belief base B_a and its goal set G_a. Let $G \in G_a$ be such that $B_a \not\models G$. $Missing(B_a, G)$, is defined as follows :*

$$Missing(B_a, G) = \bigwedge_{l : l \in premise(G) \text{ and } B_a \not\models l} l$$

$Missing(B_a, G)$ is the conjunction of all the literals in the premise of G which cannot be deduced from B_a (i.e., which are not yet believed by the agent). Therefore, in the particular case in which $B_a^m = \emptyset$, $Missing(B_a, G) = premise(G)$, i.e., the missing piece of information to achieve G is $premise(G)$.
Notice that the notion of missing information is defined only for the goals that are not already achieved (i.e, goals such that $B_a \not\models G$). A missing information associated to a goal is then the conjunction of all the literals representing the information need to achieve that goal (not yet achieved), and only these ones. Moreover, we would like to stress that, according to Definition 4, the formula whose conclusion is G can be written as : $Missing(B_a, G) \wedge \psi_{B_a,G} \rightarrow G$ with $\psi_{B_a,G} \in L \setminus L_G$, $B_a \models \psi_{B_a,G}$ and $B_a \not\models Missing(B_a, G)$.

Proposition 1 [1]

- *Let $\varphi \in L \setminus L_G$ be a formula and $G \in G_a$ be a goal of agent a. We have that $Missing(B_a \cup \varphi, G) \subseteq Missing(B_a, G)$.*
- *If $\psi \models \varphi$ then $Missing(B_a \cup \psi, G) \subseteq Missing(B_a \cup \varphi, G)$.*
- *Let $\varphi_1 \in L \setminus L_G$, $\varphi_2 \in L \setminus L_G$ be two formulas and $G \in G_a$ be a goal of agent a. We have that $Missing(B_a \cup (\varphi_1 \wedge \varphi_2), G) = Missing((B_a \cup \varphi_1) \cup \varphi_2), G)$.*

Definition 5 (Multiset of missing information)
Let a be an agent whose belief base is B_a and whose goal set is G_a. The multiset [2] of missing information to achieve the goals in G_a is : $Missing(B_a, G_a) = \{Missing(B_a, G_1), \ldots, Missing(B_a, G_k)\}$ with $\{G_1, \ldots, G_k\} = \{G_i \in G_a \text{ and } B_a \not\models G_i\}$.

There is therefore as much missing information as there are unachieved goals, i.e., the cardinality of $Missing(B_a, G_a)$ corresponds to the number of goals that are not yet achieved.

Example 1 *Let us consider a propositional language whose letters are : p, q, r, G_1 and G_2 respectively meaning "I know the main papers about modal logic", "I know the main papers about BDI agents", "I know the Python language", "I can start writing the state of the art" and "My Monday's class is prepared". Let us consider $G_a = \{G_1, G_2\}$ and $B_a = \{p\} \cup \{p \wedge q \rightarrow G_1, r \rightarrow G_2\}$. We have that, $Missing(B_a, G_1) = q$, $Missing(B_a, G_2) = r$ and therefore, $Missing(B_a, G_a) = \{q, r\}$. This means that, in order to achieve its goals, the agent lacks knowledge about BDI agents and about the Python language.*

The following proposition shows that adding a belief to the belief base B_a does not increase the number of missing conjunctions. Moreover, if this does not reduce it either, then it does not increase their size. Finally, if adding a belief to the belief base B_a reduces the number of missing conjunctions, then this means that such new belief allows to achieve one or more goals.

Proposition 2 *For all formula (piece of information) $\varphi \in L \setminus L_G$, we have :*

- *$|Missing(B_a \cup \varphi, G_a)| \leq |Missing(B_a, G_a)|$.*
- *$\forall \varphi$ if $|Missing(B_a \cup \varphi, G_a)| = |Missing(B_a, G_a)|$ then there is a bijection $f : Missing(B_a \cup \varphi, G_a) \rightarrow Missing(B_a, G_a)$ such that $\forall \psi \in Missing(B_a \cup \varphi, G_a) \ \ \psi \subseteq f(\psi)$.*

1. Proofs are omitted due to length limitation
2. Reminder : a multiset is a set whose elements can have several occurrences, such as $\{p, q, p\}$.

— *If* $|Missing(B_a \cup \varphi, G_a)| < |Missing(B_a, G_a)|$ *then* $\exists G_i \in G_a$ *such that* $Missing(B_a, G_i) \in Missing(B_a, G_a)$ *and* $B_a \cup \varphi \models G_i$.

3 A Binary Approach

In this section, we characterize useful information for an agent in view of achieving its goals in two different ways. According to this binary approach, a piece of information is useful or not.

Definition 6 *Let a be an agent with its belief base B_a and its set of goals G_a. Formula $\varphi \in L \setminus L_G$ is U^1-useful for agent a iff $Missing(B_a \cup \varphi, G_a) \prec^1 Missing(B_a, G_a)$. We use the notation $U^1_{G_a,B_a}\varphi$ or, more simply, $U^1\varphi$, when there is no ambiguity.*

According to this definition, a formula φ in $L \setminus L_G$ is useful for a in view of achieving its goals G_a iff being aware of φ allows a to reduce its information need either by reducing the number of missing conjunctions or by simplifying them. Restricting useful information to formulas of $L \setminus L_G$ only amounts (i) to restricting to information the agent must acquire in order to achieve its goals and (ii) to rule out the fact that a goal can be achieved by a other than through the acquisition of information recommended in the formulas whose aims are the conclusions.

Definition 7 *Let a be an agent with its belief base B_a and its goals G_a. The formula $\varphi \in L \setminus L_G$ is U^2-useful for a iff $Missing(B_a \cup \varphi, G_a) \prec^2 Missing(B_a, G_a)$. We use the notation $U^2_{G_a,B_a}\varphi$ or $U^2\varphi$ when there is no ambiguity.*

According to this second definition, a formula φ of $L \setminus L_G$ is U^2-useful for a if knowing φ allows a to reduce its information need either by reducing the amount of missing conjunctions or by reducing their size. However, the two previous definitions, based on different pre-orders, are equivalent as shown by the following proposition.

Proposition 3
$U^1\varphi \iff U^2\varphi$.

We will denote by $U\varphi$ the useful information.

Example 2 Example 1 (continued) $Missing(B_a, G_a) = \{q, r\}$. $Missing(B_a \cup \{r\}, G_a) = \{q\}$. $Missing(B_a \cup \{q\}, G_a) = \{r\}$. $Missing(B_a \cup \{q \wedge r\}, G_a) = \emptyset$. *Therefore,* Ur, Us *and* $U(q \wedge r)$. *In addition, if x is a propositional letter of the language, we have $U(r \wedge x)$ which means that $r \wedge x$ is useful. Indeed, knowing Python and Java is useful for the agent because it allows the agent to achieve G_2.*

The last remark in this example shows a limitation of this binary model. Indeed, r is useful and so is $r \wedge x$ because, like r, it reduces the agent's need for information. However, this could be questionable because $r \wedge x$ contains x, which does not intervene in reducing the agent's need for information. In other words, reading a document on Python and Java, certainly allows the agent to acquire useful information about Python to prepare the class, but leads the agent to read content about Java, not useful for achieving its goals. This limitation is emphasized by the following proposition.

Proposition 4 *Let φ_1 and φ_2 be two formulas of $L \setminus L_G$. If $U\varphi_1$ then $U(\varphi_1 \wedge \varphi_2)$.*

Some more results are given below.

Proposition 5
— *If φ is not useful then* $Missing(B_a \cup \varphi, G_a) = Missing(B_a, G_a)$
— *If* $\exists \psi \in Missing(B_a, G_a)$ *such that* $\varphi \models \psi$ *then* $U\varphi$.
— $U\varphi \not\Rightarrow Missing(B_a, G_a) \models \varphi$
— $Missing(B_a, G_a) \models \varphi \not\Rightarrow U\varphi$

The first point of this proposition shows that adding unnecessary information to the agent's belief base does not change missing information. The second point shows that any information that implies missing information is useful. In particular, any missing information is useful. The reverse is obviously not true. See example 2 : $r \wedge x$ is useful but does not belong to $Missing(B_a, G_a)$. Therefore, all missing information is useful, but some useful information is not missing. The third point illustrates the fact that useful information is not necessarily a logical consequence of the $Missing(B_a, G_a)$ set. Finally, the fourth point illustrates the fact that there are logical consequences of $Missing(B_a, G_a)$ set that are not useful.

4 An Ordinal Approach

In this section we are interested in a notion of *relative usefulness* by defining, in two different ways, a pre-order between the formulas. To compare two formulas φ_1 and φ_2, we compare the two sets of information that is missing once the piece of information is added to the belief base, i.e., we compare $Missing(B_a \cup \varphi_1, G_a)$ and $Missing(B_a \cup \varphi_2, G_a)$, by using either of the pre-orders $\preceq^1$ and $\preceq^2$. Here, the obtained definitions will not be equivalent (see Example 3).

Definition 8 *Let a be an agent, B_a be its belief base and G_a be its set of goals. Let φ_1 and φ_2 be two formulas of $L \setminus L_G$. φ_1 is at least as useful for a as φ_2, denoted by $\varphi_2 \preceq^1_u \varphi_1$, iff $Missing(B_a \cup \varphi_1, G_a) \preceq^1 Missing(B_a \cup \varphi_2, G_a)$. φ_1 is strictly more useful for a than φ_2, denoted by $\varphi_2 \prec^1_u \varphi_1$, iff $\varphi_2 \preceq^1 \varphi_1$ and $\varphi_1 \not\preceq^1 \varphi_2$. Finally, φ_1 is as useful for a as φ_2, denoted by $\sim^1_u$, iff $\varphi_2 \preceq^1_u \varphi_1$ and $\varphi_1 \preceq^1_u \varphi_2$.*

According to this definition, if one piece of information allows to achieve more goals than another, then it is more useful. If it makes it possible to achieve the same number

of goals but if, for at least one goal, it makes it possible to reduce missing information, then it is more useful.
Obviously, $\varphi_2 \prec_u^1 \varphi_1$ iff $Missing(B_a \cup \varphi_1, G_a) \prec^1 Missing(B_a \cup \varphi_2, G_a)$ and $\varphi_2 \sim_u^1 \varphi_1$ iff $Missing(B_a \cup \varphi_1, G_a) = Missing(B_a \cup \varphi_2, G_a)$. $\preceq_u^1$ is a pre-order on all the propositional formulas but not a total pre-order. For example, in Example 3 below, $p \wedge q$ and $p \wedge r$ are incomparable. Indeed $Missing(B_a \cup (p \wedge q), G_a) = \{r\}$ and $Missing(B_a \cup (p \wedge r), G_a) = \{q\}$ and $\{r\} \not\preceq^1 \{q\}$ and $\{q\} \not\preceq^1 \{r\}$.

Definition 9 *Let a be an agent, B_a be its belief base and G_a be its set of goals. Let φ_1 and φ_2 be two formulas of $L \setminus L_G$. φ_1 is at least as useful for a as φ_2, denoted by $\varphi_2 \preceq_u^2 \varphi_1$, iff $Missing(B_a \cup \varphi_1, G_a) \preceq^2 Missing(B_a \cup \varphi_2, G_a)$. φ_1 is strictly more useful for a than φ_2, denoted by $\varphi_2 \prec_u^2 \varphi_1$, iff $\varphi_2 \preceq_U^2 \varphi_1$ and $\varphi_1 \not\preceq_U^2 \varphi_2$. Finally, φ_1 is as useful for a as φ_2, denoted by $\sim_u^2$, iff $\varphi_2 \preceq_u^2 \varphi_1$ and $\varphi_1 \preceq_u^2 \varphi_2$.*

According to this definition, if one piece of information allows to achieve more goals than another, then it is more useful. If it achieves the same number of goals and if the missing information is generally shorter, then it is more useful.These two definitions are not equivalent as shown below.

Example 3 *Let us suppose that : $B_a = \{p \wedge q \rightarrow G_1, p \wedge r \rightarrow G_2\}$ and $G_a = \{G_1, G_2\}$. We have for instance, $Missing(B_a \cup (p \wedge x), G_a) = \{q, r\}$ and $Missing(B_a \cup r, G_a) = \{p \wedge q, p\}$. Thus $r \prec_u^2 (p \wedge x)$ but $r \not\prec_u^1 (p \wedge x)$.*

Proposition 6 *If $\psi \models \varphi$ then $\varphi \preceq_U^1 \psi$ and $\varphi \preceq_U^2 \psi$.*

In particular $\varphi_1 \preceq_U^1 \varphi_1 \wedge \varphi_2$ and $\varphi_1 \preceq_U^2 \varphi_1 \wedge \varphi_2$. That is to say $\varphi_1 \wedge \varphi_2$ is at least as useful, in the sense of $\preceq_U^1$ (and of $\preceq_U^2$) than φ_1. However, we do not have $\varphi_1 \prec_U^1 \varphi_1 \wedge \varphi_2$ neither $\varphi_1 \prec_U^2 \varphi_1 \wedge \varphi_2$ as shown in the previous examples where $p \sim^1 p \wedge x$ and $p \sim^2 p \wedge x$.

5 A Numerical Approach

In this section, we follow a numerical approach by associating each piece of information with a usefulness degree. To begin with, we state some rationality postulates such a measure must satisfy. The general case will not be treated, and we will limit ourselves to calculating the degree of usefulness *of conjunctions of positive literals*. Let φ be a conjunction of positive literals. We define :

- $Cons(B_a, \varphi) = \{l \; literal \; of \; L \setminus L_G : B_a \cup \varphi \models l\}$
- $N_1(\varphi) = \Sigma_{G \in G_a} |Cons(B_a, \varphi) \cap Missing(B_a, G)\}|$
- $N_2(\varphi) = \Sigma_{G \in G_a} |Missing(B_a, G) \setminus Cons(B_a, \varphi)|$
- $N_3(\varphi) = |\varphi \setminus \cup_{G \in G_a} Missing(B_a, G)|$

$Cons(B_a, \varphi)$ is the set of all the literals that are deducible after adding φ to B_a. $N_1(\varphi)$ counts the literals common to $Cons(B_a, \varphi)$ and to the missing information. The larger the $N_1(\varphi)$, the more φ reduces the missing information to achieve the goals. $N_2(\varphi)$ counts the literals of missing information that are not in $Cons(B_a, \varphi)$. Notice that $N_2(\varphi) = \Sigma_{G \in G_a} |Missing(B_a, G)| - N_1(\varphi)$. Therefore, if $N_1(\varphi)$ increases, $N_2(\varphi)$ decreases. $N_3(\varphi)$ counts the literals of φ that are not literals of missing information. Adding them is therefore not useful to achieve the goals.
Let us consider again agent a whose belief base is B_a and goal set is G_a.

Definition 10 *The set of goals that a formula φ allows the agent to achieve is :*

$$E_{B_a, G_a}(\varphi) = \{G \in G_a, B_a \not\models G \; and \; B_a \cup \varphi \models G\}$$

We note $E(\varphi)$ when there is no ambiguity.

Let $U(\varphi)$ be a real representing how much φ is useful for a. We consider the following postulates :

- **(P1)** $|E(\varphi_1)| < |E(\varphi_2)| \implies U(\varphi_1) < U(\varphi_2)$
- **(P2)** $|E(\varphi_1)| = |E(\varphi_2)|$ and $N_1(\varphi_1) > N_1(\varphi_2) \implies U(\varphi_1) > U(\varphi_2)$
- **(P3)** $|E(\varphi_1)| = |E(\varphi_2)|$ and $N_1(\varphi_1) = N_1(\varphi_2)$ and $N_3(\varphi_1) < N_3(\varphi_2) \implies U(\varphi_1) > U(\varphi_2)$
- **P4** $|E(\varphi_1)| = |E(\varphi_2)|$ and $N_1(\varphi_1) = N_1(\varphi_2)$ and $N_3(\varphi_1) = N_3(\varphi_2) \implies U(\varphi_1) = U(\varphi_2)$

According to **(P1)**, the higher the number of goals that a formula makes it possible to achieve, the higher its usefulness degree. According to **(P2)**, **(P3)** and **(P4)**, when two formulas allow to achieve the same number of goals (whether the goals are the same, different or even no goals at all), then the more a formula reduces missing information the more useful it is. Moreover, in case of equality, the most useful information is the one which brings the least useless information ; finally, if they have the same number of useless pieces of information, then they have the same usefulness degree. These postulates are consistent because their premises are incompatible.
Notice that, according to these postulates, if $N_1(\varphi_1) = N_1(\varphi_2)$ and $N_3(\varphi_1) = N_3(\varphi_2)$ then $U(\varphi_1) = U(\varphi_2)$.
In the following, we provide the definition of a usefulness measure U which satisfies these postulates.

Definition 11 *Let a be an agent whose goals are in G_a and let φ be a conjunction of positive literals. We define the usefulness degree [3] by :*

$$U(\varphi) = \frac{1}{|G_a|+1} \left[|E(\varphi)| + \frac{N_1(\varphi)}{N_1(\varphi) + N_2(\varphi) + \frac{N_3(\varphi)}{N_3(\varphi)+1}} \right]$$

The intuitive idea behind this definition is as follows. The usefulness of information can be seen as a calculation of the similarity between the information the agent needs to achieve its goals and the piece of information that arrives. The more direct or indirect elements (that can be deduced)

3. Such a degree should be noted $U_{B_a, G_a}(\varphi)$ but we will note it $U(\varphi)$ when there is no ambiguity.

there are in common between the two, the more useful the information will be. We would like to stress that this fact allows to account for the serendipity factor [Tom00] in the definition of usefulness. Indeed, an agent gets (asks for) a piece of information to achieve a given goal, but if the received piece of information helps also achieving other goals then this fact is considered in the computation of the usefulness. However, the number of common elements is not always enough to distinguish the degrees of usefulness between two pieces of information. Indeed, in some cases it would also be necessary to take into account their differences. We have been inspired by Tversky's idea [Tve77], according to which, in order to calculate the similarity between two objects A and B, we should consider, in addition to what they have in common, what distinguishes them, i.e., the features of A which are not features of B and vice-versa. This is the reason why we have considered these three values, $N_1(\varphi)$, $N_2(\varphi)$ and $N_3(\varphi)$, in our definition.

Remark 1 *We can notice that for any conjunction of positive literals φ we have :*

$$0 \leq \frac{N_1(\varphi)}{N_1(\varphi)+N_2(\varphi)+\frac{N_3(\varphi)}{N_3(\varphi)+1}} \leq 1.$$

$$\frac{N_1(\varphi)}{N_1(\varphi)+N_2(\varphi)+\frac{N_3(\varphi)}{N_3(\varphi)+1}} = 1 \implies E(\varphi) = G_a.$$

Proposition 7 *The measure $U(\varphi)$ proposed in Definition 11 satisfies postulates* **(P1)–(P4)**.

Example 4 *Let us consider : $B_a = \{q\} \cup \{p \wedge q \rightarrow G_1, p \wedge r \rightarrow G_2\}$ et $G_a = \{G_1, G_2\}$. We have then $Missing(B_a, G_a) = \{p, p \wedge r\}$. We obtain $U(p \wedge r) = 1$, $U(p) = 5/6$, $U(p \wedge q) = U(p \wedge x) = 11/14$, $U(r) = 1/6$, $U(q \wedge r) = 1/7$, $U(q \wedge x) = U(q) = 0$. In other words, $p \wedge r$ is the piece of information that has the maximal degree of usefulness, which is explained by the fact that adding $p \wedge r$ allows to achieve both goals G_1 and G_2. The usefulness of p is lower than the usefulness of $p \wedge r$ but it is higher than those of the other formulas, because adding p allows to achieve a goal (G_1). On the other hand, $p \wedge q$, is less useful than p because of q : the agent already knows q therefore q is not useful anymore for the agent because not novel. The same reasoning holds for $p \wedge x$. Formula r instead is less useful because it only reduces missing information regarding one single goal. It is easy to understand that $q \wedge r$ is less useful than r once more because of the unnecessary information q. Obviously, q and $q \wedge x$ are not useful at all because they do not help progressing towards a goal.*

Proposition 8 *If $U(\varphi_1) = U(\varphi_2)$ then $|E(\varphi_1)| = |E(\varphi_2)|$ and $N_1(\varphi_1) = N_1(\varphi_1)$, $N_2(\varphi_1) = N_2(\varphi_1)$, $N_3(\varphi_1) = N_3(\varphi_1)$.*

$N_1(\varphi)$ quantifies the useful part of φ for the agent, while $N_2(\varphi)$ quantifies the agent's disappointment (lack of needed information) towards φ and, finally, $N_3(\varphi)$ quantifies the disturbance caused to the agent by the unexpected and unnecessary content of φ. Our definition of usefulness takes these three aspects into consideration. By this proposition, the only way two formulas can have the same usefulness is by having the same values for these three parameters. This shows that definition 11 does not permit any compensation : a variation of one of these three values cannot be compensated by the variation of the others.

Particular Cases

— **When $B_a^m = \emptyset$:** In the case where $B_a^m = \emptyset$, i.e., when the only beliefs of the agent concern the agent's needs in terms of information about the way to achieve its goals, we have : $Missing(B_a, G) = premise(G)$ and $Cons(B_a, \varphi) = \varphi$. $U(\varphi)$ can then be written as :

$$U(\varphi) = \frac{1}{|G_a+1|} \cdot \left[|E(\varphi)| + \frac{N_1(\varphi)}{K+\frac{N_3(\varphi)}{N_3(\varphi)+1}} \right]$$

with $E(\varphi) = \{G \in G_a, premise(G) \subseteq \varphi\}$, $N_1(\varphi) = \Sigma_{G \in G_a} |\varphi \cap Premise(G)\}|$, $K = \Sigma_{G \in G_a} |premise(G)|$, $N_3(\varphi) = |\varphi \setminus \cup_{G \in G_a} Premise(G)|$

— **When $B_a^m = \emptyset$ and G_a is a singleton :** In this case, the agent has a single goal, G_0, and its only beliefs is the formula which expresses the information need for achieving that single goal. $U(\varphi)$ can then be written as follows :

$$U(\varphi) = \frac{1}{2} \cdot \left(n(\varphi) + \frac{|\varphi \cap premise(G_0)|}{|premise(G_0)| + \frac{|\varphi \setminus premise(G_0)|}{|\varphi \setminus premise(G_0)|+1}} \right)$$

with $n(\varphi) = 1$ if $premise(G_0) \subseteq \varphi$ and $n(\phi) = 0$ otherwise.

Example 5 *Take $premise(G_0) = a \wedge b$. Then we have $U(c) = 0$, $U(a \wedge c) = 1/5$, $U(a) = 1/4$, $U(a \wedge b \wedge c) = 9/10$, $U(a \wedge b) = 1$. In other words, c is not useful at all because knowing c does not allow the agent to reach or get closer to its goal. $a \wedge c$ is a little more useful, because even if knowing c is not useful to the agent, knowing a allows it to get a little closer to its goal. a is more useful than $a \wedge c$ because it does not add unnecessary information. $a \wedge b \wedge c$ is even more useful because even if it adds unnecessary information, it allows the agent to achieve its goal. Finally, $a \wedge b$ is the most useful because it allows the agent to reach its goal and does not add any unnecessary information.*

6 An Example of Application to Information Retrieval

In this section, we will first recall some relevance dimensions in information retrieval which have been used in the literature [dCPDP09] to propose documents to a user (who now takes the place of what we called "agent" in the above general framework). We will then compare those dimensions with the usefulness measure we are proposing here. However, to have a fair comparison, we need to reformulate those dimensions in a logical setting [**?**].

6.1 A Refresher on Relevance Measures

Formally, in the vector space model, a piece of information or, more generally, a document d, can be represented as a vector of T elements, $d = [w_{1d}, \ldots, w_{|T|d}]$. The user interests are represented by a vector $q = [w_{1q}, \ldots, w_{|T|q}]$, $|T|$ being the size of the term vocabulary used. Different choices have been made in the literature regarding the values of w_{id}, for example : simply based on the presence or absence of a word in the document, in this case the vector contains values in $\{0, 1\}$, or based on the frequency of the word in the document and in the whole repository (TF-IDF) [BR11]. Here, we will consider the three relevance dimensions used in [dCPDP09].

Aboutness The measure of *aboutness* (*t*opical relevance) is calculated by the standard cosine-similarity [SM84] :

$$\text{Aboutness}(d, q) = \frac{\sum_{i=1}^{|T|}(w_{iq}.w_{id})}{\sqrt{\sum_{i=1}^{|T|} w_{iq}^2 \cdot \sum_{i=1}^{|T|} w_{id}^2}}. \quad (1)$$

Coverage The *coverage* criterion measures how strongly the user interests are included in a document [PBV07].

$$\text{Coverage}(d, q) = \frac{\sum_{i=1}^{|T|} \min(w_{iq}, w_{id})}{\sum_{i=1}^{|T|} w_{iq}}. \quad (2)$$

This function produces the maximum value 1 when the non null elements in q's vector also belong to d's vector. It produces the value zero when the two vectors have no common element. Moreover, the value of the function increases with the increase of the number of common elements.

Appropriateness This dimension allows to measure how appropriate or how seemly a document is with respect to the user interests [dCPDP09].

$$\text{Appropriateness}(d, q) = 1 - \frac{\sum_{i=1}^{|T|} |w_{iq} - w_{id}|}{|T|}. \quad (3)$$

According to this definition, a piece of information is considered *fully appropriate* if it covers all the user interests. However, if in addition it covers other subjects, it is considered *less appropriate*.

6.2 Reformulation in Logic

We can consider a user query in information retrieval as the information need associated to a goal. This way, the premise of the goal can be represented by a formula. Let φ and ψ be two conjunctions of positive literals of a propositional language. We have :

$$\begin{aligned} \text{Aboutness}(\varphi, \psi)) &= \frac{|\varphi \cap \psi|}{\sqrt{|\varphi| \cdot |\psi|}}, \\ \text{Coverage}(\varphi, \psi) &= \frac{|\varphi \cap \psi|}{|\psi|}, \\ \text{Appropriateness}(\varphi, \psi) &= 1 - \frac{|\varphi \setminus \psi| + |\psi \setminus \varphi|}{|L|}. \end{aligned}$$

After replacing the premises of the agent's goal by the formula ψ, the measure defined in definition 11 is then rewritten as follows :

$$U(\varphi, \psi) = \frac{1}{2} \cdot \left(n(\varphi) + \frac{|\varphi \cap \psi|}{|\psi| + \frac{|\varphi \setminus \psi|}{|\varphi \setminus \psi| + 1}} \right)$$

with $n(\varphi) = 1$ if $\psi \subseteq \varphi$ and $n(\phi) = 0$ otherwise.
More precisely, we consider a propositional language L that has $|T|$ propositional letters $p_1 \ldots p_{|T|}$ and a letter G_0 representing the goal of the user. A document d can then be represented by a formula noted φ_d defined as : $\varphi_d = \bigwedge_{i=1,\ldots,|T| \text{ and } w_{i,d}=1} p_i$. A query q can also be represented by a formula noted $premise(G_0)$ defined by $\psi_q = \bigwedge_{i=1,\ldots,|T| \text{ and } w_{i,q}=1} p_i$.
The following proposition allows us to reformulate in logic the three IR measures we have considered from the literature.

Proposition 9

$$\begin{aligned} \text{Aboutness}(d, q) &= \text{Aboutness}(\varphi_d, \psi_q) \\ \text{Coverage}(d, q) &= \text{Coverage}(\varphi_d, \psi_q) \\ \text{Appropriateness}(d, q) &= \text{Appropriateness}(\varphi_d, \psi_q) \end{aligned}$$

Example 6 *Let us consider again Example 5, with the propositional language whose letters are a, b, c and G_0. $\psi = a \wedge b$ and let us consider the five formulas : $\varphi_1 = c$, $\varphi_2 = a \wedge c$, $\varphi_3 = a$, $\varphi_4 = a \wedge b \wedge c$, and $\varphi_5 = a \wedge b$. The following table summarizes the values of the four measurements.*

φ	$About$	Cov	$Approp$	U
$\varphi_1 = c$	0	0	0	0
$\varphi_2 = a \wedge c$	1/2	1/2	1/3	1/5
$\varphi_3 = a$	$\frac{1}{\sqrt{2}}$	1/2	2/3	1/4
$\varphi_4 = a \wedge b \wedge c$	$\frac{2}{\sqrt{6}}$	1	2/3	9/10
$\varphi_5 = a \wedge b$	1	1	1	1

A number of observations emerge from these results. First of all, we notice that two formulas can have identical degrees of coverage without their degrees of usefulness being identical. Thus, $Coverage(\varphi_2) = Coverage(\varphi_3)$ but $U(\varphi_2) \neq U(\varphi_3)$. Similarly, two formulas may have identical degrees of appropriateness without their degrees of usefulness being identical. Thus, $Appropriatemess(\varphi_3) = Appropriateness(\varphi_4)$ but $U(\varphi_3) \neq U(\varphi_4)$. We also notice that a and $a \wedge b \wedge c$ have identical appropriateness values although for different reasons : $appriopriateness(a) = 2/3$ because a says nothing about b, whereas this is part of the user's information need, and $appriopriateness(a \wedge b \wedge c) = 2/3$ because $a \wedge b \wedge c$, although providing all the information the user need to achieve his/her goal, it provides unnecessary information, c. On the other hand, these different reasons lead to different degrees of usefulness and, in particular, $U(a)$ is much lower than $U(a \wedge b \wedge c)$. Indeed, by definition, U favors information that allows the user need to be satisfied

(this is fully the case with $a \wedge b \wedge c$ whereas it is partially the case with a). Even if $a \wedge b \wedge c$ provides unnecessary information, namely c, the user will be able to achieve his/her goal with it, unlike with a.

The following proposition provides some comparisons between the U measure and the IR ones.

Proposition 10 *Let φ and ψ be two conjunctions of literals.*

- $U(\varphi, \psi) = \text{Aboutness}(\varphi, \psi) = \text{Appropriateness}(\varphi, \psi) = 1 \Longleftrightarrow \varphi = \psi$.
- $\text{Coverage}(\varphi, \psi) = 1 \Longleftrightarrow \psi \subseteq \varphi$.
- $U(\varphi, \psi) = \text{Aboutness}(\varphi, \psi) = \text{Appropriateness}(\varphi, \psi) = \text{Coverage}(\varphi, \psi) = 0 \Longleftrightarrow \varphi \cap \psi = \emptyset$.
- $\text{Coverage}(\varphi_1, \psi) < \text{Coverage}(\varphi_2, \psi) \Longrightarrow U(\varphi_1, \psi) < U(\varphi_2, \psi)$
- $\text{Appropriateness}(\varphi_1, \psi) \leq \text{Appropriateness}(\varphi_2, \psi)$ *and* $\text{Coverage}(\varphi_1, \psi) = \text{Coverage}(\varphi_2, \psi) \Longrightarrow U(\varphi_1, \psi) \leq U(\varphi_2, \psi)$.

7 Conclusion and Future Work

We have proposed three approches to define the notion of usefulness for a cognitive agent. A *binary approach*, which allows to classify a piece of information as being *useful or not*. An *ordinal approach*, which allows to compare two pieces of information in order to establish which one is more useful. Two different operators have been proposed in this case : a pre-order operator and a total order operator. However, and like for the binary approach, the proposed ordinal approach does not allow to consider unnecessary information. This is accounted for by the third approach by means of a numerical definition of usefulness. We have compared, through an easy to understand example, three IR measures from the literature with our numerical measure. The results of the comparision show that our numerical definition of usefulness, based on the cognitive aspects of the user, allows to capture in a single value different dimensions, without the need for eliciting an explicit priority order on the dimensions from the user. In addition, it allows to somehow account for the *seredipity* factor (see Example 4). Moreover, it also allows to account for *novelty* with respect to the user's beliefs, not only with respect to the past user interactions as usual in the literature (see again Example 4, in which the fact that a piece of information contains information already known by the user diminishes its usefulness).

An application of our framework that would be interesting to investigate is its use to reduce the needs to coordinate multiple *assistive agents* advising the same user [STK19]. Other possible applications would be in the case of the Information Flow Problem in multi-agent systems, in which there is a need to ensure an adequate exchange of information within a system [BTJGF18], and in the case of BDI personal medical assistant agents, where one critical requirement is to (automatically) produce an accurate documentation [CMR+18]. We also plan to extend our framework to allow the calculation of usefulness for more general formulas.

Acknowledgements The second author acknowledges support of the project PEPS AIRINFO funded by the CNRS.

Références

[BR11] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. *Modern Information Retrieval - the concepts and technology behind search, Second edition*. Pearson Education Ltd., Harlow, England, 2011.

[BTJGF18] Luis BÃžrdalo, AndrÃl's Terrasa, Vicente JuliÃąn, and Ana GarcÃ■a-Fornes. The information flow problem in multi-agent systems. *Engineering Applications of Artificial Intelligence*, 70 :130 – 141, 2018.

[CKC+08] Charles L. A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2008, Singapore, July 20-24, 2008*, pages 659–666, 2008.

[CMR+18] Angelo Croatti, Sara Montagna, Alessandro Ricci, Emiliano Gamberini, Vittorio Albarello, and Vanni Agnoletti. Bdi personal medical assistant agents : The case of trauma tracking and alerting. *Artificial Intelligence in Medicine*, 2018.

[dCMLVC98] Michael da Costa Móra, José Gabriel Pereira Lopes, Rosa M. Vicari, and Helder Coelho. Bdi models and systems : Bridging the gap. In *Proceedings of ATAL'98*, pages 11–27, 1998.

[dCPDP09] Célia da Costa Pereira, Mauro Dragoni, and Gabriella Pasi. Multidimensional relevance : A new aggregation criterion. In M. Boughanem, C. Berrut, J. Mothe, and C. Soulé-Dupuy, editors, *ECIR*, volume 5478 of *Lecture Notes in Computer Science*, pages 264–275. Springer, 2009.

[FSP+14] Frédéric Flouvat, Jérémy Sanhes, Claude Pasquier, Nazha Selmaoui-Folcher, and Jean-François Boulicaut. Improving pattern discovery relevancy by deriving constraints

from expert models. In *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, pages 327–332, 2014.

[Gri75] Herbert P. Grice. Logic and conversation. In Peter Cole and Jerry L. Morgan, editors, *Syntax and Semantics : Vol. 3 : Speech Acts*, pages 41–58. Academic Press, New York, 1975.

[HS13] Xiaoli Huang and Dagobert Soergel. Relevance : An improved framework for explicating the notion. *JASIST*, 64(1) :18–35, 2013.

[Min98] Jack Minker. An overview of cooperative answering in databases. In *Flexible Query Answering Systems, Third International Conference, FQAS'98, Roskilde, Denmark, May 13-15, 1998, Proceedings*, pages 282–285, 1998.

[PBV07] Gabriella Pasi, Gloria Bordogna, and Robert Villa. A multi-criteria content-based filtering system. In *SIGIR '07 : Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 775–776, 2007.

[RG91] Anand S. Rao and Michael P. Georgeff. Modeling rational agents within a BDI-architecture. In *KR'91*, pages 473–484, 1991.

[SM84] Gerard Salton and Michael McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill Book Company, 1984.

[STK19] Budhitama Subagdja, Ah-Hwee Tan, and Yilin Kang. A coordination framework for multi-agent persuasion and adviser systems. *Expert Syst. Appl.*, 116 :31–51, 2019.

[Tom00] Elaine G. Toms. Serendipitous information retrieval. In *DELOS*, 2000.

[Tve77] Amos Tversky. Features of similarity. *Psychological Review*, 84(4) :327–352, 1977.

[ZPM18] Ziqi Zhang, Johann Petrak, and Diana Maynard. Adapted textrank for term extraction : A generic method of improving automatic term extraction algorithms. *Procedia Computer Science*, 137 :102 – 108, 2018. Proceedings of the 14th International Conference on Semantic Systems 10th âĂŞ 13th of September 2018 Vienna, Austria.

CELTIC/EDAIN : une approche de modélisation et de supervision d'expériences interactives

Damien Mondou[1] Armelle Prigent[1] Arnaud Revel[1]

[1] La Rochelle Université - Laboratoire Informatique, Image, Interraction (L3i)
23 Avenue Albert Einstein, 17000 La Rochelle

prenom.nom@univ-lr.fr

Résumé

Cet article présente une nouvelle approche de modélisation et de supervision d'expérience interactive. Cette approche a été utilisée pour concevoir un jeu sérieux avec un robot Nao. Ce jeu permet aux plus jeunes, de découvrir de manière ludique l'exposition dédiée à l'ethnographie au muséum d'histoire naturelle de La Rochelle et l'exposition consacrée à l'archéologie au musée Sainte Croix de Poitiers. La phase de modélisation du jeu est divisée en deux parties. La première consiste à définir des comportements atomiques, regroupable en patterns. La seconde étape vise à définir des agents implémentant les patterns précédemment définis, en spécifiant les contenus et les comportements réels exécutés sur le procédé contrôlé, dans notre cas le robot Nao. Le premier objectif de cette approche est d'externaliser les contenus du jeu et les comportements réels du procédé (les différents postures et gestures de Nao) en base de données. Le second objectif est de pouvoir définir un jeu sérieux sans aucune contrainte concernant le procédé piloté.

Mots Clef

Modélisation formelle, réseaux d'automates, robotique, interaction homme-robot.

Abstract

This paper presents a new approach to designing and supervising an interactive experience. The approach is implemented by creating a serious game with a Nao robot. This game allows youth to discover in a fun way the exhibition dedicated to the ethnographic artifacts of La Rochelle's natural history museum and the exhibition dedicated to archaeology at the Sainte Croix Museum in Poitiers. The design phase of the game is divided into two steps. The first step defined the atomic behaviors grouped within the pattern. In the second step, the agents implementing these patterns were created ; they specified the contents and behaviors to be executed on the controlled process, in our case the Nao robot. The first objective is to externalize the contents of the game (e.g. robot speech) and the real behaviors of the process (e.g. the different postures and gestures of the Nao) in a database. The second objective is to be able to define a serious game without constraints on the process piloted.

Keywords

Formal model, automatas network, robotic, human-robot interaction.

1 Introduction

Les musées sont continuellement en recherche de nouvelles manières d'améliorer l'expérience des visiteurs, de leur permettre de découvrir les collections et d'acquérir de nouvelles connaissances. Ainsi, depuis plusieurs années, le jeu sérieux a démontré son potentiel pour créer de l'engagement et amener les utilisateurs à améliorer leur compréhension d'un sujet [15]. Les avantages de l'apprentissage par le jeu ont donc été démontrés [33, 13] et de nombreux sites culturels se sont équipés de fonctionnalités ludiques dans cet objectif. Ces enjeux sont particulièrement forts pour le jeune public qui peut montrer un désintérêt pour les musées. Les dispositifs digitaux ludiques représentent un vecteur d'engagement efficace pour ces générations.
Ainsi, on trouve aujourd'hui, dans ces lieux de culture, des systèmes basés sur des écrans mobiles [9], des approches de réalité augmentée [24], de transmedia et des solutions d'immersion virtuelle [7] pour permettre aux visiteurs de naviguer dans les contenus.
Intégrer des systèmes à base de robots est une nouvelle perspective particulièrement intéressante. En effet, les robots deviennent de plus en plus présents dans notre vie quotidienne (bien souvent dans un contexte de support à l'humain). Il peut s'agir de robots d'accueil comme au King's College de Londres où la réceptionniste est un robot nommé Inkha ou dans les supermarchés pour guider les consommateurs dans leur recherche de produits ou fournir des conseils nutritionnels, tels que LoweBot [28], Aiko Chihira [31], et Pepper [5]. Les robots ont également intégrés des oeuvres artistiques [2, 18] dans le cadre de projets transdisciplinaires entre artistes et scientifiques.
Ainsi, les robots investissent aujourd'hui les musées et les lieux de culture pour constituer une alternative aux dispo-

sitifs numériques de diffusion d'information ou de création d'engagement auprès des visiteurs. L'intérêt de cette nouvelle approche tient dans plusieurs points. Elle permet d'une part une interaction plus naturelle avec les visiteurs (il est plus aisé de visiter le musée et jouer sans avoir à être équipé d'une tablette, d'un téléphone ou d'autres équipements qui pourraient interférer avec la visite et la découverte des oeuvres). D'autre part, elle ouvre la voie à une dynamique sociale entre les visiteurs (pouvant échanger entre eux quand bien même ils sont de simples spectateurs de l'interaction). Enfin, les systèmes de vision par ordinateur, intégrés au robot, permettent de détecter la présence de visiteurs et le robot peut alors les interpeller pour les inviter au jeu. Ainsi, un simple visiteur qui n'a pas fait la démarche de jouer sur tablette ou de s'équiper d'un dispositif de réalité virtuelle peut devenir un joueur pour une partie de sa visite.

1.1 L'interaction robotique dans les lieux de culture

Un certain nombre d'expérimentations ont déjà été menées pour analyser la faisabilité et l'impact de l'introduction des robots dans les lieux de culture. Ainsi, depuis 2014, deux robots sont en charge d'accueillir les visiteurs au Musée National des Sciences et Innovations émergentes à Tokyo et peuvent également produire du contenu verbal. A l'exposition Eppur Si Muove au Musée d'Art Moderne de Luxembourg, un robot Nao sur une base de robot mobile est chargé de guider le public et de présenter les œuvres [16]. En France, dans le cadre d'un projet sur l'esthétisme artificiel, le robot Berenson a été déployé au musée du quai Branly. Ce robot se comporte comme un critique d'art ; il exprime une émotion lorsqu'il se trouve devant une oeuvre. Son opinion évoluera grâce aux réactions des visiteurs présents autour de lui [6]. De plus, certains musées, en raison de leur architecture, ne sont pas accessibles aux personnes à mobilité réduite (et ne peuvent réaliser les travaux nécessaires). Le musée du château d'Oiron en France a appréhendé ce problème et, au premier étage du musée, a présenté un robot (Norio) qui peut être contrôlé à distance par un joystick à partir du rez-de-chaussée, ce qui permet aux personnes handicapées de découvrir les œuvres qui leur sont inaccessibles [17]. Sous une autre forme, un projet actuellement mené au L3i consiste à piloter un robot à partir d'un casque de réalité virtuelle [1]. Cette plateforme, destinée actuellement à la recherche et l'aide aux victimes en milieu hostile à l'homme pourrait être adaptée à la visite d'espaces culturels pour les personnes à mobilité réduite.
Nous avons donc, au travers de notre projet "Musées 3.0" et en collaboration avec le muséum d'Histoire Naturelle de La Rochelle et le musée Sainte Croix de Poitiers, développé une expérience interactive pour les jeunes visiteurs au travers d'un jeu sérieux. Ce jeu leur permet de découvrir, d'une manière intéressante et ludique, une partie des collections, au travers d'un quiz oral proposé par un robot Nao. Le robot Nao, initialement développé par la société française Aldebaran (aujourd'hui Softbank Robotics), dispose d'un ensemble de capteurs (notamment une caméra, un sonar et des capteurs tactiles) qui lui permettent de percevoir l'environnement dans lequel il évolue. Ce robot est également capable d'interagir verbalement avec l'humain au travers de ses microphones et haut-parleurs (le robot intégrant les API de reconnaissance et de synthèse vocale). Le jeu, quant à lui, consiste en un jeu de piste dirigé par le robot qui pose des questions aux jeunes joueurs. Ces derniers partent alors dans le musée à la recherche de la réponse et retournent vers Nao pour répondre lorsqu'ils l'ont trouvée.

1.2 Une approche générique de modélisation et de supervision

Concevoir une scénarisation de qualité pour une expérience interactive peut soulever un certain nombre de défis. D'une part, il est parfois complexe de produire une arborescence de cas suffisamment riche pour offrir à l'utilisateur une sensation de liberté dans ses choix d'interaction. De la même manière, une trop grande liberté de l'utilisateur dans une arborescence importante réduit la capacité du concepteur à analyser la qualité de chacune des exécutions possibles. Que ce soit dans l'univers des jeux [11], des systèmes de communication [20] ou de pilotage de robots [25], la nécessité d'utiliser une modélisation de l'activité est primordiale. Cette dernière permet de disposer des méthodes nécessaires à l'anticipation des chemins possibles de l'utilisateur, à la composition dynamique de comportement favorisant une arborescence plus large et à l'analyse des traces d'interaction à posteriori. Ainsi, l'utilisation de réseaux de Petri par exemple [3, 10, 36] ou d'automates à états finis [29, 35] est répandue dans l'état de l'art. En effet, ces approches permettent de garantir une haute qualité d'expérience par la vérification de certaines propriétés (de sécurité, d'accessibilité ou de vivacité notamment).
C'est dans cette dynamique que [27] propose CITE, un framework dédié complet destiné à faciliter la phase de modélisation d'un système interactif et garantissant une souplesse suffisante pour atteindre les objectifs de complexité et d'extensibilité. Il s'agit ici de prendre en compte toutes les dimensions de l'interaction (le temps, l'espace, l'interaction et le contenu) en utilisant un modèle formel capable de construire dynamiquement l'arborescence du scénario à partir de la description des agents, de leurs comportements et des contextes de leur exécution. Nous décrivons ici la structure de CELTIC implémentant les dimensions CIT (Contenu, Interaction et Temps) du modèle CITE basée sur deux couches de description (la dimension E, correspondant à la prise en compte des lieux de l'interaction n'est pas présentée ici). Le processus de supervision dynamique consiste à contrôler l'expérience interactive au regard du modèle formel (basé sur des automates à états finis). L'architecture complète de pilotage (appelée EDAIN) a été définie et implémentée. Nous avons prouvé l'effica-

1. Tests réalisés sur un robot Pepper et un casque HTC Vive.

cité de ce modèle dans le cadre d'expériences en présence de public.
L'un des avantages de ce modèle est qu'il permet de produire, après exécution, une trace complète de l'expérience de chaque utilisateur, modélisée sous la forme d'un automate temporisé. Nous montrerons ici, l'intérêt de cette information pour l'analyse des comportements et pour le potentiel qu'elle peut représenter pour l'apprentissage dynamique en vue de l'amélioration du modèle. En effet, elle constitue une première étape d'intégration d'un processus visant à affiner la valeur des paramètres du modèle en fonction des exécutions passées via un bouclage de pertinence.

2 CELTIC : éditeur de modélisations d'expériences interactives

CELTIC (Common Editor for Location Time Interaction and Content) est un éditeur générique de production de modèle pour des expériences interactives. Il peut s'agir d'un jeu, d'une application ou comme dans notre cas d'étude, d'une expérience de jeu avec un robot. L'objectif est de proposer une modélisation générique et modulaire qui permette de mettre en oeuvre une supervision et une analyse des traces des utilisateurs. Ce modèle vise de plus à prendre en compte les différentes composantes de l'interaction : le temps, l'espace et le contenu. La littérature propose différentes approches de modélisations permettant une supervision et une adaptation que ce soit dans l'univers du web [12, 34] ou de la scénarisation de jeu vidéo [21, 22, 23]. Ces travaux reposent pour la plupart sur des modélisations formelles garantissant une plus grande souplesse dans le contrôle de l'exécution. Ainsi, comme cité précédemment, les réseaux de Petri sont souvent utilisés pour modéliser et vérifier les activités asynchrones, en particulier les jeux sérieux [1, 26]. Des approches de logiques linéaires ont été mises en œuvre pour représenter la consommation de ressources dans le contexte du jeu [8], ou des techniques à base d'automates ont été utilisées pour garantir à l'exécution d'atteindre un état souhaité par analyse d'accessibilité [30]. Cependant, ces modèles se révèlent souvent complexes à manipuler car ils restent bas-niveaux et monolithiques (ils nécessitent, en effet, de modéliser l'intégralité de l'arborescence d'interaction). Une solution est alors de permettre une modélisation offrant une construction dynamique du modèle. Une première approche de supervision dynamique basée sur un modèle générique a été proposée dans [29], et a donné naissance à un cadre de supervision appelé *#Telling*. La modélisation, ici divisée en trois couches, peut superviser l'activité en fonction des scènes d'exécution. Les comportements atomiques des entités sont modélisés par des automates à états finis ensuite composés dynamiquement pour représenter le comportement global d'une entité dans une scène. Dans cette approche, l'objectif est de laisser une possibilité importante de choix à l'utilisateur tout en garantissant la qualité du cadre narratif et ce en facilitant la tâche du concepteur. Bien que l'efficacité de ce modèle ait été prouvée, elle est limitée par certains points. Elle ne permet pas de prendre en compte la dimension temporelle, ni dans la définition du comportement, ni dans le passage d'une situation à une autre. Enfin, la gestion de contenu n'est pas prise en charge et reste intégrée au modèle, ce qui complexifie lourdement la tâche de conception. Nous présentons ici une extension de cette dernière approche qui facilite le contrôle et permet l'externalisation du contenu et la représentation des contraintes temporelles.

2.1 Principe du modèle à deux couches

Notre approche est, elle aussi, basée sur des réseaux d'automates temporisés à états finis , très adaptés pour représenter des systèmes synchrones avec des contraintes de temps. Nous simplifions la tâche de modélisation par une méthode en deux étapes. La modélisation d'une expérience interactive est divisée en deux parties :

— tout d'abord, la description de l'expérience est réalisée de manière abstraite, en définissant les entités réutilisables (les comportements et les patterns), modélisées par des automates temporisés. Cette étape crée la *couche déclarative* ;
— par l'instanciation des entités génériques dans des agents, regroupés dans des contextes d'exécution ordonnés, la *couche d'implémentation* est alors définie.

Le processus de création d'un système interactif est illustré dans les figures 1 et 2 et détaillé dans les sections 2.2 et 2.3.

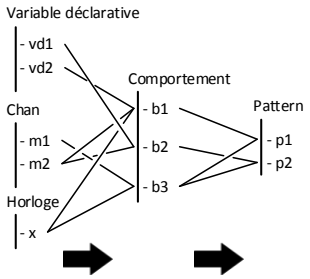


FIGURE 1 – Couche déclarative du modèle.

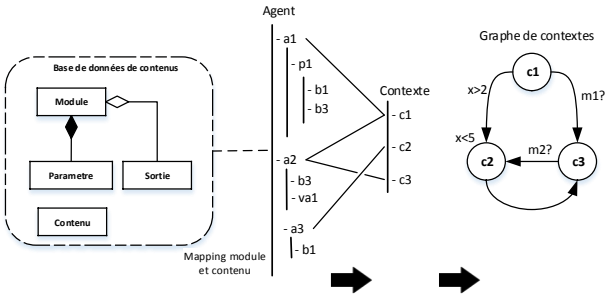


FIGURE 2 – Couche d'implémentation du modèle.

2.2 La couche déclarative

La couche déclarative est celle dans laquelle nous définissons les entités génériques du système modélisé, à savoir les comportements atomiques qui peuvent être exécutés. Ces comportements sont associés à des variables déclaratives (des entiers, des chaînes de caractères ou des booléens) et à des messages (channels), qui sont notamment les signaux permettant de synchroniser plusieurs comportements pour la composition dynamique. Les comportements sont ensuite regroupés en patterns qui représentent un ensemble de comportements, spécifiques à l'entité, qui peuvent être réutilisés par le principe d'héritage multiple.
L'enjeu majeur de cette couche déclarative est de permettre la représentation de comportements atomiques et de patterns de comportements. Ceux-ci peuvent être réutilisés et composés dynamiquement au sein d'agents dans la couche d'implémentation.

Les comportements et les patterns. Dans notre approche, un comportement atomique est une transition d'automate temporisé [19] qui viendra s'agréger avec tous les autres comportements possibles de l'agent. Par exemple, nous souhaitons représenter un premier comportement permettant au robot de se lever (dans un délai compris entre 2 et 5 unités de temps) et un second lui permettant de réaliser une reconnaissance vocale. Les modèles de ces comportements sont représentés sur la figure 3.

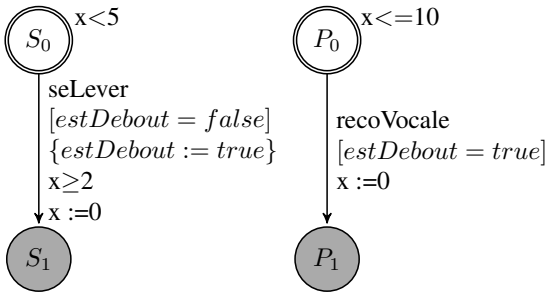


FIGURE 3 – Automates temporisés des comportements *seLever* et *recoVocale*

Les patterns décrivent un ensemble de comportements susceptibles d'être exécutés conjointement au sein de l'application et sont implémentés par des agents. Un automate temporisé du pattern est alors obtenu par le produit synchronisé de l'ensemble des automates des comportements qui le composent (selon l'algorithme de synchronisation décrit dans [4]). Cette composition permet au système d'évoluer par la synchronisation de deux entités ou par leur évolution individuelle. Un exemple d'automate obtenu à partir des comportements définis précédemment est donné à la figure 4.

L'éditeur. La conception du modèle dans cette couche déclarative repose sur un éditeur actuellement en cours de développement dans sa seconde version basée sur des technologie web afin de permettre une édition accessible à tous

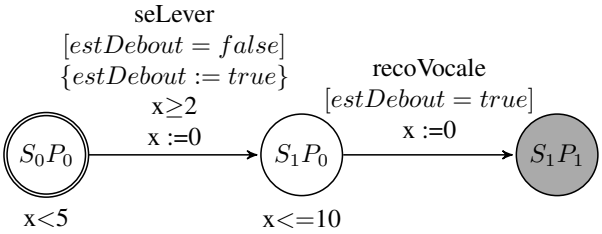


FIGURE 4 – Automate temporisé du pattern

sur un serveur distant. Cet éditeur produit un fichier XML de scénario. Ainsi, pour les comportements et le pattern décrits précédemment, l'éditeur produit la couche déclarative présentée à la figure 5.

```
1  <declarative_layer>
2    <clock id="X"/>
3    <declarative_variable id="estDebout" type="bool"/>
4    <behavior id="seLever" .../>
5    <behavior id="recoVocale" .../>
6    <behavior id="bouger" .../>
7    <behavior id="parler" .../>
8    <pattern id="recoUp" ...>
9      <behavior id="seLever" priority="1"/>
10     <behavior id="recoVocale" priority="2"/>
11   </pattern>
12 </declarative_layer>
```

FIGURE 5 – Déclaration d'un pattern

2.3 La couche d'implémentation

La couche d'implémentation utilise les comportements et patterns décrits en amont pour concevoir les entités impliquées dans l'expérience interactive : les agents et les contextes d'exécution. Il s'agit, dans un premier temps, de construire des agents qui implémentent des comportements et des patterns. Ces agents sont ensuite liés dans des contextes d'exécution représentant une situation spécifique dans lequel les agents interagissent entre eux.

Les agents. Un agent peut implémenter certains des comportements atomiques décrits dans la couche déclarative. Il peut aussi implémenter le rôle d'un ou plusieurs patterns grâce au mécanisme d'héritage multiple.
Chaque comportement composant l'agent doit correspondre à une spécification de la part du concepteur. Ainsi, grâce aux variables d'agents, le concepteur va pouvoir appliquer un ensemble de contraintes, conditionnant l'exécution du comportement. La dernière tâche a effectuer est l'association d'un module à chaque comportement. La base de données possède une table *Module* contenant l'ensemble des modules exécutables sur le processus piloté. Par exemple, pour le comportement *recoVocale* (ligne 9 de la figure 6), le concepteur va associer le module correspondant (dans notre cas le module 1). Ce module, pour être exécuté, impose la spécification de plusieurs paramètres (présent dans la table *Parametre* de la base de données) :

— *dictionnary* (ligne 10). Ce dictionnaire correspond

au vocabulaire proposé à Nao pour la reconnaissance vocale ;

— *time* (ligne 11). Une durée d'écoute pendant laquelle le joueur pourra répondre à la question posée (durée pendant laquelle le robot sera à l'écoute).

Chaque module renvoie à la fin de son exécution une liste de valeurs de sorties (présent dans la table *Sortie*). Le concepteur doit alors indiquer dans quelle variable du modèle chaque valeur retournée sera sauvegardée. Dans notre exemple, le module associé au comportement *recoVocale* retourne le mot reconnu (*wordRecognized*) et un taux de confiance (*confidence*). Ici, le concepteur a donc déclaré deux variables d'agent (ligne 3 et 4) dans lesquels vont être sauvegardés les données reçues à la fin de l'exécution, grâce à l'instruction *saveIn*. Enfin, le concepteur peut indiquer quel contenu est utilisé lors de l'exécution d'un module (en utilisant la table du même nom). Par exemple, lors de l'utilisation du module de synthèse vocale, le concepteur peut indiquer l'identifiant du discours à prononcer par le robot (ce discours étant présent dans la table des contenus).

Le concepteur peut spécialiser un agent grâce aux différents comportements qu'il implémente. L'automate de comportement de l'agent sera alors obtenu par le produit synchrone de l'ensemble des comportements. L'enjeu majeur de cette couche est de garantir la réutilisabilité des comportements. Un agent spécifique à une expérience interactive donnée peut utiliser des comportements atomiques conçus pour une autre expérience. C'est là l'un des enjeux majeurs de notre modèle.

Les figures 6 et 7 donnent une représentation XML de la déclaration d'un agent dans la couche d'implémentation et de son automate de comportements.

Les contextes et le graphe de contexte. Le contexte décrit une situation dans laquelle un certain nombre d'agents va être impliqué. Sa description se fait au travers de la liste des agents impliqués (figure 8). L'automate de contexte sera alors construit par synchronisation de l'ensemble des automates des agents présents.

Le scénario global d'exécution est produit par le graphe de contexte. Il s'agit d'un automate temporisé de haut niveau qui représente les passages entre contextes d'exécution (un exemple est donné à la figure 9).

Cette couche d'implémentation agrège d'abord les comportements dans les agents, puis les agents dans les contextes et produit dynamiquement les automates de comportement de ces derniers par synchronisation.

3 EDAIN : l'architecture de supervision

L'expérience interactive ayant été modélisée, notre outil de supervision (EDAIN) assure la gestion de l'expérience d'une part au travers d'une exécution pilotée par le parcours du graphe de contexte (et suivi des comportements générés dynamiquement au sein des contextes via la syn-

```
1  <global_variable id="out" type="string" value=""/>
2  <agent id="agentAccueil">
3    <agent_variable id="wordRecognized" type="string"
         value=""/>
4    <agent_variable id="confidence" type="double" value
         ="0.0"/>
5    <pattern id="recoUp">
6      <behavior id="seLever" rename="" module="18">
7        <moduleParameter id="posture" table="Behavior"
             id_bdd="20" saveIn=""/>
8      </behavior>
9      <behavior id="recoVocale" rename="" module="1">
10       <moduleParameter id="dictionnary" table="
             ContentDependance" id_bdd="questionID"
             saveIn="reponseCorrecte"/>
11       <moduleParameter id="time" value="10"/>
12       <output id="wordRecognized" saveIn="
             wordRecognized"/>
13       <output id="confidence" saveIn="confidence"/>
14     </behavior>
15   </pattern>
16   <behavior id="bouger" rename="" module="3">
17     <moduleParameter id="gesture" table="Behavior"
           id_bdd="9"/>
18     <output id="out" saveIn="out"/>
19   </behavior>
20   <behavior id="parler" rename="accueilPublic" module
         ="2">
21     <moduleParameter id="speed" value="80"/>
22     <moduleParameter id="text" table="Content" id_bdd
           ="7"/>
23     <output id="out" saveIn="out"/>
24   </behavior>
25 </agent>
```

FIGURE 6 – Déclaration de l'agent

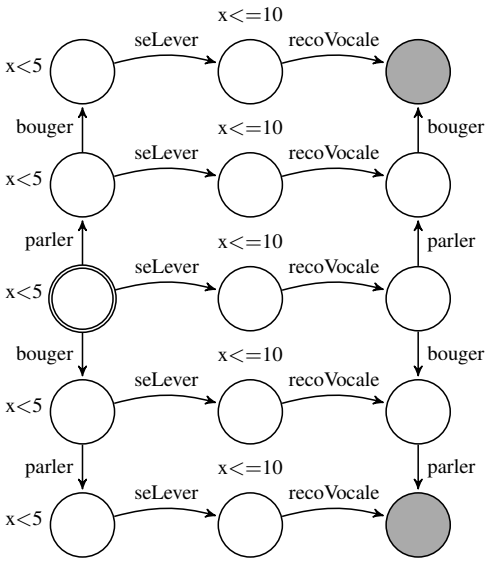


FIGURE 7 – Automate temporisé (simplifié) de l'agent

```
1  <context id="Waiting">
2    <agent id="agentAccueil"/>
3    <agent id="personneDetection"/>
4  </context>
```

FIGURE 8 – Déclaration d'un contexte dans la couche d'implémentation.

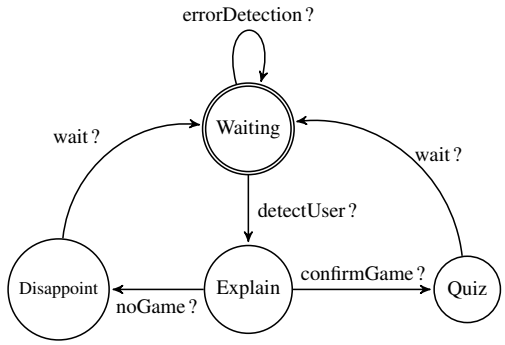


FIGURE 9 – Graphe de contexte

chronisation des agents) et d'autre part via des appels distants des comportements sur le procédé supervisé (dans notre cas : le robot Nao).

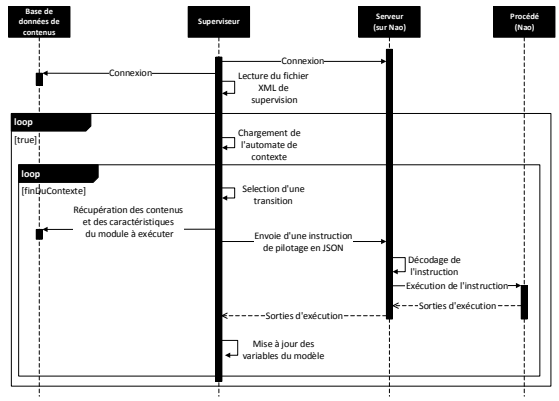


FIGURE 10 – Diagramme de séquences de la supervision.

L'objectif de notre plateforme est de dissocier la modélisation et la supervision de l'activité. Il était alors possible d'utiliser le même modèle pour contrôler différents procédés, à condition de développer un client spécifique pour chaque processus. Un diagramme de séquences de la supervision est détaillé dans la figure 10.
Cet outil de supervision utilise le fichier de spécification précédemment généré en entrée et charge le graphe de contexte. La transition d'un contexte à un autre est réalisée au travers de la réception des signaux ou via des passages de transitions spontanés par satisfaction des gardes (c'est ici la sémantique dynamique de l'automate temporisé du graphe de contexte qui détermine la réalisation de l'exécution).

4 Expérimentations publiques et analyses des situations types

Les expérimentations menées dans les différents lieux de culture ont plusieurs objectifs. Tout d'abord, elles nous permettent d'éprouver la méthode de modélisation proposée et de vérifier la généricité du modèle de scénario, la réutilisabilité des éléments de la couche déclarative d'une expérimentation à une autre et la facilité avec laquelle les contenus sont décorrélés du modèle (au travers de l'architecture s'appuyant sur la base de données présentée à la figure 2). Une autre validation porte sur l'exactitude de notre algorithme de composition dynamique des comportements pour construire les agents et les comportements du système au sein des contextes et sur la qualité du processus de supervision permettant d'exécuter le graphe de contexte. Enfin, nous souhaitons pouvoir récolter des données sur les exécutions de tous les utilisateurs afin d'exploiter la structure du modèle et mettre en oeuvre des méthodes de fouille de données pour extraire automatiquement des informations sur les comportements des usagers.
Nous avons réalisé deux expérimentations publiques pour des évènements tels que la fête de la science (Muséum d'Histoire Naturelle de La Rochelle) ou les Off de la Gamer Assembly (Poiters).

FIGURE 11 – Expérimentation au Muséum d'Histoire Naturelle - 2017.

La première expérimentation a eu lieu lors de la fête de la science 2017 (figure 11). Le scénario créé pour l'occasion avait pour objectif de faire découvrir certaines oeuvres exposées, sous forme de quiz, proposé par les robots Nao. Un total de 46 joueurs a pu découvrir les oeuvres au travers de ce jeu. La seconde expérimentation, dédiée au musée Sainte Croix de Poitiers, s'est déroulée lors de la Gamers Assembly 2018. Nous proposions à cette occasion, un nouveau scénario interactif, faisant intervenir les robots Nao et des jeux développés sur tablettes pour l'occasion. Au travers de deux séances, nous avons reçu un nombre total de 11 joueurs.

4.1 Hypothèse de réutilisabilité

Les comportements et patterns de comportements définis pour le robot Nao concernent un certain nombre d'actions liées aux capteurs et effecteurs de cette plateforme. Il peut s'agir de détecter une présence humaine, de parler, d'écouter une réponse de l'utilisateur par exemple. Les comportements décrits dans la couche déclarative pour la première expérimentation ont été réutilisés sans difficulté pour la seconde expérimentation (environ 70% de réutilisation). Nous avons donc pu redéfinir de manière relati-

vement simple un nouveau scénario sur la base des éléments existants. De plus, d'un lieu de culture à un autre, les contenus manipulés dans le scénario changent (puisqu'ils s'appuient sur les collections des musées). La structure de stockage des textes dans la base nous permet aujourd'hui d'utiliser un comportement générique (par exemple parler) et de le connecter rapidement à un nouveau contenu intégré dans la base de données. Cette nouvelle architecture nous préserve des difficultés de construction des scénarios d'exécution que pouvaient poser l'outil propriétaire de Nao (Choregraphe), qui présentait une dépendance au contenu très forte et posait de grandes problématiques pour la réutilisabilité des scénarios.

4.2 Algorithme de supervision

Au cours de la première expérimentation publique, nous avons validé l'algorithme de supervision dans sa version non-temporisée (les comportements et le graphe de contexte ne contenaient pas d'horloges). L'algorithme s'est avéré efficace et a permis de réaliser un pilotage de l'activité cohérent avec la modélisation réalisée. L'extension mise au point au cours de la seconde expérimentation a donc consisté à intégrer ces contraintes temporelles dans le modèle et à en tenir compte pour la supervision.

4.3 Analyse des comportements utilisateurs

Nous avons intégré, dans notre superviseur, un système d'observation de l'expérience pour générer des traces d'exécution. Dans l'exemple de la figure 12, les lignes 1, 3 et 5 correspondent à l'exécution d'un comportements sur le robot avec une liste de paramètres propre à chaque module (à noter que les paramètres *text* et *gesture* de la ligne 1 ont des valeurs entières, correspondants aux identifiants de la base de données). Les lignes 2, 4 et 6 correspondent quant à elles au retour d'exécution des modules sur le robot. Par exemple, la ligne 2 nous apprend que le joueur 3 a répondu "oui" à la question posée avec un taux de reconnaissance de 52 %.

```
1 13 oct. 2017 14:26:52; joueur:3 question; parameters: [
    speed: 100, text: 8, dictionnary: oui,non, time:
    10, gesture: 9]
2 13 oct. 2017 14:27:06; joueur:3 questionBack;
    parameters: [confidence:0.521499991417,
    wordRecognized:oui]
3 13 oct. 2017 14:27:07; joueur:3 reponseOut1;
    parameters: [gesture: 9, speed: 100, text: 9]
4 13 oct. 2017 14:27:12; joueur:3 reponseOut1Back;
    parameters: [out: ok]
5 13 oct. 2017 14:27:13; joueur:3 poserQuestion;
    parameters: [speed: 100, text: 1, gesture: 9]
6 13 oct. 2017 14:27:23; joueur:3 poserquestionBack;
    parameters: [out: ok]
```

FIGURE 12 – Exemples d'observations obtenus lors de l'expérimentation.

L'objectif à terme est d'utiliser ces traces d'exécution pour analyser le comportement de l'utilisateur final. En effet, lors de la conception de l'expérience interactive, il est difficile pour le concepteur de prévoir tous les cas d'utilisation possibles en situation réelle. Nous souhaitons donc, grâce à ces traces, utiliser un système de clustering visant à effectuer une analyse qualitative de l'expérience a posteriori. Nous utilisons pour cela les traces issues de la première expérimentation (muséum). En effet, lors de celle-ci, des cas d'erreurs sont apparus à plusieurs reprises :

- des erreurs de reconnaissances vocales. Ces erreurs étaient en particulier dues à la résonance de la salle et du nombre importants de personnes qui s'y trouvaient ;
- des abandons. Une longue file d'attente s'est créée au fil de l'après midi qui a découragé un certain nombre de joueurs ;
- des erreurs de reconnaissance faciale liées à l'éclairage ou au mauvais positionnement du joueur.

L'objectif est d'utiliser un algorithme pour identifier automatiquement des clusters. Ceux-ci seront ensuite utilisés afin d'identifier dans quelle situation un joueur se trouve lors d'une prochaine expérimentation afin d'intégrer le bouclage de pertinence et améliorer le modèle (notamment le seuil de confiance pour la reconnaissance).
Nous disposons de 4000 traces représentant les parties de 46 joueurs. A partir de ces traces, nous recréons le graphe modélisant l'évolution d'un joueur dans notre jeu. Nous identifions ainsi quatre clusters émergeant, représentant les parties terminées, les abandons, les erreurs de reconnaissance vocale et les erreurs d'identification. Leur répartition est donnée dans le tableau suivant, sachant qu'un graphe peut appartenir au plus à deux clusters (par exemple, la partie a pu être terminée mais des problèmes de reconnaissance vocale sont apparus).

Parties terminées	Problèmes de reco. vocale	Problèmes d'identification	Abandons
39%	28%	7%	54%

TABLE 1 – Vérité terrain de la première expérimentation.

Nous utilisons un algorithme d'apprentissage non supervisé pour effectuer la clusterisation qui doit pouvoir identifier automatiquement le nombre optimal de clusters. Toutes ces contraintes nous ont amenées à utiliser l'algorithme de clusterisation Affinity propagation [14]. Nous déterminons trois vecteurs caractéristiques différents pour effectuer la clusterisation :

- vecteur "states" : vecteur identifiant pour chaque graphe, le nombre d'occurrences de chaque états du graphe ;
- vecteur "transitions" : vecteur identifiant pour chaque graphe, le nombre d'occurrences de chaque transitions du graphe ;
- vecteur "states + transitions" : couplage des deux vecteurs précédents.

Pour chaque vecteur de caractéristiques, nous ajoutons une donnée concernant la durée finale de l'expérience.
L'utilisation des deux premiers vecteurs par l'algorithme

Affinity propagation permet d'identifier 5 clusters. Cependant, aucune sémantique particulière n'en ressort. En effet, la donnée sur le temps de la partie a beaucoup influencé l'algorithme. Nous avons donc utilisé le troisième vecteur qui permet quand à lui d'identifier 6 clusters, que nous avons étiqueté a posteriori de la façon suivante :

- Abandon sans erreur particulière ;
- Partie terminée avec erreur de reconnaissance vocale ;
- Abandon avec problème d'identification des joueurs (le joueur était mal placé devant le robot) ;
- Abandon avec erreur de reconnaissance vocale ;
- Erreur d'identification ;
- Partie terminée sans erreur particulière.

Sur la base de cette clusterisation, la prochaine étape consistera à identifier dynamiquement dans quelle situation le joueur se trouve afin de parer aux problèmes éventuellement rencontrés, de façon dynamique.

5 Conclusion

Nous avons présenté ici notre modèle de conception d'expériences interactives et la mécanique de supervision permettant de l'utiliser pour piloter le système. Des expériences publiques ont été menées. Elles nous ont permis de valider notre approche et d'entamer des travaux sur l'analyse des comportements utilisateurs via la détection de clusters dans les traces.

De plus, ces expérimentations nous ont prouvé à quelle point la prise en compte du temps est importante pour obtenir une interaction de qualité. En effet, plus encore que dans les systèmes informatiques classiques, le paramétrage des temps de parole du robot et des délais au cours desquels il écoute une réponse verbalisée par le public est primordiale. Par exemple, nous avons constaté que la rapidité de réponse d'un utilisateur est la plupart du temps dépendante de son âge. Ainsi, un enfant va répondre très rapidement (parfois même avant que la séquence d'enregistrement du robot soit déclenchée), alors qu'une personne plus âgée prend davantage de temps avant de verbaliser sa réponse (et souvent alors que cette même séquence d'enregistrement est terminée). De plus, le choix des délais d'attente du robot avant de réagir à la présence d'un visiteur, sa vitesse de parole ou de mouvement doivent être paramétrés avec précaution car ces éléments ont un impact fort sur la perception du public [32]. C'est dans cette dynamique que nous avons intégré au framework une prise en compte des contraintes temporelles pour les comportements des agents et pour la gestion des passages entre contextes. Au delà de cela, nous envisageons d'utiliser les traces utilisateurs pour déterminer par apprentissage automatique les délais d'interaction appropriés aux types de visiteurs et mettre en place un raffinement dynamique du modèle via un bouclage de pertinence. C'est ce qui constitue aujourd'hui la perspective principale de nos travaux. Enfin, nos premiers tests concernant le pilotage d'un robot Pepper via la plateforme CELTIC/EDAIN ont été concluant. En effet, une partie du jeu a pu être déployé sur le robot Pepper sans aucune modification du modèle.

Références

[1] Manuel Araújo and Licínio Roque. Modeling games with petri nets. *Breaking New Ground : Innovation in Games, Play, Practice and Theory - Proceedings of DiGRA 2009*, 01 2009.

[2] Elise Aspord, Joffrey Becker, and Emmanuelle Grangier. *Link human/robot*. Van Dieren eds, 2014.

[3] Franciny M. Barreto, Joslaine Cristina Jeske de Freitas, and Stéphane Julia. A timed petri net model to specify scenarios of video games. In Shahram Latifi, editor, *Information Technology - New Generations*, pages 467–473, Cham, 2018. Springer International Publishing.

[4] Johan Bengtsson and Wang Yi. Timed automata : Semantics, algorithms and tools. In *Lectures on Concurrency and Petri Nets*. Springer Berlin Heidelberg, 2004.

[5] Daniel Van Boom. Pepper the humanoid robot debuts in france. publié sur le site Cnet le 21/10/2015.

[6] Sofiane Boucenna, Philippe Gaussier, Pierre Andry, and Laurence Hafemeister. A robot learns the facial expressions recognition and face/non-face discrimination through an imitation game. *International Journal of Social Robotics*, 6(4) :633–652, Nov 2014.

[7] Marcello Carrozzino and Massimo Bergamasco. Beyond virtual museums : Experiencing immersive virtual reality in real museums. *Journal of Cultural Heritage*, 11(4) :452 – 458, 2010.

[8] Ronan Champagnat, Pascal Estraillier, and Armelle Prigent. Adaptative execution of game : Unfolding a correct story. In *International Conference on Advances in Computer Entertainment Technology*, ACE 06, New York, 2006. ACM.

[9] Tanguy Coenen, Lien Mostmans, and Kris Naessens. Museus : Case study of a pervasive cultural heritage serious game. *J. Comput. Cult. Herit.*, 6(2) :8 :1–8 :19, May 2013.

[10] Hugo Costelha and Pedro Lima. Robot task plan representation by petri nets : Modelling, identification, analysis and execution. *Autonomous Robots*, 33, 11 2012.

[11] G. W. de Oliveira, S. Julia, and L. M. Soares Passos. Game modeling using workflow nets. In *2011 IEEE International Conference on Systems, Man, and Cybernetics*, pages 838–843, Oct 2011.

[12] Roberto De Virgilio. AML : A modeling language for designing adaptive web applications. *Personal and Ubiquitous Computing*, 16(5) :527–541, 2012.

[13] Diane Dietze. *Playing and learning in early childhood education*. Wadsworth Publishing Company, 2011.

[14] Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315(5814) :972–976, 2007.

[15] Juho Hamari, David J. Shernoff, Elizabeth Rowe, Brianno Coller, Jodi Asbell-Clarke, and Teon Edwards. Challenging games help students learn : An empirical study on engagement, flow and immersion in game-based learning. *Computers in Human Behavior*, 54 :170 – 179, 2016.

[16] Patrick Henaff. Entre art et science : Guido, un robot guide espiègle au musée d'art moderne de luxembourg. *session vidéo, Journées Nationales de la Recherche en Robotique (JNRR)*, october 2015.

[17] Mathilde Khlat. Norio, the robot guide of the oiron castle. Publié sur le site de Tourmag le 11/12/2014.

[18] Matthieu Lapeyre, Pierre Rouanet, and Pierre-Yves Oudeyer. Poppy : a New Bio-Inspired Humanoid Robot Platform for Biped Locomotion and Physical Human-Robot Interaction. In *Proceedings of the 6th International Symposium on Adaptive Motion in Animals and Machines (AMAM)*, Darmstadt, Germany, March 2013.

[19] Kim G. Larsen, Paul Pettersson, and Wang Yi. Uppaal in a nutshell. *STTT*, 1 :134–152, 1997.

[20] Thi Thieu Le, Luigi Palopoli, Roberto Passerone, and Yusi Ramadian. Timed-automata based schedulability analysis for distributed firm real-time systems : A case study. *Int. J. Softw. Tools Technol. Transf.*, 15(3) :211–228, June 2013.

[21] A. Liapis, H. P. Martínez, J. Togelius, and G. N. Yannakakis. Adaptive game level creation through rank-based interactive evolution. In *2013 IEEE Conference on Computational Inteligence in Games (CIG)*, pages 1–8, Aug 2013.

[22] Y Louchart and Ruth Aylett. Emergent narrative, requirements and high-level architecture. In *In Proceedings of the 3rd Hellenic Conference on Artificial Intelligence*, page 308, 2004.

[23] Brian Magerko. Building an interactive drama architecture. In *In First International Conference on Technologies for Interactive Digital Storytelling and Entertainment*, pages 226–237, 2003.

[24] T. Miyashita, P. Meier, T. Tachikawa, S. Orlic, T. Eble, V. Scholz, A. Gapel, O. Gerl, S. Arnaudov, and S. Lieberknecht. An augmented reality museum guide. In *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, ISMAR '08, pages 103–106, Washington, DC, USA, 2008. IEEE Computer Society.

[25] A. Miyazawa, P. Ribeiro, W. Li, A. Cavalcanti, and J. Timmis. Automatic property checking of robotic applications. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3869–3876, Sep. 2017.

[26] Stéphane Natkin and Liliana Vega. A Petri Net Model for the Analysis of The Ordering of Actions in Computer Games. In *GAME ON 2003*, France, January 2003. London, October 2003.

[27] Armelle Prigent and Arnaud Revel. CITE – Content Interaction Time and spacE : a hybrid approach to model man-robot interaction for deployment in museums. *EAI Endorsed Transactions on Creative Technologies*, 4(13), November 2017.

[28] Prnewswire. Lowe's introduces lowebot - the next generation robot to enhance the home improvement shopping experience in the bay area. Publié sur le cite PnNewsWire le 30 août 2016, 2016.

[29] Nicolas Rempulski. *Synthèse dynamique de superviseur pour l'exécution adaptative d'applications interactives*. PhD thesis, Université de La Rochelle, 2013.

[30] Nicolas Rempulski, Armelle Prigent, Vincent Courboulay, Matthieu Perreira Da Silva, and Pascal Estraillier. Adaptive Storytelling Based On Model-Checking Approaches. *International Journal of Intelligent Games & Simulation (IJIGS)*, 5(2) :33–42, November 2009.

[31] Reuters. Humanoid robot starts work at japanese department store. Publié sur le site de Reuters le 20 avril 2015.

[32] Arnaud Revel and Pierre Andry. Emergence of structured interactions : from a theoretical model to pragmatic robotics. *Neural Networks*, 22(2) :116–125, January 2009.

[33] Ingrid Pramling. Samuelsson and Marilyn. Fleer. *Play and learning in early childhood settings : international perspectives*. Springer Dordrecht ; London, 2008.

[34] Chian Wang, Dao Zhi Wang, and Jia Li Lin. ADAM : An adaptive multimedia content description mechanism and its application in web-based learning. *Expert Systems with Applications*, 37(12) :8639–8649, 2010.

[35] Rui Wang, Yong Guan, Houbing Song, Xinxin Li, Xiaojuan Li, Zhiping Shi, and Xiaoyu Song. A formal model-based design method for robotic systems. *IEEE Systems Journal*, PP :1–12, 09 2018.

[36] V. A. Ziparo, L. Iocchi, D. Nardi, P. F. Palamara, and H. Costelha. Petri net plans : A formal model for representation and execution of multi-robot plans. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1*, AAMAS '08, pages 79–86, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.

Resource-bounded ATL : the Quest for Tractable Fragments
En quête de fragments mécanisables pour ATL avec ressources

Francesco Belardinelli[1] Stéphane Demri[2]

[1] Department of Computing, Imperial College London & Laboratoire IBISC, Université d'Evry
[2] LSV, CNRS, ENS Paris-Saclay, Université Paris-Saclay

francesco.belardinelli@imperial.ac.uk demri@lsv.fr

Résumé

Dans ce travail, nous commençons par présenter un état de l'art des résultats sur le problème de model-checking en relation avec la logique RB±ATL, qui est une version de ATL *avec ressources. Cela nous permet d'identifier plusieurs problèmes ouverts et d'établir des relations avec les logiques à la RBTL, lorsque RB±ATL est restreinte à un unique agent. Ensuite, nous montrons que le problème de model-checking pour RB±ATL restreinte à un agent et à une ressource est* PTIME*-complet. Pour ce faire, nous faisons une léger détour en passant par les systèmes d'addition de vecteurs avec états. Nous prouvons de nouveaux résultats de complexité pour l'accessibilité d'un état de contrôle et pour la non-terminaison, lorsqu'un seul compteur est autorisé.* **Cet article a été présenté à la conférence AAMAS'19, Montréal, mai 2019.**

Mots Clef

Logiques pour systèmes multi-agent, model-checking, systèmes d'addition de vecteurs avec états, complexité.

Abstract

In this work, we begin by providing a general overview of the model-checking results currently available for the Resource-bounded Alternating-time Temporal Logic RB±ATL. This allows us to identify several open problems in the literature, as well as to establish relationships with RBTL-like logics, when RB±ATL is restricted to a single agent. Then, we show that model checking RB±ATL is PTIME*-complete when restricted to a single agent and a single resource. To do so, we make a valuable detour on vector addition systems with states, by proving new complexity results for their state-reachability and nontermination problems, when restricted to a single counter.* **This paper has been presented at the conference AAMAS'19, Montréal, May 2019.**

Keywords

Logics for agents and multi-agent systems, verification techniques for multi-agent systems, model-checking, vector addition systems with states.

1 Introduction

In recent years, logic-based languages for specifying the strategic behaviours of agents in multi-agent systems have been the object of increasing interest. A wealth of logics for strategies have been proposed in the literature, including Alternating-time Temporal Logic [AHK02], possibly with strategy contexts [LM15], Coalition Logic [Pau02], Strategy Logic [CHP07, MMPV14], among others. The expressive power of these formalisms has been thoroughly studied, as well as the corresponding verification problems, thus leading to model checking tools for game structures and multi-agent systems [CLMM14, LQR15, AdAG$^+$01, KNN$^+$08].

It is worth noticing that the computational models underlying these logic-based languages share a common feature : actions are normally modelled as abstract objects (typically a labelling on transitions) that bear no computational cost. However, if logics for strategies are to be applied to concrete multi-agent systems of interest, it is key to account for the resources actions might consume or produce. These considerations have prompted recently investigations in resource-aware logics for strategies. Obviously, there is a long tradition in resource-aware logics that dates back at least to substructural and linear logics (see e.g. [POY04]). More specifically, in this paper we follow the line of *Resource-bounded Alternating-time Temporal Logics* [ALNR14, ABLN15, ALN$^+$15, ABLN17, AL18, ABDL18], which are characterised by two main features : firstly, actions in concurrent game structures are endowed with (positive/negative) costs ; and secondly, the standard strategy operators of Alternating-time Temporal Logic (ATL) are indexed by tuples of natural numbers, intuitively representing the resource budget available to agents in the coalition. This account has proved successful in the modelling and verification of a number of multi-agent scenarios, where reasoning about resources is critical [ABLN17].

Our motivation for the present contribution is threefold. First of all, in the literature there are several gaps in the results available for the decidability and complexity of the related model checking problem. For instance, if we assume

two resources and two agents in our multi-agent system, then model checking is known to be PSPACE-hard and in EXPTIME, but no tight complexity result is available. Our long-term aim is to fill all such gaps eventually. Further, while completing this picture, it is of interest to identify model checking instances that are tractable. Although the notion of tractable problem is open to discussion, in the context of strategy and temporal logics a model checking problem decidable in polynomial time (in the size of the formula and model) falls certainly within the description. Finally, complexity results for Resource-bounded ATL appear disseminated in a number of references, and are proved by using a wealth of different techniques, thus hindering a clear vision of the state of the art. We aim at developing a unified framework based on general proof techniques. Vector addition systems with states (VASS) are key in this respect [KM69].

Our contribution in this paper is also threefold. Firstly, we give an overview of the complexity results currently available for both RB±ATL and RB±ATL*, the two most significant flavours of Resource-bounded ATL. This allows us to point out that, while for RB±ATL* we have tight complexity results for any number of resources and agents, in RB±ATL there are still several open problems, whose solution is not apparent. Secondly, we extend current model checking results for RB±ATL to a more expressive language including the release operator R too. Thirdly, we prove that model checking RB±ATL is PTIME-complete, when we reason about a single resource and a single agent. Since we show that this setting is tantamount to the Computation Tree Logic CTL with a single resource, our result means that we can reason about resources in CTL at no extra computational cost. Most interestingly, to prove this main contribution we establish new complexity results for the state-reachability and nontermination problems in VASS with a single counter. The latter can be seen as self-standing contributions in formal methods.

Structure of the paper. In Sect. 2, we present background notions on resource-bounded concurrent game structures and ATL-like logics. In Sect. 2.3, we show that the resource-bounded logics RBTL* and RB±ATL* restricted to a single agent have the same expressive power. In Sect. 3, we prove the main theoretical contributions of the paper. Specifically, in Sect. 3.1 we review the state of the art on model checking RB±ATL. Then, in Sect. 3.2 we show that the state-reachability and nontermination problems for VASS with a single counter are decidable in PTIME. Finally, in Sect. 3.3 we leverage on our new results for 1-VASS to prove that model checking RB±ATL with a single agent and a single resource is PTIME-complete. Sect. 4 concludes the paper, discusses the complexity of RB±ATL* fragments, and evokes directions for future work.

This paper has been presented at the conference AAMAS'19, Montréal, May 2019.

2 Preliminaries

Below, we introduce preliminary notions on models for resource-bounded logics, as well as the logical languages themselves. Our presentation follows closely [ABDL18]. In the rest of the paper, $\mathbb{N}$ (resp. $\mathbb{Z}$) is the set of natural numbers (resp. integers) and $[m, m']$ with $m, m' \in \mathbb{Z}$ is the set $\{j \in \mathbb{Z} \mid m \leq j \leq m'\}$. For a finite or infinite sequence $u \in X^\omega \cup X^*$ of elements in some set X, we write u_i for the $(i+1)$-th element of u, i.e., $u = u_0 u_1 \ldots$. For $i \geq 0$, $u_{\leq i}$ is the prefix of u of length $i+1$, i.e., $u_{\leq i} = u_0 u_1 \ldots u_i$ and $u_{\geq i}$ is the suffix of u defined as $u_{\geq i} = u_i u_{i+1} \ldots$. The length of a finite or infinite sequence $u \in X^\omega \cup X^*$ is denoted as $|u|$, where $|u| = \omega$ for $u \in X^\omega$.

2.1 Resource-bounded CGS

Resource-bounded CGS are concurrent game structures [AHK02] enriched with counters and a cost function that assigns a cost (either positive or negative) to every action, thus updating the values of the counters as the system executes. Hereafter we follow closely [ABDL18] and assume a countably infinite set AP of propositional variables (or atoms).

Definition 1 (RB-CGS) *A* resource-bounded concurrent game structure *is a tuple* $M = \langle Ag, S, Act, r, act, cost, \delta, L\rangle$ *such that :*

- Ag *is a finite, non-empty set of agents (by default* $Ag = [1, k]$ *for some* $k \geq 1$*);*
- S *is a finite, non-empty set of* states $s, s', \ldots$ *;*
- Act *is a finite, non-empty set of* actions *with a distinguished action* `idle`*;*
- $r \geq 1$ *is the number of* resources*;*
- $act : S \times Ag \to \wp(Act) \setminus \{\emptyset\}$ *is the* protocol function*, such that for all* $s \in S$ *and* $a \in Ag$, $\texttt{idle} \in act(s, a)$*;*
- $cost : S \times Ag \times Act \to \mathbb{Z}^r$ *is the (partial)* cost function*; that is,* $cost(s, a, \texttt{a})$ *is defined only when* $\texttt{a} \in act(s, a)$*, and moreover, we assume that* $cost(s, a, \texttt{idle}) = \vec{0}$*;*
- $\delta : S \times (Ag \to Act) \to S$ *is the (partial)* transition function *such that* δ *is defined for state* s *and map* $f : Ag \to Act$ *only if for every agent* $a \in Ag$, $f(a) \in act(s, a)$*;*
- $L : AP \to \wp(S)$ *is the* labelling function.

Intuitively, a resource-bounded CGS describes the interactions of a group Ag of agents, who are able to perform the actions in Act according to the protocol function act. The execution of a joint action entails a transition in the system, as specified by the function δ. Moreover, on each transition the values of the r resources are updated according to the cost of the joint action. The `idle` action is introduced in [ALNR14, ALNR17] and it is often advantageous in terms of computational complexity (see e.g. [ABLN17, ABDL18] or Section 3). An RB-CGS M is *finite* whenever L is restricted to a finite subset of AP. The

size $|M|$ of a finite M is the size of its encoding when integers are encoded in binary and, maps and sets are encoded in extension using a reasonably succinct encoding.
Given a coalition $A \subseteq Ag$ and state $s \in S$, a *joint action available to A in s* is a map $f : A \to Act$ such that for every agent $a \in A$, $f(a) \in act(s, a)$. The set of all such joint actions is denoted as $D_A(s)$. Given a state $s \in S$, the set of joint actions available to Ag is simply denoted as $D(s)$, and the function δ is defined only for such joint actions. We write $f \sqsubseteq g$ if $Dom(f) \subseteq Dom(g)$, and for every agent $a \in Dom(f)$, $g(a) = f(a)$. Given a joint action $f \in D_A(s)$, we write $out(s, f)$ to denote the set of *immediate outcomes* :

$$\{s' \in S \mid \text{ for some } g \in D(s), f \sqsubseteq g \text{ and } s' = \delta(s, g)\}.$$

Further, given a joint action $f \in D_A(s)$ and a state s, the cost of a transition from s by f (w.r.t. coalition A) is defined as

$$cost_A(s, f) \stackrel{\text{def}}{=} \sum_{a \in A} cost(s, a, f(a)).$$

A *computation* λ is a finite or infinite sequence $s_0 \xrightarrow{f_0} s_1 \xrightarrow{f_1} s_2 \ldots$ such that for all $0 \leq i < |\lambda| - 1$ we have $s_{i+1} = \delta(s_i, f_i)$.

2.2 The logics RB±ATL* and RB±ATL

To specify the strategic properties of agents in resource-bounded CGS, we present the logics RB±ATL* and its fragment RB±ATL, which are extensions of ATL* and ATL respectively, introduced in [ALNR14, ALN+15] to explicitly account for the production and consumption of resources by agents. Once more, in the presentation of RB±ATL* and RB±ATL we follow [ABDL18].
Syntax. Given a finite set Ag of agents and a number $r \geq 1$ of resources, we write RB±ATL*(Ag, r) to denote the resource-bounded logic with agents from Ag and r resources, whose models are resource-bounded CGS with the same parameters.

Definition 2 (RB±ATL*) *The* state-formulas ϕ *and* path-formulas ψ *in RB±ATL*(Ag, r) are built according to the following BNF :*

$$\begin{aligned} \phi &::= p \mid \neg\phi \mid \phi \wedge \phi \mid \langle\langle A^{\vec{b}} \rangle\rangle \psi \\ \psi &::= \phi \mid \neg\psi \mid \psi \wedge \psi \mid \mathsf{X}\,\psi \mid \psi\,\mathsf{U}\,\psi, \end{aligned}$$

where $p \in AP$, $A \subseteq Ag$, and $\vec{b} \in (\mathbb{N} \cup \{\omega\})^r$. The formulas in RB±ATL$(Ag, r)$ are understood as the state-formulas.*

Clearly, RB±ATL* extends ATL* by indexing the strategic operator $\langle\langle A \rangle\rangle$ with tuple $\vec{b}$, whose intuitive meaning is that the coalition A can achieve their goal by using at most $\vec{b}$ resources. Alternatively, $\vec{b}$ can be understood as the initial budget of the computations, which is the interpretation followed along the paper. Then, the value ω plays the role of an infinite supply of the resource.

The dual operator $[\![A^{\vec{b}}]\!]$ is introduced as $[\![A^{\vec{b}}]\!]\psi \stackrel{\text{def}}{=} \neg\langle\langle A^{\vec{b}} \rangle\rangle\neg\psi$. The linear-time operators X and U have their standard readings ; while the propositional connectives $\vee$, $\rightarrow$, and temporal operators *release* R, *always* G, and *eventually* F are introduced as usual. For instance, $\phi\,\mathsf{R}\,\psi \stackrel{\text{def}}{=} \neg(\neg\phi\,\mathsf{U}\,\neg\psi)$, and therefore $\phi\,\mathsf{R}\,\psi$ shall be equivalent to $\mathsf{G}\,\psi \vee (\neg\phi\,\mathsf{U}\,(\phi \wedge \psi))$.
We also consider the fragment RB±ATL(Ag, r) of RB±ATL*(Ag, r), where path formulas are restricted by $\psi ::= \mathsf{X}\,\phi \mid \phi\,\mathsf{U}\,\phi \mid \phi\,\mathsf{R}\,\phi$.

Remark 1 *Differently from [ABDL18], we explicitly consider the* release *operator* R *in our definition of RB±ATL. Indeed, in [LMO08] it is proved that, differently from the case of the Computation Tree Logic* CTL*, it is not possible to express* R *in terms of* X *and* U *in* ATL*. This proof can be quite easily adapted to the case of RB±ATL by considering the subclass of CGS assigning the cost* 0 *to all actions. Hence, we explicitly introduce the operator* R*. In Section 3.3, we will prove that this extra expressivity comes at no cost in terms of the complexity of the verification problem.*

Semantics. We provide a formal interpretation of the languages RB±ATL* and RB±ATL by using resource-bounded CGS. Specifically, we need a formal notion of resource-bounded strategy for the interpretation of strategic operators $\langle\langle A^{\vec{b}} \rangle\rangle$. To start with, a *(memoryful) strategy* F_A for coalition A is a map from the set of finite computations to the set of joint actions of A such that $F_A(s_0 \xrightarrow{f_0} s_1 \ldots \xrightarrow{f_{n-1}} s_n) \in D_A(s_n)$. A computation $\lambda = s_0 \xrightarrow{f_0} s_1 \xrightarrow{f_1} s_2 \ldots$ *respects strategy* F_A iff for all $i < |\lambda|$, $s_{i+1} \in out(s_i, F_A(s_0 \xrightarrow{f_0} s_1 \ldots \xrightarrow{f_{i-1}} s_i))$. A computation λ that respects F_A is *maximal* if it cannot be extended further while respecting the strategy. In the present context, maximal computations starting in state s and respecting F_A are infinite and we denote the set of all such computations by $Comp(s, F_A)$.
Given a bound $\vec{b} \in (\mathbb{N} \cup \{\omega\})^r$ and a computation $\lambda = s_0 \xrightarrow{f_0} s_1 \xrightarrow{f_1} s_2 \ldots$ in $Comp(s, F_A)$, let the *resource availability* $\vec{v}_i$ *at step* $i < |\lambda|$ be defined as : $\vec{v}_0 = \vec{b}$ and for all $i < |\lambda| - 1$, $\vec{v}_{i+1} = cost_A(s_i, f_i) + \vec{v}_i$ (assuming $n + \omega = \omega$ for every $n \in \mathbb{Z}$). Then, λ is $\vec{b}$*-consistent* iff for all $i < |\lambda|$, $\vec{v}_i \in (\mathbb{N} \cup \{\omega\})^r$ (negative values are not allowed). If $\vec{b}(i) = \omega$, we actually have an infinite supply of the i-th resource, thus not constraining the behaviour of agents with respect to that particular resource. Since the resource availability depends only on the agents in A, in [ABDL18] this is called the *proponent restriction condition* (see also [ABLN15]). Without this restriction about the action costs of the opponent coalitions, the model-checking problem can be shown undecidable when the number of agents is unbounded, see e.g. [ABLN15]. The set of all the $\vec{b}$-consistent (infinite) computations is denoted by $Comp(s, F_A, \vec{b})$. A $\vec{b}$*-strategy* F_A *with respect to*

s is a strategy such that $Comp(s, F_A) = Comp(s, F_A, \vec{b})$.

Definition 3 (Satisfaction relation) *We define the satisfaction relation $\models$ for a state $s \in S$, an infinite computation λ, $p \in AP$, a state-formula ϕ, and a path-formula ψ as follows (clauses for Boolean connectives are standard and thus omitted) :*

$(M, s) \models p$ *iff* $s \in L(p)$
$(M, s) \models \langle\langle A^{\vec{b}} \rangle\rangle \psi$ *iff* *for some $\vec{b}$-strategy F_A w.r.t. s, $\forall \lambda \in Comp(s, F_A)$, $(M, \lambda) \models \psi$*
$(M, \lambda) \models \phi$ *iff* $(M, \lambda_0) \models \phi$
$(M, \lambda) \models \mathsf{X}\,\psi$ *iff* $(M, \lambda_{\geq 1}) \models \psi$
$(M, \lambda) \models \psi\,\mathsf{U}\,\psi'$ *iff* *for some $i \geq 0$, $(M, \lambda_{\geq i}) \models \psi'$, and $\forall 0 \leq j < i$, $(M, \lambda_{\geq j}) \models \psi$*

Clearly, ATL^* and ATL [AHK02] can be seen as fragments of $\mathsf{RB}{\pm}\mathsf{ATL}^*$ and $\mathsf{RB}{\pm}\mathsf{ATL}$ respectively. In particular, the unindexed strategic operator $\langle\langle A \rangle\rangle$ can be expressed as $\langle\langle A^{\vec{\omega}} \rangle\rangle$.
In the sequel, we consider the following decision problem.

Definition 4 (Model Checking) *Let $k, r \geq 1$, ϕ a formula in $\mathsf{RB}{\pm}\mathsf{ATL}^*([1, k], r)$ (resp. $\mathsf{RB}{\pm}\mathsf{ATL}([1, k], r)$), M be a finite RB-CGS for $Ag = [1, k]$ and r resources, and let s be a state in M. The model checking problem amounts to decide whether $(M, s) \models \phi$.*

We conclude this section with a remark on the case of a single agent, which will be prominent in what follows.

Remark 2 *In the case of a single agent, that is, for $Ag = \{1\}$, in our languages we only have modalities $\langle\langle Ag^{\vec{b}} \rangle\rangle$ and $\langle\langle \emptyset^{\vec{b}} \rangle\rangle$, as well as duals $[\![Ag^{\vec{b}}]\!]$ and $[\![\emptyset^{\vec{b}}]\!]$, for $\vec{b} \in (\mathbb{N} \cup \{\omega\})^r$. By Definition 3, the meaning of these operators is as follows :*

$(M, s) \models \langle\langle \emptyset^{\vec{b}} \rangle\rangle \psi$ *iff for every computation λ from s, $(M, \lambda) \models \psi$*
$(M, s) \models [\![\emptyset^{\vec{b}}]\!] \psi$ *iff for some computation λ from s, $(M, \lambda) \models \psi$*
$(M, s) \models \langle\langle Ag^{\vec{b}} \rangle\rangle \psi$ *iff for some $\vec{b}$-consistent computation λ from s, $(M, \lambda) \models \psi$*
$(M, s) \models [\![Ag^{\vec{b}}]\!] \psi$ *iff for every $\vec{b}$-consistent computation λ from s, $(M, \lambda) \models \psi$*

Notice that the semantics of operators $\langle\langle \emptyset^{\vec{b}} \rangle\rangle$ and $[\![\emptyset^{\vec{b}}]\!]$ corresponds to the meaning of modalities A and E in CTL^*; whereas $\langle\langle Ag^{\vec{b}} \rangle\rangle$ and $[\![Ag^{\vec{b}}]\!]$ can be used to introduce resource-bounded counterparts $\mathtt{E}^{\vec{b}}$ and $\mathtt{A}^{\vec{b}}$ of modalities E and A. In Section 2.3, we show that $\mathsf{RB}{\pm}\mathsf{ATL}^*$ for the single agent case is basically equivalent to a different resource-bounded logic RBTL^* introduced in [BF09].

One of our goals is to provide a framework for the complexity classification of (fragments of) $\mathsf{RB}{\pm}\mathsf{ATL}(Ag, r)$, as well as extensions such as $\mathsf{RB}{\pm}\mathsf{ATL}^*(Ag, r)$. Mainly, we focus on bounding the number of agents or resources, proving novel results along the way.

2.3 When RBTL^* comes into play

Below, we present a resource-bounded temporal logic that extends CTL^* by adding resources [BF10, BF09]. Then, we show that this logic is essentially the same as single-agent $\mathsf{RB}{\pm}\mathsf{ATL}^*$ described in Example 2. While such a result is not surprising, apparently it has so far been overlooked in the literature [1]. Such an equivalence allows us to apply results for single-agent $\mathsf{RB}{\pm}\mathsf{ATL}$ to RBTL as well. We first introduce the syntax and semantics of RBTL^* as given in [BF09].

Definition 5 *Given $r \geq 1$, the* state-formulas ϕ *and* path-formulas ψ *in RBTL^* are built according to the following BNF :*

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid \langle\vec{b}\rangle\psi$$

$$\psi ::= \phi \mid \neg\psi \mid \psi \wedge \psi \mid \mathsf{X}\,\psi \mid \psi\,\mathsf{U}\,\psi,$$

where $p \in AP$ and $\vec{b} \in (\mathbb{N} \cup \{\omega\})^r$. Formulas in RBTL^* are all and only the state-formulas generated by the BNF.

The fragment RBTL of RBTL^* is obtained by restricting path formulas just like in the case of $\mathsf{RB}{\pm}\mathsf{ATL}$: $\psi ::= \mathsf{X}\,\phi \mid \phi\,\mathsf{U}\,\phi \mid \phi\,\mathsf{R}\,\phi$. In [ABDL18] the interpretation of RBTL^* is given on a particular class of models, based on vector addition systems with states :

Definition 6 (Model) *A model for RBTL^* is a tuple $A = \langle Q, r, R, L \rangle$ s.t. (i) (Q, r, R) is a* vector addition system with states *(VASS), that is,*

1. *Q is a non-empty finite set of* control states *;*
2. *$r \geq 1$ is the number of* counters *;*
3. *R is a finite subset of $Q \times \mathbb{Z}^r \times Q$;*

and (ii) $L : AP \to \wp(Q)$ is a labelling function.

In a model A, a *pseudo-run* λ is an infinite sequence $(q_0, \vec{v}_0) \to (q_1, \vec{v}_1) \to \ldots$ such that for all $i \geq 0$, there exists $(q, \vec{u}, q') \in R$ such that $q_i = q$, $q_{i+1} = q'$, and $\vec{v}_{i+1} = \vec{u} + \vec{v}_i$. A pseudo-run λ is a *run* iff for all $i \geq 0$, $\vec{v}_i \in (\mathbb{N} \cup \omega)^r$.

Definition 7 (Satisfaction relation) *We define the satisfaction relation $\models$ in model A, for state $q \in Q$, run λ, $p \in AP$, state-formula ϕ, and path-formula ψ as follows (clauses for Boolean connectives are immediate and thus omitted) :*

$(A, q) \models p$ *iff* $q \in L(p)$
$(A, q) \models \langle\vec{b}\rangle\psi$ *iff* *for some run λ from $(q, \vec{b})$, $(A, \lambda) \models \psi$*
$(A, \lambda) \models \phi$ *iff* $(A, \lambda_0) \models \phi$
$(A, \lambda) \models \mathsf{X}\,\psi$ *iff* $(A, \lambda_{\geq 1}) \models \psi$
$(A, \lambda) \models \psi\,\mathsf{U}\,\psi'$ *iff* *for some $i \geq 0$, $(A, \lambda_{\geq i}) \models \psi'$, and for all $0 \leq j < i$, $(A, \lambda_{\geq j}) \models \psi$*

1. Indeed, in [ABDL18], complexity results are given independently for both RBTL^* and single-agent $\mathsf{RB}{\pm}\mathsf{ATL}^*$, even though the two logics can be translated one into the other.

Next, we prove that the logics RBTL* and RB$\pm$ATL$^*(\{1\}, r)$ with a single agent are semantically equivalent, in the sense that truth-preserving translations exist between models and formulas. First, consider the translation map τ from RBTL* to RB$\pm$ATL$^*(\{1\}, r)$ such that τ is the identity on AP, it is homomorphic for Boolean and temporal operators, and $\tau(\langle\vec{b}\rangle\psi) \stackrel{\text{def}}{=} \mathtt{E}^{\vec{b}}\tau(\psi)$. Actually, it can be shown that τ is a bijection between RBTL* and RB$\pm$ATL$^*(\{1\}, r)$. Not only that, but τ is a bijection between RBTL and RB$\pm$ATL$(\{1\}, r)$ as well. Further, given a resource-bounded CGS $M = \langle\{1\}, AP, S, Act, r, act, cost, \delta, L\rangle$ with a single agent 1, define the associated model $A_M = \langle S, r, R, L\rangle$ for RBTL* such that

— R is the set of tuples $(q, \vec{u}, q')$ such that $\delta(q, \mathtt{a}) = q'$ for some action $\mathtt{a} \in act(q, 1)$ with $cost(q, 1, \mathtt{a}) = \vec{u}$.

Symmetrically, given a model $A = \langle Q, r, R, L\rangle$, define the associated single-agent, resource-bounded CGS $M_A = \langle\{1\}, Q, R, r, act, cost, \delta, L\rangle$ such that for every $q \in Q$,

— $act(q, 1) = \{(q', \vec{u}, q'') \in R \mid q = q'\}$;
— for every $(q, \vec{u}, q') \in act(q, 1)$, $cost(q, 1, (q, \vec{u}, q')) = \vec{u}$;
— for every $(q, \vec{u}, q') \in act(q, 1)$, $\delta(q, (q, \vec{u}, q')) = q'$.

We now state the following auxiliary lemma, whose proof follows immediately by the definitions of A_M and M_A above.

Lemma 1 *1. Given a single-agent, resource-bounded CGS M and state $s \in S$, for every $\vec{b}$-consistent computation λ in M, in A_M there exists a run λ' from $(s, \vec{b})$ such that for every $i \geq 0$, $(\lambda_i, \vec{v}_i) \to (\lambda_{i+1}, \vec{v}_{i+1})$ with $\vec{v}_{i+1} = \vec{v}_i + \vec{u}$ for $\vec{u} = cost(\lambda_i, 1, \mathtt{a}_i)$ and $\lambda_i \xrightarrow{\mathtt{a}_i} \lambda_{i+1}$.*

2. Given a model A for RBTL* and state $q \in Q$, for every run λ from $(s, \vec{b})$, in M_A there exists a $\vec{b}$-consistent computation λ' such that for every $i \geq 0$, $\lambda_i \xrightarrow{(\lambda_i, \vec{u}, \lambda_{i+1})} \lambda_{i+1}$ for $(\lambda_i, \vec{v}_i) \to (\lambda_{i+1}, \vec{v}_{i+1})$ and $\vec{v}_{i+1} = \vec{u} + \vec{v}_i$.

By using Lemma 1 we can finally prove that RB$\pm$ATL$^*(\{1\}, r)$ and RBTL* are closely related semantically.

Theorem 1 *(1) For every ϕ in RBTL* and model A with state $q \in Q$, $(A, q) \models \phi$ iff $(M_A, q) \models \tau(\phi)$. (2) For every ϕ' in RB$\pm$ATL* and single-agent, resource-bounded CGS M with state $s \in S$, $(M, s) \models \phi'$ iff $(A_M, s) \models \tau^{-1}(\phi')$.*

Consequently, RBTL* and the restriction of RB$\pm$ATL* to a single agent are essentially the same logic in the sense that their translations are semantically faithful when single-agent RB-CGS are understood as RBTL* models (i.e., a VASS with a valuation). A similar result holds for RBTL and single-agent RB$\pm$ATL. This result is particularly relevant in the light of Section 3, where we dig deeper into the verification of single-agent, resource-bounded logics.

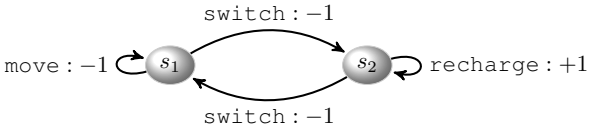


FIGURE 1 – The resource-bounded CGS in Example 1. Transitions with action `idle` are omitted.

Example 1 *We illustrate the formal machinery introduced so far, particularly the single-agent case, with a toy example. We consider a scenario in which a rover is exploring an unknown area. At any time the rover can choose between two modes : either it moves around or it recharges its battery through a solar panel, but it cannot do both things at the same time. Moving around consumes one energy unit at every time step, whereas the rover can recharge of one energy unit at a time. Switching between these modes requires one energy unit.*
This simple scenario can be modelled as the resource-bounded CGS $M = \langle\{rover\}, \{s_1, s_2\}, \{\mathtt{move}, \mathtt{recharge}, \mathtt{switch}, \mathtt{idle}\}, 1, act, cost, \delta, L\rangle$ depicted in Figure 1, where in particular :

— $act(s_1, rover) = \{\mathtt{move}, \mathtt{switch}, \mathtt{idle}\}$ *and* $act(s_2, rover) = \{\mathtt{recharge}, \mathtt{switch}, \mathtt{idle}\}$;
— $cost(s_1, rover, \mathtt{move}) = cost(s_1, rover, \mathtt{switch}) = cost(s_2, rover, \mathtt{switch}) = -1$ *and* $cost(s_2, rover, \mathtt{recharge}) = +1$;
— $\delta(s_1, \mathtt{move}) = s_1$, $\delta(s_1, \mathtt{switch}) = s_2$, $\delta(s_2, \mathtt{recharge}) = s_2$, *and* $\delta(s_2, \mathtt{switch}) = s_1$;
— $AP = \{moving\}$ *and* $L(moving) = \{s_1\}$.

Even in such a simple scenario with a single agent, we can express interesting properties such as "no matter what the rover does, at any time it has a strategy, with an initial budget of at most b energy units, such that it will eventually be moving". This specification can be expressed in RB$\pm$ATL as

$$[\![\{rover\}^\omega]\!]\,\mathsf{G}(\langle\!\langle\{rover\}^b\rangle\!\rangle\,\mathsf{F}\,moving) \tag{1}$$

Next, we show that specifications such as (1), concerning a single agent and a single resource, can be efficiently verified in PTIME.

3 Model-checking RB$\pm$ATL$(\{1\}, 1)$

This section is devoted to the technical developments of our main theoretical results. Specifically, in Section 3.1 we review the known complexity results for model checking RB$\pm$ATL and its fragments. Then, in Section 3.2 we prove that the control-state reachability and nontermination problems for vector addition systems with states (VASS) with one counter are decidable in PTIME. These results are then used in Section 3.3 to show that the model-checking problem for RB$\pm$ATL$(\{1\}, 1)$ is also in PTIME. Thus, our

contribution shows that reasoning about a single resource in RB±ATL with a single agent comes at no extra computational cost compared to CTL.

3.1 Model Checking Results for RB±ATL

In Table 1, we summarize the main complexity results available in the literature for RB±ATL(Ag, r), depending on the number $|Ag|$ of agents and the number r of resources. The result in boldface is original from this contribution. All the results hold in the presence of R instead of G, except the PSPACE upper bound from [ALNR17, Theo. 2].

For an unbounded number of resources and at least two agents, the model-checking problem is known to be 2EXPTIME-complete. This result follows from Theorem 2 (membership) and Theorem 3 (hardness) in [ABDL18]. When restricted to a single agent, the problem becomes EXPSPACE-complete [ABDL18, Th. 4].

For a fixed number of resources greater than four and at least two agents, the model-checking problem is again EXPTIME-complete. The upper bound follows from [ABDL18, Cor. 1], while the lower bound derives from the complexity of the control-state reachability problem for alternating VASS [CS14], which can be simulated by using two agents only [ABDL18, Th. 3]. Further, for a fixed amount of resources greater than two, and two agents, the model-checking problem is in EXPTIME [ABDL18, Cor. 1]. In the case of a single agent, the same problem is in PSPACE [ABDL18, Cor. 2] ; whereas it is PSPACE-hard in both cases, as we can reduce to it the control-state reachability problem for 2-VASS, which is PSPACE-complete [BFG$^+$15].

Finally, in the case of a single resource, the problem is known to be in PSPACE [ALNR17, Th. 2] (the result is established for a language with G and it is plausible to extend it to R). For the case of a single agent, model checking is in PTIME, which is the main theoretical contribution of this section. It is therefore PTIME-complete as model checking CTL is already PTIME-hard (see, e.g., [Sch03, DGL16]). The characterisation of the complexity for one resource and at least two agents is still open : currently, neither the proof of the PTIME upper bound in Section 3.3, nor the PSPACE-hardness results from [JS07] and [FLL$^+$17, Sect. 5] could be advantageously used to close this complexity gap.

3.2 Decision problems for 1-VASS

In order to show that the model-checking problem for RB±ATL$(\{1\}, 1)$ is PTIME-complete, we establish that two well-known decision problems on vector addition systems with states (VASS), when restricted to a single counter, can be solved in polynomial time. More precisely, we show that the control-state reachability and nontermination problems for 1-VASS are in PTIME, whereas, for instance, the control-state reachability problem for VASS is EXPSPACE-complete in general [Lip76, Rac78]. Although control-state reachability is a subproblem of the covering problem, that has been quite studied (see, e.g., [AH09, BS11, Dem13]), to the best of our knowledge there is no result in the literature on the upper bound when restricted to a single counter. Hereafter, we provide formal arguments for tractability by appropriately tuning and correcting the proof technique dedicated to the boundedness problem for 1-VASS from [RY86]. Note also that in [GHLT16], the updates in the BVASS (extending VASS) are restricted to the set $\{-1, 0, +1\}$ (see [GHLT16, Def. 1]). Therefore the upper bound in [GHLT16] does not extend to our present case where updates are arbibtrary integers encoded in binary. When updates are arbitrary integers encoded in binary (as done herein), the relevant problems for 1-BVASS are known to be PSPACE-complete [FLL$^+$17].

We recall the notion of VASS as given in Definition 6, so a VASS is a structure $V = (Q, r, R)$, where R is a finite set of transitions. A *configuration* of a VASS V is defined as a pair $(q, \vec{x}) \in Q \times \mathbb{N}^r$ (ω is discarded in this section). Given $(q, \vec{x})$, $(q', \vec{x'})$ and a transition $t = q \xrightarrow{\vec{u}} q'$, we write $(q, \vec{x}) \xrightarrow{t} (q', \vec{x'})$ whenever $\vec{x'} = \vec{u} + \vec{x}$. Then, $(q_0, \vec{x_0})$ is called the *initial* configuration.

An r-VASS is a VASS with r counters. We present two standard decision problems on VASS that play a crucial role in solving the model checking problem for RB±ATL$(Ag, 1)$.

Control state reachability problem CREACH(VASS) :

Input : a VASS V, a configuration $(q_0, \vec{x}_0)$, and a control state q_f.

Question : is there a finite run with initial configuration $(q_0, \vec{x}_0)$ and with final configuration with state q_f ?

Nontermination problem NONTER(VASS) :

Input : a VASS V and a configuration $(q_0, \vec{x}_0)$.

Question : is there an infinite run with initial configuration $(q_0, \vec{x}_0)$?

Other classical decision problems for VASS have been considered in the literature (see e.g. recent developments about the reachability problem in [Sch16, CLL$^+$18]), but in this paper we only need to tame the control-state reachability and nontermination problems for 1-VASS in order to solve the model-checking problem for RB±ATL$(\{1\}, 1)$.

Definition 8 (Simple Run, Path, and Loop) *A* simple run $\rho = (q_0, \vec{x_0}), \ldots, (q_k, \vec{x_k})$, $k \geq 0$, *is a finite run such that no control state appears twice. A* simple path *is a sequence of transitions $t_1 \ldots t_k$ such that no control state occurs more than once. A* simple loop *is a sequence of transitions $t_1 \ldots t_k$ such that the first control state of t_1 is equal to the second control state of t_k (and it occurs nowhere else) and no other control state occurs more than once.*

In a 1-VASS, a simple loop is *(strictly) positive* if the cumulated effect is (strictly) positive. Given a run $\rho = (q_0, x_0), \ldots, (q_k, x_k), \ldots$ and $\alpha \geq 0$, we write $\rho^{+\alpha}$ to denote the sequence $(q_0, x_0 + \alpha), \ldots, (q_k, x_k + \alpha), \ldots$. If ρ is a run, the sequence $\rho^{+\alpha}$ is also a run. The following lemma provides 1-VASS with a characterisation of runs ending in a distinguished final state.

$r\backslash\|Ag\|$	∞	**2**	**1**
∞	2EXPTIME-c. [ABDL18, Th. 2 and 3]		EXPSPACE-c. [ABDL18, Th. 4]
$\geq$ **4**	EXPTIME-c. [ABDL18, Cor. 1]		PSPACE-h. [BFG$^+$15] in PSPACE [ABDL18, Cor. 2]
3	PSPACE-h. [BFG$^+$15] in EXPTIME [ABDL18, Cor. 1]		
2			
1	PTIME-h. (from ATL) in PSPACE [ALNR17, Th. 2]		PTIME-h. (from CTL) **in PTIME (Th. 4)**

TABLE 1 – The complexity of model checking $\mathsf{RB}\pm\mathsf{ATL}(Ag, r)$.

Lemma 2 *Let V be a 1-VASS, (q_0, x_0) an initial configuration, and q_f a location. There is a finite run from (q_0, x_0) to configuration (q_f, x_f) for some $x_f \geq 0$ iff (1) either $q_0 = q_f$; or*

2. *there is a simple path $(q_0, x_0), \ldots, (q_k, x_k)$ with $q_k = q_f$; or*
3. *we have that*
   - *there is a simple run $(q_0, x_0), \ldots, (q_n, x_n)$,*
   - *there is a strictly positive simple loop $t_1 \ldots t_\beta$ such that $(q_n, x_n) \xrightarrow{t_1 \ldots t_\beta} (q_n, x_n + \alpha)$ is a run ($\alpha > 0$),*
   - *there is a simple path starting at q_n and ending at q_f.*

As illustration, Figure 2 presents a 1-VASS V, and witness runs and path for the positive instance $(V, (q_0, 7), q_f)$ of CREACH(1-VASS). By contrast, the configuration $(q_0, 5)$ cannot reach q_f.

Proof First, it is not difficult to check that if either (1), (2) or (3) holds, then there is a finite run from (q_0, x_0) to configuration (q_f, x_f) for some $x_f \geq 0$. By way of example, firing the strictly positive simple loop at least $(|Q| \times \max\{|\ u\ |\colon q \xrightarrow{u} q' \text{ is a transition}\}$ times, allows to pursue the run following the path from q_n to q_f.
Conversely, let us suppose that $\rho = (q_0, x_0), \ldots, (q_k, x_k)$ is a run with $q_k = q_f$. If $q_0 = q_f$, then the witness run can be reduced to (q_0, x_0). Otherwise, either ρ is a simple run and condition (2) holds, or there are $0 \leq i < j \leq k$ such that $q_i = q_j$. In case $x_i \geq x_j$, the subrun $(q_i, x_i), \ldots, (q_j, x_j)$ can be removed from ρ while leading to a run reaching q_f. Typically, the suffix subrun $(q_i, x_i), \ldots, (q_j, x_j), \ldots, (q_k, x_k)$ with $\rho_\dagger = (q_j, x_j), \ldots, (q_k, x_k)$ is replaced by $\rho_\dagger^{+\alpha}$ for $\alpha = x_j - x_i$. Such a transformation can be performed as soon as the subruns correspond to the application of simple loops with negative effect. Without loss of generality, we can assume that ρ has no loop with strictly negative effect.
If ρ is not a simple run, there are $0 \leq I < J \leq |Q|$ such that $q_I = q_J$ and $x_I < x_J$. Consequently,

- there is a simple run $(q_0, x_0), \ldots, (q_I, x_I)$;
- there is a strictly positive simple loop $t_I \ldots t_{J-1}$ such that $(q_I, x_I) \xrightarrow{t_I \ldots t_{J-1}} (q_I, x_I + (x_J - x_I))$;
- there is a path from q_J to $q_k = q_f$ such that $(q_J, x_J), \ldots, (q_k, x_k)$ is a run. So, there is a simple path from q_J to q_k.

As a result, condition (3) is satisfied and the lemma holds.

The characterisation in Lemma 2 can be turned into an algorithm running in polynomial time.

Theorem 2 *The problem CREACH(1-VASS) is in* PTIME.

Proof Let V be a 1-VASS, (q_0, x_0) an initial configuration, and q_f a location. If $q_0 = q_f$, we are done. Otherwise, define values $\texttt{maxval}_q^i$ for $i \in [0, |Q|]$ and $q \in Q$ such that if there is a run $(q_0, x_0), \ldots, (q_j, x_j)$ with $q_j = q$ and $j \leq i$, then the maximal value x_j among all these runs is precisely $\texttt{maxval}_q^i$. When there is no such run, by convention $\texttt{maxval}_q^i = -\infty$. Similar values have been considered to solve the boundedness problem for 1-VASS in [RY86]. Let us compute the values $\texttt{maxval}_q^i$:

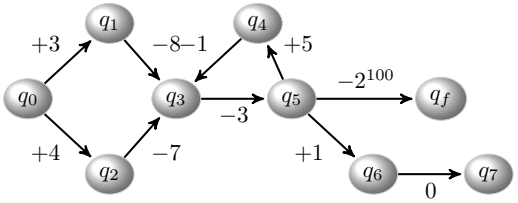
- $\texttt{maxval}_{q_0}^0 \stackrel{\text{def}}{=} x_0$ and $\texttt{maxval}_q^0 \stackrel{\text{def}}{=} -\infty$ for all $q \neq q_0$.
- For all q and $i + 1 \in [1, |Q|]$,

$$\texttt{maxval}_q^{i+1} \stackrel{\text{def}}{=} \max(\texttt{maxval}_q^i,$$

$$\{\texttt{maxval}_{q'}^j + u \in \mathbb{N} \mid j \leq i,\ q \xrightarrow{u} q' \text{ is a transition}\}).$$

The values $\texttt{maxval}_q^i$'s can be computed in polynomial time in the size of V (the number $|Q|$ of locations being an essential parameter as well as the maximal absolute value $|\ u\ |$ from updates –integers being written in binary). One can show that $\texttt{maxval}_q^i$ is indeed the maximal value as specified above.
Further, note that condition (2) in Lemma 2 is equivalent to $\texttt{maxval}_{q_f}^{|Q|} \neq -\infty$. Similarly, the three conditions in (3) from Lemma 2 are equivalent to : there are $q \in Q$ and $I < J \leq |Q|$ such that

- $\texttt{maxval}_q^I \neq -\infty$.
- $\texttt{maxval}_q^I < \texttt{maxval}_q^J$ and $\texttt{auxval}_q^0 < \texttt{auxval}_q^{J-I}$, where the values $\texttt{auxval}_{q'}^i$'s ($i \in [0, J - I]$, $q' \in Q$) are defined as follows (similarly to what is done for the $\texttt{maxval}_{q'}^j$'s) :
  - $\texttt{auxval}_q^0 \stackrel{\text{def}}{=} \texttt{maxval}_q^I$ and $\texttt{auxval}_{q'}^0 \stackrel{\text{def}}{=} -\infty$ for all $q' \neq q$.



Witness runs and path for $(V, (q_0, 7), q_f)$:

initial run : $(q_0, 7), (q_2, 11), (q_3, 4)$

"> 0 loop" : $(q_3, 4), (q_5, 1), (q_4, 6), (q_3, 5)$

final path : $q_3 \to q_5 \to q_f$

FIGURE 2 – Witness runs and path from Lemma 2(3)

— For all q' and $i+1 \in [1, J-I]$,

$$\texttt{auxval}_{q'}^{i+1} \stackrel{\text{def}}{=}$$

$$\max(\texttt{auxval}_{q'}^{i},$$

$$\{\texttt{auxval}_{q''}^{j} + u \in \mathbb{N} \mid j \leq i,\ q' \xrightarrow{u} q'' \text{ is a transition}\}).$$

— There is a simple path starting at q and ending at q_f.

The first two points above can be checked in PTIME, and the third one in NLOGSPACE as it is an instance of the standard graph reachability problem GAP. So, CREACH(1-VASS) is in PTIME.

Note that the values $\texttt{auxval}_{q'}^{i}$'s in the proof of Theorem 2 are necessary to guarantee that the values $\texttt{maxval}_q^I$ and $\texttt{maxval}_q^J$ are obtained following a common subrun until reaching the configuration $(q, \texttt{maxval}_q^I)$ [2]. Now, let us turn to the characterisation of runs and paths witnessing nontermination.

Lemma 3 *Let V be a 1-VASS and (q_0, x_0) an initial configuration. There is an infinite run starting at (q_0, x_0) iff*

— *there is a simple run $(q_0, x_0), \ldots, (q_n, x_n)$; and*

— *there is a positive simple loop $t_1 \ldots t_k$ such that $(q_n, x_n) \xrightarrow{t_1 \ldots t_k} (q_n, x_n + \alpha)$ is a run ($\alpha \geq 0$).*

Proof Clearly, the satisfaction of the two conditions implies that there is an infinite run starting at (q_0, x_0) : just consider the run generated by $(t_1 \ldots t_k)^\omega$ from configuration (q_n, x_n). Let us prove the other direction, similarly to what is done in the proof of Lemma 2. Suppose that $\rho = (q_0, x_0), \ldots, (q_k, x_k), \ldots$ is an infinite run. Without loss of generality, we can assume that ρ has no simple loop with strictly negative effect. There are $n \geq 0$ and $q \in Q$ such that $q_n = q$, $\{i \in \mathbb{N} \mid q_i = q\}$ is infinite and $(q_0, x_0), \ldots, (q_n, x_n)$ is a simple run. Consider some $J > I \geq |Q|$ such that $q_J = q_I = q$ (such an index J necessarily exists). Obviously, there is a positive simple loop $t_I \ldots t_{J-1}$ such that $(q_I, x_I) \xrightarrow{t_I \ldots t_{J-1}} (q_J, x_J)$ is a run. Hence, both conditions in the statement of the lemma hold.

2. We remark that in the proof of [RY86, Theorem 3.4] for solving the boundedness problem for 1-VASS in PTIME, a similar argument should have been used.

Once more, the characterisation in Lemma 3 can be turned into an algorithm to check nontermination, running in polynomial time.

Theorem 3 *The problem NONTER(1-VASS) is in* PTIME.

Proof Let V be a 1-VASS and (q_0, x_0) an initial configuration. Define the values $\texttt{maxval}_q^i$ for $i \in [0, |Q|]$ and $q \in Q$ such that if there is a run $(q_0, x_0), \ldots, (q_i, x_i)$ with $q_i = q$, then the maximal value x_i among all these runs is precisely $\texttt{maxval}_q^i$. Note that these values are not the same as those from the proof of Theorem 2 as we consider runs of length *exactly* i. When there is no such run, by convention $\texttt{maxval}_q^i = -\infty$.

— $\texttt{maxval}_{q_0}^0 \stackrel{\text{def}}{=} x_0$ and $\texttt{maxval}_q^0 \stackrel{\text{def}}{=} -\infty$ for all $q \neq q_0$.

— For all q and $i+1 \in [1, |Q|]$,

$$\texttt{maxval}_q^{i+1} \stackrel{\text{def}}{=} \max(\{\texttt{maxval}_{q'}^{i} + u \in \mathbb{N} \mid$$

$$q \xrightarrow{u} q' \text{ is a transition},\ \texttt{maxval}_{q'}^{i} \neq -\infty\}).$$

By convention, the maximal value of the empty set is $-\infty$.

All the values $\texttt{maxval}_q^i$'s can be computed in polynomial time in the size of V. One can show that $\texttt{maxval}_q^i$ is the maximal value as specified above. Finally, the characterisation in Lemma 3 is equivalent to : there are $q \in Q$ and $I < J \leq |Q|$ such that $\texttt{maxval}_q^I \neq -\infty$ and $\texttt{maxval}_q^I \leq \texttt{maxval}_q^J$ and $\texttt{auxval}_q^0 \leq \texttt{auxval}_q^{J-I}$, where values $\texttt{auxval}_{q'}^{i}$'s ($i \in [0, J-I]$, $q' \in Q$) are defined as

— $\texttt{auxval}_q^0 \stackrel{\text{def}}{=} \texttt{maxval}_q^I$ and $\texttt{auxval}_{q'}^0 \stackrel{\text{def}}{=} -\infty$ for all $q' \neq q$;

— for all q' and $i+1 \in [1, J-I]$,

$$\texttt{auxval}_{q'}^{i+1} \stackrel{\text{def}}{=}$$

$$\max\{\texttt{auxval}_{q''}^{j} + u \in \mathbb{N} \mid j \leq i,\ q' \xrightarrow{u} q'' \text{ is a transition}\}.$$

All conditions can be checked in polynomial time and therefore the nontermination problem for 1-VASS is in PTIME.

To conclude, by Theorem 2 and 3 both the state-reachability and nontermination problems for 1-VASS are decidable in PTIME.

3.3 Model-checking $\mathsf{RB}\pm\mathsf{ATL}(\{1\},1)$ is in PTIME

In this section we establish our main theoretical result, that is, the model-checking problem for $\mathsf{RB}\pm\mathsf{ATL}(\{1\},1)$ is PTIME-complete (forthcoming Theorem 4) by leveraging on Theorem 2 and 3. Hereafter, for every $b \in \mathbb{N} \cup \{\omega\}$, we write $\langle\langle b \rangle\rangle \phi$ instead of $\langle\langle \{1\}^b \rangle\rangle \phi$. We observe that the case of a single resource can also capture situations in which $r > 1$ resources can be converted into a unique resource (e.g., money), possibly with different rates.

As done in Section 2.3, given a resource-bounded CGS $M = (\{1\}, S, Act, 1, act, cost, \delta, L)$ with a single agent and a single resource, let us define the 1-VASS $V_M = (S, 1, R_V)$ such that $q \xrightarrow{u} q' \in R_V$ iff there is some action $\mathtt{a} \in act(q,1)$ such that $\delta(q,\mathtt{a}) = q'$ and $cost(q,1,\mathtt{a}) = u$. Similarly, we write $K_M = (S, R, L_K)$ to denote the Kripke structure such that $q \; R \; q'$ iff there is some action $\mathtt{a} \in act(q,1)$ such that $\delta(q,\mathtt{a}) = q'$ and $L_K(q) = \{q\}$ (by a slight abuse of notations, we assume that $AP = Q$). Note that, thanks to the `idle` action, K_M is a total Kripke structure, i.e., every world has at least one successor. We introduce Kripke structures as the modality $\langle\langle \omega \rangle\rangle$ amounts to forget about the costs in M, and therefore M can be understood as the Kripke structure K_M, and model checking reduces to CTL model checking. Similarly, as remarked in Section 2.3, the strategy modality $\langle\langle \emptyset^b \rangle\rangle$ behaves as the universal path quantifier A in cost-free transition systems.

We now investigate the relationship between computations in M and runs in V_M and in K_M, respectively (a variant of Lemma 1).

Lemma 4 *Let M be a RB-CGS with a single agent and a single resource.*

- **(I)** *Let $q_0 \xrightarrow{\mathtt{a}_0} q_1 \xrightarrow{\mathtt{a}_1} q_2 \cdots$ be a b-consistent computation associated to the family of resource values $(v_i)_{i\in\mathbb{N}}$. If $b \in \mathbb{N}$, then $(q_0, v_0) \to (q_1, v_1) \to (q_2, v_2) \cdots$ is an infinite run in V_M ; otherwise $q_0 \to q_1 \to q_2 \cdots$ is an infinite path in K_M.*
- **(II)** *Let $(q_0, v_0) \to (q_1, v_1) \to (q_2, v_2) \cdots$ be an infinite run in V_M. Then, there is a v_0-consistent computation $q_0 \xrightarrow{\mathtt{a}_0} q_1 \xrightarrow{\mathtt{a}_1} q_2 \cdots$ associated to the family of resource values $(v_i)_{i\in\mathbb{N}}$.*
- **(III)** *Let $q_0 \to q_1 \to q_2 \cdots$ is an infinite path in K_M. Then, there is an ω-consistent computation $q_0 \xrightarrow{\mathtt{a}_0} q_1 \xrightarrow{\mathtt{a}_1} q_2 \cdots$ in M.*

The proof of Lemma 4 follows immediately by definition, and this result is instrumental to the three following lemmas that are at the heart of the model-checking algorithm for $\mathsf{RB}\pm\mathsf{ATL}(\{1\},1)$. Given $S_1 \subseteq S$, we write $V_M^{S_1}$ (resp. $K_M^{S_1}$) to denote the restriction of V_M (resp. K_M) to the locations in S_1 only.

Lemma 5 *Let M be an RB-CGS for $\mathsf{RB}\pm\mathsf{ATL}(\{1\},1)$, $S_1 \subseteq S$ with $s \in S_1$, and $b \in \mathbb{N}$.*

- **(I)** *There is a b-consistent computation starting at s in M that visits only states in S_1 iff $(V_M^{S_1}, (s,b))$ is a positive instance of NONTER(1-VASS).*
- **(II)** *There is an ω-consistent computation starting in s in M that visits only states in S_1 iff $(K_M, s) \models \mathsf{E}\,\mathsf{G}(\bigvee_{s'\in S_1} s')$ in CTL.*

This is a consequence of Lemma 4 (which will be generalised in Lemma 7). Let us focus now on the until operator U.

Lemma 6 *Let M be a RB-CGS for $\mathsf{RB}\pm\mathsf{ATL}(\{1\},1)$, $S_1, S_2 \subseteq S$ with $s \in S$, and $b \in \mathbb{N}$.*

- **(I)** *There is a b-consistent computation starting at s in M such that its projection on S is in $S_1^* \cdot S_2 \cdot S^\omega$ (understood as an ω-regular expression) iff for some $s' \in S_2$, $(V_M^{S_1\cup S_2}, (s,b), s')$ is a positive instance of CREACH(1-VASS).*
- **(II)** *There is an ω-consistent computation starting at s in M such that its projection on S is in $S_1^* \cdot S_2 \cdot S^\omega$ iff in CTL, we have*

$$(K_M, s) \models \mathsf{E}(\bigvee_{s'\in S_1} s')\,\mathsf{U}(\bigvee_{s'\in S_2} s').$$

This is again a consequence of Lemma 4 but here, we have to use the fact that the distinguished action `idle` is enabled in any state (which is handy to extend to the infinity a finite witness run). Finally, we consider the linear-time temporal operator R.

Lemma 7 *Let M be a RB-CGS for $\mathsf{RB}\pm\mathsf{ATL}(\{1\},1)$, $S_1, S_2 \subseteq S$ with $s \in S$, and $b \in \mathbb{N}$.*

- **(I)** *There is a b-consistent computation starting at s in M such that its projection on S is in $S_2^\omega \cup ((S\setminus S_1)\cap S_2)^* \cdot (S_1 \cap S_2) \cdot S^\omega$ iff either $(V_M^{S_2}, (s,b))$ is a positive instance of NONTER(1-VASS) or for some $s' \in S_1 \cap S_2$, $(V_M^{S_2}, (s,b), s')$ is a positive instance of CREACH(1-VASS).*
- **(II)** *There is an ω-consistent computation starting at s in M such that its projection on S is in $S_2^\omega \cup ((S \setminus S_1) \cap S_2)^* \cdot (S_1 \cap S_2) \cdot S^\omega$ iff in CTL, we have*

$$(K_M, s) \models (\mathsf{E}\,\mathsf{G} \bigvee_{s'\in S_2} s') \vee \mathsf{E}(\bigvee_{s'\in (S\setminus S_1)\cap S_2} s')\,\mathsf{U}(\bigvee_{s'\in S_1\cap S_2} s').$$

By using Lemmas 5-7 we derive our main theoretical result.

Theorem 4 *The model-checking problem for $\mathsf{RB}\pm\mathsf{ATL}(\{1\},1)$ is PTIME-complete.*

PTIME-hardness is inherited from the model-checking problem for CTL.

Algorithm 1 – RB±ATL($\{1\}, 1$) model checking –

1: **procedure** GMC(M, ϕ)
2: **case** ϕ **of**
3: p: **return** $\{s \in S \mid s \in L(p)\}$
4: $\neg\psi$: **return** $S \setminus GMC(M, \psi)$
5: $\psi_1 \wedge \psi_2$: **return** $GMC(M, \psi_1) \cap GMC(M, \psi_2)$
6: $\langle\langle b \rangle\rangle \mathsf{X}\, \psi$: **return** $\{s \mid \exists\, \mathtt{a} \in act(s, 1),\ 0 \leq b + cost(s, 1, \mathtt{a}),\ \delta(s, \mathtt{a}) \in GMC(M, \psi)\}$
7: $\langle\langle \omega \rangle\rangle \mathsf{X}\, \psi$:
return $\{s \mid \exists\, \mathtt{a} \in act(s, 1),\ \delta(s, \mathtt{a}) \in GMC(M, \psi)\}$
8: $\langle\langle \emptyset^b \rangle\rangle \mathsf{X}\, \psi$:
return $\{s \mid \forall\, \mathtt{a} \in act(s, 1),\ \delta(s, \mathtt{a}) \in GMC(M, \psi)\}$
9: $\langle\langle b \rangle\rangle \psi_1 \mathsf{U}\, \psi_2$: $S_1 := GMC(M, \psi_1)$; $S_2 := GMC(M, \psi_2)$;
return $\{s \in S \mid \exists\, s' \in S_2$ s.t. $V_M^{S_1 \cup S_2}, (s, b), s'$ is a positive inst. of CREACH(1-VASS) $\}$
10: $\langle\langle \omega \rangle\rangle \psi_1 \mathsf{U}\, \psi_2$: $S_1 := GMC(M, \psi_1)$; $S_2 := GMC(M, \psi_2)$;
return $\{s \in S \mid K_M, s \models \mathsf{E}(\bigvee_{s' \in S_1} s')\, \mathsf{U}(\bigvee_{s' \in S_2} s')\}$
11: $\langle\langle \emptyset^b \rangle\rangle \psi_1 \mathsf{U}\, \psi_2$: $S_1 := GMC(M, \psi_1)$; $S_2 := GMC(M, \psi_2)$;
return $\{s \in S \mid K_M, s \models \mathsf{A}(\bigvee_{s' \in S_1} s')\, \mathsf{U}(\bigvee_{s' \in S_2} s')\}$
12: $\langle\langle b \rangle\rangle \psi_1 \mathsf{R}\, \psi_2$: $S_1 := GMC(M, \psi_1)$; $S_2 := GMC(M, \psi_2)$;
return $\{s \in S \mid V_M^{S_2}, (s, b)$ is a positive inst. of NONTER(1-VASS) $\} \cup \{s \in S \mid \exists\, s' \in S_1 \cap S_2$ s.t. $V_M^{S_2}, (s, b), s'$ is a positive inst. of CREACH(1-VASS) $\}$
13: $\langle\langle \omega \rangle\rangle \psi_1 \mathsf{R}\, \psi_2$: $S_1 := GMC(M, \psi_1)$; $S_2 := GMC(M, \psi_2)$;
return $\{s \in S \mid K_M, s \models \mathsf{E}(\bigvee_{s' \in S_1} s')\, \mathsf{R}(\bigvee_{s' \in S_2} s')\}$
14: $\langle\langle \emptyset^b \rangle\rangle \psi_1 \mathsf{R}\, \psi_2$: $S_1 := GMC(M, \psi_1)$; $S_2 := GMC(M, \psi_2)$;
return $\{s \in S \mid K_M, s \models \mathsf{A}(\bigvee_{s' \in S_1} s')\, \mathsf{R}(\bigvee_{s' \in S_2} s')\}$
15: **end case**
16: **end procedure**

Proof Let $M = (\{1\}, S, Act, 1, act, cost, \delta, L)$ be a resource-bounded CGS, and ϕ be a formula in RB±ATL($\{1\}, 1$). Let us present Algorithm 1, a polynomial-time algorithm that computes the finite set $\{s \in S \mid (M, s) \models \phi\}$ (by default, $b \in \mathbb{N}$)[3].
By induction, one can show that $(M, s) \models \phi$ iff $s \in GMC(M, \phi)$. Lemmas 5-7 are used to prove the soundness of the subroutines for U and R, with $b \in \mathbb{N} \cup \{\omega\}$, respectively. When the strategy modality is $\langle\langle \emptyset^b \rangle\rangle$, it behaves as the standard path quantifier A, which is reflected in Algorithm 1. As far as computational complexity is concerned, $GMC(M, \phi)$ is computed with a recursion depth linear in the size of ϕ and the control-state reachability and nontermination problems can be solved in polynomial time by Theorem 2 and 3. More precisely, for each occurrence of a subformula ψ of ϕ, $GMC(M, \psi)$ can be computed only once, which guarantees the overall number of calls of the form $GMC(M, \psi)$: it is sufficient to take advantage of dynamic programming and to work with a table to remember the values $GMC(M, \psi)$ already computed (omitted in the present algorithm). It is also worth observing that the instances we consider are polynomial in the sizes of M and ϕ. Finally, we take advantage of the fact that the model-checking problem for CTL including R remains in PTIME (see, e.g., [DGL16, Chapter 7]).

Consequently, reasoning about a single resource in the Computation Tree Logic CTL comes at no extra computational cost. Hence, in principle we can verify specification such as formula (1) in Example 1 efficiently. Based on the correspondences established in Theorem 1, we immediately derive the following consequence.

Corollary 5 *The model-checking problem for RBTL restricted to a single resource is* PTIME*-complete.*

4 Concluding Remarks

We investigated the complexity of the model-checking problem for Resource-bounded Alternating-time Temporal Logics. In particular, we established that RBTL* and RB±ATL$^*(\{1\}, r)$ can be understood as slight variants of the same logic. More importantly, we provided a unified view of the model-checking problems for RB±ATL, then proved that model checking RB±ATL($\{1\}, 1$) is PTIME-complete. To do so, we designed original algorithms to solve the control-state reachability and nontermination problems for 1-VASS. Hence, as far as worst-case complexity is concerned, the model-checking problems for CTL and RB±ATL($\{1\}, 1$) behave similarly.
The paper has not touched very much on the model-checking problem for RB±ATL*, for which the main results are summarised in Table 2. Unlike RB±ATL, tight complexity bounds are known for all variations on the number of agents and resources. For at least two agents, the

3. We omit the case for the operator G as G ϕ is logically equivalent to $\bot$ R ϕ.

$r \backslash \lvert Ag \rvert$	∞	**2**	**1**
∞	in 2EXPTIME [ABDL18, Th. 7]		EXPSPACE-c. [ABDL18, Th. 8]
$\geq \mathbf{1}$	2EXPTIME-h. (from ATL^*)		in PSPACE ([ABDL18, Cor. 2] & Th. 1) PSPACE-h. (from CTL^*)

TABLE 2 – The complexity of model checking $\mathsf{RB}{\pm}\mathsf{ATL}^*(Ag, r)$.

model-checking problem is 2EXPTIME-complete. The upper bound comes from [ABDL18, Th. 7], whereas the lower bound follows from the 2EXPTIME-hardness of ATL^*, which is proved by using two agents only [AHK02]. On the other hand, the problem restricted to a single agent becomes EXPSPACE-complete for an unbounded number of resources [ABDL18, Th. 8] ; while for a bounded number r, model checking $\mathsf{RB}{\pm}\mathsf{ATL}^*(\{1\}, r)$ is PSPACE-complete : the lower bound follows immediately from the PSPACE-hardness of the model-checking problem for CTL^* ; as for the upper bound, we derive it from the fact that model checking RBTL^* is in PSPACE [ABDL18, Cor. 2] and Theorem 1.

As far as future work is concerned, we plan to implement the PTIME algorithm for model checking $\mathsf{RB}{\pm}\mathsf{ATL}(\{1\}, 1)$, possibly taking advantage of the very recent results in [ACP+19], and to investigate the complexity of other meaningful fragments of $\mathsf{RB}{\pm}\mathsf{ATL}(Ag, r)$ for which tight bounds are unknown. The synthesis of parameters for the parameterised version of $\mathsf{RB}{\pm}\mathsf{ATL}(\{1\}, 1)$ (as well as for other fragments of $\mathsf{RB}{\pm}\mathsf{ATL}^*(Ag, r)$) is also worth further investigation.

Acknowledgment. We would like to thank Michael Blondin (University of Sherbrooke) for pointing us to [RY86] and for insightful feedback, as well as the anonymous referees for their suggestions and comments. F. Belardinelli acknowledges the support of the ANR JCJC Project SVeDaS (ANR-16-CE40- 0021).

Références

[ABDL18] N. Alechina, N. Bulling, S. Demri, and B. Logan. On the complexity of resource-bounded logics. *Theoretical Computer Science*, 750 :69–100, 2018.

[ABLN15] N. Alechina, N. Bulling, B. Logan, and H.N. Nguyen. On the boundary of (un)decidability : Decidable model-checking for a fragment of resource agent logic. In *IJCAI'15*, pages 1494–1501. AAAI Press, 2015.

[ABLN17] N. Alechina, N. Bulling, B. Logan, and H.N. Nguyen. The virtues of idleness : A decidable fragment of resource agent logic. *Artificial Intelligence*, 245 :56–85, 2017.

[ACP+19] S. Almagor, N. Cohen, G. Pérez, M. Shirmohammadi, and J. Worrell. Coverability in 1-VASS with disequality tests. http ://arXiv :1902.06576, February 2019.

[AdAG+01] R. Alur, L. de Alfaro, R. Grosu, T. Henzinger, A. Thomas, M. Kang, C. Kirsch, R. Majumdar F. Mang, and B-Y. Wang. jMocha : A model checking tool that exploits design structure. In *Proceedings of the 23rd International Conference on Software Engineering (ICSE01)*, pages 835–836. IEEE, 2001.

[AH09] M. Faouzi Atig and P. Habermehl. On Yen's path logic for Petri nets. In *RP'09*, volume 5797 of *Lecture Notes in Computer Science*, pages 51–63. Springer, 2009.

[AHK02] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5) :672–713, 2002.

[AL18] N. Alechina and B. Logan. Resource logics with a diminishing resource. In *AAMAS'18*, pages 1847–1849. International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, USA / ACM, 2018.

[ALN+15] N. Alechina, B. Logan, H.N. Nguyen, F. Raimondi, and L. Mostarda. Symbolic model-checking for resource-bounded atl. In *AAMAS'15*, pages 1809–1810. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

[ALNR14] N. Alechina, B. Logan, H.N. Nguyen, and F. Raimondi. Decidable model-checking for a resource logic with production of resources. In *ECAI'14*, pages 9–14, 2014.

[ALNR17] N. Alechina, B. Logan, H.N. Nguyen, and F. Raimondi. Model-checking for resource-bounded ATL with production and consumption of resources. *Journal of Computer and System Sciences*, 88 :126–144, 2017.

[BF09] N. Bulling and B. Farwer. Expressing properties of resource-bounded systems : The logics RBTL* and RBTL. In *CLIMA X*, volume 6214 of *Lecture Notes in Computer Science*, pages 22–45. Springer, 2009.

[BF10] N. Bulling and B. Farwer. On the (Un-)Decidability of Model-Checking Resource-Bounded Agents. In *ECAI'10*, pages 567–572, 2010.

[BFG+15] M. Blondin, A. Finkel, S. Göller, C. Haase, and P. McKenzie. Reachability in two-dimensional vector addition systems with states is PSPACE-complete. In *LICS'15*, pages 32–43. ACM Press, 2015.

[BS11] M. Blockelet and S. Schmitz. Model-checking coverability graphs of vector addition systems. In *MFCS'11*, volume 6907 of *Lecture Notes in Computer Science*, pages 108–119. Springer, 2011.

[CHP07] K. Chatterjee, T. Henzinger, and N. Piterman. Strategy logic. In *CONCUR'07*, volume 4703 of *Lecture Notes in Computer Science*, pages 59–73. Springer, 2007.

[CLL+18] W. Czerwinski, S. Lasota, R. Lazić, J. Leroux, and F. Mazowiecki. The Reachability Problem for Petri Nets is Not Elementary (extended abstract). *CoRR*, abs/1809.07115, 2018. To appear in STOC'19.

[CLMM14] P. Cermák, A. Lomuscio, F. Mogavero, and A. Murano. MCMAS-SLK : A model checker for the verification of strategy logic specifications. In *CAV'14*, volume 8559 of *Lecture Notes in Computer Science*, pages 525–532. Springer, 2014.

[CS14] J.B. Courtois and S. Schmitz. Alternating vector addition systems with states. In *MFCS'14*, volume 8634 of *Lecture Notes in Computer Science*, pages 220–231. Springer, 2014.

[Dem13] S. Demri. On selective unboundedness of VASS. *Journal of Computer and System Sciences*, 79(5) :689–713, 2013.

[DGL16] S. Demri, V. Goranko, and M. Lange. *Temporal Logics in Computer Science*. Cambridge University Press, 2016.

[FLL+17] D. Figueira, R. Lazić, J. Leroux, F. Mazowiecki, and G. Sutre. Polynomial-space completeness of reachability for succinct branching VASS in dimension one. In *ICALP'17*, volume 80 of *LIPIcs*, pages 119 :1–119 :14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[GHLT16] St. Göller, Ch. Haase, R. Lazic, and P. Totzke. A polynomial-time algorithm for reachability in branching VASS in dimension one. In *ICALP'16*, volume 55 of *LIPIcs*, pages 105 :1–105 :13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[JS07] P. Jančar and Z. Sawa. A note on emptiness for alternating finite automata with a one-letter alphabet. *Information Processing Letters*, 104(5) :164–167, 2007.

[KM69] Richard M. Karp and Raymond E. Miller. Parallel program schemata. *Journal of Computer and System Sciences*, 3(2) :147 – 195, 1969.

[KNN+08] M. Kacprzak, W. Nabialek, A. Niewiadomski, W. Penczek, A. Pólrola, M. Szreter, B. Woźna, and A. Zbrzezny. Verics 2007 - a model checker for knowledge and real-time. *Fundamenta Informaticae*, 85(1) :313–328, 2008.

[Lip76] R.J. Lipton. The reachability problem requires exponential space. Technical Report 62, Department of Computer Science, Yale University, 1976.

[LM15] F. Laroussinie and N. Markey. Augmenting atl with strategy contexts. *Information and Computation*, (245) :98–123, 2015.

[LMO08] F. Laroussinie, N. Markey, and G. Oreiby. On the expressiveness and complexity of ATL. *Logical Methods in Computer Science*, 4(2), 2008.

[LQR15] A. Lomuscio, H. Qu, and F. Raimondi. MCMAS : A model checker for the verification of multi-agent systems. *Software Tools for Technology Transfer*, 2015. http ://dx.doi.org/10.1007/s10009-015-0378-x.

[MMPV14] F. Mogavero, A. Murano, G. Perelli, and M.Y. Vardi. Reasoning about strategies : On the model-checking problem. *ACM Transactions on Computational Logic*, 15(4) :34 :1–34 :47, 2014.

[Pau02] M. Pauly. A modal logic for coalitional power in games. *Journal of Logic and Computation*, 12(1) :149–166, 2002.

[POY04] D.J. Pym, P.W. O'Hearn, and H. Yang. Possible worlds and resources : the semantics of BI. *Theoretical Computer Science*, 315(1) :257–305, 2004.

[Rac78] C. Rackoff. The covering and boundedness problems for vector addition systems. *Theoretical Computer Science*, 6(2) :223–231, 1978.

[RY86] L. Rosier and H.-C. Yen. A multiparameter analysis of the boundedness problem for vector addition systems. *Journal of Computer and System Sciences*, 32 :105–135, 1986.

[Sch03] Ph. Schnoebelen. The complexity of temporal logic model checking. In *AIML'02*, pages 437–459. King's College Publication, 2003.

[Sch16] S. Schmitz. The complexity of reachability in vector addition systems. *SIGLOG News*, 3(1) :4–21, 2016.

Modélisation des stratégies de génération de choix variables chez la souris

Marwen Belkaid[1†] Jérémie Naudé[2] Philippe Faure[2*] Olivier Sigaud[1*]

[1] Sorbonne Université, CNRS, ISIR, 75005 Paris, France
[2] Sorbonne Université, INSERM, CNRS, NPS - IBPS, 75005 Paris, France
[*] Contribution égale
[†] Auteur correspondant : marwen.belkaid@upmc.fr

Résumé

Mieux comprendre les processus cognitifs impliqués dans la prise de décision et les comportements adaptatifs est fondamental pour la communauté de l'intelligence artificielle. Dans cet article, nous étudions les stratégies cognitives permettant la génération de choix aléatoires chez la souris. Nous utilisons un algorithme d'apprentissage par renforcement (RL) pour modéliser leur comportement dans une tâche qui récompense des séquences de choix non-répétitives. Nos résultats montrent que les souris sont capables de résoudre la tâche et suggérent qu'elles le font en ajustant les paramètres de leur processus de décision.

Mots Clef

RL, exploration aléatoire, variabilité des choix.

Abstract

Better understanding the cognitive processes involved in decision-making and adaptive behaviors is fundamental for the artificial intelligence community. In this paper, we study the cognitive strategies allowing the generation of random choices by mice. We use a reinforcement learning (RL) algorithm to model the animals' behavior in a task rewarding non-repetitive choice sequences. Our results show that mice can solve the task and suggest they do so by adjusting the parameters of their decision-making process.

Keywords

RL, random exploration, choice variability.

1 Introduction

L'apprentissage par renforcement est un excellent exemple de brassage fructueux entre intelligence artificielle et neurosciences. En effet, les algorithmes de différences temporelles ont été inspirés par le comportement animal dans l'étude de conditionnement [1]. Au cours des deux dernières décennies, ces algorithmes ont à leur tour fourni un cadre formel fondamental pour l'analyse des données comportementales et neurales en neuroscience [2, 3]. Ce cadre décrit bien les mécanismes par lesquels les individus exploitent leur connaissance de l'environnement et répètent des actions qui maximisent les récompenses. Cependant, les principes sous-jacents à la génération de variabilité dans le processus de prise de décision sont encore mal compris. Dans le cadre de l'apprentissage par renforcement, la variabilité des choix est associée au processus d'exploration. L'exploration "dirigée" vise à recueillir des informations sur les contingences environnementales, tandis que l'exploration aléatoire introduit une variabilité indépendamment de celles-ci. Des études ont montré que les animaux sont en mesure d'accroître la variabilité de leurs choix, en particulier lorsque les règles changent [3] ou qu'ils doivent tromper des prédictions faites sur leurs décisions [4, 5]. Cependant, l'utilisation systématique de contingences probabilistes ou d'environnements volatils rend difficile d'isoler expérimentalement une génération intrinsèque de variabilité des conditions environnementales.

Dans cette étude, nous utilisons une nouvelle expérience sur souris et modélisons le comportement des animaux avec un modèle RL pour tester l'hypothèse selon laquelle les animaux peuvent ajuster de manière adaptative le caractère aléatoire de leur comportement.

2 Expériences sur souris

Nous avons entraîné des souris sur une tâche nécessitant la génération d'une séquence complexe de choix [6]. Dans une arène ouverte, les souris pouvaient recevoir une récompense dans trois zones cibles. Ne pouvant pas recevoir deux récompenses successives au même endroit, les animaux devaient réaliser une série de choix en choisissant la prochaine cible parmi les deux alternatives restantes. L'attribution de la récompense dépendait de la complexité de la séquence. Plus précisément, nous avons estimé la mesure de complexité LZ [7] de sous-séquences de choix de taille 10 (9 choix passés + choix suivant) à chaque essai. Malgré sa difficulté, cette tâche est totalement déterministe, ce qui la différencie des autres approches [4, 5]. Nous avons pu déterminer que 25% des séquence de longueur 10 n'étaient pas récompensés au dernier choix. Donc, théoriquement, si une estimation correcte de la complexité des séquences conduit à un taux de réussite de 100%, une sélection aléatoire pure a un taux de réussite de 75%, et une séquence répétitive (e.g. ABCABC...) conduit à un très faible taux de réussite.

Nous avons constaté que les souris augmentaient progressi-

Plages des hyperparamètres pour la recherche aléatoire

Label	Plage	Pas	Description
m	$[0, 9]$	1	Taille de la mémoire
τ	$[1, 20]$	continue	Température du Softmax
κ	$[0, 1]$	continue	Coût du demi-tour

Plages des hyperparamètres pour la recherche aléatoire en grille

Label	Plage	Pas	Description
m	$[0, 9]$	1	Taille de la mémoire (pas d'ambigüité)
	$[0, 7]$	1	Taille de la mémoire (ambigüité faible)
	$[0, 5]$	1	Taille de la mémoire (ambigüité moyenne)
α	$2^{-i}, i \in [0, 10]$	1	Taux d'apprentissage
τ	$2^{i}, i \in [-4, 4]$	1	Température du Softmax
κ	$[0.5, 0.95]$	0.05	Coût du demi-tour

TABLE 1 – Plages des valeurs utilisées pour l'optimisation des hyperparamètres

vement la variabilité de leurs séquences de choix [6]. Cela a été démontré par l'augmentation corrélée au fil des sessions de la complexité des séquences et du taux de réussite qui en résulte, ainsi que l'augmentation du taux de demi-tours (i.e. changement de direction) dans les séquences.

3 Méthodes

3.1 Modèle d'apprentissage par renforcement

Nous avons représenté la tâche par un processus de décision markovien (MDP) avec trois états $s \in \{\text{A, B, C}\}$ et trois actions $a \in \{\text{GoToA, GoToB, GoToC}\}$, correspondant respectivement aux emplacements où la récompense est attribuée et aux transitions entre ces emplacements. Les valeurs d'état-action $Q(s, a)$ sont apprises à l'aide de la règle de Rescorla-Wagner [8] :

$$\Delta Q(\mathbf{s_t}, a_t) = \alpha(\mathcal{U}_{t+1} - Q(\mathbf{s_t}, a_t)) \tag{1}$$

où $\mathbf{s_t} = [s_t, s_{t-1}, ..., s_{tm}]$ est l'état en cours qui peut inclure la mémoire jusqu' au m^{ime} emplacement passé, a_t l'action en cours, α la vitesse d'apprentissage et $\mathcal{U}$ la fonction utilitaire définie comme suit :

$$\mathcal{U}_{t+1} = \begin{cases} (1-\kappa).r_{t+1} & \text{si } \; s_{t+1} = s_{t-1} \\ r_{t+1} & \text{sinon} \end{cases} \tag{2}$$

où r est la fonction de récompense et κ le paramètre de coût de demi-tour modélisant le coût moteur ou tout biais en défaveur de l'action menant l'animal à son emplacement précédent. Le coût de demi-tour était nécessaire pour reproduire les trajectoires stéréotypées observées chez la souris à la fin de la phase d'entraînement [6].
La sélection de l'action a été effectuée à l'aide d'une règle softmax, ce qui signifie que, dans l'état $\mathbf{s_t}$, l'action a_t est sélectionnée avec la probabilité :

$$P(a_t|\mathbf{s_t}) = \frac{e^{Q(\mathbf{s_t}, a_t)/\tau}}{\sum_a e^{Q(\mathbf{s_t}, a)/\tau}} \tag{3}$$

où τ est le paramètre de température. Ce paramètre réduit la sensibilité à la différence dans les valeurs d'actions, augmentant ainsi la quantité de bruit ou le caractère aléatoire de la prise de décision. Le coût du demi-tour κ a l'effet inverse puisqu'il représente un biais comportemental qui limite le caractère aléatoire des choix. Nous appelons *randomness* l'hyperparamètre défini comme $\rho = \tau/\kappa$.
Une version standard d'apprentissage par renforcement revient à n'inclure aucune mémoire des emplacements précédents et à utiliser un coût du demi-tour nul.
Nous avons également utilisé des variantes du modèle pour manipuler l'ambiguïté de la représentation d'états, i.e. si chacun des emplacements A, B, C pouvait être représenté par $n \geq 1$ états. Pour simplifier, nous avons utilisé trois niveaux d'ambiguïté avec $n = 1$, 2 et 3 pour tous les emplacements. Cela nous a permis de présenter une preuve de l'impact potentiel de l'utilisation d'une représentation d'état parfaite dans notre modèle.

3.2 Optimisation des hyperparamètres

Les principaux résultats d'ajustement de modèle (*model fitting*) présentés dans cet article ont été obtenus en optimisant les hyperparamètres vis-à-vis du comportement des souris session par session indépendamment. Ce processus visait à déterminer quelles valeurs des deux hyperparamètres m et $\rho = \tau/\kappa$ font que le modèle se comporte comme une souris en termes de taux de réussite (pourcentage d'actions récompensées) et de complexité (variabilité des décisions). Notre objectif principal était de distinguer deux stratégies : répéter des séquences récompensées ou choisir au hasard. Par conséquent, nous avons momentanément laissé de côté la question de la vitesse d'apprentissage et n'avons considéré que le comportement du modèle après la convergence. Le taux d'apprentissage α a ici été fixé à 0.1.
Les hyperparamètres ont été sélectionnés par recherche aléatoire (*random search*) [9] (voir les plages répertoriées dans le tableau 1). Le modèle a été exécuté pour 2.10^6 itérations pour chaque jeu de paramètres. Le score de *fitness*

relatif aux données moyennes des souris à chaque session a été calculé comme suit :

$$fitness = 1 - \frac{1}{2}(|\widehat{S} - \overline{S}| + |\widehat{C} - \overline{C}|) \quad (4)$$

où $\overline{S}$ et $\overline{C}$ sont respectivement le taux de réussite moyen et la complexité moyenne chez la souris et $\widehat{S}$ et $\widehat{C}$ le taux de réussite et la complexité du modèle - tous les quatre $\in [0, 1]$. Les simulations étaient suffisamment longues pour que l'apprentissage converge. Ainsi, au lieu de plusieurs exécutions pour chaque jeu de paramètres, ce qui aurait été coûteux en calcul, $\widehat{S}$ et $\widehat{C}$ ont été moyennés sur les 10 dernières sessions simulées. Nous avons considéré qu'une session simulée = 200 itérations, ce qui représente la limite supérieure du nombre d'essais réalisés par les souris au cours d'une session réelle.

Étant donné que les souris étaient systématiquement récompensées au cours de l'entraînement, leur taux de réussite dans cette condition n'était pas indicatif (i.e. toujours 100%). Ainsi, pour évaluer la capacité du modèle à reproduire des trajectoires stéréotypées circulaires lors de la dernière séance d'entraînement, nous avons remplacé $\overline{S}$ et $\widehat{S}$ dans l'équation (5) par $\overline{U}$ et $\widehat{U}$ représentant les taux moyens de demi-tour pour le modèle et les souris, respectivement.

Des simulations supplémentaires ont été réalisées avec deux objectifs : 1) tester si un seul jeu de paramètres pouvait reproduire le comportement de la souris sans qu'il ne soit nécessaire de modifier les valeurs des paramètres au cours des sessions, 2) tester l'influence de l'ambiguïté de la représentation d'état sur l'utilisation de la mémoire dans le modèle de calcul. Ces nouvelles simulations visaient à reproduire le comportement de la souris au fil des sessions, de la phase d'entraînement à la condition de complexité. Par conséquent, le taux d'apprentissage α a été optimisé en plus des hyperparamètres m et ρ susmentionnés (voir les plages répertoriées dans le tableau 1). Chaque jeu de paramètres a été testé sur 20 exécutions différentes. Chaque série est une simulation de 4000 itérations, ce qui correspond à 10 séances d'entraînement et 10 séances de complexité, car les séances simulées consistent en 200 itérations. Le score de *fitness* a été calculé comme le score moyen de la dernière séance d'entraînement et des 10 séances de complexité utilisant les équations (4) et (5). L'utilisation d'une recherche en grille (*grid search*) garantissait des valeurs comparables pour les trois niveaux d'ambiguïté. Vu le coût de calcul supplémentaire induit par des niveaux d'ambiguïté plus élevés, nous avons progressivement diminué la limite supérieure de la plage de taille de la mémoire afin d'éviter des calculs longs et inutiles dans des régions sans intérêt de l'espace de recherche.

4 Résultats

En définissant des états en tant que vecteurs incluant l'historique des localisations précédentes au lieu de la seule localisation actuelle, nous avons pu faire varier la taille de la mémoire des agents simulés et obtenir différentes solutions du modèle en conséquence. Nous avons constaté qu'en l'absence de mémoire (i.e. état = emplacement actuel), le modèle avait appris des valeurs égales pour les deux cibles dans presque tous les états. En revanche, et en accord avec les prédictions théoriques pour l'apprentissage par renforcement, avec l'historique des neuf derniers choix stockés en mémoire, le modèle était capable d'apprendre à enchainer des choix récompensés (100% succès).

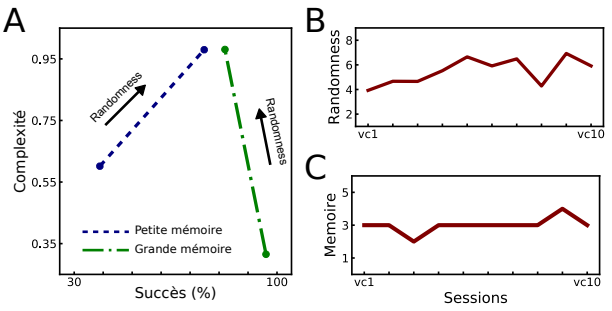


FIGURE 1 – A) Deux régimes de fonctionnement du modèle. **B)** et **C)** Valeurs optimisées des paramètres ρ et m respectivement.

Par ailleurs, la tendance à effectuer des choix aléatoires dépendait non seulement des valeurs associées aux choix actuels, mais également de la température du softmax. Cet hyperparamètre a eu des effets opposés sur le comportement du modèle selon que la taille de mémoire utilisée soit petite ou grande. Tandis que l'augmentation de la température augmentait toujours la complexité des séquences de choix, elle augmentait également le taux de réussite pour les petites tailles de mémoire, mais le diminuait pour les plus grandes mémoires (Fig 1A). Cela indique que le modèle peut trouver la solution optimale de la tâche si une mémoire de grande taille est utilisée, alors qu'une petite mémoire nécessite un niveau d'aléatoire élévé. Une limite entre les deux régimes a été trouvée entre des tailles de mémoire 3 et 4.

En optimisant le modèle vis-à-vis du comportement de la souris, nous avons constaté que l'amélioration de leurs performances au fil des sessions s'expliquait mieux par une augmentation du facteur aléatoire (paramètre *randomness*) en utilisant une petite mémoire (voir Fig 1B et 1C) – Notons que la baisse à la 8$^{\text{ième}}$ session provient des données expérimentales. Ce modèle a permis de mieux expliquer les données que l'utilisation d'un jeu de paramètres fixe tout au long des sessions (Facteur de Bayes = 3,46).

Le modèle avec une mémoire de taille 3 reproduisait au mieux le comportement de la souris, mais à peine mieux que les versions avec des mémoires plus petites. Du point de vue computationnel, une explication possible du fait que, bien qu'elle soit théoriquement suffisante, une mémoire de taille 1 obtient des résultats inférieurs à la taille 3, c'est que la représentation d'état est trop simplifiée dans le modèle. Ainsi, avec seulement trois états représentant parfaitement et sans ambiguïté chacune des cibles, l'algorithme ne peut pas prendre en compte le bruit comportemental, les erreurs et/ou les biais de la souris. Par consé-

quent, modifier la représentation de l'état du modèle pour le rendre plus réaliste devrait réduire la taille de la mémoire nécessaire pour reproduire les performances de la souris. Pour tester cette hypothèse, nous avons utilisé une variante du modèle dans laquelle nous avons manipulé l'ambiguïté de la représentation d'états : chacun des emplacements A, B, C pourrait être représenté par $n \geq 1$ états, avec $n = 1$ correspondant à des états non ambigus. Comme attendu, le modèle avait besoin d'une mémoire plus petite au fur et à mesure que l'ambiguïté de la représentation augmentait. Nous avons également constaté que le taux d'apprentissage le plus adapté était plus élevé avec des représentations ambiguës, tandis que le facteur aléatoire *randomness* demeurait inchangé quel que soit le niveau d'ambiguïté. Cela corrobore le fait que l'utilisation d'une capacité de mémoire supplémentaire par le modèle est due aux propres limites du modèle plutôt qu'à la nécessité réelle de mémoriser les choix précédents. Par conséquent, cette analyse computationnelle suggère que les souris ont adapté le paramètre *randomness* de leur système de prise de décision afin d'obtenir plus de variabilité au cours des sessions plutôt que de se souvenir des séquences de choix récompensées. Cette conclusion a été renforcée par une série d'arguments comportementaux confirmant l'absence de mémorisation de l'historique des choix dans leur stratégie [6].

5 Discussion

L'exploration et la variabilité des choix sont généralement étudiées en utilisant des tâches avec environnements stochastiques et/ou volatils. Nos résultats permettent d'avancer dans la compréhension des processus sous-jacents à la génération de la variabilité indépendamment des conditions environnementales. Confrontées à une tâche déterministe qui privilégie les séquences de choix complexes, les souris ont évité les répétitions en se rapprochant d'une sélection aléatoire.

Savoir si et comment des séquences aléatoires pourraient être générées par le cerveau a toujours été une énigme. Une première hypothèse est que, chez l'homme, le processus exploiterait la mémoire pour assurer l'égalité d'utilisation des réponses. Une seconde hypothèse suggère que le manque de mémoire pourrait aider à éliminer les biais contre-productifs. Nos résultats plaident en faveur de ce dernier point de vue : les souris n'ont pas utilisé leur mémoire, mais ont plutôt adapté leurs paramètres de prise de décision afin de maximiser le caractère aléatoire des choix.

Notons toutefois que dans notre modèle, la mémoire sert à retenir les séquences réalisées dans un horizon limité dans le but d'apprendre les associations état-action sur la base de ces représentations. D'autres modèles pourraient faire intervenir la mémoire d'une autre manière et expliquer différemment le processus d'exploration ou de génération de variabilité observé dans nos expériences. Dans de futurs travaux, nous testerons donc d'autres modèles afin d'évaluer la capacité d'une stratégie purement déterministe à générer des comportements proches de l'aléatoire et le coût computationnel minimal d'une solution de ce type.

Remerciements

Nous aimerions remercier Elise Bousseyrol, Romain Durand-de Cuttoli, Malou Dongelmans, Etienne K. Duranté, Tarek Ahmed Yahia, Steve Didienne, Bernadette Hanesse and Maxime Come qui ont réalisé les tâches expérimentales. Ce travail a été financé par le Centre National de la Recherche Scientifique CNRS UMR 8246, la Fondation pour la recherche médicale (FRM, Equipe FRM DEQ2013326488 pour P.F), l'Institut national du cancer (Grant TABAC-16-022 pour P.F.) et le Labex SMART (ANR-11-LABX-65) soutenu par les fonds du programme d'Investissements d'Avenir (ANR-11-IDEX-0004-02). Le laboratoire de P.F. et J.N. fait partie du réseau RTRA de l'École des neurosciences de Paris Île-de-France. P.F. et J.N. sont membres du LabEx Bio-Psy.

Références

[1] Richard S Sutton and Andrew G Barto. Toward a modern theory of adaptive networks : expectation and prediction. *Psychological review*, 88(2) :135, 1981.

[2] Wolfram Schultz. Getting formal with dopamine and reward. *Neuron*, 36(2) :241–263, 2002.

[3] Nathaniel D Daw, John P O'doherty, Peter Dayan, Ben Seymour, and Raymond J Dolan. Cortical substrates for exploratory decisions in humans. *Nature*, 441(7095) :876, 2006.

[4] Daeyeol Lee, Michelle L Conroy, Benjamin P McGreevy, and Dominic J Barraclough. Reinforcement learning and decision making in monkeys during a competitive game. *Cognitive Brain Research*, 22(1) :45–58, 2004.

[5] Dougal GR Tervo, Mikhail Proskurin, Maxim Manakov, Mayank Kabra, Alison Vollmer, Kristin Branson, and Alla Y Karpova. Behavioral variability through stochastic choice and its gating by anterior cingulate cortex. *Cell*, 159(1) :21–32, 2014.

[6] Marwen Belkaid, Elise Bousseyrol, Romain Durand-de Cuttoli, Malou Dongelmans, Etienne K Duranté, Tarek Ahmed Yahia, Steve Didienne, Bernadette Hannesse, Maxime Come, Alexandre Mourot, Jérémie Naudé, Olivier Sigaud, and Philippe Faure. Mice adaptively generate choice variability in a deterministic task. *BioRxiv*, 2019.

[7] Abraham Lempel and Jacob Ziv. On the complexity of finite sequences. *IEEE Transactions on information theory*, 22(1) :75–81, 1976.

[8] Robert A Rescorla and Allan R Wagner. A theory of pavlovian conditioning : The effectiveness of reinforcement and non-reinforcement. *Classical conditioning II : Current research and theory*, 1972.

[9] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb) :281–305, 2012.

An evolutionary approach to find optimal policies with an agent-based simulation
Un algorithme évolutionnaire pour trouver des politiques optimales dans un simulateur multi-agents

Abstract

In this paper, we introduce a new agent-based method to build a decision-aid tool aimed to improve policy design. In our approach, a policy is defined as a set of levers, modelling the set of actions, the means to impact a complex system.
Our method is generic, as it could be applied to any domain, and be coupled with any agent-based simulator. We could deal not only with *simple* levers (a single variable whose value is modified) but also *complex* ones (multiple variable modifications, qualitative effects, ...), unlike most optimization methods. It is based on the evolutionary algorithm CMA-ES, coupled with a normalized and aggregated fitness function. The fitness is normalized using estimated Ideal (best policy) and Nadir (worst policy) values, these values being dynamically computed during the execution of CMA-ES through a Pareto Front estimated with the ABM simulation. Moreover, to deal with complex levers, we introduce the FSM-branching algorithm, where a Finite State Machine (FSM) determines whether a complex policy can potentially be improved or has to be aborted.
We tested our method with Economic Policies on the French Labor Market (FLM), allowing the modification of multiple elements of the FLM, and we compared the results to the reference, the FLM without any policy applied. The policies studied here comprise simple and complex levers. This experience shows the viability of our approach, the efficiency of our algorithms and illustrates how this combination of evolutionary optimization, multi-criteria aggregation and agent-based simulation could help any policy-maker to design better policies.

Résumé

Dans cet article, nous créons un processus multi-agents d'aide à la décision ayant pour but d'améliorer la conception de politiques. Une politique est définie par un ensemble de leviers, modélisant les actions disponibles pour le décideur politique pour modifier un système complexe.

Notre méthode est générique et peut être appliquée à n'importe quel domaine, avec n'importe quel simulateur multi-agents. Contrairement à des processus d'optimisations plus traditionnels, nous pouvons utiliser aussi bien des leviers simples (paramètre continu du modèle dont la valeur est modifiée) que des leviers complexes (plusieurs variables, modification plus importante du modèle). Notre algorithme s'appuie sur CMA-ES, associé à des outils de décisions multi critères pour avoir une fonction de fitness agrégée et normalisée. Cette fitness est normalisée grâce aux points Idéal (meilleure politique) et Nadir (plus mauvaise politique). Ces points sont obtenus à l'aide d'un front de Pareto dynamique, calculé en continu durant le processus d'optimisation par le simulateur multi-agents. De plus, pour gérer le problème des leviers complexes, nous utilisons une machine à état (FSM) pour déterminer s'il faut continuer à optimiser une politique complexe ou l'abandonner.

Cet algorithme a été testé avec un simulateur du marché du travail français (WorkSim) en modifiant plusieurs paramètres liées aux contrats, RSA, et allocations chômages principalement. Les résultats ont été comparés à la simulation de référence, sans politique appliquée. Cette expérience montre l'intérêt de notre méthode, combinant algorithme évolutionnaire, décision multi-critères, et simulation multi-agents, et son utilité pour aider un décideur à concevoir de meilleures politiques.

Keywords

Agent-Based Simulation, Evolutionary Optimization, Multi-criteria aggregation, Policy Design, Labor Economics

1 Introduction

As artificial intelligence improves through the years, agent-based-models (ABMs) of complex systems are beginning to be used more and more, as they account for the heterogeneity of their elements, and provide a deeper understanding of the mechanisms and interactions inside those systems, than aggregated models. In this paper, we aim to provide an ABM decision-aid tool to Policy Makers, in order to improve policy design ; that is, to find the best policy according to the model and to a set of objectives, set by the policy maker.
By policy, we mean here (in the political or managerial or systemic sense) any process, set of rules or laws that affect a complex system (firm, economy, society, ...). The"optimality" of the policy will be assessed using crite-

ria, pre-defined by the Policy Maker.

Currently, two optimization approaches dominate the field of policy improvement : Reinforcement learning (RL) [13] and Black-Box Optimization (BBO) [1]. RL has been often used to control Multi-Agent Systems and improve policies (in Robotics for instance) [3; 23], whereas, to our knowledge, very few BBO have been proposed for ABM [12].

In fact, as shown in [23], RL and BBO could be quite similar, the main difference being that, for RL, the optimization operates within the complex system , with reward information being used in real time to find optimal actions. Because we want to propose a generic approach and a easy-to-use tool for any PM (who does not have to know of the system works), we favor a BBO, where the system is modeled by an ABM, simulated and taken as a black-box by the optimization algorithm. In order to take benefit from both ABM and optimization, the optimization algorithm will select a policy to be tested, and an agent-based simulator will be used to evaluate the outcomes of this policy (see Figure 2 below). We designed our method to be as generic as possible, so it could be applied to any domain, any complex system, and be coupled with any agent-based simulator.

The paper is organized as follows : we define the concepts of policies in Section 2, before explaining the whole optimization process in Section 3. In Section 4 we will describe an application of this process in an experience on economic policies on the French Labor Market (FLM), before comparing this project to other related works in Section 5 and concluding.

2 Defining Policies

2.1 Simple Policy

Broadly speaking, a **simple policy** is a set of measures that modify a given Complex System – referred to as "the System" in this paper– by changing the value of some of its parameters, denoted here **simple levers** (e.g for labor markets : minimal wage, unemployment benefits). A simple policy will be modelled here as a **set of simple levers' values**, which means that some parameters are set to these particular values (e.g minimal wage = 8 €/h). The simple levers used in our experiment on the FLM are listed in Table 3.

2.2 Complex Policy

The policy maker might want to test more elaborated policies that could not be reduced to a change of one single continuous feature. Some policy will change several (possibly many) features and/or induce qualitative effects. In that case, a **complex lever** must be used and is no longer a continuous value but a *binary variable* : one activates the policy or not. This type of binary levers is more difficult to optimize, as most of optimization algorithms are designed for continuous spaces (e.g. gradient descent). We will address the optimization of *complex levers* in section 3.3 below.

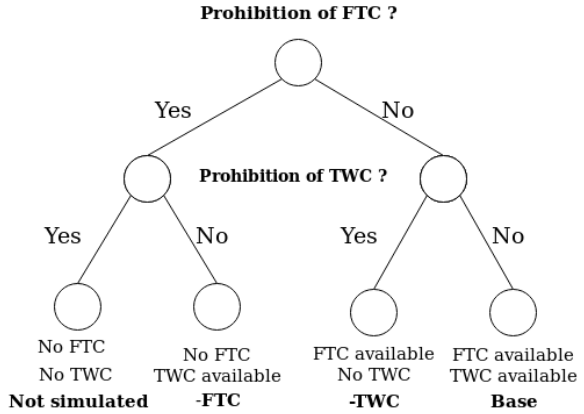


FIGURE 1 : Example of a Complex Levers tree (CL Tree). Each leaf node represents a Combination of Complex Lever values (CCL). FTC and TWC are defined in 4.1.

We define a **Complex Policy** as a *combination of simple and complex lever values*. If the policy includes n complex levers, we have 2^n possible **Combination of Complex Lever (CCL)** values. Each CCL will be a leaf node in a binary tree – called the **CL Tree**, as illustrated in Figure 1 with the case of our experiment (with $n = 2$).

2.3 Policy Evaluation

The policy maker will use the levers (simple or complex) to define the measures composing the policy, the modifications of the System that the policy enforce. Then, in order to enable an evaluation of this policy, s/he must specify the **criteria**, that are the variables (numerical and continuous) that will measure the **outcomes** of the policy, and its impact to the System. Therefore, we define an **optimal policy** as a *policy that optimizes a set of pre-defined criteria*. Eventually, the Policy Maker can weight differently the criteria, so we suppose s/he enters a set of criteria weights w_i (real positive values), eventually identical for all criteria).

3 Optimization process

The overall optimization process is depicted in Figure 2.

In accordance with our BBO approach, this process uses an evolution strategy to generate the policies, which are then simulated (the ABM simulator being considered as a black box), and evaluated with a fitness score, to improve future generations of policies. We decided to use CMA-ES [11] for the BBO algorithm as it has been shown to be one of the most efficient ones [10; 17]. Moreover it has been already successfully used with an ABM, in order to calibrate its parameters [8].

To start the optimization process, the Policy Maker has to specify the levers and the criteria s/he aims to use. CMA-ES requires to know the exact number of simple levers, and their respective bounds, to generate the population (a set of policies) that will be evaluated with the simulator. Each

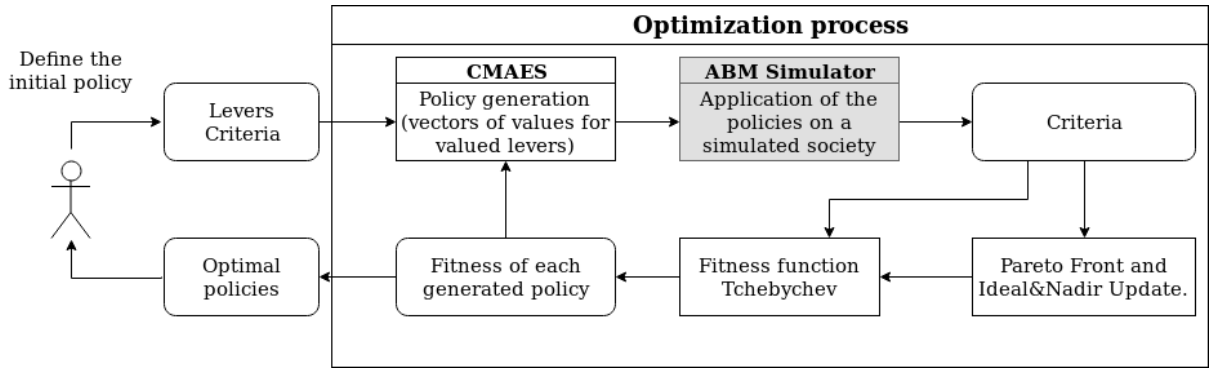


FIGURE 2 : Overall optimization process.

policy will be simulated during H ticks, H being a parameter of our method. Moreover, if the ABM is stochastic (that is often the case), we simulated the policy multiples times (set by the parameter N_S), in order to reduce the effects of this stochasticity, and the criteria will be averaged over these replications to compute the fitness score of this policy. This score helps CMA-ES to improve the next generations of policies, and increases the chance of finding the optimal one in them.

However, when designing the fitness function we must take into account that the criteria could be of very different order of magnitude : some criteria are percentages in [0,1] (like unemployment rate) while some others are a lot bigger (like the average weekly income per household). Moreover, some criteria are to be maximized (e.g. revenues, profits) while other need to be minimized (e.g. unemployment rate).

To aggregate and normalize the criteria in the fitness function, we choose an augmented weighted Tchebychev norm, as it has proven to be efficient for multiple objective programming [22; 21; 27].

3.1 The augmented weighted Tchebychev norm

Following [26] and [6], we compute the fitness function as an augmented weighted Tchebychev norm. If we have n criteria to optimize, with each criteria having a (simulated) value c_i and a weight w_i, it is given by :

$$f(c) = max_{i=1,\dots,n}(w_i.\bar{c_i}) + \epsilon \sum_{i=1}^{n}(w_i.\bar{c_i}) \qquad (1)$$

where $\bar{c_i}$ are the *normalized criteria*, given by :

$$\bar{c_i} = \begin{cases} \frac{Id_i - c_i}{Id_i - Na_i} & \text{if } c_i\text{needs to be maximized} \\ \\ \frac{c_i - Id_i}{Na_i - Id_i} & \text{if } c_i\text{needs to be minimized} \end{cases} \qquad (2)$$

Id_i is the Ideal value for criterion c_i, and Na_i is the Nadir (worst) value for criterion c_i. ϵ is a parameter (with a positive and small value). The *ideal point* is defined to be the vector of the componentwise infima of all Pareto-Optimal solutions' values, while *the Nadir point* is characterized by the componentwise supremum of these values [16].

Thanks to equation 2 , we use Ideal and Nadir points to normalize the criteria to [0,1], where a 0 corresponds to the best value and 1 the worst.

When we *minimize* f – given by Eq. 1, the criteria c_i will have to be close to Id_i (i.e. close to the best possible value of the criteria, from all the simulated points). The choice of Tchebychev norm focuses on the worst component and therefore guarantees that only feasible solutions close to the Ideal on every component will receive a good score [6]. In addition, if ϵ is chosen small enough, the practical possibility of reaching any Pareto-optimal solution is kept by an appropriate choice of weights w [26]. Moreover, the second additive component of function f ($\epsilon \sum_{i=1}^{n}(w_i.\bar{c_i}$) allows to discriminate solutions that give similar performance on criterion c_i by taking account other criterion values (c_j with $j \neq i$). This function allows us to choose the point that gives the smallest (weighted) regret, preferring a more balanced solution rather than an unbalanced one.

The weights of each criterion are defined by the decision maker in the initial policy, and are necessary to prioritize some variables over others (according to their importance) in the optimization process. Now, let us detail how Ideal and Nadir values are computed.

The Ideal point can be easily computed by optimizing each objective individually over the search space [16], while finding the Nadir point is a difficult task, and its exact values could in many cases only be approximated using heuristics [16; 25], because the components of the Nadir point are not always obtained during the search for the Ideal point described before (as finding the best value for a criteria doesn't necessarily means that its other criterias would be the worst possible, and thus, components of the Nadir point). However, even computing the Ideal point by running n single criteria optimization would be too long, especially when there are many criteria. Thus, we chose to compute the Ideal and Nadir points by incrementally estimating the Pareto (PF)[1]

[1]The Pareto Front is the set of all pareto-optimal solutions.

(PF), and taking, for each criterion, the best and worst value found among the Pareto Front solutions :

$$(Na_i, Id_i) = \begin{cases} (\min_{c \in PF}\ c_i, \max_{c \in PF}\ c_i) & \text{to maximize } c_i \\ (\max_{c \in PF}\ c_i, \min_{c \in PF}\ c_i) & \text{to minimize } c_i \end{cases} \quad (3)$$

This dynamic estimation of Ideal and Nadir allows us to constantly improve their approximated values, getting more precise as the optimization process progresses. We keep updating the Pareto Front with each new simulated policies found during the evolutionary optimization process produced by CMA-ES.

3.2 Simple Policy Optimization

For a simple policy, we use CMA-ES to find the optimal lever values. Thus, the optimization process will proceed in four main steps :

1. **New Population Generation** : the CMA-ES algorithm generates a new population of points[2], that are candidate solutions for f minimization.

2. **Updating the Pareto Front** : these new points needs to be compared to all the points in the Pareto Front (PF). For each non-dominated point *p*, we add it unless it is completely dominated by a point in PF.

3. **Recalculate Ideal and Nadir** : if at least one new point has been added to the Pareto Front, then Ideal and Nadir need to be updated because the new point may be better than the Ideal (or worse than the Nadir) on a criteria, or might have removed a point that gave the worst value for a criteria. Thus, it is needed to recalculate Ideal and Nadir after every generation if there was a change in the Pareto Front.

4. **Updating the Fitness Function** : if Ideal and Nadir were modified by the last generation, the fitness function needs to be updated to use the new Ideal and Nadir. The updated fitness function will be used to evaluate the current population, and we return to step 1.

With Ideal and Nadir normalizing every criteria, and the Tchebychev norm aggregating them, we now have a fitness function for CMA-ES to use during the optimization process. Unlike usual fitness function that are set once and never modified again, giving the same result at any time during the algorithm, the result of the augmented weighted Tchebychev norm can change over time because of the frequent updates of the Pareto front, and thus, of Ideal and Nadir.
Now that we have described the main stages of our optimization process, and before we present our experiment results, let us know explain how we deal with complex policies, made of complex levers.

[2]Here, a *point* is a set of simple lever values.

3.3 Complex Policy Optimization

To optimize a complex policy, made of simple and complex levers, we proceed in two steps :

1. Select a CCL (leaf node) in the CL tree to set a combination of complex lever values,

2. For this node, run the optimization process described in section 3.2 above to find the optimal values of the simple levers included in the policy.

CMA-ES doesn't work well with binary variables[3], so we cannot let CMA-ES decide on which CCL to optimize. Therefore, we designed an algorithm to monitor and lead the optimization process of the CCLs : the *branching algorithm*.

3.4 Branching algorithm

We designed the FSM-branching algorithm to select which combination of complex lever values will be chosen in the current CMA-ES in order to optimize the simple levers in the case of a complex policy. Several issues need to be considered in the creation of this algorithm :
Assessing the potential of a CCL is not really possible without exploring it a little, because you cannot predict in advance the results of an optimization (exploration) on a CCL. But it is nevertheless possible to determine approximately if a CCL could be improved : by looking at its evolution during the last iterations of CMA-ES. If these iterations have improved the optimal policy of this CCL (by reducing the fitness), then we can consider that this CCL can be improved : that is if the fitness value has been reduced by at least Z %, Z being a parameter of FSM-branching.
We propose to use a Finite State Machine (FSM) to monitor the CCL exploration. Our FSM is depicted in Figure 3, and includes three states :

Running : the CCL could potentially be improved, and has not been paused in the current iteration of the branching algorithm.

Paused : this configuration did not improve during X ticks of CMA-ES, therefore it is paused. It will be woken up at the next iteration of the branching algorithm.

Finished : this configuration has been paused Y times in a row, with no improvement. It is therefore abandoned and will never be explored again, because it can no longer be improved. *If all CCLs' optimizations are finished, the policy optimization is completed.*

Moreover, we have 3 transitions in the FSM :

Running $\rightarrow$ *Paused* : this transition occurs when the CCL is explored during X ticks of CMA-ES without significant improvement (i.e. above the required threshold)

[3]There exists a version of CMA-ES to handle integer values, but it does not work well for binary ones [9].

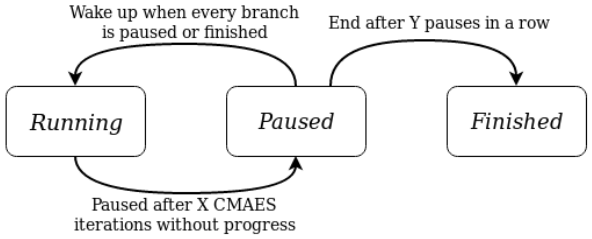


FIGURE 3 : Finite State Machine (branching algorithm).

compared to its state before exploration. The CCL is paused, and will no longer be explored as long as there are other CCLs in progress.

Paused $\rightarrow$ *Running* : (Awakening) when all CCLs are paused or finished, then the branching algorithm wakes up all paused branches, and makes them switch to the *Running* state.

Paused $\rightarrow$ *Finished* : if a CCL does not improve after Y successive awakenings, then it is aborted. The FSM of this CCL and switches to the *Finished* state : it will never be explored again in this optimization process.

Each CCL has its own FSM, and the branching algorithm has three parameters : X (the number of iterations before considering pausing the CCL), Y (the number of awakenings with no improvements to abort the CCL) and Z the minimal required fitness decrease (in %) to consider that a CCL has improved its optimization.

The FSM-branching has the following qualities :

Termination : it is not possible for policies to improve endlessly, as the more optimized the policy, the more difficult it is to improve it even more. There will therefore inevitably be a time when there will be no improvement after Y awakenings, and the CCL will therefore be abandoned.

Potential exploitation : this algorithm evaluates the potential of a CCL to know if it should be explored, and will therefore explore all CCLs that have a potential for improvement.

Adaptive resources allocation : No CCL will monopolize all the computing resources (unless it is the only one in progress), and all improvable CCLs will be explored one after the other. In addition, potential-free CCLs will quickly be abandoned, leaving more time for potential CCLs.

To end this presentation of the FSM Branching, let us now explain **why parallelization is not efficient here**. Each CCL has a CMA-ES for itself as they cannot share the same evolution strategy, because each policy could have very different impacts on the System. As each CCL progresses independently, it could be possible to parallelize all the CMA-ES, so that each optimization progress at the same time, and then no branching algorithm would be required. However, each population generated by CMA-ES needs to be evaluated by the simulator, and it takes $N_P \times N_S$ simulations to do so, with N_P being the size of the population generated, and N_S the number of simulations done for a single policy (as the model is stochastic, it is necessary in order to reduce effects of randomness on the result of the policy). For example, in the experience described in the next section, we had $N_P = 48$ and $N_S = 16$, and thus we had 768 simulations per generated population for each CCL. Those 768 simulations need to be finished to update the Pareto Front and evaluate the population. Since all CCLs share the same unique Pareto Front, the process would need to wait for each CCL to finish their simulations to proceed, losing most of the advantages of a parallelization. Not to mention that this will add complex synchronization issues. All in all, this makes parallelization simply inefficient to implement our algorithms, so we decided to focus on optimizing one CCL at the time, and using multiple threads to do the 768 simulations as fast as possible using parallelization on a cluster.
The aim of the FSM-branching algorithm is precisely to allow the optimization process to share the computing resources between the different configurations, while favouring the most promising CCLs.

3.5 Overview of our optimization algorithm

The flow diagram in Figure 4 below summarizes the whole process. The set $\mathcal{P}_c^*$ is made of the best policies found for CCL c (see section 3.6 below).

3.6 Policy Comparison Protocol

To end the presentation of our policy optimization method, we need to explain how we should evaluate and compare the policies with each other, which is one the main goals in policy design. Because of the stochasticity of both CMA-ES and of the ABM simulator, it is important to replicate not only the simulations, but also the optimization process as CMA-ES may have found a sub-optimal policy because of some events during the optimization process. Therefore we launched multiple instances (N_R) of this optimization process, and then we need to compare all these generated policies, using the Policy Comparison Protocol, which proceeds as follows :

1. During the optimization process, we build a set $\mathcal{P}_c^*$ of best policies for each CCL c, the set of policies that have improved the fitness function at least once during the execution of CMA-ES. These policies are stored as a list, sorted in an increasing order of fitness value, so that the first item stores the optimal policy[4].

[4]We have decided, at this stage, to store multiple best policies and not only the optimal one, for two main reasons : (1) the order in this set depends on the Ideal and Nadir of the optimization process, and may change when the Ideal and Nadir is updated by a new generation, causing

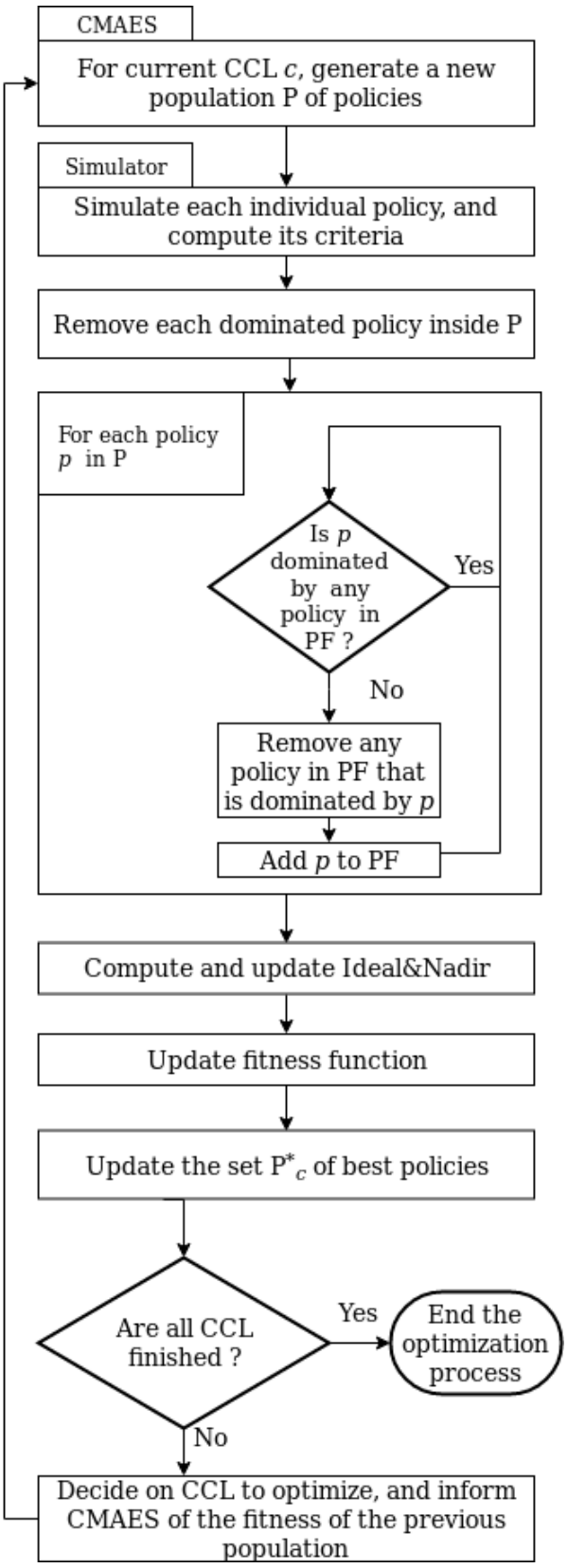


FIGURE 4 : Flowchart of our optimization algorithm.

2. We launched multiple (i.e. $N_R = 7$) full optimization runs[5].

3. So now we have a set $\mathcal{P}_c^{*^r}$ for each CCL c and each run r. Then, we build the union of all these sets (21 in this experiment)
$\mathcal{P}^* = \bigcup_{r=1}^{N_R} \bigcup_{c=1}^{N_C} \mathcal{P}_c^{*^r}$, from which we compute a Global Pareto Front, the Global Ideal and Global Nadir of all these policies, and thus, giving us a Global Fitness Function Gf that allows us to compare all the policies found in the sets $\mathcal{P}_c^{*^r}$.

4. In particular, for each CCL c, we compute the optimal policies as the one giving the lowest Global Fitness in the set $\mathcal{P}_c^{*^r}$.

In our experiment, we obtained 358 policies in $\mathcal{P}^*$. The Global Pareto Front contained 268 policies, and each CCL has an optimal policy, given by the best fitness score (with Gf) out of all its policies inside the Pareto Front.

4 Experience and Results

The goal of this experience is meant to be a proof of concept of our methodology to find optimal policies, and is meant to show how to analyze the results of the optimization process.

4.1 Experience settings

We chose to apply our methodology to the case of labor policies in France. To do so, we use our ABM Labor Market Simulator, WorkSim [8][6]
In this paper, the experience focuses on the utilization of the three available labor contract types that form the core of the FLM [19] : *Open Ended Contract (OEC)*, *Fixed Term Contract (FTC)* and *Temporary Help Contract (TWC)*[18] [7]. Using our optimization method, we aim to study how the

a change in the fitness function, and thus in P_b^*'s order; and (2) many best policies have a very close fitness value (difference below 0.01% in our experiment) and could all be considered near the optimum.

[5]By *full run*, we mean a fully completed optimization run, i.e. when the algorithm in Figure 4 is completed for all the CCLs (i.e. $N_C = 3$.

[6]WorkSim is a comprehensive model of the labor market. The stock-flow accounting of individuals, based on gross flows, is complete and endogenous. It is supplemented by a stock-flow accounting of jobs for further analysis. The institutional environment is modeled and based on labor law, which sets constraints on the possible decisions at the microeconomic level, taking into account the specific characteristics of each agent, worker or employer. It implements search on both sides of the market with multi-jobs firms, inter-temporal decision processes under bounded rationality, anticipations of demand shocks, learning, endogenous contract choices, endogenous salaries and productivities, different types of human capital. WorkSim is calibrated on a large number of targets of the French labor market, using CMA-ES.

[7]Let us briefly summarize the main features of these labor contracts. The Main OEC features are : no duration limit, probationary period, no firing costs for the first year, no termination costs if quitting, variable firing costs when firing. The Main FTC features are : maximum duration of 18 months including the possibility to be renewed once, a grace period after the termination of the contract during which the job cannot be filled, a small probationary period, allowance at the end of the contract : 10 % of total gross salary. FTC cannot be broken without heavy penalties (paying the remaining salary part). TWC is a special type of FTC since

labor market could be optimized depending on these available contracts. Hence we will study three different configurations (depicted in Figure 1 above) :

Base : All three contract types are available.

-TWC : Temporary Working Contracts are prohibited (i.e. OEC and FTC only).

-FTC : Fixed Term Contracts are prohibited (i.e. OEC and TWC only).

Notice that we removed the case where FTC and TWC were disabled, because past experiences have shown that the labor market is severely sub-optimal when no fixed term contracts (FTC or TWC) exist [7]. Similarly, we did not add another complex lever to analyze the effects of the prohibition of OEC, as OECs are the core of the labor market, and there is no point in trying to remove them.
Each of these configurations are defined by the same simple levers, and the same criteria, and will be evaluated with the same fitness function, as *the Pareto Front, Ideal and Nadir are shared by all these CCLs* (see section 3.6 above). This allows us to easily compare the results of each CCL as they follow the same format : levers, criteria, and fitness function are all identical.
The parameters of our method are listed in Table 1. The criteria[8] (with their weights) are listed in Table 2, and the 10 simple levers [9] we chose are listed in Table 3. Moreover, all the Labor Market Simulator's parameters are calibrated to reproduce many FLM statistics for the year 2014, the agents are initialized to reproduce the real FLM at a scale of $1/2300$. Each simulation takes around 3 minutes to run the $H = 416$ ticks[10], which means that with 16 simulations per point, it takes around 50 minutes for each CMA-ES iteration to end (on a 48-cores computer grid, so each of the 48 CMA-ES population runs on a single core). In the case of our experiment, it took around 380 iterations of CMA-ES (shared between the CCL) – around 14 days – to complete the optimization process.

4.2 Overall comparison of policies

To give an overview of the optimal policies outcomes, we displayed them in a radar chart, as shown in Figure 5. Note that,in this Figure, we display the inverted values $(1 - x)$ of the normalized criteria value x in order to have 1 for the optimal value for a normalized criteria, and 0 for the worst value, as it is commonly done in radar charts, and allows us to compare the configurations easily, criteria by criteria, or as a whole. The criteria are sorted by descending criteria weight (clockwise). The **Reference** is obtained by running the FLM Simulator without any optimization (no policy applied, no modifications of the FLM), and will serve as a baseline to compare the policies.
For a more quantitative analysis, we use the criteria outcomes displayed in Table 2. First of all, 3 criteria did not result in significant differences (shaded in gray in the Table) and therefore we ignored them for our analysis.

Name	Description	Value
	General	
N_C	Nb. of CCL	3
N_R	Nb. of full runs	7
	CMA-ES	
σ	Step size	0.3
N_P	Population size	48
N_S	Nb. simulations per evaluation	16
H	Duration of each simulations	416 ticks
	Tchebychev	
ϵ	Weight of sum component	0.001
	FSM-Branching	
X	Nb.of CMA-ES iter. before pausing	3
Y	Nb. of pauses before aborting	15
Z	Required fitness improvement	1 %
	Simulator	
N_A	Total number of agents	23000
	Nb. of individuals	20000
	Nb .of firms	3000

TABLE 1 : List of parameters for our optimization method, with their values for our experiment.

Now, how could we rank the 4 policies ? We propose to perform a **weighted Condorcet voting** [24], using the following ranking function based on the criteria weights.

For each CCL c, we count the number of times it strictly dominates a CCL d, using the following weight-based preference relation :

$$c \succ d \iff \sum_{i:(c_i > d_i)} w_i > \sum_{j:(d_j > c_j)} w_j \qquad (4)$$

where d_i is the value of CCL d on criterion i.

We found – from our results – that $-FTC$ *is the Condorcet winner* and we obtained the following strict order :

$$-FTC \succ Base \succ -TWC \succ Ref$$

However, the Condorcet voting method is purely ordinal. To account also for the criteria outcomes, we compute an **aggregated score** S_w as the weighted sum of the normalized criteria, by limiting the sum to criteria giving signi-

the employer is a Temporary Help Agency. The agency provides workers to the client firm for a mission, and is paid by regular firms to find these workers. It usually finds a suitable employee faster than regular firms, and also screens and train sometimes workers better than firms.

[8] We selected here 9 frequently used labor market measures (see e.g. `http://www.oecd.org/sdd/labour-stats/` for definitions.).

[9] These levers have been derived from the economical programs of the candidates at the last French presidential election.

[10] In our Labor Simulator, one tick corresponds to one week in reality. Thus, we chose to run each simulation during 8 years to ensure that a steady state has been reached.

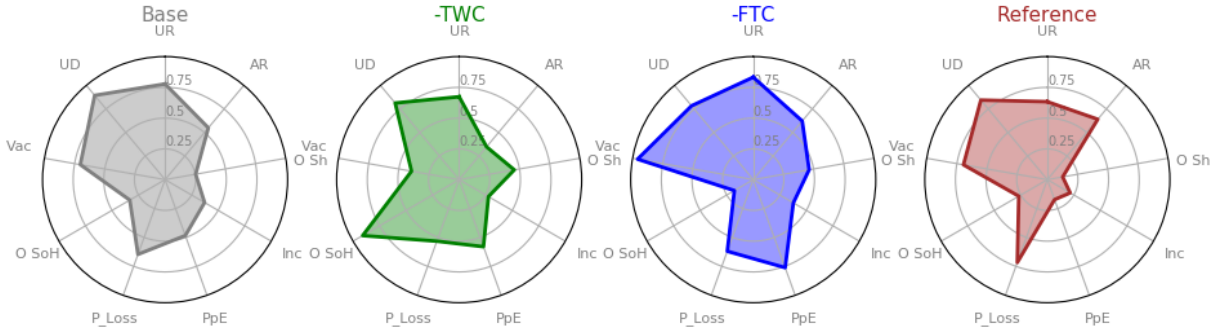


FIGURE 5 : Radar charts of the final optimal policies found for each configuration. The (abbreviated) criteria are displayed in the same order as for Table 2, clockwise.

w	Criteria	**Base**	**-FTC**	**Ref**	**-TWC**
10	Unemployment rate (%)	8.6 *0.21*	**8.1** *0.16*	9.9 *0.37*	9.5 *0.32*
5	Global activity rate (%)	71.9 *0.44*	72.5 *0.37*	72.6 *0.36*	70.1 *0.64*
3	OEC share (%) (from all active contracts)	83.5 *0.74*	**85.5** *0.53*	82.3 *0.87*	85.4 *0.53*
3	Median income per household share (in €/month)	1587 *0.62*	1590 *0.61*	1512 *0.78*	1544 *0.71*
3	Average firm profit per employee (in €/week)	1052 *0.51*	**1093** *0.23*	1004 *0.83*	1066 *0.41*
2	Yearly Probability of being fired (when on OEC)	5.7 *0.34*	5.8 *0.37*	5.7 *0.28*	5.9 *0.46*
1	Weekly Share of OEC hires (%)	**8.13** *0.66*	6.6 *0.81*	7.4 *0.73*	13.9 *0.08*
1	Job vacancy rate (%)	6.92 *0.29*	**3.21** *0.03*	7.17 *0.3*	11.38 *0.6*
1	Unemployment duration	**72** *0.1*	84 *0.21*	79 *0.16*	81 *0.18*
	S_W	6,9	**4,9**	10	6,8

TABLE 2 : List of the 9 criteria and weight values w. We display the optimal raw values values for our 3 CCL and the Reference. The numbers in italics are the normalized values (0=best), and the list line displays their S_w score. The shaded lines indicates no significant difference between the policies. The best value for each criteria is displayed in bold.

ficant differences. These S_W scores confirmed the above order, they are displayed in Table 2 (last line).

To briefly comment this result, we first observe that the optimization actually worked, as the configurations outperforms the Reference. Although they obtained the same aggregated score S_W, $Base$ dominates $-TWC$ in the Condorcet voting, because it performed better on the criterion with the highest weight (Unemployment rate). Of course, we could obtain very different rankings if we change the weight values : any decision-aid tool is sensitive to the way the policy maker will weight his/her criteria. Moreover, these weights may be difficult for a policy maker to determine accurately. Such an issue is outside the scope of this paper, but let us just mention that we could combine our approach with weight elicitation techniques [4] to facilitate these settings.

4.3 Comparison between the two best policies

To end this results section, we focus on the comparison of the two best policies : $-FTC$ and $Base$. If we look at the simple lever values found by our optimization process (Table 3), we can see that $Base$ and $-FTC$ converged towards a similar values for several levers : they both raised the minimum wage (SMIC), the RSA base value (that acts as a work welfare benefit), the unemployment minimum daily allocation, and the reduction coefficient of insurance charges. All these levers impact the working class as they improve their income (during employment and unemployment), and their employability (reduction of employer's labor charges). However, $-FTC$ and $Base$ differ significantly in 4 levers : TWC renewals (doubled in $-FTC$) , maximum TWC duration (divided by 2), Unemployment maximal benefit for 50+ (/2) and unemployment maximal allocation divided by 2 as well. It is beyond the scope of this paper to discuss the underlying economic and social consequences of these choices. However, from a methodological point of view, it is important to point out that these levers values must be taken into account when evaluating

and comparing policies, as they entail political choices and have direct impacts on the system entities (firms, individuals, the State,... in our case).

Finally, looking at the criteria values, we found that $-FTC$ outperforms Base significantly on 4 criteria : Unemployment (146k fewer unemployed individuals), OEC share (+594k OEC contracts), a slight increase in profit per employee (+4 %) and a vacancy rate divided by more than 2. Overall $-FTC$ seems to reduce unemployment (-546k unemployed compared with the Reference), improve income for households , profits for firms and strongly reduces the vacancy rate, which shows than the job matching is much more efficient with $-FTC$.

How to explain such a good performance of $-FTC$? Why the removal of FTC contracts seems to perform better that the removal of TWC contracts ? To explain this result, we take advantage of our agent-based approach and looked at the outputs of our Labor Simulator. We found that individuals chose TWC jobs to gain experience, and obtain a permanent contract more easily after accumulating enough experience, while companies "abuse" less fixed-term contracts (in $-FTC$), and use temporary contracts only for rapid labour needs to satisfy an increase in demand. In addition, the recruitment of TWC jobs is much faster than FTC, and its higher cost is therefore offset by the absence of vacancy costs (i.e. costs induced when a job remains vacant, eventually for a long time) . These observations will have to be further studied, as our aim in this paper is mainly to illustrate how our methodology works in a real case.

Levers	$Base$	$-FTC$	Ref
Reduction coeff. for insurance charges	2.47	2.75	1.6
SMIC (minimum wage in €/h)	9.2	9.7	8.05
Nb of allowed FTC renewals	2	N/A	2
Nb of allowed TWC renewals	1	3	2
Maximum duration of FTC (in weeks)	79	37	72
Unemploy. max benefit duration (50yo+)	166	84	144
Unemploy. max benefit duration (< 50yo)	100	106	96
Unemploy. max. daily allocation (in €)	347	122	241
Unemploy. min. daily allocation (in €)	36	39	29
RSA base value (in €/ month)	670	656	500

TABLE 3 : List of the ten simple levers used in our experiment on FLM, and their optimal values for the 2 best CCL and the Reference.

5 Related Works

We chose CMA-ES over the other Single-Objective Optimization (SOO) methods, as it has shown to be the most efficient and fastest algorithms [10]. However, since our decision problem is multi-objective by nature, why did we not use a Multi-Objective Optimization (MOO) approach ? First of all, MOO are generally much slower than SOO [15], up to the point that MOO are almost impossible to use in practice when the number of criteria rises [5]. This is particularly problematic for our approach that used a quite detailed agent-based simulation to evaluate each outcome, and therefore demanding in terms of computation time. A second argument that favors a SOO approach over MOO concerns our need to allow the policy maker to weight his/her criteria : this is easily done in SOO with a weighted aggregation, and impossible with MOO.

Furthermore, the combination of CMA-ES with a Pareto-Front computation has already been proposed in the CMA-PAES model [20], this method combining a MOO version of CMA-ES with the Pareto Archived Evolution Strategy [14] to improve the solution generation that was sub-optimal in PAES. Like our approach, CMA-PAES dynamically builds a Pareto Front to obtain the optimal policies but, like many MOO methods, it does not help to choose the best one among these. Moreover, CMA-PAES does not handle weights, nor complex levers (we would obtain a single Pareto Front for each CCL and could not compare them). In fact our aims differ : CMA-PAES seeks exhaustive optimal policies (the most diverse one) while we look for a single (or a reduced set) of best optimal policies (e.g. Condorcet winner).

Finally, we proposed a method to compute Ideal and Nadir, using a simulated estimation of the Pareto Front. While the Ideal point can be obtained in a MOO by optimizing each criteria separately, the Nadir point is more complicated. Thus, Nadir-Soco [25] estimates the Nadir point, using the "Extreme-To-Nadir" method that focuses on knee points [2]. Nadir-Ejor [16] finds the Nadir by optimizing Q sub-problems derived from the initial one, that are the optimization of the $Q - 1$ combinations of the Q objectives separately. This is a very costly approach in term of time and computing resource.

Moreover, these two methods implicitly imply that the criteria are somehow negatively correlated (improving one criteria must deteriorate the other ones), but this is not the case in all decision problems, where some criteria could be independent. Hence our approach is more generic as it does not rely on such assumption.

6 Conclusion

In this paper, we proposed a new approach to aid policy design by combining agent-based simulation with evolutionary optimization and multi-criteria aggregation techniques. We introduced a *new optimization method based* on CMA-ES coupled with a dynamic estimation of the Pareto Front. This estimation is done by the agent-based simulation, and enables to compute Ideal and Nadir points, these points being used to normalize the fitness function.

One of the advantages of our approach is its genericity : it could be applied to any number of criteria, any set of criteria weights, any agent-based simulator. Moreover, unlike existing methods, we do not only deal with simple policies (made of quantitative levers) but can also process complex policies, made of several configurations (CCL), for which we proposed the *FSM-branching* algorithm to optimize the computation of the configurations.

We also introduced an algorithm to compare policies after the completion of the optimization process, We illustrated our method in a real case, the French Labor Market, where we used a fairly detailed Labor Market Simulator to study how new arrangements of labor contracts could improve important criteria (e.g. employment). The method selected an unexpected solution (to remove FTC contracts), and in that case it clearly dominated the others, being even a Condorcet winner[11] . Moreover, one benefit of our approach is that it allows both to find an optimal solution but also an estimated Pareto Front that allow a policy maker to study other solutions (close to the optima).
Future works will focus on a more in-depth exploration of the behaviour of our algorithms : the sensitivity of its parameters (in Table 1), and a benchmark comparison with MOO. Like every optimization approaches, we also aim to study deeper how we could estimate the quality of the optimal policy, and its unicity, although these questions are known to be very difficult, both theoretically and in terms of the required computational resources.

[11]Of course, this would not always be the case, our method does not guarantee in any way to find such a winner for any application.

Références

[1] Charles Audet. 2014. *A Survey on Direct Search Methods for Blackbox Optimization and Their Applications*. Springer New York, New York, NY, 31–56. https://doi.org/10.1007/978-1-4939-1124-0_2

[2] Jurgen Branke, Kalyanmoy Deb, Henning Dierolf, and Matthias Osswald. 2004. Finding knees in multi-objective optimization. In *In the Eighth Conference on Parallel Problem Solving from Nature (PPSN VIII). Lecture Notes in Computer Science*. Springer-Verlag, 722–731.

[3] Lucian Buşoniu, Robert Babuška, and Bart De Schutter. 2010. *Multi-agent Reinforcement Learning : An Overview*. Springer Berlin Heidelberg, Berlin, Heidelberg, 183–221. https://doi.org/10.1007/978-3-642-14435-6_7

[4] Adiel Teixeira de Almeida, Jonatas Araujo de Almeida, Ana Paula Cabral Seixas Costa, and Adiel Teixeira de Almeida-Filho. 2016. A new method for elicitation of criteria weights in additive models : Flexible and interactive tradeoff. *European Journal of Operational Research* 250, 1 (2016), 179 – 191. https://doi.org/10.1016/j.ejor.2015.08.058

[5] K. Deb and H. Jain. 2014. An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I : Solving Problems With Box Constraints. *IEEE Transactions on Evolutionary Computation* 18, 4 (Aug 2014), 577–601. https://doi.org/10.1109/TEVC.2013.2281535

[6] Lucie Galand and Patrice Perny. 2006. Search for Compromise Solutions in Multiobjective State Space Graphs. In *Proceedings of the 2006 Conference on ECAI 2006 : 17th European Conference on Artificial Intelligence August 29 – September 1, 2006, Riva Del Garda, Italy*. IOS Press, Amsterdam, The Netherlands, The Netherlands, 93–97. http://dl.acm.org/citation.cfm?id=1567016.1567042

[7] Oliver Goudet, Jean-Daniel Kant, and Gérard Ballot. 2014. Forbidding Fixed Duration Contracts : Unfolding the Opposing Consequences with a Multi-Agent Model of the French Labor Market. In *Advances in Artificial Economics*. Springer, 151–167.

[8] Olivier Goudet, Jean-Daniel Kant, and Gérard Ballot. 2017. WorkSim : A Calibrated Agent-Based Model of the Labor Market Accounting for Workers' Stocks and Gross Flows. *Comput. Econ.* 50, 1 (June 2017), 21–68. https://doi.org/10.1007/s10614-016-9577-0

[9] Nikolaus Hansen. 2011. *A CMA-ES for Mixed-Integer Nonlinear Optimization*. Research Report RR-7751. INRIA. https://hal.inria.fr/inria-00629689

[10] Nikolaus Hansen, Anne Auger, Raymond Ros, Steffen Finck, and Petr Pošík. 2010. Comparing Results of 31 Algorithms from the Black-box Optimization Benchmarking BBOB-2009. In *Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO '10)*. ACM, New York, NY, USA, 1689–1696. https://doi.org/10.1145/1830761.1830790

[11] Nikolaus Hansen and Andreas Ostermeier. 2001. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation* 9, 2 (2001), 159–195.

[12] Stephan Hutterer, Michael Affenzeller, and Franz Auinger. 2012. Evolutionary Optimization of Multi-agent Controlstrategies for Electric Vehicle Charging. In *Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO '12)*. ACM, New York, NY, USA, 3–10. https://doi.org/10.1145/2330784.2330786

[13] Leslie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. 1996. Reinforcement Learning : A Survey. *CoRR* cs.AI/9605103 (1996). http://arxiv.org/abs/cs.AI/9605103

[14] J. Knowles and D. Corne. 1999. The Pareto archived evolution strategy : a new baseline algorithm for Pareto multiobjective optimisation. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, Vol. 1. 98–105 Vol. 1. https://doi.org/10.1109/CEC.1999.781913

[15] R.T. Marler and J.S. Arora. 2004. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization* 26, 6 (01 Apr 2004), 369–395. https://doi.org/10.1007/s00158-003-0368-6

[16] M.Ehrgott and D.Tenfelde-Podehl. 2003. Computation of ideal and Nadir values and implications for their use in MCDM methods. *European Journal of Operational Research* 151 (2003), 119–139. https://doi.org/10.1016/S0377-2217(02)00595-7

[17] Olaf Mersmann, Mike Preuss, and Heike Trautmann. 2010. Benchmarking Evolutionary Algorithms : Towards Exploratory Landscape Analysis. In *Proceedings of the 11th International Conference on Parallel Problem Solving from Nature : Part I (PPSN'10)*. Springer-Verlag, Berlin, Heidelberg, 73–82. http://dl.acm.org/citation.cfm?id=1885031.1885040

[18] Francois Michon. 2008. France : Temporary agency work and collective bargaining in the EU. (18 Dec. 2008). https://www.eurofound.europa.eu/publications/report/2008/france-temporary-agency-work-and-collective-

[19] Thierry Pénard, Michel Sollogoub, and Valérie Ulrich. 1999. Hiring contracts, Wage, and Job Satisfaction : Theory and evidence on French low qualified youths. (June 1999). https://perso.univ-rennes1.fr/thierry.penard/biblio/jole.pdf

[20] Shani Rostami and Alex Shenfield. 2012. CMA-PAES : Pareto archived evolution strategy using covariance matrix adaptation for Multi-Objective Optimisation. In *2012 12th UK Workshop on Computational Intelligence (UKCI)*. 1–8. https://doi.org/10.1109/UKCI.2012.6335782

[21] Ralph E. Steuer. 1989. *Multiple Criteria Optimization : Theory, Computation, and Application*. Krieger. https://books.google.fr/books?id=tSA_PgAACAAJ

[22] Ralph E. Steuer and Eng-Ung Choo. 1983. An Interactive Weighted Tchebycheff Procedure for Multiple Objective Programming. *Math. Program.* 26, 3 (Oct. 1983), 326–344. https://doi.org/10.1007/BF02591870

[23] Freek Stulp and Olivier Sigaud. 2013. Policy Improvement : Between Black-Box Optimization and Episodic Reinforcement Learning. In *Proceedings JFPDA*. 1–15.

[24] W.D. Wallis. 2014. *The Mathematics of Elections and Voting*. Springer International Publishing.

[25] Handing Wang, Shan He, and Xin Yao. 2015. Nadir Point Estimation for Many-Objective Optimization Problems Based on Emphasized Critical Regions. *Soft Computing* (Nov 2015). https://doi.org/10.1007/s00500-015-1940-x

[26] Andrzej P. Wierzbicki. 1986. On the completeness and constructiveness of parametric characterizations to vector optimization problems. *Operations-Research-Spektrum* 8, 2 (01 Jun 1986), 73–87. https://doi.org/10.1007/BF01719738

[27] Andrzej P. Wierzbicki. 1999. *Reference Point Approaches*. Springer US, Boston, MA, 237–275. https://doi.org/10.1007/978-1-4615-5025-9_9

Peut-on rire en IA ?

Révision de croyances et modélisation de plaisanteries

Florence Dupin de Saint-Cyr
florence.bannay@irit.fr

Henri Prade
henri.prade@irit.fr

IRIT-CNRS, Université Paul Sabatier,
118, route de Narbonne, 31062 Toulouse Cedex 9

La plus perdue de toutes les journées est celle où l'on n'a pas ri.
Nicolas Chamfort*

Résumé

Il est des sujets parmi les activités cognitives humaines que l'intelligence artificielle étudie peu : comprendre le rire en est un exemple. Cet article, exploratoire et préliminaire, cherche à identifier les mécanismes à l'œuvre dans les mots d'esprit et les blagues à caractère narratif. Il semble que dans nombre d'entre eux, un mécanisme de révision de croyances opère, conduisant à une conclusion inattendue, et assurant la chute finale de la plaisanterie.

Mots Clef

Humour, révision de croyances, surprise.

Abstract

There are some human cognitive activities that have remained little studied in artifial intelligence : understanding laughter is an example. This paper, exploratory and preliminary, tries to identify the mechanisms at work in quips and narrative jokes. It seems that in many cases a belief revision mechanism is operating, leading to an unexpected conclusion, through the punchline of the jest.

Keywords

Humor, belief revision, surprise.

1 Introduction

A côté des multiples tâches spécifiques auxquelles s'est attaqué, avec succès, l'intelligence artificielle (IA) depuis ses débuts, comme la formalisation de différents types de raisonnement, la résolution de problèmes de satisfaction de contraintes, ou l'interprétation d'images, il en est d'autres qui ont été beaucoup moins étudiées, sans doute parce qu'elles correspondaient à des enjeux jugés moins importants, moins sérieux, ou qu'elles paraissaient plus délicates. Ainsi deux exemples de ces domaines particulièrement complexes et difficiles à appréhender sont l'humour et l'art. Ces deux questions présentent d'ailleurs quelques similarités. Dans les deux cas un "émetteur", le blagueur ou l'artiste présente un contenu expressif à un "récepteur", le public, supposé capable d'apprécier ce qu'on lui destine. Humour et art ont aussi en commun de souvent jouer avec, voire de bousculer, les interdits et les préjugés, au travers de leur propos. Dans les deux cas, il semble qu'il y ait un effet de surprise [25, 33] pour le public lors de la découverte de ce qui est présenté. Très schématiquement, tandis que l'art s'adresse d'abord à notre sensibilité et suscite nos émotions, il semble que la blague sollicite avant tout une appréhension logique du monde (on parle de "comprendre une plaisanterie"). De plus, la subtilité de la mécanique humoristique stimule l'intelligence et fait réfléchir. L'humour parlerait à notre intelligence réflexive, spéculative, tandis que l'art aiguillonnerait d'abord la part réactive de notre intelligence, et convoquerait davantage différentes formes de ressenti [8]. Sans doute les choses ne sont pas aussi simples, et de fait nombre d'œuvres d'art ont une dimension conceptuelle [22]. Peut-être, plus subtilement, on pourrait se risquer à dire que dans l'énonciation d'une blague on installe une situation présentant une incohérence que la chute va résoudre de manière surprenante, tandis que l'art cultiverait plus l'ambiguïté, le décalage, l'excès [36, 26]. Cependant l'humour comme l'art posent des problèmes de perception, d'analyse et de synthèse pour le public, dans les deux cas particulièrement difficiles à modéliser.

La littérature en IA sur l'humour est assez réduite. Une grande part des articles concernent la reconnaissance de situations d'ironie, ou de rire dans des textes, dans des signaux audio ou des images, voire l'évaluation d'un niveau de drôlerie, e.g., [29]. Quelques autres travaux s'intéressent à la génération de jeux de mots, ou à la dérivation d'acronymes détournés, e.g., [31]. Marvin Minsky [20] discute

*Sébastien-Roch Nicolas de Chamfort (1741-1794) Maximes, Pensées, Caractères et Anecdotes. Garnier-Flammarion, Collection GF, n°188, 1968. Il s'agit de la maxime n°80.

la théorie freudienne des plaisanteries [14], et met en évidence que au-delà des questions d'interdits, l'humour est aussi affaire de connaissance sur la connaissance, et de reconnaissance et de suppression d'incohérences.

L'ambition, modeste, de cet article est d'abord d'essayer de cerner les contours d'un domaine peu exploré, et plus particulièrement, dans une blague ou un bon mot, de comprendre et de modéliser le mécanisme provoquant l'effet susceptible de déclencher le rire. Nous nous en tiendrons à cette étape d'analyse sans considérer le problème de la synthèse de plaisanteries.

Nous passons d'abord en revue les notions liées au rire dans la section 2 pour ensuite s'intéresser à la modélisation en IA de la notion de surprise en section 3, la section 4 présente l'essentiel de notre contribution qui concerne la formalisation du mécanisme à l'œuvre dans la plupart des plaisanteries. L'article se termine par un retour sur quelques articles d'IA touchant à l'humour et conclut.

2 Les notions liées au rire

Tout le monde connait la phrase de Rabelais *Rire est le propre de l'homme* (« Mieulx est de ris que de larmes escripre, pour ce que rire est le propre de l'homme. » [24]). Rabelais s'inscrit dans la tradition d'Aristote qui souligne que le rire est une qualité spécifiquement humaine.

Selon Attardo et Raskin [5] il y a trois principales théories sur l'humour : la théorie du soulagement (l'humour aiderait à relâcher la tension nerveuse), la théorie de la supériorité (l'humour permettrait de se sentir supérieur), et la théorie de l'incongruité (l'humour consiste à désobéir aux schémas mentaux et attentes). Cette dernière théorie est la théorie dominante qui est aussi la perspective dans laquelle nous nous plaçons dans cet article (même si on peut arguer que la chute de la blague, amenant un certain retour à la cohérence, apaise une dissonance cognitive, ce qui semble aller dans le sens de la première théorie).

Beaucoup d'auteurs ont essayé de cerner les notions liées au rire telles que l'humour, le comique, l'ironie, le calembour, la plaisanterie, le cocasse, ou la dérision. Très tôt on a compilé des recueils de bons mots et de blagues, e.g., [4, 21]. [1] Ceux-ci opèrent généralement une classification thématique qui met en évidence les différentes variétés de sujet ou de registre (chez le docteur, l'argent, la religion, l'étranger, le sexe, ...) ou de forme (histoires drôles, calembours, devinettes, contrepèteries, ...) propices à l'élaboration de plaisanteries.

1. Et les blagues d'il y a près de 2000 ans [4] font encore souvent (sou)rire : *C'est un intellectuel plein d'humour qui est à court d'argent et qui vend ses livres. Il écrit alors à son père : « J'ai une bonne nouvelle : je commence à vivre de mes livres »*, ou encore : *C'est un intellectuel qui s'est acheté un pantalon. Comme il est serré et qu'il a du mal à l'enfiler, il se fait épiler les jambes.* Le lecteur pourra à titre d'exercice formuler ces blagues dans le langage que nous proposons en Section 4. Notons que la première blague en utilisant les termes « intellectuel plein d'humour » gâche un peu son potentiel comique, et illustre déjà notre Propriété 2 puisque la surprise de sa chute est d'autant moins forte qu'on s'attend dès le début à ce que cette personne pleine d'humour en fasse preuve.

Certaines choses semblent être drôles par elle-même (le nez rouge du clown, une personne qui tombe, un langage amphigourique, ...), c'est la matière du comique, que met en évidence Bergson dans son célèbre livre sur le rire [7]. Son ouvrage comme son sous-titre l'indique est consacré à la signification du comique. Il identifie trois critères nécessaires au rire :

- l'objet du rire devrait se rapporter directement ou indirectement à des êtres humains ;
- le rieur devrait avoir mis de côté ses émotions ;
- le rire serait un acte social.

Selon Bergson « Est comique tout arrangement d'actes et d'événements qui nous donne, insérées l'une dans l'autre, l'illusion de la vie et la sensation nette d'un agencement mécanique ». Ainsi le rire proviendrait de l'incongruité de plaquer du mécanique sur du vivant. Bergson distingue cinq effets comiques :

- le diable à ressort : « Passons alors au théâtre. C'est par celui de Guignol que nous devons commencer. Quand le commissaire s'aventure sur la scène, il reçoit aussitôt, comme de juste, un coup de bâton qui l'assomme. Il se redresse, un second coup l'aplatit. Nouvelle récidive, nouveau châtiment. Sur le rythme uniforme du ressort qui se tend et se détend, le commissaire s'abat et se relève, tandis que le rire de l'auditoire va toujours grandissant. » ;
- le pantin à ficelles : lorsque l'on voit qu'une personne qui se croit libre est en réalité le jouet d'une autre personne ;
- l'effet boule de neige : engrenage involontaire d'actions de plus en plus énormes ;
- la répétition ;
- l'inversion : l'arroseur arrosé par exemple.

Tout ce qui précède ressort du comique qui est à distinguer de l'ironie. L'ironie est plus subtile. « L'ironie remet tout en question ; par ses interrogations indiscrètes elle ruine toute définition, dérange à tout moment la pontifiante pédanterie prête à s'installer dans une déduction satisfaite. Grâce à l'ironie, la pensée respire plus légèrement quand elle s'est reconnue, dansante et grinçante, dans le miroir de la réflexion » [16]. Elle avance souvent masquée, procède par sous-entendu, et il faut être capable de la déceler, de la reconnaître. C'est d'ailleurs un problème en Traitement Automatique des Langues dans le cadre de la reconnaissance d'opinion. Pour surmonter cette difficulté, on a pu proposer pour l'écrit la création et l'usage d'un point d'ironie : ؟ [2]. L'ironie peut se faire agressive en restant sur un mode plaisant [1], et devenir complexe dans son expression littéraire [35].

L'ironie se distingue du comique et de l'humour, dont il est maintenant question. Dans son article [25] sur les mots d'esprit, Raccah établit un parallèle entre métaphore et bon mot qui tous les deux reposent sur un effet 'manipulatoire' au plan de la communication. Il différencie le comique

et l'humour par le fait que le premier fait rire du comportement volontaire ou non du locuteur alors que le second nécessite trois conditions :

— un énoncé ;

— une intention de faire rire ;

— que le propos cause le rire.

Raccah propose comme explication d'un bon mot qui nous fait rire « la sensation d'être tombé dans un piège inattendu et inévitable : c'est la *chute* qui nous fait tomber dans ce piège ».
Dans notre étude, nous nous restreignons à l'humour et donc nous ne traiterons ni de l'ironie ni des situations comiques comme celles étudiées par Bergson. Cependant il serait possible de retrouver des éléments de la définition de Raccah dans les situations comiques : par exemple dans la situation du diable à ressort, le spectateur peut penser que ce n'est pas possible de continuer à donner des coups de bâtons au gendarme ; la surprise, et donc le rire, vient du fait que ça ne s'arrête pas. Pour le pantin à ficelle, le spectateur peut supposer que le pantin va comprendre qu'il est manipulé, la surprise viendrait de ce qu'il ne s'en rend pas compte ...

3 Les surprises en IA

La surprise naît de la survenue de quelque chose jugée pratiquement impossible. La première version de la théorie des possibilités [10] qu'avait proposée l'économiste anglais G. L. S. Shackle [28] reposait sur la notion de degré de surprise associé à un évènement, qui était en fait un degré d'impossibilité de cet évènement, calculé à partir d'une distribution de possibilités reflétant notre connaissance incertaine de l'état du monde considéré. Un fait est alors d'autant plus surprenant qu'il est moins cohérent avec ce qu'on s'imaginait possible. Formellement,
$\texttt{degré de surprise}(A) = 1 - \texttt{possibilité}(A)$.

En IA, la question des surprises a été étudiée par Lorini et Castelfranchi [19]. Ces auteurs ont formulé deux notions de surprise différentes :

— *mismatch-based surprise* qui traduit l'incompatibilité entre ce que l'on perçoit et ce que l'on s'attendait à percevoir (*scrutinized expectation*)

— *astonishment* ou *surprise in recognition* qui traduit la difficulté d'accepter ce que l'on perçoit car cette chose perçue est très peu plausible dans l'absolu.

Dans le deuxième cas, les auteurs évoquent le fait que la personne dans cette situation peut marquer un temps d'arrêt, d'incompréhension car elle a de la difficulté à intégrer la nouvelle perception, « elle ne peut pas y croire ». Dans les deux cas le degré de surprise ou d'étonnement est associé à une probabilité. Dans le premier cas, la surprise est d'autant plus grande que la probabilité associée à l'attente initiale est élevée ; dans le second cas, l'étonnement est d'autant plus grand que la chose perçue est improbable.
Lorini et Castelfranchi proposent aussi une formalisation en logique modale avec probabilités (*logic of probabilistic quantified beliefs*) dont la sémantique est celle donnée par Fagin et Halpern [13] à laquelle un opérateur d'action standard est ajouté ainsi que des constructeurs pour parler de ce qui en train d'être examiné (en mémoire de travail : *representation under scrutiny*) et des données perçues. Ils ont modélisé les deux types de surprise dans cette logique. Ils ont ensuite proposé de modéliser l'intégration cognitive qui suit une surprise par un opérateur de changement de croyance qu'ils nomment *update process*. Ainsi chez ces auteurs la notion de surprise est d'abord définie par une forme d'inconsistance par rapport à des attentes explicites ou par rapport à des connaissances a priori. Puis une fois la surprise arrivée, l'opérateur doit mettre à jour ses connaissances.
Dans notre étude, nous proposons une approche différente puisque nous utilisons le changement de croyance pour définir la surprise (et non pour modéliser son intégration chez l'auditeur) : la nouvelle information (la chute de l'histoire) génère une révision des croyances dont le résultat est surprenant dans le sens où il est inconsistant avec ce que l'on croyait au départ. Pour plus de détails sur la révision de croyances et la mise à jour, le lecteur pourra consulter l'appendice A.

Le terme surprise est également apparu dans le cadre de l'extrapolation de scénario [11], l'objectif est de minimiser les surprises : dans le sens où une trajectoire est d'autant plus surprenante qu'elle ne respecte pas les lois statiques et dynamiques du monde étant données des observations ou des informations sur l'occurrence d'évènements à différents instants. Cette minimisation des surprises est une extension de celle introduite dans les opérateurs d'extrapolation de croyances [12] (dans un contexte plus restreint où l'on suppose que les fluents sont par défaut inertes et les surprises sont des changements imprévus de valeurs).
Notons que dans cet article nous nous restreindrons à considérer des situations statiques dans lesquels les nouvelles connaissances apprises sont surprenantes, c'est pourquoi dans cet article la notion de surprise ne correspondra pas à une évolution surprenante du monde, mais plutôt à une évolution surprenante des connaissances sur le monde. Ainsi, notre travail se situe donc dans le domaine de la révision de croyances plutôt que dans le domaine de la mise à jour [34, 18].

4 Formalisation des plaisanteries

Dans l'humour, selon Raccah [25], le piège est créé grâce au récit utilisé dans un but manipulatoire. Avec l'humour, il y a une manipulation de ce que l'on veut faire croire à l'auditeur afin qu'il tombe dans le piège. Dans cet article sur l'humour, nous proposons simplement de traduire l'analyse de Raccah en termes de révision de croyances. Ainsi, la manipulation de l'auditeur sera vue comme un processus en deux phases :

— La première phase est une révision des croyances de

l'auditeur avec des informations incomplètes destinées à suggérer une conclusion (qui s'avèrera fausse).

— La deuxième phase correspond à l'arrivée de la chute de l'histoire, qui d'une part surprend car elle est incompatible ou incongrue par rapport à la conclusion provisoire précédente, ce qui se traduira dans notre modèle par une inconsistance. D'autre part, la chute semble inéluctable ce que nous exprimons par le fait qu'elle explique logiquement les informations initiales.

Le deuxième enseignement que l'on tire de l'article de Raccah porte sur l'intensité du caractère drôle : selon Raccah, « le trait est d'autant plus drôle que le piège était inattendu et inévitable ». Cela nous incite à proposer un degré de drôlerie qui sera fonction de la différence de niveau de normalité entre les mondes obtenus dans les deux phases d'une part et de l'inéluctabilité de l'explication que fournit la chute à la situation initiale d'autre part.

4.1 Langage

On considère un langage propositionnel $\mathcal{L}$, où les propositions sont notées par des symboles grecs en minuscule ; les symboles $\bot$, $\top$, $\vee$, $\wedge$, $\neg$, $\rightarrow$, $\equiv$, $\vdash$, $\models$ dénotent respectivement la contradiction, la tautologie, les connecteurs logiques « ou », « et », « non », l'implication matérielle, l'équivalence logique, l'inférence classique, la satisfaction. On utilise $\rightsquigarrow$ pour représenter une règle par défaut : $a \rightsquigarrow b$ qui est interprétée comme la contrainte [2] $N(b|a) > 0$ signifiant qu'il est plus plausible que b soit vrai en présence de a plutôt que b soit faux [6]. Cette écriture suppose l'existence d'une relation de plausibilité sur les mondes, que nous représenterons par une distribution de possibilité π. Dans ce contexte, une base de connaissances K est un ensemble de règles par défaut $\mathcal{L} \times \mathcal{L}$ de la forme $a \rightsquigarrow b$ où $a, b \in \mathcal{L}$.
Dans la suite, nous utilisons un opérateur de *révision de croyances* [3, 17] dont les principes sont rappelés en appendice. La révision des croyances K par l'information φ est notée $K \circ \varphi$.

4.2 Formalisation

Dans cette section, nous proposons une formalisation *statique* des plaisanteries en logique propositionnelle, c'est-à-dire que l'on considère que la plaisanterie décrit une situation α et que la chute vient compléter cette description par une information β. Les informations α et β sont décrites par des propositions. Nous considérons que la plaisanterie est écoutée par un auditeur dont la base de connaissances est un ensemble de formules propositionnelles dénotée K.

Définition 1 (énoncé, énoncé à tiroir)
Une énoncé simple est une paire (α, β) de formules de $\mathcal{L}$. Une énoncé à tiroir est un n-uplet avec $n > 2$, $(\varphi_1, \ldots, \varphi_n)$, de formules de $\mathcal{L}$.

La situation cognitive induite par un énoncé est décrite par la base de connaissances K révisée itérativement par les éléments de l'énoncé. La chute d'une plaisanterie est considérée comme *surprenante* si le résultat de la révision des connaissances K par la description initiale de la situation est contradictoire avec ce que l'on obtient après révision par la chute.

Définition 2 (énoncé surprenant)
Un énoncé (α, β) est surprenant *relativement à K ssi*

$$(K \circ \alpha) \textit{ consistant}^{3} \textit{ et } (K \circ \alpha) \cup (K \circ (\alpha \wedge \beta)) \vdash \bot$$

Un énoncé à tiroir $(\varphi_1, \ldots, \varphi_n)$ est surprenant à l'étape i relativement à K ssi

$$(K \circ \psi) \cup (K \circ (\psi \wedge \varphi_i) \vdash \bot$$

avec $\psi = \varphi_1 \wedge \varphi_2 \ldots \wedge \varphi_{i-1}$ et $K \circ \psi$ consistant.

Pour comprendre la plaisanterie il faut que sa logique paraisse implacable une fois la chute révélée, c'est donc que la chute est à la fois tout à fait recevable et explique la situation. En d'autres termes, si l'on avait su β dès le début, il aurait expliqué α. Ce que l'on peut traduire ainsi :

Définition 3 (énoncé à chute révélatrice)
Étant donné un énoncé (α, β), sa chute β est révélatrice *relativement à K ssi*

$$(K \circ \beta) \textit{ consistant et } K \circ \beta \models \alpha$$

On peut penser que le rire vient soulager la tension de dissonance cognitive provoquée par l'inconsistance entre ce que l'on s'attendait à entendre et la chute de l'histoire. Ce soulagement ne peut avoir lieu qu'une fois l'histoire comprise, la chute jouant le rôle d'une révélation.

Définition 4 (énoncé potentiellement drôle) *Un énoncé est* potentiellement drôle *ssi il est surprenant et sa chute est révélatrice.*

Cette définition d'énoncé potentiellement drôle est directement inspirée de Raccah [25], à ceci près que nous avons essayé de définir le caractère révélateur de la chute plutôt que son côté inévitable (qui pourrait se comprendre comme $K \circ \alpha \vdash \beta$, niant le caractère surprenant de β).

Notons qu'on a écrit « potentiellement drôle » car on peut imaginer des exemples d'énoncés surprenants à chutes révélatrices mais qui ne sont pas nécessairement drôles pour leur auditeur. Par exemple, une découverte scientifique peut être surprenante et révélatrice, tout comme une

2. Cette contrainte utilise une mesure de nécessité conditionnelle. En théorie des possibilités [10], les mesures de nécessité sont duales des mesures de possibilité Π : $N(a) = 1 - \Pi(\neg a)$. La contrainte $N(b|a) > 0$ s'écrit de manière équivalente en $\Pi(a \wedge b) > \Pi(a \wedge \neg b)$ ce qui exprime que, dans le contexte a, b vrai est la situation la plus normale.

3. Cette condition est automatiquement assurée par les postulats habituels de la révision (voir Appendice) sitôt que α est consistant.

histoire policière bien ficelée, ou encore certaines révélations politiques, pour les publics concernés. Cette remarque montre que d'autres ingrédients que le mécanisme à base de révision de croyances ci-dessus sont également nécessaires à la survenue du rire. Ainsi, on peut citer les éléments évoqués en Section 2, tels que l'intention de faire rire, l'absence d'empathie, ou encore le jeu avec des tabous.

Après cette dernière définition, nous sommes en mesure de formaliser un premier exemple (issu d'un recueil d'histoires « drôles » [21]) exprimé en langage naturel que nous avons transposé en logique. Dans cet article, nous faisons abstraction de l'étape du passage de l'énoncé en langage naturel à une représentation logique.

Exemple 1 *Un homme vient de se faire renverser par une voiture. Le conducteur sort de la voiture et dit : Vous êtes bien chanceux on est juste devant le bureau du médecin. Oui ! sauf que le médecin c'est moi !*

Modélisation :
$\alpha = blesse \wedge proche\_medecin$,
$\beta = blesse \wedge medecin\_meme$,
$K = \left| \begin{array}{l} blesse \wedge proche\_medecin \rightsquigarrow sera\_vite\_soigne \\ blesse \wedge \neg medecin\_soignant \rightsquigarrow \neg sera\_vite\_soigne \\ medecin\_meme \rightsquigarrow proche\_medecin \\ blesse \wedge medecin\_meme \rightsquigarrow \neg medecin\_soignant \end{array} \right.$
Calculons maintenant $K \circ \alpha$, les modèles les plus plausibles satisfont $\varphi = \alpha \wedge sera\_vite\_soigne$.
Les modèles les plus plausibles de $K \circ (\alpha \wedge \beta)$ satisfont $\psi = proche\_medecin \wedge medecin\_meme \wedge \neg medecin\_soignant \wedge \neg sera\_vite\_soigne$. La chute est donc bien surprenante puisque $\varphi \wedge \psi \vdash \bot$.
D'autre part, la chute explique α :

$$K \circ \beta \vdash blesse \wedge proche\_medecin$$

$proche\_medecin$ correspond à la partie de l'énoncé qui nous avait manipulé pour nous faire penser que le blessé était chanceux.

La propriété suivante montre l'importance de l'ordre narratif dans la plaisanterie.

Propriété 1 (ne pas réveler la chute avant la fin !) *Si (α, β) est un énoncé potentiellement drôle pour un auditeur sachant K, alors (β, α) n'est pas potentiellement drôle pour cet auditeur.*

Preuve : Puisque (α, β) est potentiellement drôle, c'est que la chute est révélatrice, c'est-à-dire que $K \circ \beta \nvdash \bot$ et $K \circ \beta \models \alpha$. Mais ceci empêche l'énoncé (β, α) d'être surprenant, et donc d'être potentiellement drôle. En effet, en utilisant les postulats K5 et K6, $K \circ (\beta \wedge \alpha) \equiv (K \circ \beta) \cup \{\alpha\}$ donc $K \circ \beta$ est consistant avec $K \circ (\beta \wedge \alpha)$. □

La propriété suivante indique la vertu de la concision.

Propriété 2 (les blagues les plus courtes...) *Si (α, β) est un énoncé potentiellement drôle sachant K alors l'ajout d'$\alpha' \in \mathcal{L}$, tel que $\alpha \nvdash \alpha'$, peut rendre l'énoncé $(\alpha \wedge \alpha', \beta)$ non potentiellement drôle.*

Preuve : En effet, soit $K = \{o \rightsquigarrow v, m \rightarrow o, m \rightsquigarrow \neg v, o \wedge a \rightsquigarrow m\}$ une base de connaissances exprimant dans l'ordre que : généralement les oiseaux (o) volent (v), les manchots (m) sont des oiseaux, généralement les manchots (m) ne volent pas, la plupart des oiseaux de l'Antarctique (a) sont des manchots. Imaginons un énoncé (o, m), il est à la fois surprenant puisque $K \circ o \models o \wedge \neg m \wedge \neg a \wedge v$ et $K \circ (o \wedge m)) \models o \wedge m \wedge \neg v$ et la chute est révélatrice puisque $K \circ m \models o$. Cependant en ajoutant l'information que l'oiseau est de l'Antarctique on peut créer une situation non surprenante vis à vis de $\beta = m$, ainsi $(o \wedge a, m)$ n'est plus un énoncé surprenant. D'autre part, si on ajoute une information qui n'a pas de rapport avec β alors on peut obtenir une chute qui n'est pas complètement révélatrice, considérons $\alpha' = t$, où t signifie « s'appelle Titi » alors $K \circ m \vdash o$ ne permet pas de déduire que $K \circ m \vdash t$ l'énoncé a donc une chute qui n'est pas nécessairement révélatrice. □

Pour se convaincre de la validité de la Propriété 2, on peut considérer le cas particulier $\alpha' = \beta$. Ceci correspondrait dans l'Exemple 1 à préciser que le conducteur vient de renverser un médecin traitant devant le cabinet de ce même médecin. Dans ce cas, la chute « Oui ! sauf que le médecin c'est moi ! » n'est plus drôle (car on le sait déjà). Néanmoins la nouvelle histoire peut être trouvée drôle différemment avec $\alpha'' =$ *Un homme vient de renverser son propre médecin traitant devant son cabinet* et $\beta'' =$ *« Vous êtes bien chanceux, on est juste devant le bureau de mon médecin »*. Ici, l'énoncé est bien surprenant cependant la révélation ne peut venir que si l'on perçoit l'ironie de l'utilisation du mot chanceux.

On peut remarquer que la Propriété 2 n'impose pas que la chute soit la plus courte possible, on peut donc affiner la définition de « drôle » en « efficace » en imposant la minimalité.

Définition 5 (concision de la chute) *Un énoncé (α, β) est efficace ssi il est potentiellement drôle et pour tout β' t.q. $\beta \models \beta'$, (α, β') n'est pas potentiellement drôle.*

Revenons sur l'Exemple 1, cet exemple peut aussi être vu comme une plaisanterie à tiroir $(\varphi_1, \varphi_2, \varphi_3, \varphi_4)$ avec $\varphi_1 = blesse$, $\varphi_2 = chanceux$, $\varphi_3 = proche\_medecin$, $\varphi_4 = medecin\_meme$. En effet, dire « Vous êtes bien chanceux » après avoir vu que la personne est blessé peut déjà provoquer le rire. Ensuite la révélation d'être « devant le bureau d'un médecin » permet de comprendre. Enfin la seconde partie de la plaisanterie correspond à ce qui a déjà été analysé. Notons qu'ici dans cette première partie d'énoncé itéré φ_2 est surprenant mais non révélateur, c'est φ_3 qui explique φ_2. Ainsi pour une plaisanterie à tiroir,

le caractère surprenant et le caractère révélateur ne sont pas nécessairement simultanés. En voici un autre exemple d'après [21].

Exemple 2 *Un gars et une bonne femme rentrent dans le cabinet du médecin. Celui-ci, se tournant vers la dame, lui dit :*
— Si vous êtes malade, déshabillez-vous, je vous prie... mais la fille fait des manières. Elle a des réticences. Elle regarde le gars par en-dessous. Alors le médecin répète :
— Mais enfin, madame, déshabillez-vous ! Je suis docteur. La pudeur n'est pas de mise ici ! Alors elle se met à gigoter et tout d'un coup, elle fond en larmes. Décontenancé, le médecin demande au gars :
— Qu'est-ce qu'elle a, votre femme ? Elle est toujours aussi nerveuse ?
— Je ne sais pas, dit l'autre. Je viens juste de faire sa connaissance dans votre salle d'attente....

Modélisation :
$\varphi_1 = ensemble, \quad \varphi_2 = reticences, \quad \varphi_3 = \neg couple$
$K = \left| \begin{array}{l} ensemble \rightsquigarrow couple \\ couple \rightsquigarrow \neg reticences \\ \neg couple \rightsquigarrow reticences \end{array} \right.$
Calculons maintenant $K \circ \varphi_1$*, les modèles les plus plausibles satisfont* $\psi = ensemble \wedge couple \wedge \neg reticences$*.* $K \circ (\varphi_1 \wedge \varphi_2)$ *satisfait* $reticences$*.* φ_2 *est donc surprenante puisque* $\psi \wedge (K \circ (\varphi_1 \wedge \varphi_2)) \vdash \bot$*.*
D'autre part la chute φ_3 *explique* φ_2 *:* $(K \circ \varphi_3) \models reticences$*.*

Enfin, pour donner corps à l'intuition de Raccah selon laquelle la plaisanterie est d'autant plus drôle que la surprise et le caractère inévitable sont grands, on peut proposer l'évaluation suivante :

Définition 6 (degré de drôlerie) *Le degré de drôlerie d'une plaisanterie* (α, β) *est :*

$$\max(Incons((K \circ \alpha) \cup (K \circ (\alpha \wedge \beta)), N(\alpha | K \circ \beta)))$$

Cette définition suppose que les règles par défauts de K sont codées en logique possibiliste [6], $Incons(A)$ représentant le niveau d'inconsistance de la base de connaissance possibiliste A. Le deuxième terme est la nécessité conditionnelle de α sachant $K \circ \beta$, reflètant l'inéluctabilité de la révélation.

De plus, n'oublions pas que d'autres éléments peuvent contribuer au degré de drôlerie. On peut penser à l'effort cognitif nécessaire pour comprendre la plaisanterie, le registre, les effets comiques du narrateur [4], l'humeur de l'auditeur, sa capacité d'inhibition de ses émotions...

4. En particulier, on peut citer les décalages de niveau de langage présents dans les parodies burlesques, comme dans le poème de Georges Fourest parodiant Le Cid qui se termine ainsi :

« Dieu ! » soupire à part soi la plaintive Chimène,
« qu'il est joli garçon l'assassin de Papa ! »

La Négresse Blonde (1909)

Cette section a proposé la modélisation d'un aspect important des plaisanteries en termes de révision de croyances, illustré sur quelques exemples. Il est clair qu'une validation de la pertinence de ces idées demanderait leur vérification sur un corpus conséquent de blagues. Même si les auteurs ont l'impression que de fait beaucoup de blagues fonctionnent de cette manière.

Il faut enfin noter que d'autres mécanismes peuvent être à l'œuvre dans les blagues narratives. Ainsi un procédé qu'on rencontre relativement souvent, est la mise en parallèle de plusieurs situations, ce qui s'apparente à l'analagie, où le raisonnement n'est plus déductif, mais l'effet de surprise final toujours aussi grand. En voici un exemple [5] :
Dans le cadre de fouilles jusqu'à 100 m de profondeur, dans le sous-sol russe, les scientifiques ont trouvé des vestiges de cuivre qui dataient d'environ 1000 ans.
Les Russes en ont conclu que leurs ancêtres disposaient déjà il y a 1000 ans d'un réseau de fil de cuivre.
Les Américains, pour faire bonne mesure, ont également procédé à des fouilles dans leur sous-sol jusqu'à une profondeur de 200 m. Ils y ont trouvé des restes de verre. Il s'est avéré qu'ils avaient environ 2000 ans.
Les Américains en ont conclu que leurs ancêtres disposaient déjà il y a 2000 ans d'un réseau de fibre de verre numérique. Et cela, 1000 ans avant les Russes !
Une semaine plus tard, en Belgique on a publié le communiqué suivant : « Suite à des fouilles dans le sous-sol belge jusqu'à une profondeur de 500 m les scientifiques belges n'ont rien trouvé du tout. Ils en concluent que les Anciens Belges disposaient déjà il y a 5000 ans d'un réseau Wifi ! »

5 Autres travaux d'IA sur l'humour

Comme souligné dans le livre de Raskin [27], peu de chercheurs se sont consacrés exclusivement à la recherche sur l'humour. On peut cependant noter des publications dans les conférences d'intelligence artificielle comme celle de Hempelmann et collègues dans FLAIRS en 2006 [15] qui proposent d'utiliser les ontologies pour analyser et générer des plaisanteries automatiquement selon des scripts prédéfinis. Ces auteurs se basent sur la théorie en linguistique computationelle d'Attardo et Raskin [5] selon laquelle les six principales ressources participant à la drôlerie d'une plaisanterie sont (par ordre décroissant d'importance) : l'opposition de script (SO), le mécanisme logique (LM), la situation (SI), la cible (TA), la stratégie narrative (NS) et le langage (LA). Les auteurs choisissent ensuite de se consacrer aux deux premiers, et c'est aussi ce que nous avons proposé dans cet article puisque nous inspirant de l'article de Raccah cité plus haut, nous nous sommes basés sur la surprise et le caractère révélateur de la chute. La surprise est liée à la contradiction entre les deux phases de la plaisanterie (SO), le caractère révélateur est associé au mécanisme logique (LM).

5. Tiré de `http://www.bernardwerber.com/interactif/histoiresdroles.php?page=2&tri=1`

On peut également citer les travaux de Stock et Strapparava [31] publiés à l'IJCAI en 2003 qui présentent HAHACRONYM un système de production d'acronymes humoristiques à partir d'acronymes existants (par exemple, F B I - Federal Bureau of Investigation devient Fantastic Bureau of Intimidation, ou encore MIT - Massachusetts Institute of Technology devient Mythical Institute of Theology). Notons qu'il s'agit de la génération d'un énoncé surprenant et révélateur au sens où il explique la série de lettres d'une façon inattendue.
Shahaf et collègues [29] ont travaillé sur l'analyse automatique du degré de drôlerie de la légende humoristique associée à une image, ils ont utilisé une base de milliers de légendes recueillies à l'occasion d'un concours organisé par le New York Times pour élire la légende la plus drôle associée à une même image. Les auteurs ont fait annoter les légendes par paires en demandant aux participants d'indiquer la plus drôle des deux légendes pour chaque paire. Un classifieur de légendes a ensuite été construit qui prédit étant donné un dessin et deux légendes, quelle légende est la plus drôle. Puis en simulant un tournoi le système permet d'élire la légende la plus drôle de toutes. Les caractéristiques utilisées pour formuler la tâche en termes d'apprentissage sont : le caractère inhabituel du langage utilisé, le caractère imprévisible de la légende, la complexité grammaticale. Là encore, on retrouve la nécessité du caractère surprenant pour la drôlerie.

6 Conclusion

Il arrive que certains articles ou écrits à allure scientifique, soient des plaisanteries [32, 9, 23]. De deux choses l'une, soit le présent article est une plaisanterie à propos d'articles scientifiques, soit c'est un article scientifique à propos des plaisanteries ¿! Cela dépend de la surprise que cette conclusion provoque chez le lecteur par rapport à ce qu'il s'attendait à trouver. ¿!

Appendice : Révision de croyances

Dans le cadre du changement de croyances, l'article d'Alchourròn, Gärdenfors et Makinson (AGM) [3] a introduit le concept de révision de croyances. La révision de croyances consiste à déterminer ce qu'il reste des anciennes croyances après l'arrivée d'une nouvelle information. Les croyances sont représentées par des expressions dans un langage formel. AGM ont défini trois types de changement de croyances, la contraction, l'expansion et la révision. L'expansion consiste simplement à ajouter l'information sans vérifier sa cohérence avec les connaissances antérieures. La contraction permet de retirer une information. La révision consiste à ajouter une information tout en préservant la cohérence. Cette dernière opération est nécessaire puisque l'incohérence entraîne un état de croyance inexploitable. Le principal apport de l'article d'AGM est la définition d'un ensemble de postulats qui doivent être satisfait par tout opérateur de révision dit *rationnel*. Comme remarqué dans [30] ces postulats sont fondés sur trois principes :
— un principe de consistance (le résultat doit être consistant),
— un principe de minimisation du changement (on doit modifier le moins possible les croyances initiales),
— un principe de priorité à la nouvelle information (la nouvelle information doit être vérifiée après la révision).
Nous rappelons ci-dessous l'ensemble des postulats énoncés par Katsuno et Mendelzon [18] qui sont équivalents à ceux d'AGM, mais permettent plus facilement de relier un opérateur de révision à une relation de distance entre interprétations. Plus formellement, un opérateur de révision KM associe à un ensemble de formules K et à une formule φ (représentant la nouvelle information), un autre ensemble de croyances noté $K \circ \varphi$. Pour être considéré *rationnel* l'opérateur $\circ$ doit satisfaire les postulats KM :
$K1$: $K \circ \varphi \models \varphi$
$K2$: si $K \cup \{\varphi\}$ satisfiable, alors $K \circ \varphi \equiv K \cup \{\varphi\}$.
$K3$: si φ est satisfiable, alors $K \circ \varphi$ l'est aussi.
$K4$: Si $K_1 \equiv K_2$ et $\varphi_1 \equiv \varphi_2$ alors $K_1 \circ \varphi_1 \equiv K_2 \circ \varphi_2$
$K5$: $(K \circ \varphi) \cup \{\psi\} \models K \circ (\varphi \wedge \psi)$
$K6$: si $(K \circ \varphi) \cup \{\psi\}$ est satisfiable alors $K \circ (\varphi \wedge \psi) \models (K \circ \varphi) \cup \{\psi\}$
$K1$ impose que la nouvelle information doit être vérifiée après la révision. $K2$ impose que lorsque la nouvelle information n'est pas contradictoire avec les croyances initiales alors la révision est une simple expansion. $K3$ exprime que si la nouvelle information est consistante alors l'ensemble de croyances révisées l'est aussi. $K4$ exprime qu'un opérateur de révision est indépendant de la syntaxe. Ces 4 premiers postulats sont les postulats basiques de révision, les deux derniers expriment la minimisation du changement. $K5$ implique que réviser par une conjonction $\varphi \wedge \psi$ doit donner un état de croyances plus précis que ce que l'on peut déduire logiquement de K révisé par φ auquel on a ajouté par union l'information ψ. $K6$ signifie que lorsqu'on révise K par $\varphi \wedge \psi$, toute déduction logique à partir de ψ et $K \circ \varphi$ fera partie de la croyance révisée à condition que ψ ne soit pas contradictoire avec $K \circ \varphi$.
Katsuno et Mendelzon ont énoncé le théorème de représentation suivant, permettant d'exprimer une révision en termes de modèles proches :

Théorème 1 ([17]) $\circ$ *satisfait les postulats* $(K1)$ *-* $(K6)$ *ssi il existe une fonction associant fidèlement à chaque état épistémique* φ *un pré-ordre total* $\leq_\varphi$ *tel que :*

$$Mod(\varphi \circ \psi) = \min(Mod(\psi), \leq_\varphi)$$

L'association est fidèle lorsqu'elle associe à toute formule φ un pré-ordre $\preceq_\varphi$ qui privilégie strictement les interprétations satisfaisant φ aux autres.
Dans le contexte d'une révision de croyances, l'information φ est une nouvelle connaissance sur le monde qui est considéré comme statique. Lorsque φ représente une évolution du monde et non pas une évolution des connaissances sur le monde, alors l'opération est appelée *mise à*

jour [34, 18]. La mise à jour possède ses propres postulats et son propre théorème de représentation, mais cette question va au-delà de la modélisation utilisée dans cet article.

Références

[1] René Marill Albérès. *Le Comique et l'Ironie*. Classiques Hachette, Thèmes et Parcours Littéraires, 1973.

[2] Alcanter de Brahm. *L'Ostensoir des Ironies*. Rumeur des âges (1996), précédé de Le point sur l'ironie par Pierre Schoentjes, 1899.

[3] Carlos E Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change : Partial meet contraction and revision functions. *The J. of Symbolic Logic*, 50(2) :510–530, 1985.

[4] Anonyme. *Va te marrer chez les Grecs (Philogelos)*. Traduction, notes et postface par Arnaud Zucker. Mille et une Nuits, 2008.

[5] Salvatore Attardo and Victor Raskin. Script theory revis(it)ed : Joke similarity and joke representation model. *Humor-International Journal of Humor Research*, 4(3-4) :293–348, 1991.

[6] Salem Benferhat, Didier Dubois, and Henri Prade. Practical handling of exception-tainted rules and independence information in possibilistic logic. *Appl. Intell.*, 9(2) :101–127, 1998.

[7] Henri Bergson. *Le Rire : Essai sur la Signification du Comique*. publie.net, 1900 (réédition 2011).

[8] Jean-François Bonnefon and Henri Prade. Qu'est-ce qui (nous) fait signe ? In Mario Borillo, editor, *Dans l'Atelier de l'Art : Expériences Cognitives*, pages 186–205. Champ Vallon, 2010.

[9] Miguel de Unamuno. *La Cocotologie. Notes pour un traité*. Editions Self, Paris, 1946.

[10] Didier Dubois and Henri Prade. *Possibility Theory. An Approach to Computerized Processing of Uncertainty*. Plenum Press, New York and London, 1988. With the collaboration of H. Farreny, R. Martin-Clouaire and C. Testemale.

[11] Florence Dupin de Saint-Cyr. Scenario update applied to causal reasoning. In *Proc. Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'08)*, pages 188–197. AAAI Press, septembre 2008.

[12] Florence Dupin de Saint-Cyr and Jérôme Lang. Belief extrapolation (or how to reason about observations and unpredicted change). *Artificial Intelligence*, 175 :760–790, janvier 2011.

[13] Ronald Fagin and Joseph Y Halpern. Reasoning about knowledge and probability. *J. of the ACM (JACM)*, 41(2) :340–367, 1994.

[14] Sigmund Freud. Humour. In *The Standard Edition of the Complete Psychological Works of Sigmund Freud, Volume XXI (1927-1931) : The Future of an Illusion, Civilization and its Discontents, and Other Works*, pages 161–166. 1961.

[15] Christian Hempelmann, Victor Raskin, and Katrina E Triezenberg. Computer, tell me a joke... but please make it funny : Computational humor with ontological semantics. In G. Sutcliffe and R.Goebel, editors, *Proc. 19th Int. Florida Artificial Intelligence Research Society Conf., FLAIRS'06*, pages 746–751. AAAI Press, 2006.

[16] Vladimir Jankélévitch. *L'Ironie*. Flammarion, 1964.

[17] Hirofumi Katsuno and Alberto O Mendelzon. Propositional knowledge base revision and minimal change. *Artificial Intelligence*, 52(3) :263–294, 1991.

[18] Hirofumi Katsuno and Alberto O Mendelzon. On the difference between updating a knowledge base and revising it. *Belief revision*, 29 :183, 2003.

[19] Emiliano Lorini and Cristiano Castelfranchi. The cognitive structure of surprise : looking for basic principles. *Topoi*, 26(1) :133–149, 2007.

[20] Marvin Minsky. Jokes and the logic of the cognitive unconscious. In Lucia Vaina and Jaakko Hintikka, editors, *Cognitive Constraints on Communication - Representations and Processes*, pages 175–200. D. Reidel Publ. Comp., 1984.

[21] Hervé Nègre. *Dictionnaire des Histoires Drôles*. Fayard, 1970.

[22] Henri Prade. Monde(s) et représentation(s). quelques propos sur les rapports entre art et science. In Mario Borillo, editor, *Approches Cognitives de la Création Artistique*, pages 49–56. Mardaga, 2005.

[23] Georges Pérec. *Cantatrix sopranica L. et autres écrits scientifiques*. Éditions du Seuil, coll. « Librairie du XXe siècle », 1980 (réédité en 1995).

[24] François Rabelais. *La vie très horrificque du grand Gargantua, père de Pantagruel, jadis composée par M. Alcofribas abstracteur de quintessence*. La Sirène (réédition 1919), 1534.

[25] Pierre-Yves Raccah. Humour et métaphore : quelques éléments d'une analogie pour la construction d'un sens inattendu : Illustration sur un corpus de citations de George Bernard Shaw. *Revue de Sémantique et Pragmatique, Presses de l'Univ. d'Orléans*, 2015.

[26] Vilayanur S. Ramachandran. The artful brain. In *The Internet and the University : Forum 2004*, pages 169–198. 2004. https ://www.educause.edu/ir/library/pdf/ffpiu049.pdf.

[27] Victor Raskin. *The Primer of Humor Research*, volume 8. Walter de Gruyter, 2008.

[28] George Lennox Sharman Shackle. *Decision, Order and Time in Human Affairs*. (2nd edition), Cambridge University Press, UK, 1961.

[29] Dafna Shahaf, Eric Horvitz, and Robert Mankoff. Inside jokes : Identifying humorous cartoon captions. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1065–1074. ACM, 2015.

[30] Léa Sombé. A glance at revision and updating in knowledge bases. *International J. of Intelligent Systems*, 9(1) :1–27, 1994.

[31] Oliviero Stock and Carlo Strapparava. Getting serious about the development of computational humor. In *Proc. IJCAI'03*, volume 3, pages 59–64, 2003.

[32] Thémiseul de Saint-Hyacinthe. *Le Chef d' Œuvre d'un Inconnu*. Réédition Aubanel (1965), 1714.

[33] M.-A. Williams. Aesthetics and the explication of surprise. *Languages of Design. Formalisms for word, image and sound*, 3 :145–157, 1996.

[34] Marianne Winslett. Reasoning about action using a possible models approach. In *Proc. AAAI'88*, page 89–93, 1988.

[35] Monique Yaari. *Ironie Paradoxale et Ironie Poétique*. Summa Publications, Birmingham, Alabama, 1988.

[36] Semir Zeki. *Inner Vision. An Exploration of Art and the Brain*. Oxford University Press, 1999.

Efficiency, Sequenceability and Deal-Optimality in Fair Division of Indivisible Goods

Aurélie Beynier
Sorbonne Université, CNRS, LIP6
F-75005 Paris, France
aurelie.beynier@lip6.fr

Sylvain Bouveret
Univ. Grenoble Alpes, CNRS, LIG
Grenoble, France
sylvain.bouveret@imag.fr

Michel Lemaître
Formerly ONERA
Toulouse, France
michel.lemaitre.31@gmail.com

Nicolas Maudet
Sorbonne Université, CNRS, LIP6
F-75005 Paris, France
nicolas.maudet@lip6.fr

Simon Rey
Sorbonne Université, ENS P-S
Paris, Cachan, France
srey@ens-paris-saclay.fr

Parham Shams
Sorbonne Université, CNRS, LIP6
F-75005 Paris, France
parham.shams@lip6.fr

ABSTRACT

In fair division of indivisible goods, using sequences of sincere choices (or picking sequences) is a natural way to allocate the objects. The idea is as follows: at each stage, a designated agent picks one object among those that remain. Another intuitive way to obtain an allocation is to give objects to agents in the first place, and to let agents exchange them as long as such "deals" are beneficial. This paper investigates these notions, when agents have additive preferences over objects, and unveils surprising connections between them, and with other efficiency and fairness notions. In particular, we show that an allocation is sequenceable if and only if it is optimal for a certain type of deals, namely cycle deals involving a single object. Furthermore, any Pareto-optimal allocation is sequenceable, but not the converse. Regarding fairness, we show that an allocation can be envy-free and non-sequenceable, but that every competitive equilibrium with equal incomes is sequenceable. To complete the picture, we show how some domain restrictions may affect the relations between these notions. Finally, we experimentally explore the links between the scales of efficiency and fairness.

KEYWORDS

Multiagent Resource Allocation; Fair Division; Efficiency

ACM Reference Format:
Aurélie Beynier, Sylvain Bouveret, Michel Lemaître, Nicolas Maudet, Simon Rey, and Parham Shams. 2019. Efficiency, Sequenceability and Deal-Optimality in Fair Division of Indivisible Goods. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 9 pages.

1 INTRODUCTION

In this paper, we investigate fair division of indivisible goods. In this problem, a set of indivisible objects or goods has to be allocated to a set of agents, taking into account the agents' preferences about the objects. This classical collective decision making problem has plenty of practical applications, among which the allocation of space resources [9, 30], of tasks to workers in crowdsourcing market systems [34], papers to reviewers [26] or courses to students [19].

This problem can be tackled from two different perspectives. The first possibility is to resort to a benevolent entity in charge of collecting in a *centralized* way the preferences of all the agents. This entity then computes an allocation that takes into account these preferences and satisfies some fairness (*e.g.* envy-freeness) and efficiency (*e.g.* Pareto-optimality) criteria, or optimizes a well-chosen social welfare ordering. The second possibility is to have a *distributed* point of view, *e.g.* by starting from an initial allocation and letting the agents negotiate to swap their objects [23, 35]. A somewhat intermediate approach consists in allocating the objects to the agents using a *protocol*, which allows to build an allocation interactively by asking the agents a sequence of questions. Protocols are at the heart of works mainly concerning the allocation of divisible resources (cake-cutting) [16], but have also been studied in the context of indivisible goods [14, 16].

In this paper, we focus on a particular allocation protocol: *sequences of sincere choices* (also known as *picking sequences*). This very simple protocol works as follows. A central authority chooses a sequence of agents before the protocol starts, having as many agents as the number of objects (some agents may appear several times in the sequence). Then, each agent appearing in the sequence is asked to choose in turn one object among those that remain. For instance, according to the sequence $\langle 1, 2, 2, 1\rangle$, agent 1 will choose first, then agent 2 will pick two objects in a row, and agent 1 will take the last object. This protocol, used in a lot of everyday situations, has been studied for the first time by Kohler and Chandrasekaran [29]. Later, Brams and Taylor [17] have studied a particular version of this protocol, namely alternating sequences, in which the sequence of agents is restricted to a balanced ($\langle 1, 2, 2, 1\ldots\rangle$) or strict ($\langle 1, 2, 1, 2\ldots\rangle$) alternation of agents. Bouveret and Lang [11] have further formalized this protocol, whose properties (especially related to game theoretic aspects) have later been characterized by Kalinowski *et al.* [27, 28]. Finally, Aziz *et al.* [4] have studied the complexity of problems related to finding whether a particular assignment (or bundle) is achievable by a particular class of picking sequences. Picking sequences have also been considered by Brams and King [15], that focus on a situation where the agents have ordinal preferences. They make an interesting link between this protocol and Pareto-optimality, showing, among others, that picking sequences always result in a Pareto-optimal allocation, but also that every Pareto-optimal allocation can be obtained in this way.

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

In this paper, we elaborate on these ideas and analyze the links between sequences, certain types of deals among agents, and some efficiency and fairness properties, in a more general model in which the agents have numerical additive preferences on the objects. Our main contributions are the following. We give a formalization of the link between allocations and sequences of sincere choices, highlighting a simple characterization of the sequenceability of an allocation (Section 3). Then, we show that in this slightly more general framework than the one by Brams and King, Pareto-optimality and sequenceability are not equivalent anymore (Section 4). By unveiling the connection between sequenceability and cycle deals among agents (Section 5), we obtain a rich "scale of efficiency" that allows us to characterize the degree of efficiency of a given allocation. Interestingly, some domain restrictions have significant effects on this hierarchy (Section 6). We also highlight (Section 7) a link between sequenceability and another important economical concept: the competitive equilibrium from equal income (CEEI). Another contribution is the experimental exploration of the links between the scale of efficiency and fairness properties (Section 8).

2 MODEL AND DEFINITIONS

The aim of the fair division of indivisible goods, also called MultiAgent Resource Allocation (MARA), is to allocate a finite set of *objects* $O = \{o_1, \ldots, o_m\}$ to a finite set of *agents* $N = \{1, \ldots, n\}$. A *sub-allocation* on $O' \subseteq O$ is a vector $\vec{A}^{|O'} = \langle A_1^{|O'}, \ldots, A_n^{|O'} \rangle$ of *bundles* of objects, such that $\forall i \forall j$ with $i \neq j$: $A_i^{|O'} \cap A_j^{|O'} = \emptyset$ (a given object cannot be allocated to more than one agent) and $\bigcup_{i \in N} A_i^{|O'} = O'$ (all the objects from O' are allocated). $A_i^{|O'} \subseteq O'$ is called agent i's *share* on O'. $\vec{A}^{|O''}$ is a sub-allocation of $\vec{A}^{|O'}$ when $A_i^{|O''} \subseteq A_i^{|O'}$ for each agent i Any sub-allocation $\vec{A}^{|O}$ on the entire set of objects will be written $\vec{A}$ and just called *allocation*.

Any satisfactory allocation must take into account the agents' preferences on the objects. Here, we will make the classical assumption that these preferences are *numerically additive*. Each agent i has a *utility function* $u_i : 2^O \to \mathbb{R}^+$ measuring her satisfaction $u_i(A)$ when she obtains share A, which is defined as follows:

$$u_i(A) \stackrel{def}{=} \sum_{o_k \in \pi} u_i(o_k),$$

where $u_i(o_k)$ is the weight given by agent i to object o_k. This assumption, as restrictive as it may seem, is made by a lot of authors [8, 31, for instance] and is considered a good compromise between expressivity and conciseness.

Definition 2.1. An instance of the *additive multiagent resource allocation problem* (add-MARA instance) is a tuple $I = \langle N, O, u \rangle$, where N and O are as defined above and $u : N \times O \to \mathbb{R}^+$ is a mapping. Here, $u_i(o_k)$ is the weight given by agent i to object o_k.

We say that the agents' preferences *are strict on objects* if, for each agent i and each pair of objects $o_k \neq o_l$, we have $u_i(o_k) \neq u_i(o_l)$. Similarly, we say that the agents' preferences are *strict on shares* if, for each agent i and each pair of shares $A \neq A'$, we have $u_i(A) \neq u_i(A')$. Note that strict preferences on shares entail strict preferences on objects; the converse is false.

We will denote by $\mathcal{P}(I)$ the set of allocations for I. We will also use the notation $o_i o_j \ldots$ as a shorthand for bundle $\{o_i, o_j, \ldots\}$.

Definition 2.2. Given an agent i and a set of objects O', we will write $\text{best}(O', i) = \text{argmax}_{o_k \in O'} u_i(o_k)$ the objects from O' having the highest weight for i (they will be called *top* objects of i).

A (sub-)allocation $\vec{A}^{|O'}$ is said *frustrating* if no agent receives one of her top objects in $\vec{A}^{|O'}$ (formally: $\text{best}(O', i) \cap A_i^{|O'} = \emptyset$ for each agent i), and *non-frustrating* otherwise.

In the following, we will consider a particular way of allocating objects to agents: sequences of sincere choices.

Definition 2.3. Let $I = \langle N, O, u \rangle$ be an add-MARA instance. A *sequence of sincere choices* (or simply *sequence* when the context is clear) is a vector of N^m. We will denote by $\mathcal{S}(I)$ the set of possible sequences for the instance I.

Let $\vec{\sigma} \in \mathcal{S}(I)$ be a sequence of agents and let i_t be the σ^{th} agent of the sequence. $\vec{\sigma}$ is said to *generate* allocation $\vec{A}$ if and only if $\vec{A}$ can be obtained as a possible result of the non-deterministic[1] Algorithm 1 on input I and $\vec{\sigma}$.

Algorithm 1: Execution of a sequence

Input: an instance $I = \langle N, O, u \rangle$ and a sequence $\vec{\sigma} \in \mathcal{S}(I)$
Output: an allocation $\vec{A} \in \mathcal{P}(I)$

```
A ← empty allocation (such that ∀i ∈ N : A_i = ∅);
O_1 ← O;
for t from 1 to m do
    i ← i_t;
    choose object o_t ∈ best(O_t, i) ;
    A_i ← A_i ∪ {o_t} ;
    O_{t+1} ← O_t \ {o_t}
```

Definition 2.4. An allocation $\vec{A}$ is said to be *sequenceable* if there exists a sequence $\vec{\sigma}$ that generates $\vec{A}$, and *non-sequenceable* otherwise. For a given instance I, we will write $\mathcal{R}(I)$ the binary relation defined by $(\vec{\sigma}, \vec{A}) \in \mathcal{R}(I)$ if and only if $\vec{\sigma}$ can be generated by $\vec{A}$.

Example 2.5. Let I be the instance represented by the following weight matrix:[2]

$$\begin{pmatrix} 8 & 2 & 1 \\ 5 & 1 & 5 \end{pmatrix}$$

For instance, sequence $\langle 2, 1, 2 \rangle$ generates two possible allocations: $\langle o_1, o_2 o_3 \rangle$ and $\langle o_2, o_1 o_3 \rangle$, depending on whether agent 2 chooses object o_1 or o_3 that she both prefers. Allocation $\langle o_1 o_2, o_3 \rangle$ can be generated by three sequences. Allocations $\langle o_1 o_3, o_2 \rangle$ and $\langle o_3, o_1 o_2 \rangle$ are non-sequenceable.

For any instance I, $|\mathcal{S}(I)| = |\mathcal{P}(I)| = n^m$. Also note that the number of objects allocated to an agent by a sequence is the number of times the agent appears in the sequence.

The notion of frustrating allocation and sequenceability were already implicitly present in the work by Brams and King [15], and sequenceability has been extensively studied by Aziz *et al.* [4] with a focus on sub-classes of sequences (*e.g.* alternating sequences). However, a fundamental difference is that in our setting, the preferences

[1] The algorithm contains an instruction **choose** splitting the control flow into several branches, building all the allocations generated by $\vec{\sigma}$.
[2] In this example and the following ones, we represent instances by a matrix where the value at row i and column o_k represents the weight $w(i, o_k)$.

might be non strict on objects, which entails that the same sequence can yield different allocations (in the worst case, an exponential number), as Example 2.5 shows.

3 SEQUENCEABLE ALLOCATIONS

We have seen in Example 2.5 that some allocations are non-sequenceable. We will now formalize this and give a precise characterization of sequenceable allocations. That is, we will try to identify under which conditions an allocation is achievable by the execution of a sequence of sincere choices. We first start by noticing that in every sequenceable allocation, the first agent of the sequence gets a top object, so every frustrating allocation is non-sequenceable. However, being non-frustrating is not a sufficient condition for an allocation to be sequenceable, as the following example shows:

Example 3.1. Consider this instance:

$$\begin{pmatrix} \textcircled{9} & 8 & 2 & \textcircled{1} \\ 2 & \textcircled{5} & \textcircled{1} & 4 \end{pmatrix}$$

In allocation $\mathbb{A} = \langle \mathbb{A}_1\mathbb{A}_4, \mathbb{A}_2\mathbb{A}_3 \rangle$, each agent receives her top object. However, after $\mathbb{A}_1$ and $\mathbb{A}_2$ have been allocated (they must be allocated first by all sequences generating $\mathbb{A}$), the dotted sub-allocation remains. This sub-allocation is obviously non-sequenceable because it is frustrating. Hence $\mathbb{A}$ is not sequenceable either.

This property of containing a frustrating sub-allocation exactly characterizes the set of non-sequenceable allocations:

Proposition 1. *Let $\mathbb{I} = \langle \mathcal{N}, \mathcal{O}, \mathbb{A} \rangle$ be an instance and $\mathbb{A}$ be an allocation of this instance. The two following statements are equivalent:*

(A) *$\mathbb{A}$ is sequenceable.*

(B) *No sub-allocation of $\mathbb{A}$ is frustrating (in every sub-allocation, at least one agent receives a top object).*

Proof. (B) $\Rightarrow$ (A). Let us suppose that for all subsets of objects $O' \subseteq O$, at least one agent gets one of her top objects in $\mathbb{A}^{|O'}$. We will show that $\mathbb{A}$ is sequenceable. Let $\mathbb{A}$ be a sequence of agents and $\overrightarrow{O} \in \left(2^O\right)^m$ be a sequence of bundles jointly defined as follows:

- $O_1 = O$ and $\mathbb{A}_1$ is an agent that receives one of her top objects in $\mathbb{A}^{|O_1}$;
- $O_{t+1} = O_t \setminus \{\mathbb{A}_t\}$, where $\mathbb{A}_t \in \text{best}(O_t, \mathbb{A}_t)$ and $\mathbb{A}_t$ is an agent that receives one of her top objects in $\mathbb{A}^{|O_t}$, for $\mathbb{A} \geq 1$.

From the assumption on $\mathbb{A}$, we can check that $\mathbb{A}$ is well-defined. Moreover, $\mathbb{A}$ is one of the allocations generated by $\mathbb{A}$.

(A) $\Rightarrow$ (B) by contraposition. Let $\mathbb{A}$ be an allocation containing a frustrating sub-allocation $\mathbb{A}^{|O'}$. Suppose that there exists a sequence $\mathbb{A}$ generating $\mathbb{A}$. We can notice that in a sequence of sincere choices, when an object is allocated to an agent, all the objects that are strictly better for her have already been allocated at a previous step. Let $\mathbb{A}_k \in O'$, and let $\mathbb{A}$ be the agent that receives $\mathbb{A}_k$ in $\mathbb{A}$. Since $\mathbb{A}^{|O'}$ is frustrating, there is another object $\mathbb{A}_l \in O'$ such that $\mathbb{A}(\mathbb{A}\mathbb{A}_l) > \mathbb{A}(\mathbb{A}\mathbb{A}_k)$. As we have seen, $\mathbb{A}_l$ is necessarily allocated before $\mathbb{A}_k$ in the execution of $\mathbb{A}$. Let $\mathbb{A}$ be the agent who receives $\mathbb{A}_l$. Using the same argument for $\mathbb{A}$ and $\mathbb{A}_l$ we find another object $\mathbb{A}_p \in O'$ allocated before $\mathbb{A}_l$ in the sequence. Iterating this argument, since O' is finite, we will eventually find an object which has been encountered before. This creates a cycle in the precedence relation of the objects in the execution of the sequence. Contradiction: no sequence can thus generate $\mathbb{A}$. □

Besides characterizing a sequenceable allocation, the proof of Proposition 1 gives a practical way of checking if an allocation is sequenceable, and, if it is the case, of computing a sequence that generates this allocation.

Proposition 2. *Let $\mathbb{I} = \langle \mathcal{N}, \mathcal{O}, \mathbb{A} \rangle$ be an instance and $\mathbb{A}$ be an allocation of this instance. We can decide in time $\mathrm{O}(\mathbb{A} \times \mathbb{A}^2)$ if $\mathbb{A}$ is sequenceable.*

The proof is based on the execution of Algorithm 2. This algorithm is similar in spirit to the one proposed by Brams and King [15] but is more general because (i) it can involve non-strict preferences on objects, and (ii) it can conclude with non-sequenceability.

Algorithm 2: Sequencing an allocation

Input: $\mathbb{I} = \langle \mathcal{N}, \mathcal{O}, \mathbb{A} \rangle$ and $\mathbb{A} \in \mathcal{P}(\mathbb{I})$
Output: a sequence $\mathbb{A}$ generating $\mathbb{A}$ or **NonSeq**

1 $(\mathbb{A}, O') \leftarrow (\langle\rangle, O)$;
2 **for** $\mathbb{A}$ **from** *1* **to** $\mathbb{A}$ **do**
3 **if** $\exists \mathbb{A}$ *such that* $\text{best}(O', \mathbb{A}) \cap \mathbb{A}_i \neq \emptyset$ **then**
4 Append $\mathbb{A}$ to $\mathbb{A}$;
5 let $\mathbb{A}_k \in \text{best}(O', \mathbb{A}) \cap \mathbb{A}_i$;
6 $O' \leftarrow O' \setminus \{\mathbb{A}_k\}$;
7 **else return NonSeq** ;
8 **return** $\mathbb{A}$;

Proof. We show that Algorithm 2 returns a sequence $\mathbb{A}$ generating the input allocation $\mathbb{A}$ if and only if there is one. Suppose that the algorithm returns a sequence $\mathbb{A}$. Then, by definition of the sequence (in the loop from line 2 to line 7), at each step $\mathbb{A}$ $\mathbb{A} = \mathbb{A}_t$ can choose an object in $\mathbb{A}_i$, that is one of her top objects. Conversely, suppose the algorithm returns **NonSeq**. Then, at a given step $\mathbb{A}$, $\forall \mathbb{A}$ $\text{best}(O', \mathbb{A}) \cap \mathbb{A}_i = \emptyset$. By definition, $\mathbb{A}^{|O'}$ is therefore, at this step, a frustrating sub-allocation of $\mathbb{A}$. By Proposition 1, $\mathbb{A}$ is thus non-sequenceable. The loop from line 2 to line 7 runs in time $\mathrm{O}(\mathbb{A} \times \mathbb{A})$, because searching for the top objects in the preferences of each agent can be completed in time $\mathrm{O}(\mathbb{A})$. This loop being executed $\mathbb{A}$ times, the algorithm runs in $\mathrm{O}(\mathbb{A} \times \mathbb{A}^2)$. □

4 PARETO-OPTIMALITY

An allocation is Pareto-optimal if no other allocation dominates it. In our context, allocation $\mathbb{A}'$ dominates allocation $\mathbb{A}$ if for all agent $\mathbb{A}$ $\mathbb{A}_i(\mathbb{A}'_i) \geq \mathbb{A}_i(\mathbb{A}_i)$ and $\mathbb{A}_j(\mathbb{A}'_j) > \mathbb{A}_j(\mathbb{A}_j)$ for at least one agent $\mathbb{A}$ When an allocation is generated from a sequence, in some sense, a weak form of efficiency is applied to build the allocation: each successive (picking) choice is "locally" optimal. This raises a natural question: is every sequenceable allocation Pareto-optimal?

This question has already been extensively discussed independently by Aziz *et al.* [3] and Bouveret and Lemaître [13]. We complete the discussion here to give more insights about the implications of the previous results in our framework.

Brams and King [15, Proposition 1] prove the equivalence between sequenceability and Pareto-optimality. However, they have

a different notion of Pareto-optimality, because the agents' preferences are given as linear orders over *objects*. To be able to compare bundles, these preferences are lifted on subsets using the *responsive set extension* $\succ_{RS}$. This extension leaves many bundles incomparable and leads to define *possible* and *necessary* Pareto-optimality. Brams and King's notion is possible Pareto-optimality. Aziz *et al.* [2] show that, given a linear order $\succ$ on objects and two bundles $\mathbb{X}$ and $\mathbb{X}'$, $\mathbb{X} \succ_{RS} \mathbb{X}'$ if and only if $\mathbb{X}(\mathbb{X}) > \mathbb{X}(\mathbb{X}')$ for *all* additive utility functions $\mathbb{X}$ compatible with $\succ$ (that is, such that $\mathbb{X}(\mathbb{X}_k) > \mathbb{X}(\mathbb{X}_l)$ if and only if $\mathbb{X}_k \succ \mathbb{X}_l$). This characterization of responsive dominance yields the following reinterpretation of Brams and King's result: an allocation $\overrightarrow{\mathbb{X}}$ is sequenceable if and only if for each other allocation $\overrightarrow{\mathbb{X}}'$, there is a set $\mathbb{X}_1, \ldots, \mathbb{X}_n$ of additive utility functions, respectively compatible with $\succ_1, \ldots, \succ_n$ such that $\mathbb{X}_i(\mathbb{X}_i) > \mathbb{X}_i(\mathbb{X}_i')$ for at least one agent $\mathbb{X}$

The latter notion of Pareto-optimality is very weak, because (unlike in our context) the set of additive utility functions is not fixed — we just have to find one that works. Under our stronger notion, the equivalence between sequenceability and Pareto-optimality no longer holds.[3]

Example 4.1. Let us consider the following instance:

$$\begin{pmatrix} 5 & 4 & 2 \\ 8 & 2 & 1 \end{pmatrix}$$

The sequence $\langle 1, 2, 2 \rangle$ generates allocation $\overrightarrow{\mathbb{X}} = \langle \mathbb{X}_1, \mathbb{X}_2\mathbb{X}_3 \rangle$ giving utilities $\langle 5, 3 \rangle$. $\overrightarrow{\mathbb{X}}$ is then sequenceable but it is dominated by $\overrightarrow{\mathbb{X}}' = \langle \mathbb{X}_2\mathbb{X}_3, \mathbb{X}_1 \rangle$, giving utilities $\langle 6, 8 \rangle$ (and generated by $\langle 2, 1, 1 \rangle$). Observe that, under ordinal linear preferences, $\overrightarrow{\mathbb{X}}'$ would not dominate $\overrightarrow{\mathbb{X}}$, but they would be incomparable.

The last example shows that a sequence of sincere choices does not necessarily generate a Pareto-optimal allocation. What about the converse? We can see, as a trivial corollary of the reinterpretation of Brams and King's result in our terminology, that the answer is positive *if the preferences are strict on shares*. The following result is more general, because it holds even without this assumption:

Proposition 3 ([3, 13]). *Every Pareto-optimal allocation is sequenceable.*

As already noticed by Aziz *et al.* [3], the proof follows from an adaptation of Brams and King's Proposition 1 (necessity part of the proof) [15]. However, we find useful to give the proof, because it is more general than the previous one, and will be reused in subsequent results of this paper. Before giving this proof, we illustrate it on a concrete example [12, Example 5].

Example 4.2. Let us consider the following instance:

$$\begin{pmatrix} \dagger(12) & 15 & \dagger 11 & (7) & 2 \\ 2 & 12 & (7) & \dagger 15 & \dagger(11) \\ 15 & \dagger(20) & 9 & 2 & 1 \end{pmatrix}$$

The circled allocation $\overrightarrow{\mathbb{X}}$ is not sequenceable: indeed, every sequence that could generate it should start with $\langle 3, 1, \ldots \rangle$, leaving the frustrating sub-allocation $\overrightarrow{\mathbb{X}}$ in a dotted box.

[3] Actually, since it is known [1, 21] that testing Pareto-optimality with additive preferences in coNP-complete, and that testing sequenceability is in P (Proposition 2), they cannot be equivalent unless P = coNP.

Let us consider agent 1 for instance. Since the suballocation is frustrating, she does not receive $\mathbb{X}_3$ (which is her top object), but agent 2 does. This latter agent, however, does not get her top object, $\mathbb{X}_4$, because agent 1 receives it. Obviously, if agent 1 gives $\mathbb{X}_4$ to agent 2 and receives $\mathbb{X}_3$ in return, we have built a cycle in which each agent gives a regular object and receives a top one. Doing this, we have built an allocation strictly dominating $\overrightarrow{\mathbb{X}}$.

Proof. As stated in the example, we will now prove the contraposition of the proposition: every non-sequenceable allocation is dominated. Let $\overrightarrow{\mathbb{X}}$ be a non-sequenceable allocation. From Proposition 1, in a non-sequenceable allocation, there is at least one frustrating sub-allocation. Let $\overrightarrow{\mathbb{X}}$ be such a sub-allocation (that can be $\overrightarrow{\mathbb{X}}$ itself). We will, from $\overrightarrow{\mathbb{X}}$, build another sub-allocation dominating it. Let us choose an arbitrary agent $\mathbb{X}$involved in $\overrightarrow{\mathbb{X}}$, receiving an object not among her top ones in $\overrightarrow{\mathbb{X}}$. Let $\mathbb{X}_i$ be a top object of $\mathbb{X}$in $\overrightarrow{\mathbb{X}}$, and let $\mathbb{X}(\neq \mathbb{X})$ be the unique agent receiving it in $\overrightarrow{\mathbb{X}}$. Let $\mathbb{X}_j$ be a top object of $\mathbb{X}$ We can notice that $\mathbb{X}_j \neq \mathbb{X}_i$ (otherwise $\mathbb{X}$would obtain one of her top objects and $\overrightarrow{\mathbb{X}}$ would not be frustrating). Let $\mathbb{X}$ be the unique agent receiving $\mathbb{X}_j$ in $\overrightarrow{\mathbb{X}}$, and so on. Using this argument iteratively, we form a path starting from $\mathbb{X}$and alternating agents and objects, in which two successive agents and objects are distinct. Since the number of agents and objects is finite, we will eventually encounter an agent which has been encountered at a previous step of the path. Let $\mathbb{X}$be the first such agent and $\mathbb{X}_k$ be the last object seen before her in the sequence ($\mathbb{X}$is the unique agent receiving $\mathbb{X}_k$). We have built a cycle $\mathbb{X} \xrightarrow{o_k} \mathbb{X} \xrightarrow{o_{k-1}} \mathbb{X} - 1 \cdots \mathbb{X} + 1 \xrightarrow{o_i} \mathbb{X}$in which all the agents and objects are distinct, and that has at least two agents and two objects. From this cycle, we can modify $\overrightarrow{\mathbb{X}}$ to build a new sub-allocation by giving to each agent in the cycle a top object instead of another less preferred object, all the agents not appearing in the cycle being left unchanged. More formally, the following attributions in $\overrightarrow{\mathbb{X}}$ (and hence in $\overrightarrow{\mathbb{X}}$): $(\mathbb{X} \leftarrow \mathbb{X}_k)(\mathbb{X} + 1 \leftarrow \mathbb{X}_i) \cdots (\mathbb{X} \leftarrow \mathbb{X}_{k-1})$ are replaced by: $(\mathbb{X} \leftarrow \mathbb{X}_i)(\mathbb{X} + 1 \leftarrow \mathbb{X}_{i+1}) \cdots (\mathbb{X} \leftarrow \mathbb{X}_k)$ where $(\mathbb{X} \leftarrow \mathbb{X}_i)$ means that $\mathbb{X}_i$ is attributed to $\mathbb{X}$ The same substitutions operated in $\overrightarrow{\mathbb{X}}$ yield an allocation $\overrightarrow{\mathbb{X}}'$ that dominates $\overrightarrow{\mathbb{X}}$. □

Corollary 4.3. *No frustrating allocation can be Pareto-optimal (equivalently, in every Pareto-optimal allocation, at least one agent receives a top object).*

Proposition 3 implies that there exists, for a given instance, three classes of allocations: (1) non-sequenceable (therefore non Pareto-optimal) allocations, (2) sequenceable but non Pareto-optimal allocations, and (3) Pareto-optimal (hence sequenceable) allocations. These three classes define a "scale of efficiency" that can be used to characterize the allocations. What is interesting and new here is the intermediate level. We will see that this scale can be further refined.

5 CYCLE DEALS-OPTIMALITY

Pareto-optimality can be thought as a reallocation of objects among agents using improving *deals* [35, 37], as we have seen, to some extent, in the proof of Proposition 3. *Trading cycles* or *cycle deals* constitute a sub-class of deals, which is classical and used, *e.g.*, by Varian [39, page 79] and Lipton et al. [31, Lemma 2.2] in the context

of envy-freeness. Trying to link efficiency concepts with various notions of deals is thus a natural idea.

Definition 5.1. Let $\langle \mathcal{N}, O, \boxtimes \rangle$ be an add-MARA instance and $\vec{\boxtimes}$ be an allocation of this instance. A $(\mathbf{N}, \mathbf{M})$-cycle deal of $\vec{\boxtimes}$ is a sequence of transfers of items $\boxtimes = \langle (\boxtimes_1, O_1), \ldots, (\boxtimes_N, O_N) \rangle$, where, for each $\boxtimes \in \{1, \ldots, \mathbf{N}\}$, $\boxtimes_j$ denotes the $\vec{\boxtimes}^h$ agent involved in the sequence and $\boxtimes_j \in \mathcal{N}$, $O_j \subseteq \boxtimes_j$, and $|O_j| \leq \mathbf{M}$. The allocation $\vec{\boxtimes}[\leftarrow \boxtimes]$ resulting from the application of $\boxtimes$ to $\vec{\boxtimes}$ is defined as follows:

- $\boxtimes[\leftarrow \boxtimes]_{\mu_j} = \boxtimes_{\mu_j} \setminus O_j \cup O_{j-1}$ for $\boxtimes \in \{2, \ldots, \mathbf{N}\}$;
- $\boxtimes[\leftarrow \boxtimes]_{\mu_1} = \boxtimes_{\mu_1} \setminus O_1 \cup O_N$;
- $\boxtimes[\leftarrow \boxtimes]_i = \boxtimes_i$ if $\boxtimes \notin \{\boxtimes_1, \ldots, \boxtimes_N\}$.

A cycle deal $\langle (\boxtimes_1, O_1), \ldots, (\boxtimes_N, O_N) \rangle$ will be written

$$\boxtimes_1 \xrightarrow{O_1} \boxtimes_2 \ldots \boxtimes_{N-1} \xrightarrow{O_{N-1}} \boxtimes_N \xrightarrow{O_N} \boxtimes_1.$$

In other words, in a cycle deal (we omit $\mathbf{N}$ and $\mathbf{M}$ when they are not necessary to understand the context), each agent gives a subset of at most $\mathbf{M}$ items from her share to the next agent in the sequence and receives in return a subset from the previous agent. $(\mathbf{N}, 1)$-cycle deals will be denoted by $\mathbf{N}$-cycle deals. 2-cycle deals will be called *swap*-deals. Among these cycle deals, some are more interesting: those where each agent improves her utility by trading objects. More formally, a deal $\boxtimes$ will be called *weakly improving* if $\boxtimes_i(\boxtimes[\leftarrow \boxtimes]_i) \geq \boxtimes_i(\boxtimes_i)$ $\forall \boxtimes \in \mathcal{N}$ with at least one of these inequalities being strict, and *strictly improving* if all these inequalities are strict.

Intuitively, if it is possible to improve an allocation by applying an improving cycle deal, then it means that this allocation is inefficient. Reallocating the items according to the deal will make everyone better-off. It is thus natural to derive a concept of efficiency from this notion of cycle-deal.

Definition 5.2. An allocation is said to be $>$-$(\mathbf{N}, \mathbf{M})$-cycle optimal (resp. $\geq$-$(\mathbf{N}, \mathbf{M})$-cycle optimal) if it does not admit any strictly (resp. weakly) improving $(\mathbf{K}, \mathbf{M})$-cycle deal for any $\mathbf{K} \leq \mathbf{N}$.

We begin with easy observations. First, $\geq$-cycle optimality implies $>$-cycle optimality, and these two notions become equivalent when the preferences are strict on shares. Moreover, restricting the size of the cycles and the size of the bundles exchanged yield less possible deals and hence lead to weaker optimality notions.

Note that for $\mathbf{N}' \leq \mathbf{N}$ and $\mathbf{M}' \leq \mathbf{M}$ (at least one of these inequalities being strict), $>$-$(\mathbf{N}, \mathbf{M})$-cycle-optimality and $\geq$-$(\mathbf{N}', \mathbf{M}')$-cycle-optimality are incomparable. These observations show that cycle-deal optimality notions form a (non-linear) hierarchy of efficiency concepts of diverse strengths. The natural question is whether they can be related to sequenceability and Pareto-optimality. Obviously, Pareto-optimality implies both $>$-cycle-optimality and $\geq$-cycle-optimality. An easy adaptation of the proof of Proposition 3 leads to the following stronger result:

Proposition 4. *An allocation $\vec{\boxtimes}$ is sequenceable if and only if it is $>$-$\boxtimes$-cycle optimal (with $\boxtimes = |\mathcal{N}|$).*

Proof. Let $\vec{\boxtimes}$ be a non-sequenceable allocation. Then by Proposition 1, there is at least one frustrating sub-allocation in $\vec{\boxtimes}$. Using the same line of arguments as in the proof of Proposition 3 we can build a strictly improving $\boxtimes$-cycle. Hence $\vec{\boxtimes}$ is not $>$-cycle optimal. Conversely, suppose that $\vec{\boxtimes}$ admits a strictly improving $\boxtimes$-cycle deal. Then obviously this cycle yields a sub-allocation that is frustrating. □

The scale of efficiency introduced in Section 4 can then be refined with a hierarchy of $>$-cycle optimality notions below sequenceable allocations: Pareto-optimal $\Rightarrow$ sequenceable $\Leftrightarrow$ $>$-$\boxtimes$-cycle optimal $\Rightarrow$ $>$-$(\boxtimes - 1)$-cycle optimal $\Rightarrow$ ... $\Rightarrow$ $>$-swap optimal.

As for $\geq$-cycle optimality, it forms a parallel hierarchy between Pareto-optimal and non-sequenceable allocations. Note that sequenceability does not involve any $\geq$-$\boxtimes$-cycle-optimality. Thus, as soon as $\boxtimes < \boxtimes$, $\geq$-$\boxtimes$-cycle optimality and sequenceability become incomparable notions.

For instance, for 3 agents, there exist allocations which are $\geq$-swap optimal but not sequenceable and the other way around:

$$\begin{pmatrix} 2 & \dagger 1 & \textcircled{2} & \dagger 3 \\ \dagger\textcircled{3} & 3 & 1 & \textcircled{2} \\ 1 & \textcircled{2} & \dagger 3 & 1 \end{pmatrix}$$

Here the circled allocation is $\geq$-swap optimal, but not sequenceable: there exists a strictly improving 3-cycle. At the same time, the dag allocation is sequenceable (by $\langle 2, 3, 1, 1 \rangle$), but not even $\geq$-swap optimal, since 1 and 2 may agree on a weakly improving deal.

The observations previously made in this section suggest that, in some situations, the most complex cycle deals could be required to reach Pareto-optimal allocations. This is indeed the case–we now make this claim more precise. Observe that to be involved in a weakly improving cycle deal, each agent must pass at least one item, thus for a $(\boxtimes, \boxtimes)$-cycle deal, we have that $\boxtimes \leq \boxtimes - (\boxtimes - 1)$ (*i.e.* the "largest bundle" circulating in this cycle deal can be at most $\boxtimes - \boxtimes + 1$). The following generic example shows that it may be necessary to implement a $(\boxtimes, \boxtimes - \boxtimes + 1)$-cycle to reach a Pareto-optimal allocation.

	α_1	α_2	...	α_{n-1}	β_1	...	β_{m-n+1}
1	1	0	0	0	$1/(m-n+1)$	$1/(m-n+1)$	$1/(m-n+1)$
2	1	1	0	0	0	0	0
3	0	1	1	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
n	0	0	0	2	$1/(m-n+1)$	$1/(m-n+1)$	$1/(m-n+1)$

Initially, every agent $\boxtimes = 1, \ldots, \boxtimes - 1$ holds item $\boxtimes_i$, while agent $\boxtimes$ holds $\boxtimes_1, \ldots, \boxtimes_{m-n+1}$. Hence everyone enjoys utility 1. This allocation is dominated by the allocation where each agent $\boxtimes = 2, \ldots, \boxtimes$ holds $\boxtimes_{i-1}$ while agent 1 holds $\boxtimes_1, \ldots, \boxtimes_{m-n+1}$. In this allocation, the utilities of agents are instead $\langle 1, 1, \ldots, 2 \rangle$. But to obtain $\boxtimes_{n-1}$, agent $\boxtimes$ must get it from $\boxtimes - 1$ who would only release it if she gets $\boxtimes_{i-2}$, etc. In the end, agent 1 will only release $\boxtimes_1$ if she gets the full bundle $\boxtimes_1, \ldots, \boxtimes_{m-n+1}$. Overall there are $\boxtimes$ agents involved in the deal, exchanging up to $\boxtimes - \boxtimes + 1$ items. By construction, it is easy to check that no simpler cycle deal (either in terms of number of items or number of agents) would allow to reach this allocation. Furthermore, there is clearly no other allocation Pareto-dominating the initial allocation.

However, it is important to note that cycle-deals may not be sufficient to reach Pareto-optimal outcomes when there are more items than agents.

Example 5.3. Consider the following example:

$$\begin{pmatrix} ③ & ⑥ & †6 & 0 & 6 & †4 \\ †2 & 0 & ⑥ & ③ & †7 & 0 \\ 0 & †5 & 0 & †4 & ⑥ & ③ \end{pmatrix}$$

Note that in the circled allocation, all agents enjoy the same utility, $\langle 9, 9, 9\rangle$, and that it is Pareto-dominated by the dag allocation which induces the vector of utilities $\langle 10, 9, 9\rangle$. We leave it to the reader to check that no swap deal, nor 3-cycle, would be weakly improving. In fact, the only way to reach the dag allocation from this initial allocation would require to implement *simultaneously* two (3, 1)-cycle deals ($1 \longrightarrow 2 \longrightarrow 3$ and $3 \longrightarrow 2 \longrightarrow 1$).

Finally, a corollary of Propositions 2 and 4 is that checking whether an allocation is $>$-$\mathcal{X}$-cycle optimal can be made in polynomial time (by checking whether it is sequenceable).

More generally, we can observe that checking whether an allocation is $(\mathcal{X}, \mathcal{X}')$-cycle optimal can be done by iterating over all $\mathcal{X}$-uples of agents[4], and for each one iterating over all possible transfers involving less than $\mathcal{X}'$ objects. In total, there are $\mathcal{X}!\binom{n}{k}$ $\mathcal{X}$-uples of agents (which is upper-bounded by $\mathcal{X}^{k+1}$). For each $\mathcal{X}$-uple, there are at most $\left(\sum_{k''=0}^{k'} \binom{m}{k''}\right)^{k}$ possible transfers, which is again upper-bounded by $(1+\mathcal{X})^{kk'}$. Hence, in total, checking whether an allocation is $(\mathcal{X}, \mathcal{X}')$-cycle optimal can be done in time $O(\mathcal{X}^{k+1} \times (1+\mathcal{X})^{kk'})$. This is polynomial in $\mathcal{X}$ and $\mathcal{X}$ if both $\mathcal{X}$ and $\mathcal{X}'$ are bounded (as for swap deals).

6 RESTRICTED DOMAINS

We now study the impact of several preference restrictions on the hierarchy of efficiency notions introduced in Section 5.

Strict preferences on objects. When the preferences are strict on objects, then obviously every sequence generates exactly one allocation. The following proposition is stronger and shows that the converse is also true:

Proposition 5. *Preferences are strict on objects iff* $\mathcal{X}(\mathrm{I})$ *is a mapping from* $\mathcal{S}(\mathrm{I})$ *to* $\mathcal{P}(\mathrm{I})$.

Proof. If preferences are strict on objects, each agent has only one possible choice at her turn in the sequence of sincere choices and hence every sequence generates one and only one allocation.

Conversely, if preferences are not strict on objects, at least one agent (suppose w.l.o.g. agent 1) gives the same weight to two different objects $\mathcal{X}_k, \mathcal{X}_l$. Suppose that exactly $\mathcal{X}$ objects are ranked above $\mathcal{X}_k$ and $\mathcal{X}_l$. Then the sequence where agent 1 picks $\mathcal{X}+1$ items in a row, and 2 picks the $\mathcal{X} - \mathcal{X} - 1$ remaining ones obviously generates two allocations, depending on agent 1's choice at step $\mathcal{X}+1$. □

Same order preferences. We say that the agents have *same order preferences* [12] if there is a permutation $\mathcal{X} : O \mapsto O$ such that for each agent $\mathcal{X}$ and each pair of objects $\mathcal{X}_k$ and $\mathcal{X}_l$, if $\mathcal{X}(\mathcal{X}_k) < \mathcal{X}(\mathcal{X}_l)$ then $\mathcal{X}\,(\mathcal{X}, \mathcal{X}(\mathcal{X}_k)) \geq \mathcal{X}\,(\mathcal{X}, \mathcal{X}(\mathcal{X}_l))$.

Proposition 6. *All the allocations of an instance with same order preferences are sequenceable (and actually cycle-deal optimal). Conversely, if all the allocations of an instance are sequenceable, then this instance has same order preferences.*

[4]We do not need to also run through all cycles of strictly less than k agents: such a cycle can be simulated just by appending at the end some agents whose role is just to pass the objects they receive to the next agent.

Proof. Suppose that the agents have same order preferences, and let $\vec{\mathcal{X}}$ be an arbitrary allocation. In every sub-allocation of $\vec{\mathcal{X}}$ at least one agent obtains a top object (because the preference order is the same among agents) and hence cannot be frustrating. By Proposition 1, $\vec{\mathcal{X}}$ is sequenceable.

Conversely, let us assume for contradiction that there are two distinct objects $\mathcal{X}_k$ and $\mathcal{X}_l$ and two distinct agents $\mathcal{X}$ and $\mathcal{X}$ such that $\mathcal{X}\,(\mathcal{X}, \mathcal{X}_k) > \mathcal{X}\,(\mathcal{X}, \mathcal{X}_l)$ and $\mathcal{X}\,(\mathcal{X}, \mathcal{X}_k) < \mathcal{X}\,(\mathcal{X}, \mathcal{X}_l)$. The sub-allocation $\vec{\mathcal{X}}^{|\{o_k, o_l\}}$ such that $\mathcal{X}_i^{|\{o_k, o_l\}} = \{\mathcal{X}_l\}$ and $\mathcal{X}_j^{|\{o_k, o_l\}} = \{\mathcal{X}_k\}$ is frustrating. By Proposition 1, every allocation $\vec{\mathcal{X}}$ containing this frustrating sub-allocation is non-sequenceable. □

Let us now characterize the instances for which $\mathcal{X}(\mathrm{I})$ is a one-to-one correspondence.

Proposition 7. *For a given instance* I*, the following two statements are equivalent.*

(A) *Preferences are strict on objects and in the same order.*
(B) *The relation* $\mathcal{X}(\mathrm{I})$ *is a one-to-one correspondence.*

The proof is a consequence of Propositions 5 and 6.

Single-peaked preferences. An interesting domain restriction are single-peaked preferences [10, 22], which, beyond voting, is also relevant in resource allocation settings [6, 20]. Formally, in this context, single-peakness can be defined as follows.

There exists a linear order $\triangleright$ over the set of objects O. Let $\mathcal{X}\mathcal{X}\mathcal{X}(\mathcal{X})$ be the preferred object of $\mathcal{X}$. An agent $\mathcal{X}$ has *single-peaked* preferences wrt. $\triangleright$ if, for any two objects $(\mathcal{X}_k, \mathcal{X}_l) \in O$ such that either $\mathcal{X}\mathcal{X}\mathcal{X}(\mathcal{X}) \triangleright \mathcal{X}_l \triangleright \mathcal{X}_k$ or $\mathcal{X}_k \triangleright \mathcal{X}_l \triangleright \mathcal{X}\mathcal{X}\mathcal{X}(\mathcal{X})$ (*i.e.* lying on the same "side" of the agent's peak), it is the case that $\mathcal{X}$ prefers $\mathcal{X}_l$ over $\mathcal{X}_k$.

Interestingly, when preferences are single-peaked, the hierarchy of $\mathcal{X}$-cycle optimality collapses at the second level:

Proposition 8. *If the preferences are single-peaked and additive, then an allocation* $\vec{\mathcal{X}}$ *is* $\geq$-$\mathcal{X}$*-cycle optimal iff it is swap-optimal.*

Proof. ([20, revisited]) First, note that $\geq$-$\mathcal{X}$-cycle optimality trivially implies swap-optimality. Let us now show the converse.

Let us consider for the sake of contradiction an allocation $\vec{\mathcal{X}}$ that is swap-optimal and such that there exists a $\geq$-$\mathcal{X}$-cycle $\mathcal{X}$, with $\mathcal{X} \leq \mathcal{X}$. Without loss of generality, let us suppose that $\mathcal{X} = \langle(\mathcal{X}_1, \{\mathcal{X}_1\}), \ldots, (\mathcal{X}_k, \{\mathcal{X}_k\})\rangle$. We show by induction on $\mathcal{X}$, the length of $\mathcal{X}$, that such a cycle can not exist.

Base case: $\mathcal{X} = 2$ A 1-cycle of length $\mathcal{X} = 2$ is a swap-deal but as $\vec{\mathcal{X}}$ is swap-optimal, no improving swap-deal exists in $\vec{\mathcal{X}}$ hence the contradiction.

Induction step: Let us assume that for each $\mathcal{X}'$ such that $2 \leq \mathcal{X}' \leq \mathcal{X} - 1$, no $\geq$-$\mathcal{X}'$-cycle exists in $\vec{\mathcal{X}}$ and let us show that no cycle of length $\mathcal{X}$ exists.

To exhibit a contradiction we will need to use the following necessary condition [7]: to be single-peaked, a profile U needs to be worst-restricted, *i.e.* for any triple of objects $\mathrm{O} = (\mathcal{X}_a, \mathcal{X}_b, \mathcal{X}_c) \in O^3$ there always exists an object $\mathcal{X}_j \in \mathrm{O}$ such that there exists an agent $\mathcal{X}$ with $\mathcal{X}_j \notin \mathrm{argmin}_{o_k \in O}\, \mathcal{X}\,(\mathcal{X}, \mathcal{X}_k)$ [36].

Because $\mathcal{X}$ is a $\geq$-$\mathcal{X}$-cycle, for all agent $\mathcal{X}_i \neq \mathcal{X}_1$ involved in $\mathcal{X}$ we have $\mathcal{X}_{i-1} \succ_{\mu_i} \mathcal{X}_i$ and $\mathcal{X}_k \succ_{\mu_1} \mathcal{X}_1$. As no $\geq$-$\mathcal{X}'$-cycle exists, with $\mathcal{X}' < \mathcal{X}$, for all agents $\mathcal{X}_i \neq \mathcal{X}_1$ involved in $\mathcal{X}$ and for all objects $\mathcal{X}_l$ in

$\mathbb{O}, \mathbb{O}_l \neq \mathbb{O}_i$ and $\mathbb{O}_l \neq \mathbb{O}_{i-1}$, we have $\mathbb{O}_i \succ_{\mu_i} \mathbb{O}_l$. Moreover for all objects $\mathbb{O}_l$ in $\mathbb{O}, \mathbb{O}_l \neq \mathbb{O}_1$ and $\mathbb{O}_l \neq \mathbb{O}_k$, we have $\mathbb{O}_1 \succ_{\mu_1} \mathbb{O}_l$. If the preferences do not respect these conditions, a $\geq$-$\mathbb{O}'$-cycle exists with $\mathbb{O}' < \mathbb{O}$.

Because the profile is worst-restricted, for all the triple of objects O in $\{\mathbb{O}_1, \ldots, \mathbb{O}_k\}$, at most two resources of O can be ranked last among O by the agents. Let us call $\mathbb{O}_w$ one of these objects ranked last by agent $\mathbb{O}_l$ and held by agent $\mathbb{O}_w$. Thanks to the previous paragraph, we know that $best(O, \mathbb{O}_w) = \mathbb{O}_{w-1}$ and so, because her preferences are single-peaked, $\mathbb{O}_w$ puts $\mathbb{O}_{w+1}$ in last position among $\mathbb{O}_{w-1}, \mathbb{O}_w, \mathbb{O}_{w+1}$. The same holds for agent $\mathbb{O}_{w+1}$ who ranks $\mathbb{O}_{w-1}$ in last position among $\mathbb{O}_{w-1}, \mathbb{O}_w, \mathbb{O}_{w+1}$ (because $\mathbb{O}\mathbb{O}\mathbb{O}(\mathbb{O}_{w+1}) = \mathbb{O}_w$). Therefore when we focus only on the three objects $\mathbb{O}_{w-1}, \mathbb{O}_w, \mathbb{O}_{w+1}$, each of them is ranked last among them by one agent which violates the condition of worst-restriction. The contradiction is set, no $\geq$-$\mathbb{O}$-cycle exists in $\vec{\mathbb{O}}$. □

Together with Proposition 4, Proposition 8 gives another interpretation of sequenceability in this domain:

Corollary 6.1. *If preferences are single-peaked (and additive), then an allocation $\vec{\mathbb{O}}$ is sequenceable if and only if it is swap-optimal.*

Proposition 1 by Damamme *et al.* [20] is much stronger than our Corollary 6.1, as it shows that swap-optimality is actually equivalent to Pareto-efficiency *when each agent receives a single resource*. Unfortunately, in our context where each agent can receive several items, this is no longer the case, as the following example shows:

Example 6.2. Consider this instance, single-peaked with respect to $1 \triangleright \cdots \triangleright 6$:

$$\begin{pmatrix} \dagger(1) & (2) & 3 & 4 & 5 & \dagger 6 \\ 1 & \dagger 3 & (4) & (5) & \dagger 6 & 2 \\ 1 & 2 & \dagger 4 & \dagger 5 & (6) & (3) \end{pmatrix}$$

The circled allocation is swap-optimal, but Pareto-dominated by the allocation marked with dags.

7 ENVY-FREENESS AND CEEI

The use of sequences of sincere choices can also be motivated by the search for a *fair* allocation protocol. Here, we will focus on two fairness properties and analyze their link with sequenceability.

The first of these notions is probably one of the most prominent fairness properties: envy-freeness [25, 38, 39].

Definition 7.1. Let $\mathbb{I}$ be an add-MARA instance and $\vec{\mathbb{O}}$ be an allocation. $\vec{\mathbb{O}}$ verifies the *envy-freeness property* (or is simply *envy-free*), when $\mathbb{O}_i(\mathbb{O}_i) \geq \mathbb{O}_i(\mathbb{O}_j)$, $\forall(\mathbb{O}\mathbb{O}) \in \mathcal{N}^2$ (no agent strictly prefers the share of any other agent).

The notion of *competitive equilibrium* is an old and well-known concept in economics [24, 40]. If equal incomes are imposed among the stakeholders, this concept becomes the *competitive equilibrium from equal incomes* [32], yielding a very strong fairness concept that has been recently explored both in artificial intelligence and in economics [12, 19, 33].

Definition 7.2. Let $\mathbb{I} = (\mathcal{N}, O, \mathbb{O})$ be an add-MARA instance, $\vec{\mathbb{O}}$ an allocation, and $\vec{\mathbb{O}} \in [0,1]^m$ a vector of prices. A pair $(\vec{\mathbb{O}}, \vec{\mathbb{O}})$ is said to form a *competitive equilibrium from equal incomes (CEEI)* if

$$\forall \mathbb{O} \in \mathcal{N} : \mathbb{O}_i \in \operatorname{argmax}_{\pi \subseteq O} \left\{ \mathbb{O}_i(\mathbb{O}) : \sum_{o_k \in \pi} \mathbb{O}_k \leq 1 \right\}.$$

In other words, $\mathbb{O}_i$ is one of the maximal shares that $\mathbb{O}$ can buy with a budget of 1, given that the price of each object $\mathbb{O}_k$ is $\mathbb{O}_k$.

We will say that allocation $\vec{\mathbb{O}}$ is a CEEI if there exists a vector $\vec{\mathbb{O}}$ such that $(\vec{\mathbb{O}}, \vec{\mathbb{O}})$ forms a CEEI.

As Bouveret and Lemaître [12] and Brânzei *et al.* [18] have shown, with additive preferences, every CEEI allocation is envy-free. In this section, we investigate the question of whether an envy-free or CEEI allocation is necessarily sequenceable. For envy-freeness, the answer is negative.

Proposition 9. *There exist non-sequenceable envy-free allocations, even if the agents' preferences are strict on shares.*

Proof. A counterexample with strict preferences on shares is given in Example 4.2 above, for which we can check that the circled allocation $\vec{\mathbb{O}}$ is envy-free and non-sequenceable. □

Concerning CEEI, it is already well known that any CEEI allocation is Pareto-optimal (hence sequenceable) if the preferences are strict on shares [12]. This is also a consequence of the First Welfare Theorem introduced by Babaioff *et al.* [5] for indivisible goods.

However, surprisingly, this result does not hold anymore if the preferences are not strict on shares, as the following example shows:

$$\begin{pmatrix} \dagger(2) & \dagger 3 & 3 & (2) \\ 2 & 3 & \dagger(4) & 1 \\ 0 & (4) & 2 & \dagger 4 \end{pmatrix}$$

The circled allocation is CEEI (with prices 0.5, 1, 1, 0.5) but is ordinally necessary (hence also additively) dominated by the allocation marked with †.

In spite of this negative result, we can still guarantee a certain level of efficiency for CEEI allocations:

Proposition 10. *Every CEEI allocation is sequenceable.*

Proof. We will show that no allocation can be at the same time non-sequenceable and CEEI. Let $\vec{\mathbb{O}}$ be a non-sequenceable allocation. We can use the same terms and notations than in the proof of Proposition 3, especially concerning the dominance cycle.

Let C be the set of agents concerned by the cycle. $\vec{\mathbb{O}}$ contains the following shares:

$$\mathbb{O}_i = \{\mathbb{O}_k\} \cup \mathbb{O}_i \quad \mathbb{O}_{i+1} = \{\mathbb{O}_i\} \cup \mathbb{O}_{i+1} \quad \ldots. \quad \mathbb{O}_k = \{\mathbb{O}_{k-1}\} \cup \mathbb{O}_k$$

whereas the allocation $\vec{\mathbb{O}}'$ that dominates it, contains:

$$\mathbb{O}'_i = \{\mathbb{O}_i\} \cup \mathbb{O}_i \quad \mathbb{O}'_{i+1} = \{\mathbb{O}_{i+1}\} \cup \mathbb{O}_{i+1} \quad \ldots. \quad \mathbb{O}'_k = \{\mathbb{O}_k\} \cup \mathbb{O}_k$$

the other shares being unchanged from $\vec{\mathbb{O}}$ to $\vec{\mathbb{O}}'$.

Suppose that $\vec{\mathbb{O}}$ is CEEI. This allocation must satisfy two kinds of constraints. First, $\vec{\mathbb{O}}$ must satisfy the price constraint. If we write $\mathbb{O}(\mathbb{O}) \stackrel{def}{=} \sum_{o_k \in \pi} \mathbb{O}_k$, we have, $\forall \mathbb{O} \in C, \mathbb{O}(\mathbb{O}_i) \leq 1$ (1).

Next, $\vec{\mathbb{O}}$ must be optimal: every share having a higher utility for an agent than her share in $\vec{\mathbb{O}}$ costs strictly more than 1. Provided that $\forall \mathbb{O} \in C : \mathbb{O}_i(\mathbb{O}'_i) > \mathbb{O}_i(\mathbb{O}_i)$ (because $\vec{\mathbb{O}}'$ substitutes more preferred objects to less preferred objects in $\vec{\mathbb{O}}$), this constraint can be written as $\forall \mathbb{O} \in C, \mathbb{O}(\mathbb{O}'_i) > 1$ (2).

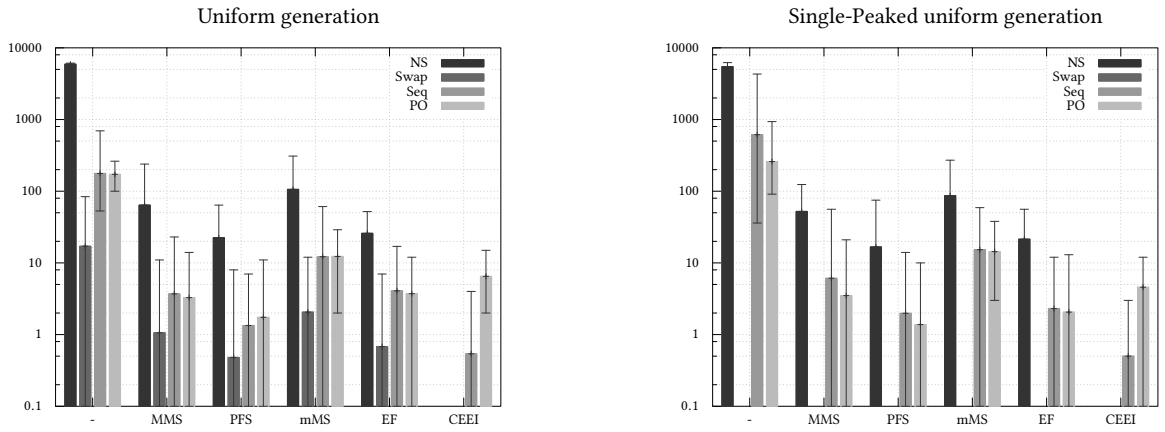


Figure 1: Distribution of the number of allocations by pair of (efficiency, fairness) criteria.

By summing equations (1) and (2), provided that all shares are disjoint, we obtain

$$\mathbb{X}(\bigcup_{j\in C} \mathbb{X}_j) \leq |C| \quad \text{and} \quad \mathbb{X}(\bigcup_{j\in C} \mathbb{X}'_j) > |C|$$

Yet, $\bigcup_{j\in C} \mathbb{X}_j = \bigcup_{j\in C} \mathbb{X}'_j$ (because the allocation $\vec{\mathbb{X}}'$ is obtained from $\vec{\mathbb{X}}$ by simply swapping objects between agents in C). The two previous equations are contradictory. □

8 EXPERIMENTS

We have exhibited in Sections 4 and 5 a "hierarchy of allocation efficiency" made of several steps: Pareto-optimal (PO), sequenceable (Seq), {cycle-deal-optimal}, non-sequenceable (NS). A natural question is to know, for a given instance, which proportion of allocations are located at each level of the scale. We give a first answer by experimentally studying the distribution of allocations between the different levels. For cycle-deal optimality, we focus on the simplest type of deals, namely, >-swap-deals. We thus have a linear scale of efficiency concepts, from the strongest to the weakest: PO → Seq → Swap → NS. We also analyze the relation between efficiency and various notions of fairness by linking this latter scale with the 6-level scale of fairness introduced by Bouveret and Lemaître [12]: CEEI → Envy-Freeness (EF) → min-max share (mMS) → proportionality (PFS) → max-min share (MMS) → NS. We generate 50 add-MARA instances involving 3 agents-8 objects, using two different models. For both models, a set of weights are uniformly drawn in the interval $[\![0, 100]\!]$ and the instances are then normalized. For the second model, these weights are reordered afterwards to make the preferences single-peaked. For each instance, we generate all 6561 allocations, and identify for each of them the *highest* level of fairness and efficiency satisfied. The average number of allocations with min-max interval is plotted as a box for each level on a logarithmic scale in Figure 1.

Note that some fairness and efficiency tests require to solve NP-hard or coNP-hard problems (MMS, mMS, and PO tests). These tests are delegated to an external ILP solver. This is especially interesting for the CEEI test which is known to be NP-hard [18], and for which, to the best of our knowledge, no practical method had been described before. The implementation is available as a fully documented and tested Free Python library.[5]

We note several interesting facts. First, a majority of allocations do not have any efficiency nor fairness property (first black bar on the left). Second, the distribution of allocations on the efficiency scale seems to be related to the fairness criteria: a higher proportion of swap-optimal or sequenceable allocations are found among envy-free allocations than among allocations that do not satisfy any fairness property, and for CEEI allocations, there are even more Pareto-optimal allocations than just sequenceable ones. Lastly, the absence of vertical bar for swap-optimality in the experiments concerning single-peaked preferences confirms the results of Corollary 6.1: in this context, no allocation can be swap-optimal but not Sequenceable; hence, all the allocations that are swap-optimal are contained in the bars concerning sequenceable or Pareto-optimal allocations. Similarly, the absence of bars for swap-optimality and NS (non-sequenceable) in both graphs for the CEEI case confirms the result of Proposition 10.

9 CONCLUSION

In this paper, we have shown that picking sequences and cycle-deals can be reinterpreted to form a rich hierarchy of efficiency concepts. Many interesting questions remain open, such as the complexity of computing cycle-deals or the link between efficiency concepts and social welfare. One could also think of further extending the efficiency hierarchy by studying restrictions on possible sequences (*e.g.* alternating) or extending the types of deals to non-cyclic ones.

ACKNOWLEDGMENTS

This work is partially supported by the ANR project 14-CE24-0007-01 - CoCoRICo-CoDec.

REFERENCES

[1] Haris Aziz, Péter Biró, Jérôme Lang, Julien Lesca, and Jérôme Monnot. 2016. Optimal Reallocation Under Additive and Ordinal Preferences. In *Proceedings of*

[5] Available at: https://gricad-gitlab.univ-grenoble-alpes.fr/bouveres/fairdiv.

the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'16). International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 402–410.

[2] Haris Aziz, Serge Gaspers, Simon Mackenzie, and Toby Walsh. 2015. Fair assignment of indivisible objects under ordinal preferences. *Artificial Intelligence* 227 (2015), 71–92.

[3] Haris Aziz, Thomas Kalinowski, Toby Walsh, and Lirong Xia. 2016. Welfare of Sequential Allocation Mechanisms for Indivisible Goods. In *22nd European Conference on Artificial Intelligence (ECAI 2016)*, Gal A. Kaminka, Maria Fox, Paolo Bouquet, Eyke Hüllermeier, Virginia Dignum, Frank Dignum, and Frank van Harmelen (Eds.). IOS Press, 787–794.

[4] Haris Aziz, Toby Walsh, and Lirong Xia. 2015. Possible and necessary allocations via sequential mechanisms. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*. AAAI Press, 468–474.

[5] Moshe Babaioff, Noam Nisan, and Inbal Talgam-Cohen. 2017. Competitive equilibria with indivisible goods and generic budgets. *arXiv preprint arXiv:1703.08150* (2017).

[6] Sophie Bade. 2018. Matching with single-peaked preferences. *Journal of Economic Theory* 180 (2018), 81–99.

[7] Miguel A Ballester and Guillaume Haeringer. 2011. A characterization of the single-peaked domain. *Social Choice and Welfare* 36, 2 (2011), 305–322.

[8] Nikhil Bansal and Maxim Sviridenko. 2006. The Santa Claus problem. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing (STOC '06)*. ACM, New York, NY, USA, 31–40.

[9] Nicola Bianchessi, Jean-François Cordeau, Jacques Desrosiers, Gilbert Laporte, and Vincent Raymond. 2007. A Heuristic for the Multi-satellite, Multi-orbit and Multi-user Management of Earth Observation Satellites. *European Journal of Operational Research* 177, 2 (March 2007), 750–762.

[10] David Black. 1948. On the rationale of group decision-making. *The journal of political economy* (1948), 23–34.

[11] Sylvain Bouveret and Jérôme Lang. 2011. A general elicitation-free protocol for allocating indivisible goods. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI-11)*, Toby Walsh (Ed.). IJCAI/AAAI, Barcelona, Spain, 73–78.

[12] Sylvain Bouveret and Michel Lemaître. 2016. Characterizing conflicts in fair division of indivisible goods using a scale of criteria. *Autonomous Agents and Multi-Agent Systems* 30, 2 (2016), 259–290.

[13] Sylvain Bouveret and Michel Lemaître. 2016. Efficiency and Sequenceability in Fair Division of Indivisible Goods with Additive Preferences. Proceedings of the Sixth International Workshop on Computational Social Choice (COMSOC'16). (2016).

[14] Steven J. Brams, Marc D. Kilgour, and Christian Klamler. 2012. The undercut procedure: an algorithm for the envy-free division of indivisible items. *Social Choice and Welfare* 39, 2-3 (2012), 615–631.

[15] Steven J. Brams and Daniel King. 2005. Efficient Fair Division—Help the Worst off or Avoid Envy? *Rationality and Society* 17, 4 (2005), 387–421.

[16] Steven J. Brams and Alan D. Taylor. 1996. *Fair Division — From Cake-cutting to Dispute Resolution*. Cambridge University Press.

[17] Steven J. Brams and Alan D. Taylor. 2000. *The Win-win Solution. Guaranteeing Fair Shares to Everybody*. W. W. Norton & Company.

[18] Simina Brânzei, Hadi Hosseini, and Peter Bro Miltersen. 2015. Characterization and computation of equilibria for indivisible goods. In *International Symposium on Algorithmic Game Theory*. Springer, 244–255.

[19] Eric Budish. 2011. The Combinatorial Assignment Problem: Approximate Competitive Equilibrium from Equal Incomes. *Journal of Political Economy* 119, 6 (dec 2011), 1061–1103.

[20] Anastasia Damamme, Aurélie Beynier, Yann Chevaleyre, and Nicolas Maudet. 2015. The Power of Swap Deals in Distributed Resource Allocation. In *The 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015)*. Istanbul, Turkey, 625–633.

[21] Bart de Keijzer, Sylvain Bouveret, Tomas Klos, and Yingqian Zhang. 2009. On the complexity of efficiency and envy-freeness in fair division of indivisible goods with additive preferences. In *Proceedings of the 1st International Conference on Algorithmic Decision Theory (ADT'09) (Lecture Notes in Artificial Intelligence)*. Springer Verlag, Venice, Italy, 98–110.

[22] Edith Elkind, Martin Lackner, and Dominik Peters. 2016. Preference Restrictions in Computational Social Choice: Recent Progress. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*. 4062–4065.

[23] Ulle Endriss, Nicolas Maudet, Fariba Sadri, and Francesca Toni. 2006. Negotiating Socially Optimal Allocations of Resources. *Journal of Artificial Intelligence Research* 25 (2006), 315–348.

[24] Irving Fisher. 1892. *Mathematical Investigations in the Theory of Value and Prices, and Appreciation and Interest*. Augustus M. Kelley, Publishers.

[25] Duncan K. Foley. 1967. Resource Allocation and the Public Sector. *Yale Economic Essays* 7, 1 (1967), 45–98.

[26] Judy Goldsmith and Robert H Sloan. 2007. The AI conference paper assignment problem. In *Proc. AAAI Workshop on Preference Handling for Artificial Intelligence, Vancouver*. 53–57.

[27] Thomas Kalinowski, Nina Narodytska, and Toby Walsh. 2013. A Social Welfare Optimal Sequential Allocation Procedure. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI-13)*, Francesca Rossi (Ed.). IJCAI/AAAI, Beijing, China, 227–233.

[28] Thomas Kalinowski, Nina Narodytska, Toby Walsh, and Lirong Xia. 2013. Strategic Behavior when Allocating Indivisible Goods Sequentially. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI-12)*. AAAI Press, Bellevue, WA, 452–458.

[29] David A Kohler and R Chandrasekaran. 1971. A class of sequential games. *Operations Research* 19, 2 (1971), 270–277.

[30] Michel Lemaître, Gérard Verfaillie, and Nicolas Bataille. 1999. Exploiting a Common Property Resource under a Fairness Constraint: a Case Study. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*. Stockholm, Sweden, 206–211.

[31] Richard Lipton, Evangelos Markakis, Elchanan Mossel, and Amin Saberi. 2004. On Approximately Fair Allocations of Divisible Goods. In *Proceedings of the 5th ACM Conference on Electronic Commerce (EC-04)*. ACM, New York, NY, 125–131.

[32] Hervé Moulin. 2003. *Fair Division and Collective Welfare*. MIT Press.

[33] Abraham Othman, Tuomas Sandholm, and Eric Budish. 2010. Finding approximate competitive equilibria: efficient and fair course allocation. In *Proceedings of the 9th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS-10)*, Wiebe van der Hoek, Gal A. Kaminka, Yves Lespérance, Michael Luck, and Sandip Sen (Eds.). IFAAMAS, Toronto, Canada, 873–880.

[34] Senjuti Basu Roy, Ioanna Lykourentzou, Saravanan Thirumuruganathan, Sihem Amer-Yahia, and Gautam Das. 2015. Task assignment optimization in knowledge-intensive crowdsourcing. *VLDB J.* 24, 4 (2015), 467–491.

[35] Tuomas W. Sandholm. 1998. Contract Types for Satisficing Task Allocation: I. Theoretical Results. In *Proceedings of the AAAI Spring Symposium: Satisficing Models*, Sandip Sen (Ed.). AAAI Press, Menlo Park, California, 68–75.

[36] Amartya K Sen. 1966. A possibility theorem on majority decisions. *Econometrica* (1966), 491–499.

[37] Lloyd Shapley and Herbert Scarf. 1974. On cores and indivisibility. *Journal of mathematical economics* 1, 1 (1974), 23–37.

[38] Jan Tinbergen. 1953. *Redeljke Inkomensverdeling*. N. V. DeGulden Pers., Haarlem.

[39] Hal R. Varian. 1974. Equity, Envy and Efficiency. *Journal of Economic Theory* 9 (1974), 63–91.

[40] Léon Walras. 1874. *Éléments d'économie politique pure ou Théorie de la richesse sociale* (1st ed.). L. Corbaz.

Modèles d'apprentissage automatique de la persistance aux médicaments : application au cancer du sein

Thomas Janssoone, Pierre Rinder, Clémence Bic, Dorra Kanoun and Pierre Hornus

***Abstract*—L'adhésion aux traitements médicamenteux, c'est-à-dire le fait de prendre ses médicaments conformément à la posologie, a été au centre des attentions ces dernières années. L'Organisation Mondiale de la Santé souligne dans ses rapports [1] que le fait de ne pas respecter le plan de traitement est en réalité un problème majeur, car cela compromet gravement l'efficacité de thérapie à long terme et augmente le coût des services de santé. En effet, dans les pays développés, environ 50% seulement des patients atteints de maladies chroniques suivent correctement leurs traitements. Dans cet article, nous présentons nos travaux sur la modélisation de la consommation de médicaments par les patientes dans les traitements du cancer du sein. Nous nous concentrons sur la persistance au traitement qui indique si le patient a arrêté son parcours de soins avant la fin prévue. Nous détaillons les différentes étapes de notre approche. À partir des données de remboursement du système de santé français, nous reconstruisons les parcours de soins des patients. Ensuite, des méthodes statistiques sont utilisées pour prédire la non-persistance des hormonothérapies et estimer les variables explicatives des décisions de nos modèles. Nous montrons ainsi que les variables explicatives de notre étude sont conforme aux études médicales antérieures sur les facteurs de non persistance. Nous détaillons ensuite la comparaison de plusieurs méthodes d'apprentissage automatique pour prédire un arrêt de traitement illégitime et discutons leurs limites, en particulier sur l'interprétabilité de leurs résultats.[2]**

***Index Terms*—Persistence, Adherence, analyse de survie, machine learning, EHR SNIIRAM**

I. Introduction

Durant les dernières décennies, le recours aux traitements oraux s'est fortement développé[5, 15]. Ainsi, en cancérologie, ces traitements oraux devraient atteindre 25% des médications en 2025[3]. Ce recours aux traitements oraux anticancer fait des problèmes d'adhésion un sujet majeur. Dans leur revue de l'état de l'art, Krueger et al. [10] ont souligné l'importance de ce sujet, car *"la non-adherence (ou la non-persistance) peut entraîner une augmentation de la morbidité, de la mortalité et des coûts de la santé"*. Entre autres, ce document complète la définition proposée par Osterberg and Blaschke [16] en tant que *"L'adhérence (ou le respect) d'un traitement médicamenteux correspond à la [...] mesure dans laquelle les patients prennent leurs médicaments tels que prescrits par leur médecin."*. Le terme adhérence est un concept composite qui englobe généralement les notions d'observance et de persistance. L'observance thérapeutique décrit, en général, dans quelle mesure le patient se conforme à la prescription (nombre de prise, posologie, etc.). La persistance représente le respect de la durée du traitement jusqu'à son terme et ce, sans interruption de celui-ci. Cet article se concentre sur la non-persistance en médecine en tant que mesure du fait qu'un patient arrête de suivre un traitement avant la fin de la période recommandée.

Pour prévenir ces arrêts illégitimes, une solution courante consiste à mettre en place des programmes de soutien aux patients comprenant, par exemple, 1) des ateliers pédagogique avec des échanges d'informations et de conseils pour les patients, 2) des sessions de soutien et de coaching dispensées par des infirmières (par téléphone ou en face-à-face), et 3) l'envoi d'informations aux professionnels de santé traitant le patient. Ces programmes peuvent se montrer très efficaces : par exemple, Krolop et al. [9] a montré que le coaching des pharmaciens a amélioré la persistance de 12% et le système de rappel par SMS de Spoelstra et al. [17] a augmenté de 10% la persistance au traitement.
Pourtant, il existe deux principales limites à ces interventions : 1) L'utilisation de l'intervention humaine est efficace, mais très coûteux en temps et en personnel ce qui limite sa portée, 2) L'utilisation des technologies numériques (notifications et explications) est trop générique et peut être perçu comme trop intrusif (rappels quotidiens) ce qui peut désintéresser ou angoisser des patients. Pour optimiser la pertinence de ces interventions, nous proposons d'utiliser des techniques d'apprentissage automatique sur les données de consommation des patientes atteintes d'un cancer du sein. Nous cherchons à estimer le risque de non-persistance au traitement à différents moments du parcours de soins. Le but est de pouvoir prédire les moments les plus appropriés

Thomas Janssoone, Pierre Rinder, Clémence Bic et Pierre Hornus sont à Sêmeia company, 9 cour de petites écuries, 75011 Paris, France e-mail: (cf http://semeia.io).

Dorra Kanoun travaille à la Clinique Pasteur, Toulouse, France.

[1] http://www.who.int/chp/knowledge/publications/Persistence_full_report.pdf

[2] Ces travaux ont été présentés aux workshop AI for Good et ML for Health de NeurIPS 2019

[3] http://www.unicancer.fr/patients/quelle-prise-charge-cancers-2020

pour intervenir auprès des patients à risque, ceux qui ont réellement besoin d'aide. Ainsi, les populations bénéficieront d'un accompagnement adapté à leurs profils et à leurs besoins, et les interventions humaines seront réservées à des situations réellement critiques.

Pour déterminer les catégories de patients à risque et les moments appropriés pour les contacter, nous développons des modèles prédictifs basés sur des données pseudonymisées. Ces modèles prédictifs sont entraînés avec les données de remboursement de l'assurance-maladie française (SNIIRAM).
Dans la suite de cet article, nous examinons d'abord les approches précédentes, puis nous présentons nos modèles et discutons de nos résultats.

II. État de l'art

Compte tenu de l'importance du phénomène de non-persistance, de nombreuses enquêtes ont tenté d'en identifier les facteurs déterminants afin d'améliorer les interventions et donc le respect du traitement. L'examen de nombreuses publications scientifiques à base d'observations fournit une évaluation quantitative intéressante de la recherche menée sur le sujet [3, 12]. Cette méta-analyse indique que l'âge croissant des patients et le niveau de complexité du traitement (médicaments multiples, injections, ...) sont des facteurs augmentant le risque de non-persistance. De même, le manque d'éducation et, plus encore, les faibles revenus sont liés à une augmentation de ce risque. Une autre étude met en évidence l'impact de la santé mentale des patients et montre que les épisodes dépressifs ont un impact très négatif sur la conformité du patient aux prescriptions des professionnels de santé [4].

Par ailleurs, d'autres études de DiMatteo montrent que d'autres facteurs influencent également la persistance. Par exemple, en faisant la distinction entre la gravité objective de la maladie du patient et la prise de conscience de la gravité de sa pathologie par le patient, il souligne que les connaissances du patient de sa maladie influencent le niveau d'observance et non la gravité réelle de la maladie. Cela souligne l'importance du rôle de la pédagogie des patients pour renforcer leurs persistances aux traitements. De même, une méta-analyse montre ainsi l'influence de l'entourage du patient (soutien de son conjoint, de sa famille, de ses proches et de l'environnement social au sens large) dans le suivi correct de ses traitements [2].
Ces études fournissent des indications a priori pour la détection de profils de risque de non-persistance. Dans le même temps, ils soulignent l'intérêt d'identifier et d'accompagner ces patients dans la prise de leurs médicaments.

Enfin, Franklin et al. [6] soulignent la difficulté d'utiliser ces informations pour prédire la persistance. Ils évaluent différentes approches, basées sur des régressions logistiques, pour définir trois catégories de prédicteurs de persistance. Ils montrent alors que l'utilisation d'informations de recensement ou de données de transaction conduit à une mauvaise prévision. Cependant, ils soulignent que l'indication de la persistance au cours du premier mois du traitement augmente considérablement la précision des résultats. Lo-Ciganic et al. [11] confirment cette nuance sur le poids de chaque variable de prédiction de persistance. Ils utilisent des points saillants de forêts de survie aléatoires pour trouver des seuils de persistance spécifiques à un patient afin de distinguer les risques d'hospitalisation. Là encore, les principales variables sont liées à l'historique du patient et aux transactions précédentes.

Nous proposons dans cet article d'explorer ces solutions afin de prédire le risque d'un arrêt illégitime au cours d'un traitement.

III. La base de données SNIIRAM

A. Introduction

Afin d'améliorer le recours à l'intervention humaine et d'optimiser l'efficacité de technologies numériques (applications, chatbot, ...), nous proposons l'utilisation de techniques d'apprentissage automatique sur les données de consommation de patientes atteintes d'un cancer du sein. L'objectif est de catégoriser les patients en classes de risque en fonction de leurs caractéristiques afin de connaître les moments les plus appropriés pour les contacter. Ainsi, les personnes bénéficieront d'un accompagnement adapté à leurs profils et à leurs besoins, et les interventions humaines seront réservées aux situations pour lesquelles elles sont réellement nécessaires. Pour déterminer ces catégories de patients à risque et ces moments appropriés, nous développons des modèles prédictifs basés sur des données pseudonymisées. Nous avons retravaillé ces données pour fournir différentes représentations d'un parcours de soins puis les avons analysées à l'aide d'algorithmes d'apprentissage automatique.

Ces modèles prédictifs sont entraînés et testés sur les données de remboursement du système de santé français (SNIIRAM). Le SNIIRAM est l'une des plus grandes bases structurées de données de la santé au monde. L'utilisation de ces données massives permet l'application de modèles complexes et la détection de signaux faibles. Les données utiles sont, par exemple, les hospitalisations, les achats de médicaments ou les informations contextuelles sur le patient (âge, services gouvernementaux, informations géographiques, ...). Plus de détails peuvent être trouvés dans [18]. Des travaux antérieurs ont déjà montré l'intérêt de l'extraction massive de données pour faciliter le diagnostic, soit en prenant

toutes les informations pour une approche "statique" [14], ou, plus récemment, en intégrant également des informations dynamiques [13]. D'autres études ont été menées sur les déterminants de l'observance, en particulier pour le cancer du sein.

Notre étude porte sur le cancer du sein chez les femmes sur une partie des données du SNIIRAM. La cohorte de l'étude comprend 50% de femmes (tirées au sort) répondant aux critères suivants :

- diagnostiquée avec un cancer du sein
- ayant acheté au moins l'une des molécules suivantes pour la période étudiée : *Anastrozole*, *Capecitabine*, *Cyclophosphamide*, *Etoposide*, *Everolimus*, *Exemestane*, *Lapatinib*, *Letrozole*, *Megestrol*, *Melphalan*, *Tamoxifen*, *Toremifene* and *Vinorelbine*

L'extraction concerne les consommations entre 2013 et 2015 et se compose de trois catégories principales :

- Transactions pharmaceutiques (molécule, nombre de doses, date, ...)
- Hospitalisations (diagnostic, date de début, date de fin, ...)
- Informations sur le patient (âge, service, date du diagnostic éventuel de l'affection longue durée(*ALD*) , pathologies, ...)

Un pré-traitement a été effectué sur la variable ancienneté de l'ALD qui représente le nombre de jours écoulés depuis le diagnostic de la maladie (indiqué dans ALD 30). Cette variable a la particularité de contenir des valeurs extrêmes, qui biaisent l'estimation pour les modèles supposant un effet linéaire. C'est pourquoi un logarithme lui est appliqué pour éliminer ce biais. Cela permet tout de même de conserver l'ordre de grandeur de la durée (en jours, semaines, mois ou années).

B. Phases de traitement

Tout d'abord, nous avons retravaillé les données brutes pour montrer les différentes phases du traitement. Une phase est une période d'absorption continue d'une molécule ou d'hospitalisations pour chimiothérapie ou radiothérapie. Cela permet de reconstruire le cheminement du patient.

Suivant la recommandation d'oncologues, le critère permettant d'identifier une fin de phase est l'existence d'une période de deux mois après le délai médian couvert par le dernier achat (ou hospitalisation) sans nouvel achat de la molécule (ou hospitalisation de même type). La date de la dernière dose théorique est obtenue en calculant l'intervalle médian entre deux achats de la molécule ou deux hospitalisations du même type : ce comportement médian est considéré comme conforme au dosage. La fin de cette période après la dernière boîte achetée correspond à la date de la dernière prise théorique. Ainsi, le délai médian est de 30 jours entre deux achats d'une boite de 30 doses de tamoxifène. Pour les médicaments, les jours d'hospitalisation sont exclus de cette période, car le médicament est alors fourni par le personnel médical.

Par exemple : le temps médian entre deux chimiothérapies est de trois semaines. S'il y a une période de 2 mois et 3 semaines sans chimiothérapie, ceci est considéré comme une pause dans la phase.

Une phase du traitement est considérée comme censurée par l'un des arrêts légitimes (décès, changement de traitement, problème cardiaque grave ou début de soins palliatifs) si cet événement survient moins de deux mois après la date de la dernière dose théorique. Par exemple, si un patient a acheté une boîte de 30 comprimés le 1êr janvier 2007, l'événement doit avoir lieu avant le 31 mars (30 + 2x30). La date de la mort est présente dans les données initiales, les commutateurs sont identifiés par le début d'une nouvelle phase de traitement et de soins palliatifs comme diagnostic principal (qui est repéré avec une étiquette Z515 dans la base de données). La censure des données provoquée par la fin de la période d'extraction (fin décembre 2015) est également considérée comme un arrêt légitime. Si la date de fin d'extraction des données est inférieure à deux mois à compter de la dernière consommation théorique, alors, de la même manière que pour les arrêts légitimes, la phase de traitement est considérée comme censurée. Si aucun des arrêts légitimes ne se produit, la phase est considérée comme se terminant par un arrêt illégitime, ce qui signifie un défaut de persistance.

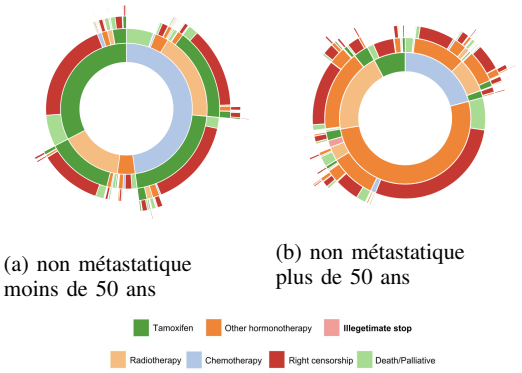
Pour chaque phase, les données suivantes sont calculées :

- Dates de début et de fin, nombre d'admissions ou d'hospitalisations, molécule ou type d'hospitalisation
- Type de fin de traitement (changement (switch), mort, arrêt, censure à droite)
- Informations sur le patient (comorbidités, nombre de consultations au cours de la première année de traitement, âge, ...)
- Interventions sur le sein (mastectomie) au cours des trois mois précédant la phase étudiée

En observant comment ces phases se succèdent, on constate la diversité des parcours des patients. La Figure 1 montre quelques *sunburst* de parcours patients selon des critères tels que l'âge et la gravité du cancer. Nous proposons d'évaluer différentes façons de déterminer si la fin d'une phase est légitime ou non. Nous nous concentrons sur la consommation de *Tamoxifene* car c'est la molécule la plus utilisée. De plus, cette molécule est prescrite jusqu'à 10 ans. Aucun patient n'est donc supposé avoir arrêté son traitement au cours de la période d'observation (3 ans) en raison de la fin de leur ordonnance (hors arrêt légitime décès, soins palliatif, passage à une autre médication).

IV. Analyse de survie

Pour mesurer le taux de non-persistance dans le temps, nous utilisons l'estimateur de Kaplan-Meier[8]. Cet estimateur utilise des statistiques non-paramétriques pour évaluer la fonction de survie sur un état. Par exemple, il est utilisé pour estimer le nombre de patients



(a) non métastatique moins de 50 ans

(b) non métastatique plus de 50 ans

(c) légende

(d) métastatique moins de 50 ans

(e) métastatique plus de 50 ans

Fig. 1: Sunbursts de parcours patients selon leurs âges et la caractérisation métastatique du cancer

vivant pendant un certain temps après un traitement, le temps nécessaire à une défaillance de la machine,... Son intérêt est de prendre en compte les données censurées, notamment par la censure de droite, chaque observation étant pondérée en fonction du nombre d'observations censurées précédemment. Les quatre facteurs de censure sont les suivants : commutation, décès, soins palliatifs et fin de l'extraction. La durée de survie en phase de traitement est ainsi estimée en prenant en compte des facteurs de censure tels que la fin de traitement légitime ainsi que la censure liée à la fin d'extraction (fin 2015).

La Figure 2 montre les variations de la fonction de risque représentant les taux d'abandon du traitement en fonction du temps. Le taux d'abandon est élevé au début de la phase, au cours des 150 premiers jours. Au cours des 5 premiers mois, la courbe est nettement supérieure au reste des valeurs. Cette période à haut risque sera donc la période la plus bénéfique pour aider les patients. L'estimateur de Kaplan-Meier nous permet d'analyser la survie, mais nous devons utiliser un modèle de régression pour examiner l'influence des facteurs sur les différentes variables.
Nous avons utilisé un modèle de Cox [1] pour identifier les caractéristiques liées à une faible persistance. La régression de Cox estime un effet fixe de chaque variable

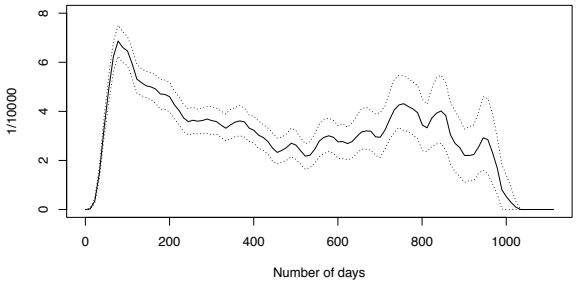


Fig. 2: La fonction de risque de Kaplan-Meier pour le Tamoxifène, qui représente le taux d'échec (ici, l'abandon du médicament) au moment t. La courbe en trait plein représente la valeur estimée et les points en pointillés représentent l'intervalle de confiance à 95% (calculé par bootstrapping).

par rapport au comportement moyen des patients. Il repose sur deux hypothèses fortes :

(1) l'effet attendu de chaque variable est linéaire
(2) l'effet de chaque variable ne varie pas dans le temps. Un exemple dans notre cas est que, si le poids trouvé pour la variable *CMU-C* est de 1.40, nous supposons qu'une personne qui bénéficie de la *CMU-C* ont 40% plus de risque d'interrompre leur traitement qu'une personne n'en bénéficient pas

Les variables explicatives du modèle de Cox sont les caractéristiques de la phase, du cheminement du patient et du profil du patient. Afin d'extraire les variables les plus significatives et d'estimer de manière fiable le coefficient associé, le modèle a été estimé à l'aide du processus itératif suivant :

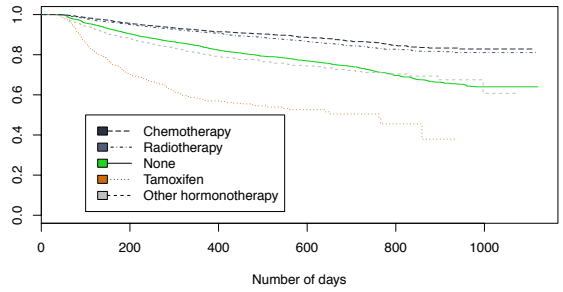
- estimation des coefficients pour l'ensemble des variables,
- sélection des variables ayant une p-valeur inférieure au seuil de 0,05
- nouvelle estimation des coefficients pour le modèle limitée aux variables sélectionnées

	coefficient calculés pour Cox	rapport de chance (exp(coefficient))	p-value
CMU-C	3.84e-01	1.47	6.6e-04
ACS	4.16e-01	1.52	1.9e-03
Temps écoulé depuis le statut ALD (log)	7.88e-02	1.08	3.9e-02
Nombre de consultation médicale	-5.70e-03	0.994	1.2e-02
Maladie psychiatrique	1.78e-01	1.19	9.3e-03
Hospitalisation récente avec diagnostic C50: tumeurs malignes du sein	-3.83e-01	0.682	3.6e-07
Dernier traitement - tamoxifene	9.12e-01	2.49	<1e-10
Dernier traitement - radiothérapie	-7.09e-01	0.492	<1e-10
Dernier traitement - chimiotherapie	-8.62e-01	0.422	<1e-10
Ménopause	1.17e-01	1.12	2.8e-02

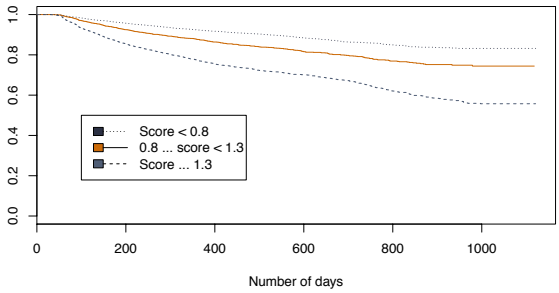
TABLE I: Poids calculés avec la régression de Cox montrant leur influence différente. Le rapport de chance indique l'impact de la variable sur le risque moyen (1.5 signifiant 50% de risque en plus de non-persistance).

Les coefficients les plus influents sont présentés dans la Table I. Nous retrouvons les caractéristiques mises en évidence dans la littérature que le modèle indique comme déterminantes pour la persistance du patient dans son traitement. Nous pouvons ainsi évaluer l'impact de l'âge, du soutien social ou d'une maladie antérieure (psychiatrie, mastectomie, ...). Nous soulignons également que l'influence du traitement antérieur à la phase actuelle a une influence majeure.

Cela nous a incités à analyser cette valeur en particulier. La Figure 3a montre les différentes fonctions de survie en fonction du traitement précédant la phase actuelle de Tamoxifène. Nous voyons trois types d'influences. Tout d'abord, le parcours classique: une hospitalisation (ici chimiothérapie ou radiothérapie) précédant la phase en cours présente le risque le plus faible d'abandon. Deuxièmement, une hormonothérapie autre que le Tamoxifène a été utilisée, correspond à un *switch* de traitement et présente un risque plus élevé. Enfin, une phase de Tamoxifène a été utilisée avant la phase actuelle. Cela suggère qu'un arrêt illégitime s'est produit avant la phase en cours et pourrait expliquer le risque le plus élevé de cette affaire. Néanmoins, cela souligne l'intérêt de notre modèle pour la recherche d'informations supplémentaires permettant de prédire l'évolution des phases d'ingestion du Tamoxifène. Comme illustré dans la Figure 3b. Nous pouvons utiliser les informations de base du patient pour calculer un score au début d'une phase Tamoxifène.



(a) Selon la phase de traitement précédente (les hospitalisations sont une chimiothérapie ou une radiothérapie)



(b) Selon le score spécifique calculé au début de la phase

Fig. 3: Exemples d'affinage des courbes de survie

Ensuite, nous sélectionnons la fonction de survie la plus précise qui donne la probabilité d'abandon pendant le nombre spécifique de jours de traitement, ce qui nous permet de prédire le risque d'abandon au fil du temps. Cependant, nous avons basé notre modèle sur l'hypothèse forte que l'effet de chaque variable ne varie pas dans le temps, comme expliqué dans item (2). Cette hypothèse de risque proportionnel peut être vérifiée à l'aide d'un test individuel de Schoenfeld et visualisée à l'aide du graphique en échelle logarithmique de la survie affiché dans le graphique. Figure 4. En utilisant les résidus de Schoenfeld, nous obtenons des valeurs de p qui vérifient cette hypothèse, à l'exception de l'hypothèse concernant le traitement précédent. Cela indique que le traitement précédent a des influences différentes sur la durée de la phase en cours et que notre hypothèse était trop forte pour cette variable. Cependant, la Figure 4 indique si les dangers sont approximativement proportionnels dans l'ensemble. Nous pouvons voir que cette hypothèse reste valable après les 20 premiers jours. Cette analyse nous donne des pistes pour améliorer notre modèle avec un accent particulier à mettre sur le début de la phase.

La prochaine étape de cette étude consiste à améliorer notre modèle avec des approches d'apprentissage automatique et d'autres modèles statistiques afin d'améliorer nos prévisions.

V. Etude de l'analyse de phase

Cette étude se concentre sur la prédiction d'un arrêt illégitime après une durée spécifique (3, 6 et 12 mois) à compter du début de la phase. Pour chaque phase, pour chaque période de temps, une étiquette indique si un arrêt illégitime s'est produit ou si le chemin de soins se poursuit correctement (pas d'arrêt ou légitime). Les données sont ensuite composées de, pour la période de 3 mois, 51220 patients avec 11,16% non adhérents, pour 6 mois, 44469 patients avec 16.53% non adhérents, pour 12 mois, 34132 patients avec 27.0% non adhérents.

Nous comparons la capacité des algorithmes

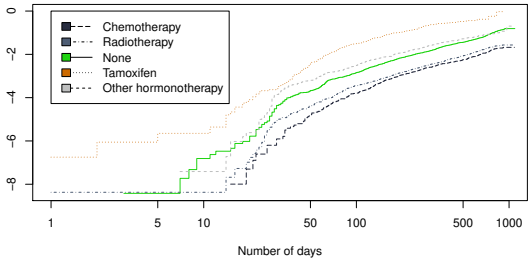
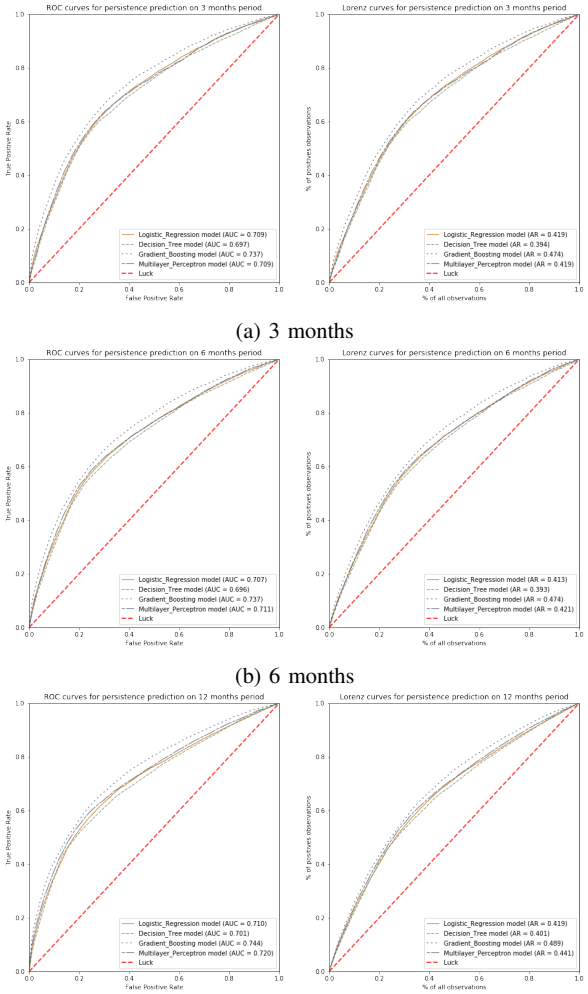


Fig. 4: Graphique en échelle logarithmique de la survie pour chaque traitement précédent. Nous observons une validation de notre modèle après les 20 premiers jours

d'apprentissage supervisé à prédire un arrêt légitime d'un illégitime avec les informations disponibles au début de la phase. Le premier modèle est une régression logistique, formée avec un nombre restreint de variables en fonction de leur valeur p (< 0.05). La régression logistique attribue à chaque variable un coefficient lié à son influence sur la tâche de classification, facilitant ainsi son interprétation. Le second modèle est un arbre de décision formé avec toutes les variables du jeu de données. Les paramètres ont été ajustés avec gridsearch. La fonction de prédiction d'un arbre de décision est facile à lire et à comprendre. Le troisième modèle est l'amplification de gradient qui, par rapport aux deux modèles précédents, est plus difficile à interpréter. Cependant, il se concentre sur les apprenants faibles et peut offrir une meilleure prédiction. Ce modèle a utilisé toutes les variables du jeu de données et les paramètres ont été ajustés avec la recherche par grille. Le dernier modèle est un perceptron multicouche (MLP). Les formations sont équilibrées pour obtenir un nombre égal de cas légitimes/illégitimes. Nous n'utilisons qu'une couche cachée, car l'ensemble de données équilibré a une taille relativement petite (9412 lignes pour la période de 3 mois). Nous avons également conservé toutes les variables pour calculer le modèle. 5 validations croisées de dossiers ont été effectuées dont les résultats sont affichés dans la Figure 5.

Les AUCs sont d'environ 0.70 avec les meilleures performances obtenues avec le Gradient Boosting. Une courbe de profit de précision cumulée (courbe de CAP ou de Lorenz) donne les deuxième et troisième mesures, indiquant la capacité d'un modèle à repérer avec précision un patient à risque. CAP $n\%$ indique le taux de bonnes classifications positives en en fonction des n prédictions aux risques les plus importants. Le Gradient Boosting et la MLP donnent les meilleurs résultats, mais il est difficile d'expliquer ces méthodes "boîte noire", ce qui pourrait limiter leurs utilisations, en particulier pour une application médicale. Ces premières approches pourraient déjà être appliquées: les soignants pourraient être informés lors de la première visite médicale et proposer davantage de soutien au patient à risque. Ces modèles de risque peuvent également être utilisés pour souligner la période avec davantage d'abandons en fonction du profil du patient et déclencher une alarme pour les atteindre si nécessaire.



(a) 3 months

(b) 6 months

(c) 12 months

Fig. 5: Courbes de Roc et Lorenz pour la prédiction de la persistance d'un arrêt illégitime n mois après le début de la phase

VI. Étude des transactions en pharmacie

L'analyse de survie et une analyse statistique des transactions en pharmacie montrent que le taux d'abandons illégitimes évolue avec le temps et en fonction d'événements antérieurs survenus dans le parcours de soins du patient. Cela a conduit à une étude plus centrée sur le patient portant sur toutes les transactions effectuées par une patiente : chaque achat dans une pharmacie, chaque hospitalisation, ... Pour chaque transaction en pharmacie d'un médicament utilisé pour traiter le cancer du sein (Tamoxifène, Exemestane, ...) , le modèle prédit si la patiente poursuivra son traitement correctement ou si un arrêt illégitime pourrait avoir lieu. Les mêmes critères utilisés dans l'étude des phases sont utilisés pour étiqueter les transactions en pharmacie. Les données peuvent donc être considérées comme une séquence d'événements (hospitalisations et transactions en pharmacie) avec des horodatages et un ensemble de caractéristiques relatives au patient. Nous pouvons diviser les données en deux catégories : *"statique"* données décrivant le patient (informations géographiques et sociologiques, âge, ...) et données *dynamiques* sur le chemin de soins actuel (derniers achats de médicaments (liés au cancer et généraux), co-pathologies, ...). Ces derniers peuvent être vus comme

une séquence d'événements sur lesquels on sait que les réseaux de neurones récurrents (RNN) fonctionnent bien, en particulier: *Long short-term memory* (LSTM [7]).

Dans cette étude, notre modèle obtient les meilleurs résultats avec l'architecture suivante : 1- un réseau LSTM est appliqué sur les informations "dynamiques" (transactions en pharmacie et hospitalisations); 2- un réseau MLP sur les informations "contextuelles" (détails sur le patient (géographique, support financier, ...)) qui ne sont pas fréquemment mises à jour dans la base de données SNIIRAM; 3- les deux sorties sont concaténées puis classées par une couche entièrement connectée. Nous avons testé des réseaux basés sur ce type d'architecture. Les entrées "dynamiques" sont des séquences de 10 observations et la sortie indique si la transaction en cours peut ne pas être suivie d'une autre (indiquant le risque d'abandon illégal). Comme 10 observations ne sont pas toujours disponibles pour chaque transaction, nous testons également différents pré-traitements de remplissage avec remplissage par des zéros ou duplication de la première observation. Dans cette étude, le remplissage des zéros a obtenu les meilleurs résultats et a été retenu pour notre processus en cours. L'objectif sous-jacent de cette étude est également de mieux comprendre l'influence de différents ensembles de fonctionnalités. Comme il est possible d'analyser l'utilisation d'informations statiques via le réseau MLP, nous nous concentrons ici sur les données dynamiques, car les RNN sont plus difficiles à comprendre. Notre stratégie consiste à former différents réseaux avec plus ou moins de données sur les caractéristiques. Notre base de référence traite uniquement des transactions en pharmacie liées à la cancérologie et des hospitalisations de longue durée. Ensuite, nous ajoutons les co-pathologies et autres transactions en pharmacie. Cependant, une information plus dynamique n'augmente pas les performances, car nous obtenons toujours un score permettant de cibler efficacement un patient dans 82 % des cas. La courbe CAP indique que, dans les premiers 20% de la population enquêtée classés entre le risque le plus élevé et le risque estimé le moins élevé, nos modèles ciblent 66% des arrêts illégitimes de toutes les transactions. C'est trois fois plus efficace que le modèle aléatoire actuel utilisé pour cibler les patients à risque, validant ainsi notre hypothèse selon laquelle les dernières transactions fournissent des informations permettant de détecter un arrêt illégitime.
Les services de santé français disposant d'un nombre limité d'heures pour appeler les patients, notre modèle pourrait doubler leur efficacité. Ces résultats pourraient également être utilisés pour demander aux pharmaciens de fournir davantage d'assistance, le cas échéant, ou pour déclencher un système basé sur SMS permettant de contacter et de motiver un patient à risque élevé. Cependant, les différentes utilisations de données que nous avons essayées n'apportent pas beaucoup d'amélioration à nos résultats. Nous pourrions utiliser des réseaux plus profonds pour gagner quelques points dans les courbes AUC et CAP. Nos premières tentatives en ce sens ont entraîné une forte instabilité dans nos résultats, pouvant venir d'un sur-apprentissage. Cela pourrait s'expliquer par le besoin de plus de données pour fournir des résultats stables à des réseaux plus complexes. Nous prévoyons de tester cette hypothèse, mais cela pourrait prendre un certain temps en raison du processus d'extraction de données SNIIRAM (qui concerne le respect de la vie privée des patients). Nous examinons également d'autres solutions, telles que l'utilisation de réseaux antagonistes génératifs (generative adversarial networks, GAN) pour augmenter la taille des données ou l'apprentissage par transfert à l'aide d'autres dossiers de santé électroniques pour améliorer les informations générales sur les parcours de soins des patients.

VII. Conclusion et discussion

Cet article explique comment utiliser différentes méthodes d'apprentissage automatique pour aider les patients à suivre leur traitement au cours d'une maladie de longue durée. Nous explorons plusieurs approches appliquées aux données de remboursement du système de santé français pour estimer le risque d'abandon illégal de médicaments. Les résultats obtenus avec des modèles simples d'observations indirectes à partir de SNIIRAM (données de remboursement) prouvent la faisabilité de notre processus. Nous validons nos premiers résultats avec les réactions d'oncologues et de chercheurs en médecine.

Nos approches visent également à être cohérentes avec les parcours de soins des patients. En utilisant le formalisme des phases de traitement ou en regardant directement les transactions en pharmacie, nous montrons que nous pourrions informer les prestataires de soins du risque potentiel d'abandon du traitement à des moments précis du traitement. Cela permet de fournir un soutien plus efficace aux moments appropriés tout en évitant le stress résultant de contacts inutiles trop fréquents et en limitant le gaspillage de ressources pour les patients à faible risque.

Les études montrent leurs capacités à prédire la non-persistance des patients. L'une des principales limites est l'intelligibilité de leur processus de décision, qui reste difficile à interpréter avec les "méthodes de la boîte noire". Ce sera un défi, car le personnel médical veut comprendre le chemin des patients. Cependant, nous validons la capacité de l'IA à estimer le risque de non-persistance. Nous étudions actuellement des méthodes pour ouvrir ces boîtes noires afin de donner un aperçu de la décision prise par les modèles.
Nous prévoyons également de développer des réseaux plus complexes qui devraient améliorer notre efficacité. Par exemple, nous pourrions examiner plusieurs patholo-

gies et utiliser une sorte de méthode d'adaptation de domaine pour détecter des modèles pertinents pour la non-persistance. L'autre défi majeur concerne l'étiquetage des données: nous envisageons d'explorer l'étiquetage automatique et la découverte d'anomalies afin de trouver des informations plus précises dans nos données.

References

[1] DR Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, 34(2), 1972.

[2] M Robin DiMatteo. Social support and patient adherence to medical treatment: a meta-analysis. *Health psychology*, 23(2):207, 2004.

[3] M Robin DiMatteo. Variations in patients' adherence to medical recommendations: a quantitative review of 50 years of research. *Medical care*, 42(3):200–209, 2004.

[4] M Robin DiMatteo, Heidi S Lepper, and Thomas W Croghan. Depression is a risk factor for noncompliance with medical treatment: meta-analysis of the effects of anxiety and depression on patient adherence. *Archives of internal medicine*, 160(14): 2101–2107, 2000.

[5] L Fallowfield, L Atkins, S Catt, A Cox, C Coxon, C Langridge, R Morris, and M Price. Patients' preference for administration of endocrine treatments by injection or tablets: results from a study of women with breast cancer. *Annals of Oncology*, 17(2):205–210, 2005.

[6] Jessica M Franklin, William H Shrank, Joyce Lii, Alexis K Krumme, Olga S Matlin, Troyen A Brennan, and Niteesh K Choudhry. Observing versus predicting: Initial patterns of filling predict long-term adherence more accurately than high-dimensional modeling techniques. *Health services research*, 51 (1):220–239, 2016.

[7] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Journal of Neural computation*, 1997.

[8] Edward L Kaplan and Paul Meier. Nonparametric estimation from incomplete observations. *Journal of the American statistical association*, 53(282), 1958.

[9] Linda Krolop, Yon-Dschun Ko, Peter Florian Schwindt, Claudia Schumacher, Rolf Fimmers, and Ulrich Jaehde. Adherence management for patients with cancer taking capecitabine: a prospective two-arm cohort study. *BMJ open*, 3(7):e003139, 2013.

[10] Kem P Krueger, Bruce A Berger, and Bill Felkey. Medication adherence and persistence: a comprehensive review. *Advances in therapy*, 22(4):313–356, 2005.

[11] Wei-Hsuan Lo-Ciganic, Julie M Donohue, Joshua M Thorpe, Subashan Perera, Carolyn T Thorpe, Zachary A Marcum, and Walid F Gellad. Using machine learning to examine medication adherence thresholds and risk of hospitalization. *Medical care*, 53(8):720, 2015.

[12] Devin M Mann, Mark Woodward, Paul Muntner, Louise Falzon, and Ian Kronish. Predictors of nonadherence to statins: a systematic review and meta-analysis. *Annals of Pharmacotherapy*, 44(9): 1410–1421, 2010.

[13] Maryan Morel, Emmanuel Bacry, Stéphane Gaïffas, Agathe Guilloux, and Fanny Leroy. Convsccs: convolutional self-controlled case series model for lagged adverse event detection. *arXiv preprint arXiv:1712.08243*, 2017.

[14] A Neumann, A Weill, P Ricordeau, JP Fagot, F Alla, and H Allemand. Pioglitazone and risk of bladder cancer among diabetic patients in france: a population-based cohort study. *Diabetologia*, 55(7): 1953–1962, 2012.

[15] VJ O'neill and CJ Twelves. Oral cancer treatment: developments in chemotherapy and beyond. *British journal of cancer*, 87(9):933, 2002.

[16] Lars Osterberg and Terrence Blaschke. Adherence to medication. n engl j med. 353:487–97, 09 2005.

[17] Sandra L Spoelstra, Barbara A Given, Charles W Given, Marcia Grant, Alla Sikorskii, Mei You, and Veronica Decker. An intervention to improve adherence and management of symptoms for patients prescribed oral chemotherapy agents: an exploratory study. *Cancer nursing*, 36(1):18–28, 2013.

[18] P Tuppin, L De Roquefeuil, A Weill, P Ricordeau, and Y Merlière. French national health insurance information system and the permanent beneficiaries sample. *Revue d'epidemiologie et de sante publique*, 58(4):286–290, 2010.