Bel-Games: A Formal Theory of Games of Incomplete Information Based on Belief Functions in the Coq Proof Assistant

Pierre Pomeret-Coquot, Hélène Fargier, Érik Martin-Dorel

To cite this version:
Pierre Pomeret-Coquot, Hélène Fargier, Érik Martin-Dorel. Bel-Games: A Formal Theory of Games of Incomplete Information Based on Belief Functions in the Coq Proof Assistant. 2022. hal-03782650v1

HAL Id: hal-03782650
https://ut3-toulouseinp.hal.science/hal-03782650v1
Preprint submitted on 21 Sep 2022 (v1), last revised 27 Feb 2023 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License
Bel-Games: A Formal Theory of Games of Incomplete Information Based on Belief Functions in the Coq Proof Assistant

Pierre Pomeret-Coquot
IRIT, Université de Toulouse III - Paul Sabatier, France

Hélène Fargier
IRIT, CNRS, Toulouse, France

Érik Martin-Dorel
IRIT, Université de Toulouse III - Paul Sabatier, France

Abstract

Decision theory and game theory are both interdisciplinary domains that focus on modeling and studying decision-making processes. On the one hand, decision theory aims to account for the possible behaviors of an agent with respect to an uncertain situation. It thus provides several frameworks to describe the decision-making processes in this context, including that of belief functions. On the other hand, game theory focuses on multi-agent decisions, typically with probabilistic uncertainty (if any), hence the so-called class of Bayesian games. In this paper, we extend Bayesian games to the theory of belief functions. We obtain a more expressive class of games we refer to as Bel games; it makes it possible to better capture human behaviors with respect to lack of information. Next, we prove an extended version of the so-called Howson–Rosenthal’s theorem, showing that Bel games can be turned into games of complete information, i.e., without any uncertainty. Doing so, we embed this class of games into classical game theory and thereby enable the use of existing algorithms. Using the Coq proof assistant, we formalize a theory of belief functions, and formally verify three different proofs of Howson–Rosenthal’s theorem.

2012 ACM Subject Classification Theory of computation → Proof theory; Theory of computation → Logic and verification; Theory of computation → Algorithmic game theory; Theory of computation → Solution concepts in game theory; Theory of computation → Representations of games and their complexity

Keywords and phrases Game of Incomplete Information, Belief Function Theory, Proof Assistant, MathComp/SSReflect

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.1

Related Version Pen-and-paper proof article: HAL [30]

Source code: GitHub repository

Funding Pierre Pomeret-Coquot: ANITI, funded by the French ”Investing for the Future – PIA3” program under the Grant agreement n°ANR-19-P13A-0004.

1 Introduction

The positioning of our work lies at the intersection of decision-making under uncertainty, game theory, and interactive theorem proving. While decision theory typically deals with single-agent decision making in an uncertain context, game theory gathers several frameworks that all involve several agents, and can model situations in many application areas such as economics, politics, logics, artificial intelligence, or biology... For example, Bayesian games
constitute one such framework in the scope of games of incomplete information (denoted “IGames” in the following), which has been well-studied by the community [14, 26].

In [10, 30], we introduce the framework of Bel games (which encompasses that of Bayesian games) based on belief functions, and generalize Howson–Rosenthal’s 2-player theorem to this framework and to any number of players. More specifically, we provide three algorithmic transforms that turn a Bel-game into a game of complete information (denoted “CGame” in the following), along with a pen-and-paper proof of correctness for these three transforms. In this paper, we formally define Bel-games using the Coq proof assistant, and present a formally-verified implementation of these transforms. The objective of this formalization effort is twofold: provide strong guarantees on the correctness of the results as well as make explicit necessary conditions and improve the sequence of proof steps; and setup a generic, formal library that allows one to formally reason about IGames. To the best of our knowledge, it is the first attempt to model IGames in a formal setting. Moreover, we focus on theoretical foundations that go beyond the classical setting of Bayesian games, relying on Belief functions theory, while addressing Nash equilibria with mixed strategies.

Related works Several formalization efforts have been carried out in game theory since 2006, each focusing on a somewhat different fragment: Vestergaard [38] then Le Roux [31], formalizing Kuhn’s existence of a Nash equilibrium in finite games in extensive form, using Coq; Lescanne et al. [22], studying rationality of infinite games in extensive form, using Coq; Martin-Dorel et al. [24], studying the probability of existence of winning strategies in Boolean finite games, using Coq; Bagnall et al. [2], formalizing well-known results of algorithmic game theory, using Coq; Dittmann [5], proving the positional determinacy of parity games, using Isabelle/HOL; Le Roux et al. [20], proving that a determinacy assumption implies the existence of Nash equilibrium in 2-player games, using Coq and Isabelle/HOL; this result being combined with that of Dittmann, using Isabelle/HOL [21].

Paper outline We start in Section 2 by reviewing a typical example of reasoning under uncertainty, where the usual framework of Bayesian games fails. In Section 3, we recall background knowledge about belief functions and formalize it in the Coq proof assistant. In Section 4, we formalize some standard notions of game theory such as hypergraphical games. Then in Section 5, we formalize Bel-games (which generalize Bayesian games), as well as the notion of Nash equilibria for these classes of IGames. In Section 6, we formally prove a generalized Howson–Rosenthal’s theorem (for n-player Bel games in lieu of 2-player Bayesian games) for three different transforms that enjoy slightly different properties. Finally, we draw some conclusions and perspectives for future work in Section 7.

2 Motivating Example: the Murder of Mr. Jones

To illustrate and motivate our work, we use the following example inspired by the murder of Mr. Jones [37], where the suspects are Peter, Paul, and Mary.

Example 1 (Peter, Quentin, and Rose). Two agents, named Agent 1 and Agent 2, are independently looking for a business association, with either Peter (P), Quentin (Q), or Rose (R). The point is that a crime has been committed, for which these three people only are suspected. Furthermore, a poor-quality surveillance video allow to estimate that there is a 50% chance that the culprit is a man (P or Q), and a 50% chance that it is a woman (R). As to the interest of the associations, making the deal with an innocent people leads to a payoff of $6k (to be shared between the people making the deal), while associating with the culprit produces no payoff ($0k). Moreover, Agent 1 is investigating about P and will know whether he is guilty before making the decision. Similarly, Agent 2 will know whether R is guilty.
3 Formalization of Belief Functions for Individual Decision Making

3.1 Knowledge Representation

The theory of belief functions is a powerful toolset from decision theory and statistics, which generalizes probability theory. It encompasses two distinct approaches for reasoning under uncertainty: Dempster-Shafer theory of evidence (DS) [4, 35] and upper-lower probability theory (ULP) [4, 39].

▶ Definition 2 (Basic probability assignment). A set-function $m : 2^Ω → [0,1]$ is a basic probability assignment (bpa, a.k.a. mass function) iff $m(∅) = 0$, $\sum_{A ∈ Ω} m(A) = 1$ and $∀ A ⊆ ω, m(A) ≥ 0$.

▶ Definition 3 (Belief function and plausibility measure). A belief function $Bel : 2^Ω → [0,1]$ is a monotonic measure which is subadditive ($∀ A, B$ distinct, $Bel(A ∪ B) ≥ Bel(A) + Bel(B)$) and verifies $Bel(∅) = 0$ and $Bel(Ω) = 1$. The dual plausibility measure $Pl$ is defined by $Pl(A) = 1 - Bel(\complement A)$, and is superadditive.

▶ Definition 4 (bpa-characterization of a belief function and a plausibility measure). Both $Bel$ and $Pl$ can be encoded by a single bpa $m : 2^Ω → [0,1]$ such that $Bel(A) = \sum_{B ⊆ A} m(B)$ and $Pl(A) = \sum_{B, B \cap A \neq ∅} m(B)$. Conversely, any bpa $m$ defines a pair of dual belief and plausibility measures.

We formalize these first definitions in Coq [3], relying on the MathComp library [23] that provides a comprehensive formalization of finite sets, functions with finite support (endowed with decidable equality) as well as big operators:

1. MathComp/SSReflect notations: finite sets over $(X : finType)$, common notations are: $A ∧ B = A ∩ B$, $A ∨ B = A ∪ B$, $¬A = A'$ and $set∅ = ∅$. Finite support functions from $(X : finType)$ to $(Y : Type)$ are typed $(ffun X → Y)$; and for any $(Z : X → Type)$, dependent funs, which map any $(x : X)$ to an element of $(Z x)$ are typed $(ffun forall x : X, Z x)$.

It appears the Bayesian approach is not relevant here: it does not capture the intended knowledge revision. Indeed, if Agent 1 learns that $P$ is innocent, the intended behavior of Agent 1 is to keep 50% chances per sex, so the probability of guilt should become 1/2 for $Q$ and 1/2 for $R$. However, in a purely Bayesian view, equiprobability would be applied and the prior probability of guilt would be 1/4 to $P$ and 1/4 to $Q$. Then, after conditioning, Agent 1 would get a probability of 1/3 to $Q$ and 2/3 to $R$. 
Bel-Games: A Formal Theory of Games of Incomplete Information

Definition focal_element (m : bpa) : pred (set W) := fun A ⇒ m A > 0.

Definition focalset (m : bpa) : {set (set W)} := {set A | focal_element m A}.

Of course, Bel and Pl can be directly expressed as sums over the focal set:

Lemma Bel_focalsetE m A : Bel m A = \sum_{B in focalset m | B \subset A} m B.

Lemma Pl_focalsetE m A : Pl m A = \sum_{B in focalset m | B \subset A} m B.

Belief functions can be interpreted from the viewpoint of two different uncertainty theories:

**DS theory** As in our example, knowledge is build from evidences about the actual state of the world ω*. There are several “parts of beliefs” which all describe a piece of information, they are encoded in the bpa m: the mass m(B) is the degree of confidence on the evidence that support ω* ∈ B, but nothing more (see Table 1). Under this interpretation, Bel(B) (resp. Pl(B)) estimates to what extent the event B is implied by (resp. is compatible with) the knowledge encoded in m — m is seen as a generalized set [7].

**ULP theory** Our example can also be understood in the perspective of the ULP theory, where Bel and Pl delimit a family of probability measures \( \mathcal{F} = \{\text{Pr} | \forall A, \text{Bel}(A) \leq \text{Pr}(A) \leq \text{Pl}(A)\} \), which gathers the probability measures that may describe the partial knowledge (also see Table 1). In this interpretation, Bel(B) (resp. Pl(B)) is the minimal (resp. maximal) value of Pr(B) in the family.

<table>
<thead>
<tr>
<th>A ⊆ Ω</th>
<th>( \emptyset )</th>
<th>( {P} )</th>
<th>( {Q} )</th>
<th>( {R} )</th>
<th>( {P,Q} )</th>
<th>( {P,R} )</th>
<th>( {Q,R} )</th>
<th>( {P,Q,R} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>m(A)</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Bel(A)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Pl(A)</td>
<td>0</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Pr_A</td>
<td>0</td>
<td>x</td>
<td>0.5 - x</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 1 Prior knowledge from Example 1 — in the DS theory, \( m \) directly describes the knowledge, in the ULP theory, \( m \) describes a family of probability measures \( (\text{Pr}_x)_{x\in[0,0.5]} \).

Next, we recall a “complexity” definition, which is used to characterize probability measures:

**Definition 6 (k-additivity).** A belief function is said to be k-additive iff the cardinality of its focal elements is at most k.

**Notation** \( "k \text{-additive " } m\) := (k_additivity m <= k) (**technicalities omitted**).

**Proposition 7** (Probability). If a belief function is 1-additive, i.e. if all focal elements are singletons, then Bel = Pl is a probability measure whose distribution is \( x \mapsto m(\{x\}) \).

Structure proba := { proba_val :> bpa ; proba_ax : 1.-additive proba_val }

Lemma PrE (p : proba) : Bel p =1 Pl p.

Definition dist (p : proba) := fun w ⇒ p [set w].

Lemma Pr_distE (p : proba) A : Bel p A = \sum_{w in A} dist p w.

Belief functions thus encompass probabilities. Though the following example makes it clearer how the use of a belief-function model can be “weaker”, and thereby, more expressive.

**Example 8** (Knowledge representation). In our running example, the prior knowledge is just \( m(\{P,Q\}) = m(\{R\}) = 1/2 \). It is expressible “as is” using belief functions, but expressing it with a probability distribution requires to make the “equi-probability assumption” (Figure 1).
3.2 Conditioning in the Belief Function Theory

Several conditioning rules for belief functions have been proposed, depending on the DS or ULP interpretation, and more generally on the kind of knowledge at work. All those conditionings generalize the Bayesian conditioning, however starting from the same “prior” bpa, they yield distinct “posteriors”: they indeed capture distinct operations such as knowledge revision, updating, focusing [8, 6, 13].

To formalize conditional events \((· | C)\) — read “given \(C\)” —, a precondition is necessary: on the technical side, it avoids division-by-zero, and on the semantics side, it means one cannot learn that an impossible event holds. Since it depends on the conditioning rule, we abstract this precondition in a predicate \(\text{revisable}\). Furthermore, every conditioning rule has to verify the \(\text{conditioning\_axiom}\) \(\text{Bel}(C | C^c) = 0\), that is, if one learns that \(C\) holds, then one also learns that no evidence for \(C^c\) can hold. Formally speaking, we define a \(\text{conditioning}\) structure which encapsulates the precondition, the conditioning algorithm itself (which turns a \(\text{revisable}\) prior in its posterior “given \(C\)” ) and a proof of the \(\text{conditioning\_axiom}\):

\[
\text{Definition}\hspace{1cm} \text{conditioning\_axiom} \hspace{0.5cm} \text{(revisable : bpa → pred \{set }W\})
\]

\[
(\text{cond : } \forall m \ C, \text{revisable m C → bpa})
\]

\[
\text{Structure}\hspace{1cm} \text{conditioning} \hspace{0.5cm} \text{:=\{} \text{revisable : bpa → pred \{set }W\}) ;
\]

\[
\text{cond\_val} \hspace{0.5cm} m \ C : \text{revisable m C → bpa} ;
\]

\[
\text{cond\_ax : @conditioning\_axiom revisable cond\_val } \text{\}}.
\]

The most common conditioning is the so-called Dempster’s conditioning [4], which captures knowledge revision (i.e. fact learning). In the DS framework, it is understood as a transfer of parts of beliefs: \(m(B)\) is transferred to \(B \cap C\) if it is not empty, or discarded otherwise (then the posterior had to be renormalized). That is, the evidence now concerns \(B\cap C\), the only possible worlds “given that \(C\) holds”. In the ULP framework, it is understood as a max-likelihood conditioning: the posterior probability family delimited by \(\text{Bel}(· | C)\) and \(\text{Pl}(· | C)\) is the conditioning of those prior probability measures which assign the maximal probability to the event \(C\), that we now know for sure.

\[
\text{Definition}\hspace{1cm} \text{Dempster\_revisable m C := Pl m C != 0}.
\]

\[
\text{Definition}\hspace{1cm} \text{Dempster\_f (m : bpa) (C : \{set }W\})
\]

\[
:= \{ \text{iffun A : \{set }W\} \Rightarrow \text{if A == set0 then 0}
\]

\[
\text{else } \sum(B : \{set }W\} \land (B \in \text{focalset m}) \land (B \cap C == A)) m B \land \text{Pl m C} \}.
\]

\[
\text{Lemma}\hspace{1cm} \text{Dempster\_bpa\_ax m C (HC : Dempster\_revisable m C) : bpa\_axiom (Dempster\_f m C).}
\]

\[
\text{Definition}\hspace{1cm} \text{Dempster\_cond m C (HC : Dempster\_revisable m C) : bpa}
\]

\[
:= \{ \text{bpa\_val := Dempster\_fun m C} ; \text{bpa\_ax := Dempster\_bpa\_ax HC } \}.
\]

\[
\text{Lemma}\hspace{1cm} \text{Dempster\_cond\_axiom : @conditioning\_axiom Dempster\_revisable Dempster\_cond.}
\]

\[
\text{Definition}\hspace{1cm} \text{Dempster\_conditioning : conditioning}
\]

\[
:= \{ \text{cond\_val := Dempster\_cond} ; \text{cond\_ax := Dempster\_cond\_axiom } \}.
\]
Example 9 (Knowledge revision). Dempster’s conditioning is the conditioning approach at work in our example (see [6] for details). Say $\omega^*$ denote the actual state of the world, which agents don’t know. Now, consider the case of Agent 1 learning that $P$ is not the murderer: Agent 1 learns that $\omega^* \notin \{P\}$, i.e., $\omega^* \in \{Q,R\}$. In this case, the evidence concerning men now only concerns $Q$, so from the viewpoint of Agent 1, the knowledge becomes $m(\{Q\}) = m(\{R\}) = 0.5$, thus excluding $P$ without changing the given odds depending on the sex of the murderer (Figure 2, center, the mass assigned to $\{P,Q\}$ was transferred to $\{P,Q\} \cap \{Q,R\} = \{Q\}$). On the contrary, now consider that Agent 2 learns that $R$ is not the murderer: Agent 2 learns $\omega^* \notin \{R\}$, i.e., $\omega^* \in \{P,Q\}$. In this case, the evidence concerning women has to be discarded, so the knowledge becomes $m(\{P,Q\}) = 1$, thus excluding $R$, and now expressing nothing more than the murderer is a man (Figure 2, right, the mass assigned to $\{R\}$ was discarded since $\{R\} \cap \{P,Q\} = \emptyset$).

Two other rules have been proposed and called strong and weak conditioning [29]; the former, also known as geometrical conditioning [36], is another “revision” rule; the latter is seldom used since it yields strange results (as for example it may be $\text{Bel}(C \mid C) < 1$).

Another well-known rule is Fagin-Halpern conditioning ([9, 16], already in [4]), which is a “focusing” rule suited for the ULP interpretation. The FH-conditioning of a bpa (which delimits a family of prior probabilities) leads to the bpa delimiting the family of posterior probabilities. At the time we are writing this paper, we did not focused on formalizing this conditioning, even if it could be interesting as a future work.

3.3 Informed Decision

A classical way to model decision processes is to use utility functions [32, 25]. Say there are several possible choices (a.k.a. actions); let $A$ denote the set of all these actions. Depending on the actual state of the world $\omega^* \in \Omega$, an action $a \in A$ may lead to different outcomes, so
that the decision maker may be satisfied or not. Let \( u : A \times \Omega \to \mathbb{R} \) be an utility function, assigning to any action \( a \in A \) and to any state of the world \( \omega \in \Omega \), the degree of satisfaction of the decision maker. Their preferences are captured by “scoring function”, which aggregates all \( \omega \)’s, considering of course the knowledge expressed over \( \Omega \) by a bpa – the action which has the best score is the best action to be chosen (from the decision maker’s point of view).

In a probabilistic setting, the classical “scoring function” is the expected utility.

A very common scoring function for belief functions is the Choquet discrete integral, which leads to the minimal expected utility, i.e., the minimal degree of satisfaction the decision maker can expect [33, 12]. Another line of thought is to consider that belief functions are particular cases of capacity measures, and to compute the global merit based on its Choquet value [11, 34]. It is easily defined using the bpa by:

\[
CEU(a) = \sum_{B \in S_m} m(B) \times \min_{\omega \in B} u(a, \omega) = \sum_{B \in S_m} m(B) \times f^{CEU}_{u(a, \cdot)}(B)
\]

This expression is a weighted sum indexed by focal elements, and it is correct to use min since a focal element cannot be empty. The function \( f^{CEU}_u \) is formalized as:

\[
\text{Definition ceu\_fun (u : W -> R) : \{set W\} -> R} \quad \text{:= fun B :> match minS u B with Some r :> r | None :> 0 end.}
\]

Another rule, axiomatized by Jaffray [17, 18], is a kind of Hurwicz criterion (i.e., a linear combination over the minimal and maximal utility reached for each focal element) – the coefficient \( \alpha \) may vary, depending on the minimal and maximal utility values over \( B \).

\[
JEU^\alpha(a) = \sum_{B \in S_m} m(B) \times (\alpha \times \min_{\omega \in B} u(a, \omega) + (1 - \alpha) \times \max_{\omega \in B} u(a, \omega)) = \sum_{B \in S_m} m(B) \times f^{JEU^\alpha}_{u(a, \cdot)}(B)
\]

\[
\text{Definition jeu\_fun (alpha : R -> R -> R) (u : W -> R) : \{set W\} -> R} \quad \text{:= fun B :> match minS u B, maxS u B with Some rmin, Some rmax :> let a := alpha rmin rmax in a * rmin + (1-a) * rmax | _, _ :> 0 end.}
\]

Last, in the Transferable Belief Model [37], the decision rule is made by recovering a “pignistic” probability distribution BetP, at the very moment where the decision is made (i.e., the equiprobability assumption is made, but after conditionings, if any). It is also expressible as a weighted sum over focal elements:

\[
TBEU(a) = \sum_{B \in S_m} m(B) \times \frac{\sum_{\omega \in B} u(a, \omega)}{|B|} = \sum_{B \in S_m} m(B) \times f^{TBEU}_{u(a, \cdot)}(B)
\]

\[
\text{Definition tbeu\_fun (u : W -> R) := fun B :> \sum_{(w in B) u w / |B|:R.}
\]

We express those three scoring functions as weighted sums of some “set-utility function” \( f \). It makes it possible to abstract them by \( XEU(a) = \sum_{B \in S_m} m(B) \times f^{XEU}_{u(a, \cdot)}(B) \).

\[
\text{Definition XEU (m : bpa) (f : \{set W\} -> R) := \sum_{B in focalset m} m B * f B.}
\]

For any two extentionally equal function \( u \) and \( v \), \( f^{XEU}_u \) and \( f^{XEU}_v \) also are extentionally equal. For the sake of the proof, we endow \( ceu\_fun \) (resp. \( jeu\_fun, tbeu\_fun \)) with a Boolean predicate asserting this property, into the structure \( CEU \) (resp. \( JEU, TBEU \)).

It can be noted that \( CEU, JEU \) and \( TBEU \) generalize the expected utility criterion: using a 1-additive bpa (i.e., a probability measure), they all lead to the expected utility value:
4 Formalization of Several Classes of Games of Complete Information

Game theory aims to model and study multi-agent decision making [25, 26]. In this paper, we focus on strategic (or simultaneous) games: agents make their own decision without knowing others’ choices, but where the outcome of the game depends on every agent’s choices.

4.1 Games of Complete Information

Let us first consider a situation where there is no uncertainty: each agent knows all the other agents involved, their possible actions and the utility they have depending on the chosen actions. One such CGame is defined by a tuple $G = (I, (A_i, u_i))_{i \in I}$ where $I$ is a finite set of agents, then for each agent $i$, $A_i$ is the set of their actions and $u_i : A \to \mathbb{R}$ is a utility function, assigning an utility value to each “profile” $a = (a_1, \ldots, a_n) \in A = A_1 \times \cdots \times A_n$. Agent $i$ prefers the profile $a$ to $a'$ iff $u_i(a) > u_i(a')$. We formalize such “profiles-for-CGames” ($a \in \prod_{i \in I} A_i$) using MathComp’s dependently-typed finite support functions; a CGame is then simply encoded by its utility functions:

$$\text{Variables} \quad (R : \text{numFieldType}) \quad (I : \text{finType}) \quad (A : \forall i : I, \text{eqType}).$$

$$\text{Definition} \quad \text{cprofile} \quad (I : \text{finType}) \quad (A : I \to \text{finType}) := \{\text{dffun} \quad \forall i : I, A_i\}.$$

$$\text{Definition} \quad \text{cgame} := \text{cprofile} A \to I \to R.$$

One of the most prominent solution concept in game theory is that of Nash equilibria [27], that is, profiles such that no agent has any incentive to move to another action:

$$\text{Definition} \quad \text{change_strategy} \quad I \quad A \quad (p : \text{cprofile} A) \quad (i : I) \quad (ai : A_i) := \text{cprofile} A \quad (j : I) \quad \text{match boolP} \quad (i == j) \quad \text{with}$$

$$\quad | \text{AltTrue} \quad h \Rightarrow \text{eq_rect} \quad A \quad ai \_ \quad (\text{eqP} \quad h)$$

$$\quad | \text{AltFalse} \quad \_ \Rightarrow p \quad j \quad \text{end}].$$

$$\text{Definition} \quad \text{Nash_equilibrium} \quad (G : \text{cgame}) \quad (a : \text{cprofile} A) := \text{bool} := (\forall i : I, [\forall ai : A_i, -- (G \quad a \quad i < G \quad (\text{change_strategy} \quad a \quad ai) \quad i)]).$$

$$\text{Definition} \quad \text{Nash_equilibrium_prop} \quad (G : \text{cgame}) \quad (a : \text{cprofile} A) := \text{Prop} := (\forall i : I, [\forall ai : A_i, -- (G \quad a \quad i < G \quad (\text{change_strategy} \quad a \quad ai) \quad i)].$$

$$\text{Lemma} \quad \text{Nash_equilibriumP} \quad G \quad a := \text{reflect} \quad (\text{Nash_equilibrium_prop} \quad G \quad a) \quad (\text{Nash_equilibrium} \quad G \quad a).$$

Example 10. Consider Example 1, but say it is known $P$ is the murderer. It is then captured by the CGame $G = (I, (A_i, u_i))_{i \in I}$ where $I = \{1, 2\}$ is the set of agents, $A_i = \{P_i, Q_i, R_i\}$ is the set of actions of Agent $i$ (choosing $P$, $Q$ or $R$) and $u_i$’s are given in Table 2. In this game, both $(Q_1, R_2)$ and $(R_1, Q_2)$ are Nash equilibria: no agent has any incentive to change.

<table>
<thead>
<tr>
<th></th>
<th>$P_2$</th>
<th>$Q_2$</th>
<th>$R_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_1$</td>
<td>(0,0)</td>
<td>(0,3)</td>
<td>(0,3)</td>
</tr>
<tr>
<td>$Q_1$</td>
<td>(3,0)</td>
<td>(2,2)</td>
<td>(3,3)</td>
</tr>
<tr>
<td>$R_1$</td>
<td>(3,0)</td>
<td>(3,3)</td>
<td>(2,2)</td>
</tr>
</tbody>
</table>

Table 2 Utility functions of Example 10 (it is known that $P$ is the murderer).
When there is some variability regarding action choices (e.g., for repeated games), it is meaningful to use mixed strategies. A mixed strategy $\rho_i$ of Agent $i$ is a probability over Agent $i$’s actions, and a mixed strategy profile $\rho = (\rho_1, \ldots, \rho_n)$ is a vector of mixed strategies:

$$
\text{Definition } \text{mixed_cprofile} := \text{cprofile (fun i = > proba_eqType R (A i)).}
$$

A mixed strategy profile $\rho$ defines a probability over the set of pure strategy profiles, namely $p_\rho(a) = \prod_{i \in I} \rho_i(a_i)$. We then embed it in a $\text{proba}$ structure:

$$
\text{Definition } \text{mk_prod_proba (p : } \forall i : X, \text{proba R (A i)) : } \{\text{ffun cprofile A -> R}}
$$

$$
:= \{\text{ffun a : cprofile A -> prod_i dist (p i) (a i)}\}. 
$$

$$
\text{Definition } \text{prod_proba (p : } \forall i : I, \text{proba R (A i)) (i0 : I) : proba R (cprofile A)).}
$$

Last, the utility of a mixed strategy profile is the expected utility w.r.t. the probability over pure strategy profiles, and the notion of Nash equilibria extends straightforwardly:

$$
\text{Definition } \text{ms_utility (G : cgame R A) (mp : mixed_cprofile) (i : I) : R}
$$

$$
:= \sum_{(p : cprofile A)} (\text{dist (prod_proba mp witnessI mp) p}) \times (G p i).
$$

$$
\text{Definition } \text{ms_Nash_equilibrium (G : cgame R A) (mp : mixed_cprofile) : Prop}
$$

$$
:= \forall i : I, \forall si : \text{proba R (A i)}, \lnot \text{ms_utility G mp i < ms_utility G (change_strategy mp si) i}.
$$

We can also use the “mixed extension” of a game (i.e., a game where agents’ actions are mixed strategies) and thereby recover previous definitions [19]:

$$
\text{Definition } \text{mixed_cgame (G : cgame R A) : cgame R (fun i = > proba_eqType R (A i))}
$$

$$
:= \text{fun mp i = > ms_utility G mp i}.
$$

$$
\text{Lemma } \text{mixed_cgameE G mp i : ms_utility G mp i = (mixed_cgame G) mp i}.
$$

$$
\text{Lemma } \text{ms_NashE (G : cgame R A) (mp : mixed_cprofile) :}
$$

$$
\text{ms_Nash_equilibrium G mp <-> Nash_equilibrium_prop (mixed_cgame G) mp}.
$$

### 4.2 Hypergraphical Games

Some CGames can be expressed succinctly as hypergraphical games [28, 40], where the utility is not defined globally, but locally, split in several local games. It produces an hypergraph, where vertices denotes agents and hyperedges denote local games. Formally, a hypergraphical game is defined by a tuple $G = (I, E, (A_i)_{i \in I}, (u_e)_{e \in E, i \in e})$, where $I$ is the set of agents, $E \subseteq 2^I$ is the set of local games (in any local game $e = \{i, j, k, \ldots\}$, agents $i, j, k, \ldots$ are playing), $A_i$ is the set of actions of agent $i$ and $u_e^i : A_e \rightarrow R$ is the utility function of agent $i$ in the local game $e$ ($A_e = \prod_{i \in e} A_i$ is the set of local profiles corresponding to $e$’s players). A hypergraphical game with 2-player local games is called a polymatrix.

In our formalization, local games are indexed by the finite type $\text{localgame} : \text{finType}$; players playing a local game $(lg : \text{localgame})$ are those who verify the boolean predicate $(\text{plays_in lg})$. That is, $\text{plays_in}$ denotes a family of sets of agents.

$$
\text{Variables } (R : \text{numFieldType}) (I : \text{finType}) (A : \forall i : I, \text{finType}).
$$

$$
\text{Variables } (\text{localgame : finType}) (\text{plays_in : localgame -> pred I}).
$$

For any local game, the type of local profiles is defined by:

$$
\text{Definition } \text{localprof (A : I -> eqType) (lg : localgame) :}
$$

$$
:= \{\text{ffun } \forall s : \{i : I \mid \text{plays_in lg i}\}, A (\text{val s})\}.
$$

In hypergraphical games, each agent chooses one action, and plays it in every local game they are involved in. The global utility of an agent is the sum of the locally obtained utilities: $u_i(a) = \sum_{e \in E} u_i^a(a_c)$, where $a_c \in A_e$ is the restriction of $a$ to indices of $e$. 

CVIT 2016
4.3 Games of Incomplete Information

Harsanyi had proposed \cite{14} a model for decision-making situations where some agents have some uncertainty about other agents, their actions, their utility functions, and more generally about any parameter some agents know and some don’t. To model such situations, the partially known parameters are expressed by “types”: each agent \(i\) has a “type” \(\theta_i\) which encodes the parameters’ values. Agent \(i\) knows (or learns) \(\theta_i\), but the other agents may ignore it. Harsanyi defined the model of IGames in a probabilistic setting, hence the so-called Bayesian games. In this setting, a probability measure expresses the knowledge on “type” configurations (it is the frame of discernment). IGames already had been extended to a possibilistic setting by Ben Amor et al. \cite{1}, and we extend it to the framework of belief functions (so it generalizes Bayesian games) in the next section.

Since agents know their own type before choosing their action, a pure strategy of Agent \(i\) becomes a function \(\sigma_i : \Theta_i \rightarrow A_i\) which assigns the chosen action to Agent \(i\)’s “type” – Agent \(i\), knowing that \(\theta_i\), will play \(\sigma_i(\theta_i) \in A_i\). A strategy profile \(\sigma = (\sigma_1, \ldots, \sigma_n)\) is a vector of such functions, telling which action every agent will play depending on their actual “type”:

\[
\text{Definition} \quad \text{iprofile } I (T : I \rightarrow \text{finType}) (A : I \rightarrow \text{eqType}) := \text{cprofile} \left(\text{fun } i \mapsto [\text{eqType of } \{\text{ffun } T i \rightarrow A i\}]\right).
\]

If the actual type configuration is \(\theta = (\theta_1, \ldots, \theta_n)\), then for any strategy profile \(\sigma\) we denote \(\sigma^\theta = (\sigma_1(\theta_1), \ldots, \sigma_n(\theta_n))\):

\[
\text{Definition} \quad \text{proj_iprofile } I (T : I \rightarrow \text{finType}) A (p : \text{iprofile } T A) := \text{fun } \theta i \mapsto [\text{ffun } i \mapsto p i (\theta i)].
\]

In the sequel, we will also need to “flatten”\(^2\) an \(\text{iprofile } T A\) that assigns to any \(i \in I\) a strategy \(\sigma_i \in (\Theta_i \rightarrow A_i)\), to get a \(\text{cprofile} \left(\text{fun } i \_t i : \{i : I \& T i \Rightarrow A (\text{val } i)\}\right)\), assigning to any dependent pair \((i, \theta_i)\) the strategy \(\sigma_i(\theta_i) \in A_i\):

\[
\text{Definition} \quad \text{iprofile_flatten } I (T : I \rightarrow \text{finType}) A (p : \text{iprofile } T A) := [\text{ffun } i \_t i \Rightarrow p (\text{proj1 } i \_t i) (\text{proj2 } i \_t i)].
\]

In Bayesian games, the utility of Agent \(i\) having the “type” \(\theta_i\) is given by the expected utility w.r.t. the conditioned probability distribution “given \(\theta_i\”. If one considers mixed strategies, then the two probability distributions (over type configurations and pure strategy profiles) are merged, so the utility becomes the expectation over the joint distribution.

5 Bel-Games

Bel games are IGames where the knowledge is expressed by a belief function – so it generalizes Bayesian games. Mathematically, a Bel game is defined by a tuple \(G = (I, (A_i, \Theta_i, u_i)_{i \in I}, m)\):

\(I\) is the finite set of agents;

\(^2\) For example, in an IGame with two players and two types per agents, \(\sigma = (\sigma_1, \sigma_2)\) is flattened to \((\sigma_1(\theta_1), \sigma_1(\theta_1'), \sigma_2(\theta_2), \sigma_2(\theta_2'))\)
\( \Theta \) is the finite set of “types” of Agent \( i \);
\( u_i : A \times \Theta \to \mathbb{R} \) is the utility function of Agent \( i \), it depends on the joint action
\((a_1, \ldots, a_n) \in A \) and on the type configuration; \((\theta_1, \ldots, \theta_n) \in \Theta \)
\( m : 2^\Theta \to [0, 1] \) is a bpa which describes the prior knowledge on type configurations.

The utility of Agent \( i \) having the “type” \( \theta_i \) is the XEU value w.r.t. the posterior knowledge
“given \( \theta_i \)” (i.e., given the event \( \{ \theta' \mid \theta_i' = \theta_i \} \subseteq 2^\Theta \)). That is, the utility depends on both the
conditioning and the decision rule. For instance, it may be:
\[
\text{CEU}_{i, \theta_i}(\sigma) = \sum_{B \subseteq 2^\Theta} m(B \mid \theta_i) \times \min_{\theta' \in B} u(\sigma(\theta'), \theta').
\]

\textbf{Example 11 (Bel game).} We now are able to express Example 1 by a Bel game \( G = (I, (A_i, \Theta_i, u_i))_{i \in I}, m \). The set of agents is \( I = \{1, 2\} \), their action sets are \( A_i = \{P_i, Q_i, R_i\} \).
Agent 1 will learn either that \( P \) is the murderer \((\omega^* \in \{P\})\) or that he is not \((\omega^* \in \{Q, R\})\):
Agent 1’s “type” set is \( \Theta_1 = \{P, \bar{P}\} \). Similarly, Agent 2 will learn either that \( R \) is the murderer
(\( \omega^* \in \{R\} \)) or that she is not \((\omega^* \in \{P, Q\})\), so \( \Theta_2 = \{\bar{R}, \bar{R}\} \). The knowledge is expressed over
\( \Theta = \Theta_1 \times \Theta_2 \); since \((\bar{P}, \bar{R}) \equiv P, (\bar{P}, \bar{R}) \equiv Q, (\bar{P}, \bar{R}) \equiv R \) and \((P, \bar{R}) \) is impossible, the knowledge
is \( m((\bar{P}, \bar{R}), (\bar{P}, \bar{R}))\) = \( m((\bar{P}, \bar{R}))\) = 0.5. Finally, utility functions are given in Table 3.

<table>
<thead>
<tr>
<th>( P_2 )</th>
<th>( Q_2 )</th>
<th>( R_2 )</th>
<th>( P_1 )</th>
<th>( Q_1 )</th>
<th>( R_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0, 0)</td>
<td>(0, 3)</td>
<td>(0, 3)</td>
<td>(2, 2)</td>
<td>(3, 3)</td>
<td>(3, 3)</td>
</tr>
<tr>
<td>(3, 0)</td>
<td>(2, 2)</td>
<td>(3, 3)</td>
<td>(0, 3)</td>
<td>(0, 0)</td>
<td>(2, 2)</td>
</tr>
<tr>
<td>(3, 0)</td>
<td>(3, 3)</td>
<td>(2, 2)</td>
<td>(3, 3)</td>
<td>(0, 3)</td>
<td>(0, 3)</td>
</tr>
</tbody>
</table>

\textbf{Table 3} Utility functions of Example 11 for \( \theta = (P, \bar{R}) \) (left, \( P \) is the murderer), \( \theta = (\bar{P}, \bar{R}) \)
(center, \( Q \) is the murderer) and \( \theta = (\bar{P}, R) \) (right: \( R \) is the murderer). The configuration \( \theta = (P, R) \) is impossible.

\textbf{Example 12 (IGame’s strategy).} Let \( \sigma = (\sigma_1, \sigma_2) \) be defined by \( \sigma_1(P) = Q_1, \sigma_1(\bar{P}) = P_1, \sigma_2(\bar{R}) = Q_2, \sigma_1(\bar{R}) = R_2 \). \( \sigma \) is a pure strategy encoding that Agent 1 will choose \( Q \) when
learning that \( P \) is the murderer, and choose \( P \) otherwise, and that Agent 2 will choose \( Q \)
when learning that \( R \) is the murderer, and choose \( R \) otherwise. Since no agent, whatever is
their type, has incentive to move to another strategy, \( \sigma \) is a Nash equilibrium.

Formally speaking, a Bel-Game is fully defined by two elements: the prior knowledge and
agents’ preferences, that is, the bpa and the utility functions. The other parameters (types
for agents \( I \), their actions \( A \) and their “types” \( T \) ) are just parameters of this pair:

\textbf{Definition} belgame (I : finType) (A : I \to eqType) (T : I \to finType) :=
(bpa R (cprofile T) \ast (cprofile A \to cprofile T \to agent \to R))%type.

For the sake of readability, we introduce two short-hands: \textit{Tconfig}, the type for “type”
configurations; and \textit{event}_\textit{ti} := \( \theta_i \mapsto \{ \theta' \mid \theta_i' = \theta_i \} \):

\textbf{Notation} \textit{Tconfig} := \{\textit{finType} of \{\textit{diff} \ \forall i : I, T i\}\}.

\textbf{Definition} \textit{event}_\textit{ti} i (ti : T i) := \{\text{set} t : \textit{Tconfig} \mid t i \equiv ti\}.

In a Bel-game, agents know their own “type” but may ignore others’ ones. For them to
reason and evaluate their utility, they will need to consider all possible “types” of other
players, and their respective knowledge. So, the prior has to be \textit{revisable} w.r.t. all possible
“types” – what we call a proper Bel-Game. We thus define the following Boolean predicate:
1:12 Bel-Games: A Formal Theory of Games of Incomplete Information

Definition proper_belgame \( A \ T \) \( (G : \) belgame \( A \ T) \) \( (\text{cond} : \) conditioning \( R \ T\text{config}) \) :
\[
\forall i : \text{agent}, [\forall t_i : T i, \text{revisable cond G.1 (event_t_i t_i)}].
\]

In a proper Bel-game, the utility of a given iprofile for Agent \( i \) whose “type” is \( \theta_i \), is the XEU value of this iprofile w.r.t. the revised knowledge:

Definition belgame_utility \( A \ T \) \( (G : \) belgame \( A \ T) \) \( (\text{cond} : \) conditioning \( R \ T\text{config}) \)
\( (\text{xeu : xeu_box R T\text{config}) (HG : proper_belgame G \text{cond}) (p : \) iprofile \( T \ A) (i : \text{agent} ) (ti : T i ) : R \) :
\[
\text{let kn := cond G.1 (event_t_i t_i) (is_revisable HG ti) in XEU kn (xeu (fun t = > G.2 (proj_iprofile p t) t i))}. \]

For Bel-games, the definition of Nash equilibria holds: an iprofile is a Nash equilibrium iff no agent with whatever “type”, has any incentive to change strategy:

Definition BelG_Nash_equilibrium_prop \( A \ T \) \( (G : \) belgame \( A \ T) \) \( (\text{cond} : \) conditioning \( R \ T\text{config}) \) \( (u : \) xeu_box \( R \_ \) \( (\text{HG : proper_belgame G \text{cond}) (p : \) iprofile \( T \ A) \) ) :
\[
\forall i : I, \forall t_i : T i, \forall a_i : A i, -(\text{belgame_utility u H p ti < belgame_utility u H (change_istrategy p ti a1) ti}). \]

6 Howson-Rosenthal-like transforms

Howson–Rosenthal’s theorem asserts the correctness of a transform, which casts a 2-player Bayesian game into an equivalent polymatrix (of complete information) [15]. This makes it possible to benefit from both theoretical and algorithmic results of classical game theory.

In the following, we formally define and prove the correctness of three Howson-Rosenthal-like transforms that we had devised in previous work [10, 30]. All these transforms cast \( n \)-player Bel-games into hypergraphical games; the produced games all have the same utility values, though different hypergraphs. These transforms can be applied depending on the conditioning and on the decision rule (cf. Table 4).

<table>
<thead>
<tr>
<th>Transform</th>
<th>Conditioning</th>
<th>XEU</th>
<th>Space</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct transform</td>
<td>Dempster’s c.</td>
<td>any</td>
<td>( O(k \times \text{Size}(G)^k) )</td>
<td>( O(k \times \text{Size}(G)^k) )</td>
</tr>
<tr>
<td>Conditioned transform</td>
<td>any</td>
<td>any</td>
<td>( O(k \times \text{Size}(G)^k) )</td>
<td>( O(k \times \text{Size}(G)^k) )</td>
</tr>
<tr>
<td>TBM transform</td>
<td>any</td>
<td>TBEU</td>
<td>( O(k \times \text{Size}(G)) )</td>
<td>( O(\text{Size}(G)) )</td>
</tr>
</tbody>
</table>

| Table 4 | Transforms, conditioning and XEU they are suited for, and their worst-case complexity w.r.t the \( k \)-additivity of the bpa and the size of the input Bel-game (from [30])

The three transforms all follow the same approach: starting from a Bel game \( G \), we construct the equivalent hypergraphical game \( \tilde{G} \), the vertices of which represent every “type” of every agent, denoted by dependent pairs \( (i, \theta_i) \), with action set \( A_i \). Its local games correspond to focal elements, so we benefit the hypergraphical game structure to compute an XEU (recall that global utility is the sum of local utilities, as XEU value is the weighted sum of utility w.r.t. focal elements). For all those transforms, let \( G \) be the input Bel-game and \( \tilde{G} \)’s vertices be the dependent pairs \( (i, \theta_i) \) with action set \( A_i \):

Variables \( (R : \) realFieldType \( ) (I : \) finType \( ) (A : I \to \text{eqType}) (T : I \to \text{finType}) \)
\( (G : \) belgame \( R \ A \ T) \).

Definition HR_agent \( : \) finType := [finType of \( i : I \& T i \)].

Definition HR_action \( (i_t_i : HR \_agent) : \) eqType := \( A (\text{projT1 i_t_i}) \).
6.1 Direct transform

The direct transform holds only for Dempster’s conditioning, which is made on-the-fly on prior focal elements. It is suited for any XEU.

Starting from a Bel Game, we construct a local game $g_B$ for each prior focal element $B$. Vertex $(i, \theta_i)$ plays in $B$ iff $\theta_i$ is possible in $B$, that is, if $\exists \theta' \in B, \theta_i = \theta'$. Its local utility in $g_B$ is the “part of XEU” computed over $B'$, the subset of $B$ on which the mass shall be transferred during Dempster’s conditioning. Formally:

**Definition 13** (Direct transform of a Bel game). The direct transform of a Bel game $G = (I, (A_i, \Theta_i, u_i)_{i \in I}, m)$ is the hypergraphical game $\tilde{G} = (\tilde{I}, \tilde{E}, (\tilde{A}(i, \theta_i))_{(i, \theta_i) \in \tilde{I}}, (\tilde{u}_{(i, \theta_i)})_{(i, \theta_i) \in \tilde{E}})$:

- $\tilde{I} = \{(i, \theta_i) \mid i \in I, \theta_i \in \Theta_i\}$,
- $\tilde{E} = (e_B)_{B \subseteq S_{\Theta}}$, where $e_B = \{(i, \theta_i) \mid \theta \in B, i \in I\}$,
- $\tilde{A}(i, \theta_i) = A_i$,
- For each $e_B \in \tilde{E}, (i, \theta_i) \in e_B$ and $\tilde{\sigma} \subseteq \tilde{A}$, let $\tilde{u}^B_{(i, \theta_i)}(\tilde{\sigma}_{e_B}) = m(B) \times f_{XEU}(B \cap \{\theta' \mid \theta_i = \theta\})/\mathrm{P}(\{\theta' \mid \theta_i = \theta\})$.

First, let $G$ be a proper Bel-game w.r.t. Dempster’s conditioning, $xeu$ be any of the XEU:

- Variable $(\text{proper}_G : \text{proper_belgame}(G \text{ Dempster\_conditioning} \text{ R Tconfig}))$
- $(xeu : \text{XEU} \text{ R Tconfig})$.

Then let $\tilde{G}$’s local games be indexed by focal elements, that is, sets of type configurations. A vertex $(i, \theta_i)$ plays in the local game $g_B$ iff $\theta_i$ is possible in $B$:

- **Definition** $\text{HR\_direct\_localgame} := \{\text{finType of} \{\text{set Tconfig}\}\}$.
- **Definition** $\text{HR\_direct\_plays\_in} (lg : \text{HR\_direct\_localgame}) (i, ti : \text{HR\_agent}) : \text{bool}$ := $[\exists \text{t : Tconfig}, [k & \text{t} \in\in lg \& \text{t} (\text{projT1 i, ti}) == \text{projT2 i, ti}]]$.

Then, the local utility functions are expressed using $\text{HRcond\_mkprofile}$ bookkeeping function, which constructs from a local profile $p$ and a “type” configuration $\theta$ the cprofile $(p(1, \theta_1), \ldots, p(n, \theta_n))$. Local utility in a local game $g_B$ is the part of the XEU computed from the prior focal element $B$. Note that Dempster’s conditioning transfers masses from $B$ to $B \cap \{\theta' \mid \theta_i = \theta\} = B'(\text{event\_ti} \theta_i)$, so the local utility correspond to an on the fly Dempster’s conditioning. The resulting HG game is finally build from local utility functions:

- **Definition** $\text{event\_ti} (\text{ti} : \text{T ti}) := \{\text{set t : Tconfig} \mid \text{ti} \in\in \text{ti}\}$.
- **Definition** $\text{HR\_direct\_mkprofile} (lg : \text{HR\_direct\_plays\_in} \text{ lg i, ti}) (i, ti : \text{HR\_agent}) : \text{profile}$ := $[\exists \text{p : HR\_direct\_localprof} \text{ lg} \rightarrow \text{HR\_direct\_localagent} \text{ lg} \rightarrow \text{R} := \text{fun} \text{ p x} \Rightarrow \text{let} (\text{i, ti}, \text{Hi, ti}) := \text{x in let} (\text{i, ti}) :=\_ i, \text{ti} \in\in \text{G.1 lg} \& \text{xeu} \text{ R} = \text{fun} \text{ t} \Rightarrow \text{G.2 (HR\_direct\_mkprofile} \text{ Hi, ti p t} \text{ ti}) \text{ t i})$
- $(\text{lg} : k : \text{event\_ti ti}) / \text{P1 G.1 (event\_ti ti)}$.

- **Definition** $\text{HR\_direct\_transform\_cgame} \text{ R HR\_action} := \text{hg\_game HR\_direct\_u}$.

Finally, we prove the correctness of this transform: it preserves utility values, thus also preserves Nash equilibria. For any $iprofile\_sigma, \text{XEU}_{(i, \theta_i)\sigma}(\sigma) = \tilde{u}_{(i, \theta_i)}(\text{iprofile\_flatten}(\sigma))$:

- **Theorem** $\text{HR\_direct\_transform\_correct} (i : I) (\text{ti} : \text{T ti}) (p : \text{iprofile T A}) : \text{belgame\_utility xeu} \text{ proper}_G \text{ p ti} = \text{HR\_direct\_transform (iprofile\_flatten p) (existT_\_ i ti)}$
- **Theorem** $\text{HR\_direct\_eqNash} (p : \text{iprofile T A}) : \text{BelG\_Nash\_equilibrium\_prop xeu} \text{ proper}_G \text{ p} \iff \text{Nash\_equilibrium\_prop HR\_direct\_transform (iprofile\_flatten p)}$. 

H. Fargier, É. Martin-Dorel and P. Pomeret-Coquot 1:13
6.2 Conditioned transform

The conditioned transform holds for any conditioning and XEU.

Starting from a Bel Game $G$, all the conditioning “given $\theta_i$” are pre-computed, let $S^*$ be the union of all posterior focal sets (i.e. the set of all possible focal elements given any $\theta_i$).

Each $B \in S^*$ leads to a local game. As for the direct transform, a vertex $(i, \theta_i)$ plays in $g_B$ if $\theta_i$ is possible in $B$. Its utility in $g_B$ is the part of XEU computed over the posterior focal element $B$. Note that $(i, \theta_i)$’s local utility in $B$ may be 0, if $B$ is not focal in the posterior “given $\theta_i$”. Formally:

Definition 14 (Conditioned transform). The conditioned transform of a Bel game $G = (I, (A_i, \Theta_i, u_i)_{i \in I}, m)$ is the hypergraphical game $\hat{G} = (\hat{I}, \hat{E}, (\hat{A}_{(i, \theta_i)})_{i \in I}, (\hat{u}^{i,\theta_i}_{e})_{(i, \theta_i) \in E,(i, \theta_i) \in E})$:

- $\hat{I} = \{(i, \theta_i) : i \in I, \theta_i \in \Theta_i\}$,
- $\hat{E} = (e_B)_{B \in S^*}$, where $e_B = \{(i, \theta_i) : \theta \in B, i \in I\}$,
- $\hat{A}_{(i, \theta_i)} = A_i$,
- For each $e_B \in \hat{E}, (i, \theta_i) \in e_B$ and $\bar{\sigma} \in \hat{A}$, let $\hat{u}^{i,\theta_i}_{e_B}(\bar{\sigma}) = u_i(|\bar{\sigma}|, \theta_i)$ in $\hat{u}^{i,\theta_i}_{e_B}$.

First let $xeu$ be any XEU, $cond$ any conditioning and $G$ be a proper belgame w.r.t $cond$:


After similar definitions for $HRcond_localgame$ and $HRcond_plays_in$, let:

Definition $HRcond_u : \forall lg, HRcond_localprof lg \Rightarrow HRcond_localagent lg \Rightarrow R$

:= fun lg p x \Rightarrow let (i_t_i, Hi_t_i) := x in let (i, ti) := i_t_i in
let kn := cond G.1 (event_ti ti) (is_revisable proper_G ti) in
kn lg * xeu (fun t => G.2 (HRcond_mkprofile Hi_t_i p t) t i) lg.

Definition $HRcond_transform : cgame R HR_action \Rightarrow hg_game HRcond_u$.

Finally, we prove the correctness of the conditioned transform:

Theorem $HRcond_transform_correct (i : I) (ti : T i) (p : iprofile T A)$:

belgame_utility proper_G p ti
= HRcond_transform (iprofile_flatten p) (existT _ i ti).

Theorem $HRcond_eqNash (p : iprofile T A)$,

BelG_Nash_equilibrium_prop xeu proper_G p
\leftrightarrow Nash_equilibrium_prop (HRcond_transform) (iprofile_flatten p).

6.3 TBM-transform

The TBM-transform is designed for the Transferable Belief Model [37], in which knowledge is first revised using Dempster’s conditioning, then at the moment of deciding, a pignistic probability distribution $BetP$ is deduced from the posterior bpa (according to the equiprobability assumption) and the decision is made according to the Expected Utility criterion. Here we benefit $BetP$’s 1-additivity to produce a low-complexity hypergraph: local games correspond to single states of the world, that is, they all involve only $n$ players.

Definition 15 (TBM-transform). The TBM transform of a Bel game $G = (I, (A_i, \Theta_i, u_i)_{i \in I}, m)$ is the hypergraphical game $G = (I, E, (A_{(i, \theta_i)})_{(i, \theta_i) \in I}, (u^{i,\theta_i}_{e})_{(i, \theta_i) \in E,(i, \theta_i) \in E})$:

- $I = \{(i, \theta_i) : i \in I, \theta_i \in \Theta_i\}$,
- $E = [Players(\{\theta\}) : \theta \in \Theta]$. 
\[ \tilde{A}_{(i, \theta_i)} = A_i, \]
\[ \tilde{u}_{(i, \theta_i)}(\sigma_x) = \text{BetP}_{(i, \theta_i)}(\theta) \times u_i(\sigma(\theta), \theta). \]

First let \( \text{cond} \) be a conditioning and \( G \) be a proper Bel-game w.r.t. \( \text{cond} \). \( \tilde{G} \)'s local games are indexed by “type” configurations, and \( (i, \theta_i) \) plays in \( g_{\theta_i} \) iff \( \theta_i = \theta'_i \):

\[
\text{Variables} \ (\text{cond} : \text{conditioning} \ R \ \text{Tconfig}) \ \ (\text{proper}_G : \text{proper}_\text{belgame} \ \text{cond}).
\]

\[
\text{Definition} \ \text{HRTBM_localgame} : \text{finType} := \text{Tconfig}.
\]

\[
\text{Definition} \ \text{HRTBM_plays_in} : \text{HRTBM_localgame} \to \text{pred} \ \text{HR_agent} := \text{fun} \ lg \ i \_ti \Rightarrow \lg (\text{projT1} i \_ti) == \text{projT2} i \_ti.
\]

Local utility is computed w.r.t. the “pignistic” distribution \( \text{BetP} \):

\[
\text{Definition} \ \text{HRTBM_u} : \forall \ lg, \ \text{HRTBM_localprof} \ lg \to \text{HRTBM_localagent} \ lg \to \text{R} := \text{fun} \ lg \ p \ x := \text{let} \ (i \_ti, _) := x \ \text{in} \ \text{let} \ (i, ti) := i \_ti \ in
\]
\[
\text{let} \ \text{btep} := \text{BetP} (\text{cond} \ G.1 \ (\text{event}_{ti} ti) \ (\text{is_revisable} \ \text{proper}_G ti)) \ \text{in}
\]
\[
\text{dist} \ \text{btep} \ lg \ast G.2 (\text{HRTBM_mkprofile} p) \ lg i.
\]

\[
\text{Definition} \ \text{HRTBM_transform} : \text{cgame} \ R \ \text{HR_action} := \text{hg_game} \ \text{HRTBM_u}.
\]

Finally, we prove the correctness of the TBM transform:

\[
\text{Theorem} \ \text{HRTBM_transform_correct} (i : I) (ti : T i) (p : iprofile T A) :
\]
\[
\text{belgame Utility} (\text{TBEU } _) \ \text{proper}_G p \ ti
\]
\[
= \text{HRTBM_transform} [\text{fun} \ j \_tj \Rightarrow p (\text{projT1} j \_tj) (\text{projT2} j \_tj)] \ (\text{existT } i \_ti).
\]

\[
\text{Theorem} \ \text{HRTBM_eqNash} (p : iprofile T A),
\]
\[
\text{BelG_Nash_equilibrium_prop} (\text{TBEU } _) \ \text{proper}_G p
\]
\[
\leftrightarrow \text{Nash_equilibrium_prop} \ \text{HRTBM_transform} (\text{iprofile_flatten} p).
\]

7 Conclusion and Perspectives

In addition to the “Bel game” model and the three transforms definition, the main contribution of this article is its formal verification in the Coq proof assistant. On the one hand, it provides strong guaranties on the correctness of the results, and allow to identify subtleties that were left implicit on the paper proof. Moreover, it helps to improve the proofs, both in their flow and in their prose. On the other hand, we provide a structured library for belief functions and hypergraphical games, as for some generic lemmas which we aim to propose for integration in MathComp/SSReflect library.

This work opens several research directions, both on the theoretic side and on the formal verification side. On one side, we aim at extending this result to other decision-theoretic approaches, as for instance partially-ordered utility aggregations for belief function and other non-additive-measure approaches (Choquet capacities of order 2, RDU). On the other side, we aim to formalize Fagin-Halpern’s conditioning rule, as for the Moebius inverse and more generally the points we left aside as they were not necessary for this result. Furthermore we like to encompass this work into a larger library on decision under uncertainty, as it helps model extension as much as reusability.

References


Bel-Games: A Formal Theory of Games of Incomplete Information


24 Érik Martin-Dorel and Sergei Soloviev. A Formal Study of Boolean Games with Random Formulas as Payoff Functions. In 22nd International Conference on Types for Proofs and


Pierre Pomeret-Coquot, Hélène Fargier, and Érik Martin-Dorel. Games of Incomplete Information: a Framework Based on Belief Functions. working paper or preprint, May 2022. URL: [https://hal-univ-tlse3.archives-ouvertes.fr/hal-03658700](https://hal-univ-tlse3.archives-ouvertes.fr/hal-03658700).


