Opponent-model search in games with incomplete information

Junkang Li, Bruno Zanuttini, Véronique Ventos

Opponent-model search in games with incomplete information*

Junkang Li[1,2] Bruno Zanuttini[2] Véronique Ventos[1]
[1]NukkAI, Paris, France
[2]Normandie Univ.; UNICAEN, ENSICAEN, CNRS, GREYC, 14 000 Caen, France
junkang.li@nukk.ai, bruno.zanuttini@unicaen.fr, vventos@nukk.ai

May 18, 2023

Abstract

Games with incomplete information are games that model situations where players do not have common knowledge about the game they play, e.g. card games such as poker or bridge. Opponent models can be of crucial importance for decision-making in such games. We propose algorithms for computing optimal and/or robust strategies in games with incomplete information, given various types of knowledge about opponent models. As an application, we describe a framework for reasoning about an opponent's reasoning in such games, where opponent models arise naturally.

1 Introduction

Most algorithmic studies in game theory focus on the computation of exact or approximate Nash equilibria, which leaves much to be desired for many reasons. Firstly, large games usually have more than one equilibrium. Which equilibria should be considered rational or reasonable has long been a subject of study: refinements of Nash equilibria (van Damme, 1991), epistemic game theory (Perea, 2012), etc. More importantly, Nash equilibrium as a solution concept has two implicit assumptions: both players have unlimited computational power for computing Nash equilibria, and each knows which equilibrium the other will choose. Needless to say, both are difficult to justify in real-life situations.

These two assumptions are also opponent models in disguise, which are models that describe or predict how an opponent reasons in a game. In this paper, we are interested in more general opponent models than those behind Nash equilibria. Such opponent models have been explicitly incorporated into game tree search algorithms (e.g. minimax, $\alpha\beta$ search, MCTS) for games with perfect information, for instance by Iida et al. (1993, 1994). The knowledge of opponent models can result in both acceleration

*

of game tree search (e.g. by pruning branches not considered by the opponents) and increase of the performance of strategies computed (e.g. by exploiting the weakness of the opponents).

In this paper, we will apply the same idea to games with incomplete information, where opponent models yield even more interesting results than in games with perfect information. We will propose different ways of taking opponent models into account, and give algorithms for computing the corresponding robust and optimal responses. We will further propose a principled method to take into account the probability that the opponent does not behave according to any of the given models. Finally, we will show an application of these models to the recursive modelling of opponents, where a level-k player assumes that their opponents reason at some level lower than k, and recursively down to level 0.

2 Related work

Equilibrium concepts in games with perfect or imperfect information have long been studied; in particular, they have been related to models of knowledge and beliefs (for each player about the others' reasoning and strategies) via the concept of rationalizability in the field of epistemic game theory. For a thorough treatment, the reader may refer for instance to the textbooks by Perea (2012) or Bonanno (2018).

When no opponent model is available, one typically considers all possible (pure or mixed) strategies. In this case, Koller and Megiddo (1992) and Koller, Megiddo, and von Stengel (1996); von Stengel (1996) study the complexity of computing maxmin strategies under a variety of settings; in particular, for mixed strategies, they give polynomial-time algorithms based on linear programming for two-player extensive-form games with perfect recall, a more general setting than ours. McMahan, Gordon, and Blum (2003) propose a double-oracle algorithm for computing optimal mixed strategies for Markov decision processes with adversarial cost functions, which can also be regarded as a polynomial-time algorithm for computing the maxmin strategies of a normal-form game. Bošanský et al. (2014) propose an algorithm that combines the ideas of linear programming and double-oracle for zero-sum extensive-form games with perfect recall, and experimentally demonstrate that it is more efficient than other algorithms when optimal mixed strategies have small supports.

Opponent models can come in diverse forms. Iida et al. (1993, 1994) propose opponent models for games with perfect information, where models are given by the evaluation function and the search depth of the opponent. A recent survey of opponent modelling approaches is provided by Albrecht and Stone (2018). Our work is related to these in the sense that we assume opponent models to be given (called "type-based reasoning" by Albrecht and Stone (2018, Sec. 4.2)). However, an important stream of work also studies the *learning* of opponent models; we refer the reader to the survey by Nashed and Zilberstein (2022).

Among opponent models, an important class is the *recursive* models, where MAX searches a strategy (at level k) assuming that MIN themselves searches a strategy (at level $k-1$) assuming that MAX searches…, etc, down to level 0. Such models have been essentially studied to capture human reasoning in games. Camerer, Ho, and

Chong (2004) propose a *cognitive hierarchy model*, where an opponent's level is modelled by a Poisson distribution on levels $0, \ldots, k-1$, and validate this model against empirical data. Wright and Leyton-Brown (2019) assess the relevance of various modelling assumptions for level 0. De Weerd, Verbrugge, and Verheij (2013) assess the efficiency of reasoning with recursive models by simulation. Such recursive models are also used in epistemic game theory to define notions such as common belief in rationality (Perea, 2012).

Finally, a closely related line of work is that about interactive POMDPs (Gmytrasiewicz and Doshi, 2005; Doshi, Gmytrasiewicz, and Durfee, 2020) for collaborative decision-making in partially observed environments. In this model, a level-k agent optimizes their behaviour given a distribution over (partially observed) physical states and over other agents' models at level $k-1$. An interesting feature of this model is that optimal behaviours at level k can be computed iteratively as a sequence of optimal policies for POMDPs, where at each iteration the other agents' model can be considered as part of the environment.

3 Background

In this paper, we focus on games in extensive form, i.e. represented by a tree. We also focus on zero-sum games with two players (MAX and MIN), but our study can be easily extended to more players and general-sum.[1] We briefly describe our setting and refer the reader to textbooks (Maschler, Solan, and Zamir, 2020, for instance) for details.

In an extensive-form game with no chance, each internal node n of the game tree is owned by a player. To each terminal node, an *outcome* (or *value*) is attached, typically a real number, which denotes the payoff for MAX (and MIN's payoff is the opposite).

We denote MAX and MIN by $+$ and $-$, respectively. Under imperfect information, an *information set* for a player $i \in \{+, -\}$ is a set of their nodes that they cannot distinguish. A *pure strategy* for i, denoted by s_i, maps each information set IS of i to an action available at IS; in particular, the same action must be chosen at all nodes in the same IS. A *mixed strategy* for i, denoted by σ_i, is a probability distribution over the set of all pure strategies of i, with the interpretation that i plays a pure strategy randomly chosen according to this distribution at the beginning of a game.

We write Σ_i^{P} (resp. Σ_i^{M}) for the set of all pure (resp. mixed) strategies of player i in a game. We also write $p_1 s_i^1 + \cdots + p_k s_i^k$ for a mixed strategy of i with support $\{s_i^1, \ldots, s_i^k\}$ and probabilities $p_1, \ldots, p_k$; in particular, a pure strategy can be regarded as a mixed strategy with singleton support. In a game with no chance, a *profile* of pure strategies $(s_+, s_-) \in \Sigma_+^{\mathrm{P}} \times \Sigma_-^{\mathrm{P}}$, uniquely determines a terminal node to be reached. The *payoff* (for MAX) under this profile, written as $u(s_+, s_-)$, is defined to be the value of this terminal node. The *expected payoff* (for MAX) under a profile of mixed strategies $(\sigma_+, \sigma_-) \in \Sigma_+^{\mathrm{M}} \times \Sigma_-^{\mathrm{M}}$ is the expectation of MAX's payoff over drawings of pure strategies.

[1] With the exception of the lexicographic setting, for which the definition of the problem does not trivially generalise.

In general, games include *chance* nodes, which can be seen as being owned by a player called Nature, who uses a behaviour strategy that is common knowledge.

3.0.1 Games with incomplete information

In this paper, we study games with incomplete information, where players do not have common knowledge about the game they play. For example, a player can be uncertain about the payoff or available actions of other players, or whether other players are themselves uncertain about the game, etc. Notable examples of such games are poker, bridge, and mahjong, where the initial distribution of cards is not common knowledge.

A game with incomplete information can be modelled as a game with imperfect information via the Harsanyi model, which uses the notion of types to define the knowledge of a player. For example, in a game of poker or bridge, the type of player is their hand. More concretely, at the beginning of a game, there is a chance node that selects a type for each player according to a common prior. Every player learns their own type but not the types of the other players. Then all players participate in a game where the form of the game tree and the outcomes can depend on each player's type, but the actions of every player only depend on their type.[2]

3.0.2 Best-defence model

We are interested in decision-making in games with incomplete information where all actions except the selection of each player's type by the chance node at the root are public. We also assume that there is no other chance node. In other words, we are interested in two-player zero-sum games with incomplete information where had the types been common knowledge, the game would be of perfect information without chance node; we call them combinatorial games with incomplete information (CGII). However, the results in this paper can be extended to any game with incomplete information with minor modifications.

Given a CGII, our goal is to find maxmin-like strategies of MAX. Such strategies are usually computed by backward induction (most typically minimax-like depth-first search) in games with perfect information. However, for a game with incomplete information, traditional backward induction is impossible since there is no non-trivial subgame: each proper subtree is connected to another one via at least one information set. Instead, an approximation of maxmin strategies can be found using the *best-defence model* (Frank and Basin, 2001), which, by assuming MIN knows MAX's type, simplifies a game where both players have incomplete information into a game where MIN has perfect information.

Throughout the paper, we study CGIIs under the best-defence model. In general, available actions of MIN depend on their type, so not every terminal node is reachable by a given type of MIN. However, one can assume that the payoff for MAX at such an unreachable terminal node is $+\infty$, which does not change the maxmin value of the game (Frank and Basin, 2001). Therefore, we assume without loss of generality that

[2] An equivalent model called the *Aumann model* uses Kripke structures where an equivalence class for player i corresponds to a type of i in the Harsanyi model. For more details, we refer the reader to the textbook by Maschler, Solan, and Zamir (2020).

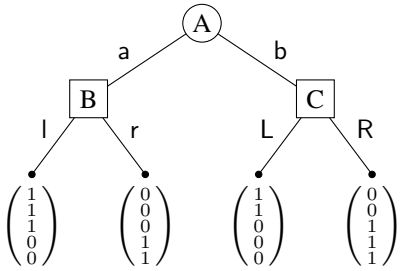


Figure 1: A CGII with 5 possible worlds.

MIN's set of actions is independent of their type, and all terminal nodes are reachable by any type of MIN.

Formally, a CGII under the best-defence model is specified by a game tree (the nodes of which are partitioned into terminal nodes, MIN's decision nodes, and MAX's decision nodes), an integer $t \geq 1$ (which denotes the number of MIN's types), a common prior $\vec{q}$ over MIN's types, and a payoff function $u : L \to \mathbb{R}^t$ that to each terminal node $n \in L$ of the game tree, assigns a vector of length t written as $\vec{u}(n) = (u(n)_1, \ldots, u(n)_t)$,[3] where $u(n)_i$ is the payoff for MAX at node n if MIN is of type i. Notice that a game with perfect information and no chance is a CGII under the best-defence model with $t = 1$, i.e. only one type of MIN.

Example. *A CGII under the best-defence model with* 5 *types of MIN is given in Figure 1, where we use square and circle to denote MAX's and MIN's nodes, respectively. Unless stated otherwise, we assume in all our examples that the common prior over MIN's types is uniform (hence each type occurs with probability* $1/5$ *in this CGII). An example of playout: if MIN plays* a *at* A *and MAX plays* l *at* B*, MAX's payoff vector will be* $(1, 1, 1, 0, 0)$*, which means MAX's gain is* 1 *if MIN is of one of the first three types, and* 0 *otherwise.*

4 Maxmin values without opponent models

In this section, we give an overview of algorithms from the literature for computing maxmin values without opponent models, which will be the basis of our algorithms for opponent-model search.

We are interested in computing the *maxmin value*

$$\underline{v}_+ :- \max_{\varsigma_+ \in \Sigma_+} \min_{s_- \in \Sigma_-^{\mathrm{P}}} u(\varsigma_+, s_-), \tag{1}$$

where Σ_+ is Σ_+^{P} or Σ_+^{M}, depending on context. We recall that u denotes the expected payoff (for MAX) with respect to type distribution and mixed strategies. Since u is linear in MIN's mixed strategies, replacing Σ_-^{P} by Σ_-^{M} in (1) will not change the value defined, hence we define the maxmin value to be against all pure strategies of MIN.

[3] In the following, we use the notation v_i for the i-th component of a vector $\vec{v}$.

The maxmin value $\underline{v}_+$ is the largest payoff MAX can guarantee by any strategy from Σ_+, no matter how MIN plays. MAX's strategies achieving this value are called *maxmin strategies*. Depending on whether we allow MAX to use mixed strategies, two notions of maxmin arise: pure maxmin and mixed maxmin. By definition, it is clear that the mixed maxmin value is no smaller than the pure maxmin value. As we will see, it is in general more difficult to compute the pure than the mixed maxmin value for a CGII. Still, in some situations, pure maxmin is more desirable or even the only viable solution concept, e.g. when outcomes are only partially ordered, or when mixed strategies are not allowed due to their probabilistic nature. Hence, we will study algorithms for both notions, with a focus on pure maxmin since algorithms for mixed maxmin only require minor modifications in the presence of opponent models.

4.0.1 A generic minimax algorithm

We will focus on algorithms for computing the maxmin *value*, but they can be easily modified to compute the corresponding maxmin *strategies*. The maxmin value of games with perfect information is typically computed by the *minimax* algorithm, a generic version of which is shown in Algorithm 1.

Algorithm 1: Generic minimax algorithm

```
1 def MiniMax(node n):
2     if n is a terminal node:
3         return eval(n)
4     else:
5         find the set of n's successor nodes C(n)
6         if n is MAX's decision node:
7             return ⋁_{n'∈C(n)} MiniMax(n')
8         else:
9             return ⋀_{n'∈C(n)} MiniMax(n')
```

This depth-first search algorithm has four parameters, which we will use to capture different algorithms in the following sections:

- V is a set of objects called *situational values*;
- eval is an evaluation function which maps each terminal node n to a value $\mathsf{eval}(n) \in V$;
- $\vee, \wedge : V \times V \to V$ are two associative binary operators, referred to as MAX's and MIN's operator, respectively.

With eval as boundary conditions, this algorithm recursively defines a situational value $\mathsf{val}(n)$ for every node n. For an instantiation of this algorithm to compute the maxmin values, one should choose the parameters as a function of the class of games under consideration, in such a way that there is a polynomial-time computable mapping from the situational value of the root $\mathsf{val}(r)$ to the maxmin value of the game.

For example, for games with perfect information, it is well-known that Algorithm 1 runs on the root yields the pure/mixed maxmin value (1) with $V :- \mathbb{R}$, $\mathsf{eval}(n) :- u(n)$, $\vee = \max$, and $\wedge = \min$.

This algorithm has several advantages: returned values for internal nodes are readily interpretable; the algorithm is extremely efficient on memory since the recursion depth is the depth of the game tree, which in general is exponentially smaller than the tree; the search can be combined with other techniques, such as heuristic functions and $\alpha\beta$ pruning (which is possible whenever $(V, \vee, \wedge)$ forms a lattice (Li et al., 2022)), move ordering, Monte Carlo techniques such as MCTS, etc.

In the following, we will present various algorithms for computing pure and mixed maxmin values, with or without opponent models. Whenever possible, we will describe them succinctly as a particular instantiation of Algorithm 1.

4.0.2 Pure maxmin

Frank and Basin (2001) show that the pure maxmin value is NP-hard to compute for CGIIs. The first exact algorithm was proposed by Ginsberg (2001), and it can be reframed as follows.

Proposition 1. *For a CGII with root r, t types of MIN, and common prior $\vec{q}$ over them, consider the instantiation of Algorithm 1 where: situational values are finite sets of vectors in $\mathbb{R}^t$; for all terminal nodes n, $\mathsf{eval}(n) :- \{\vec{u}(n)\}$; MAX's operator is set union $\cup$; MIN's operator is $\sqcap$, defined for all situational values f and g by:*

$$f \sqcap g :- \left\{ \left(\min(v_i, v_i') \right)_{1 \le i \le t} \mid \vec{v} \in f, \vec{v}\,' \in g \right\}.$$

Then it holds that

$$\underline{v}_+ := \max_{s_+ \in \Sigma_+^{\mathrm{P}}} \min_{s_- \in \Sigma_-^{\mathrm{P}}} u(s_+, s_-) = \max_{\vec{v} \in \mathsf{val}(r)} \vec{q} \cdot \vec{v}.$$

Example. *For the CGII in Figure 1, we get*

$$\begin{aligned} \mathsf{val}(B) &= \{(1,1,1,0,0), (0,0,0,1,1)\}; \\ \mathsf{val}(C) &= \{(1,1,0,0,0), (0,0,1,1,1)\}; \\ \mathsf{val}(A) &= \{(1,1,0,0,0), (0,0,1,0,0), \\ &\qquad (0,0,0,0,0), (0,0,0,1,1)\}. \end{aligned}$$

This algorithm recursively enumerates all strategies of MAX: each vector in $\mathsf{val}(n)$ implicitly represents one or several strategies of MAX for the subtree rooted at n. At the root A, given the uniform prior on MIN's types, the best vectors are $(1,1,0,0,0)$ (corresponding to MAX's strategy (l, L), by which MAX chooses l at B and L at C) and $(0,0,0,1,1)$ (corresponding to MAX's strategy (r, R)); both achieve the pure maxmin value $(\frac{1}{5}, \ldots, \frac{1}{5}) \cdot (1,1,0,0,0) = 2/5$.

Importantly, in a CGII, the expected payoff of the strategies of the subtree rooted at n may depend on that of strategies of a subtree rooted at another node n', which can be far away from n. In our example, l and R are locally optimal with respect to the

uniform prior. However, (l, R) is not optimal at the root, since it is MIN who chooses, with perfect information, either a or b as a function of their type. In other words, it is not correct to use the common prior to evaluate strategies locally at nodes B and C: the conditional probabilities of MIN's types at both B and C depend on MIN's strategy and can be different from the prior.[4]

4.0.3 Reduction of situational values

Even with non-locality, situational values, which are sets of vectors, can be reduced to accelerate the computation. If in $\mathsf{val}(n)$ a vector $\vec{v}$ is weakly dominated by another vector $\vec{v}'$, then we can discard $\vec{v}$ from $\mathsf{val}(n)$. This reduction corresponds to the elimination of weakly dominated strategies. For example, if A is an internal node of a larger CGII, then $(0, 0, 0, 0, 0)$ (corresponding to MAX's strategy (r, L)) can be discarded from $\mathsf{val}(A)$ without effect on the pure maxmin value of the larger game: MAX never does worse by playing, say, the strategy represented by $(1, 1, 0, 0, 0)$ in the subtree rooted at A.

In general, any reduction other than the elimination of dominated vectors is unsound, i.e. would yield incorrect results for at least one game. However, we will see that more reductions become sound if opponent models are available.

4.0.4 Mixed maxmin

The mixed maxmin value, defined by

$$\underline{v}_+ :- \max_{\sigma_+ \in \Sigma_+^{\mathrm{M}}} \min_{s_- \in \Sigma_-^{\mathrm{P}}} u(\sigma_+, s_-), \tag{2}$$

can be computed in polynomial time with the linear programming (LP) algorithm proposed by Koller and Megiddo (1992). This LP algorithm relies on two insights:

- The set of all mixed strategies of MAX can be represented by a system L of linear equalities, with linearly many (in the size of the game tree) variables and equalities.
- For any threshold v and any mixed strategy σ_+ of MAX represented as a solution to L, it can be verified in linear time whether $\min_{s_- \in \Sigma_-^{\mathrm{P}}} u(\sigma_+, s_-) \geq v$ by computing MIN's best responses to σ_+. This computation serves as the separation oracle in the LP.

Then the LP consists of maximising the variable v (which will yield the mixed maxmin value in (2)) under the constraints in L and the separation oracle. For more details, refer readers to Koller and Megiddo (1992).

Example. *In the game in Figure 1, the optimal mixed strategy is the uniform strategy, i.e. a uniform distribution over all* 4 *pure strategies of MAX. This strategy yields an expected payoff of at least* $1/2$*, which is the mixed maxmin value and is better than the pure maxmin value* $2/5$.

[4] This phenomenon, called *non-locality* by Frank and Basin (2001), is the culprit behind the NP-hardness of pure maxmin.

The above algorithm has been improved by von Stengel (1996); Koller, Megiddo, and von Stengel (1996). However, for simplicity, we only show modifications of the initial algorithm for taking opponent models into account. Adapting them to the improved algorithms is straightforward.

5 Opponent-model search

We now come to our main contributions, which are algorithms for finding maxmin strategies when given opponent models (OM). We will be interested in the maxmin value against a restricted set of opponent's strategies:

$$\underline{v}_+ = \max_{\varsigma_+ \in \Sigma_+} \min_{\omega_- \in \Sigma_-^{\mathrm{O}}} u(\varsigma_+, \omega_-),$$

where Σ_+ is the set of all pure or all mixed strategies for MAX, Σ_-^{O} is the set of strategies of MIN considered to be possible by the OMs, and ω_- an arbitrary strategy from Σ_-^{O}.

In general, OMs are models of the opponent's reasoning, which can come in various forms (see Section 2). As a quite general setting, we consider that an OM describes a behaviour strategy of MIN. A *behaviour strategy* for a player i maps each information set IS of i to a probability distribution over i's actions at IS. All mixed strategies can be expressed as behaviour strategies in games with perfect recall[5], and *a fortiori* in CGIIs since CGIIs are games with perfect recall. For a strategy represented by a mixed strategy or other linear representations (like sequence form (Koller and Megiddo, 1992) or evaluation function (Iida et al., 1993)), its equivalent behaviour strategy can also be computed in time linear in the size of the game tree.

Algorithmically, we assume that each OM is specified by an oracle $\mathcal{O}$ such that, for any decision node n of MIN and any type i of MIN, $\mathcal{O}(n, i)$ is the strategy at n of MIN of type i, specified as a probability distribution over MIN's actions available at n. We also assume that the OMs are given in the input and each call to the oracles takes constant time.

In this section, we consider situations where MAX is certain that MIN only considers strategies described by these OMs. This assumption will be relaxed in Section 6.

5.0.1 Single OM

We first present the simplest case, with only one OM ω_-, which means that MAX has complete knowledge of MIN's strategy. Then the game becomes a single-player game with perfect information (Koller and Megiddo, 1992), and the pure/mixed maxmin value becomes

$$\underline{v}_+ :- \max_{s_+ \in \Sigma_+^{\mathrm{P}}} u(s_+, \omega_-) = \max_{\sigma_+ \in \Sigma_+^{\mathrm{M}}} u(\sigma_+, \omega_-),$$

[5] *Perfect recall* means players never forget what they knew or did in the past. For the formal definition of perfect recall and the equivalence between mixed and behaviour strategies in games with perfect recall, see Kuhn (1953).

where the last equality is due to the linearity of u. This value can be computed by a bottom-up (i.e. depth-first) procedure, which recursively computes MAX's best strategies at each of their information set.

Since MIN's strategy is known perfectly, all MIN's decision nodes become chance nodes. As a consequence, even though MAX still does not know MIN's type, they can compute the exact probability of reaching a node under each of MIN's types and, using Bayesian updates, deduce the conditional probability of MIN's types at every node. Then MAX can choose the actions that maximise the payoff with respect to this conditional probability at any MAX's decision node.

Example. *Consider again the game on Figure 1, with ω_- defined as follows: MIN plays a if of type 1 or 2, b if of type 4 or 5, and $\frac{1}{2}a + \frac{1}{2}b$ if of type 3. Against ω_- and the uniform prior over MIN's types, MAX can compute the vector $(\frac{1}{5}, \frac{1}{5}, \frac{1}{10}, 0, 0)$ at node B, which we call the* non-normalised belief state *(NBS) at B. For instance, the first component means that the probability of the combined event that MIN is of type 1 and B is reached is $1/5$. Observe that normalising the NBS would give the posterior probability over MIN's types (for instance, $2/5$ for type 1, and 0 for type 5). Therefore, by maintaining an NBS, MAX implicitly performs Bayesian inference on MIN's types using ω_-.*

Given the NBS at B, action l *yields a higher (non-normalised) payoff of $1/2$ than* r *(with a payoff of 0) at B. Similarly, at C the NBS is $(0, 0, \frac{1}{10}, \frac{1}{5}, \frac{1}{5})$ and prescribes action* R *(with a payoff of $1/2$). At node A, MAX's payoff can be simply computed as the sum of their payoff at B and C, which yields 1. One can check that 1 is indeed the best MAX can get when playing against MIN with this particular OM, and this payoff is obtained by the strategy (l, R), which gives MAX a payoff of 1 independent of MIN's actual type.*

In general, every MAX's node n is the result of a series of MAX's actions and MIN's actions. MAX's NBS at n, written as $\overrightarrow{\mathsf{nbs}}(n)$, is computed component-wise: the i-th component is computed as the product of the probability of MIN being of type i and the probability that MIN of type i takes those actions leading to n at each of MIN's nodes that are an ancestor of n. In particular, the NBS at the root is the common prior over MIN's types. With the NBS of terminal nodes thus computed, we can then compute the best payoff for MAX.

Proposition 2. *For a CGII with root r and a single opponent model ω_-, consider the instantiation of Algorithm 1 where: $V = \mathbb{R}$; for all terminal nodes n, $\mathsf{eval}(n) := \overrightarrow{\mathsf{nbs}}(n) \cdot \vec{u}(n)$; MAX's operator is $\max$; MIN's operator is $+$. Then it holds that $\underline{v}_+ = \mathsf{val}(r)$, and the algorithm is polynomial-time.*

This algorithm can be seen as a generalisation of the OM search proposed by Iida et al. (1993), which only consider games with perfect information for which OMs are described by MIN's evaluation functions.

5.0.2 Probabilistic OMs

We now consider the case where MAX has several OMs $\omega_-^1, \ldots, \omega_-^m$ of MIN, and a probability distribution $p = (p_1, \ldots, p_m)$ over them: MIN plays the strategy ω_-^1 with

probability p_1, ω_-^2 with probability p_2, etc. In particular, the pure/mixed maxmin value is given by

$$\underline{v}_+ :- \max_{s_+ \in \Sigma_+^{\mathrm{P}}} \sum_{j=1}^{m} p_j u(s_+, \omega_-^j) = \max_{\sigma_+ \in \Sigma_+^{\mathrm{M}}} \sum_{j=1}^{m} p_j u(\sigma_+, \omega_-^j),$$

This setting is not much different from the previous one, due to the linearity of u: these OMs can be merged into one single OM describing the mixed strategy $\omega_- :- p_1\omega_-^1 + \cdots + p_m\omega_-^m$. In principle, one can traverse the game tree once and compute the behaviour strategy corresponding to ω_-, then run the single-OM algorithm from Proposition 2. Instead, we present a one-pass algorithm for probabilistic OMs, without the need to explicitly compute and store the mixed strategy ω_-. The key is to maintain not just one, but m NBSs, one $\overrightarrow{\mathsf{nbs}}_j$ for each OM ω_-^j.

Proposition 3. *For a CGII with root r and opponent models $\omega_-^1, \ldots, \omega_-^m$ distributed according to $p_1, \ldots, p_m$, consider the instantiation of Algorithm 1 where: $V = \mathbb{R}$; for all terminal nodes n, $\mathsf{eval}(n) :- \sum_{j=1}^{m} p_j \big(\overrightarrow{\mathsf{nbs}}_j(n) \cdot \vec{u}(n)\big)$; MAX's operator is $\max$; MIN's operator is $+$. Then it holds that $\underline{v}_+ = \mathsf{val}(r)$, and the algorithm is polynomial-time.*

5.0.3 Lexicographic OMs

An important subcase of search with multiple OMs is the case where MAX holds a lexicographic belief over MIN's OMs $\omega_-^1, \ldots, \omega_-^m$. For example, MAX deems that MIN most probably follows ω_-^1. Otherwise, with an infinitesimally smaller probability (compared to ω_-^1), MIN follows ω_-^2. Otherwise, with an infinitesimally smaller probability (compared to ω_-^2), MIN follows ω_-^3, etc. We define the pure/mixed maxmin value in this case to be

$$\begin{aligned}\underline{v}_+ :- &\operatorname*{lexmax}_{s_+ \in \Sigma_+^{\mathrm{P}}} \big(u(s_+, \omega_-^1), \ldots, u(s_+, \omega_-^m)\big) \\ = &\operatorname*{lexmax}_{\sigma_+ \in \Sigma_+^{\mathrm{M}}} \big(u(\sigma_+, \omega_-^1), \ldots, u(\sigma_+, \omega_-^m)\big),\end{aligned}$$

where lexmax is lexicographic maximum over vectors of length m. In other words, if there is a unique optimal strategy against ω_-^1, then this strategy is chosen; otherwise, ties are broken according to their values against ω_-^2, and so on.

This lexicographic belief can in fact be treated with the same algorithm as for probabilistic OMs. Indeed, it suffices to take the distribution over OMs to be $p = (1, \varepsilon, \varepsilon^2, \ldots, \varepsilon^{m-1})$, where ε is an indeterminate, the value of which will be taken to be 0 at the end of the computation.

We use the following operations over the set $\mathbb{R}^m[\varepsilon]$ of all real-coefficient polynomials in ε of order $m-1$: $\mathsf{eval}_{\varepsilon,m}$ defined by $\mathsf{eval}_{\varepsilon,m}(n) :- \sum_{j=1}^{m} \big(\vec{u}(n) \cdot \overrightarrow{\mathsf{nbs}}_j(n)\big)\varepsilon^{j-1}$; addition of polynomials $+_{\varepsilon,m}$; evaluation $f(x)$ of a polynomial f at a value $x \in \mathbb{R}$; and operator $\mathrm{lexmax}_{\varepsilon,m}$, which maps two polynomials $f_1, f_2 \in \mathbb{R}^m[\varepsilon]$ to f_1 if the leading coefficient of $f_1 - f_2$ is non-negative, and to f_2 otherwise.

Proposition 4. *For a CGII with root r and opponent models $\omega_-^1, \ldots, \omega_-^m$ with a lexicographic interpretation, consider the instantiation of Algorithm 1 where: $V = \mathbb{R}^m[\varepsilon]$; $\mathsf{eval} = \mathsf{eval}_{\varepsilon,m}$; MAX's operator is $\mathrm{lexmax}_{\varepsilon,m}$; MIN's operator is $+_{\varepsilon,m}$. Then it holds that $\underline{v}_+ = \mathsf{val}(r)(0)$, and the algorithm is polynomial-time.*

5.0.4 Nondeterministic OMs

We finally consider the case where MAX does not have a probability distribution over MIN's OMs: MIN's strategy is only known to be among $\omega_-^1, \ldots, \omega_-^m$. This situation is very similar to planning under adversarial cost functions (McMahan, Gordon, and Blum, 2003). The maxmin value is then

$$\underline{v}_+ :- \max_{\varsigma_+ \in \Sigma_+} \min_{1 \le j \le m} u(\varsigma_+, \omega_-^i),$$

which in general is different depending on whether Σ_+ is Σ_+^{P} or Σ_+^{M}. MIN now has (*a priori*) more agency than in the case of probabilistic OMs, since they can choose from a larger (but still limited) set of strategies.

As introduced for probabilistic OMs, for each node n, let $\overrightarrow{\mathsf{nbs}}_j(n)$ be the NBS for ω_-^j at n, which is a $t \times 1$ vector (each component corresponds to one of the t types of MIN). We can write all m NBSs as an $m \times t$ NBS matrix $\mathsf{NBS}(n) :- (\overrightarrow{\mathsf{nbs}}_1(n)^\intercal, \ldots, \overrightarrow{\mathsf{nbs}}_m(n)^\intercal)$.

Proposition 5. *For a CGII with root r and opponent models $\omega_-^1, \ldots, \omega_-^m$ with a nondeterministic interpretation, consider the instantiation of Algorithm 1 where: situational values are finite sets of vectors in $\mathbb{R}^m$; for all terminal nodes n, $\mathsf{eval}(n) :- \{\mathsf{NBS}(n) \times \vec{u}(n)\}$; MAX's operator is set union $\cup$; MIN's operator is $\uplus$, defined for all situational values f and g by $f \uplus g :- \{(v_j + v'_j)_{1 \le j \le m} \mid \vec{v} \in f, \vec{v}\,' \in g\}$. Then the pure maxmin value satisfies*

$$\underline{v}_+ :- \max_{s_+ \in \Sigma_+^{\mathrm{P}}} \min_{1 \le j \le m} u(s_+, \omega_-^j) = \max_{\vec{v} \in \mathsf{val}(r)} \min_{1 \le j \le m} v_j.$$

The algorithm above is exponential time in the worst case; it can actually be shown that this problem is NP-complete, even if MAX has perfect information (i.e. MIN only has 1 type) and there is only 2 OMs of MIN.

It can be seen that the knowledge of OMs transforms MAX's incomplete information about MIN's *type* into the one about MIN's *strategy*. Compared to Proposition 1, situational values are now sets of vectors of length m (instead of t). Each such vector implicitly represents a strategy of MAX by its expected payoff against each OM. In contrast with the case of probabilistic OMs, we cannot further collapse each vector to a single real number, since we have no distribution over the OMs. Still, reduction by weak dominance can be used just as for pure maxmin without any opponent model.

It follows that at the root, the remaining vectors are the non-dominated strategies of MAX against MIN's OMs. In other words, the algorithm computes the normal form of the game restricted to MIN's fixed m strategies, which justifies the correctness of Proposition 5 for pure maxmin. As for mixed maxmin, one can modify the separation oracle in the LP algorithm of Koller and Megiddo (1992): now the oracle only computes MIN's best responses from the m OMs.

6 Opponent models with uncertainty

We now come to our second contribution, about the case where a set of OMs of MIN is available, but MAX is not certain that MIN will behave as one of them. Without loss of generality, we focus on the case when there is a single OM ω_-, which encompasses as well the case of several OMs with a probability distribution, as discussed in Section 5.

We assume that with a probability p^∞, which is known to MAX, MIN does not follow ω_-, in which case their behaviour is arbitrary and unpredictable, and that with probability $1 - p^\infty$ MIN follows ω_-. Intuitively, p^∞ quantifies MAX's uncertainty about MIN's behaviour. This may arise for instance when MAX tries to estimate MIN's gameplay level: with $1 - p^\infty$, MIN is of a certain level with a behaviour predictable by some OM; otherwise, they have an unknown level and nothing can be assumed about their play.

This model presents a conflict between robustness and performance, well-known in the literature of linear programming with uncertain parameters or MDP planning under uncertain cost functions. MAX desires to be cautious and robust against MIN's unpredictable behaviour occurring with probability p^∞, and at the same time to improve their performance by exploiting their knowledge of the OM, which correctly predicts MIN's strategy with probability $1 - p^\infty$. Formally, we define the following modified maxmin value:

$$\underline{v}_+ :- \max_{\varsigma_+ \in \Sigma_+} \big((1 - p^\infty) u(\varsigma_+, \omega_-) + p^\infty \min_{s_- \in \Sigma_-^{\mathrm{P}}} u(\varsigma_+, s_-)\big),$$

where Σ_+ is either Σ_+^{P} or Σ_+^{M}.

Example. *Consider again Figure 1 and the OM ω_- "MIN plays* a *if of type* 1 *or* 2*,* b *if of type* 4 *or* 5*, and $\frac{1}{2}$a + $\frac{1}{2}$b if of type* 3*". The best strategy of MAX against ω_- is* (l, R) *with a payoff of* 1*. However, this strategy does not fare so well if MIN's strategy is not ω_- (or when p^∞ is close to* 1*): in the worst case, MIN plays* b *if of type* 1 *or* 2*, and* a *if of type* 4 *or* 5*. Against this strategy, MAX's expected payoff by playing* (l, R) *is only* $1/5$*. On the other hand, the pure maxmin strategy* (l, L) *only has a payoff of* $1/2$ *against ω_-, and so does the mixed maxmin strategy (which is the uniform strategy), hence neither is optimal when p^∞ is close to* 0*.*

It is clear from the example that the modified maxmin and optimal strategies depend on p^∞. We now show how to modify algorithms from the last sections to compute them.

6.0.1 Mixed strategies

We first consider the mixed strategies of MAX. The LP algorithm from Koller and Megiddo (1992) can compute the modified mixed maxmin value and an optimal strategy of MAX, with a minor modification of the separation oracle. Concretely, given a threshold v and a mixed strategy σ_+ for MAX, the separation oracle should now, apart from computing MAX's payoff v_{BR} with strategy σ_+ under MIN's best responses, also compute MAX's payoff against the OM $v_{\mathsf{OM}} = u(\sigma_+, \omega_-)$, then check whether $(1 - p^\infty)v_{\mathsf{OM}} + p^\infty v_{\mathsf{BR}} \geq v$ holds.

Example. *In the game of Figure 1 with ω_- as above, one can use this algorithm to verify that MAX's optimal strategy is* (l, R) *for $p^\infty \leq 5/8$, otherwise it is the uniform strategy. This confirms that when nondeterministic behaviour happens with a small enough probability, it is worth deviating from maxmin strategies in order to exploit the OM.*

6.0.2 Pure strategies

For pure strategies, we build on the algorithm for a single OM (Proposition 2). To cope with non-locality (because of MIN's partially unpredictable behaviour), we use situational values which are sets of ordered pairs $\langle s, \vec{v}\rangle$, with $s \in \mathbb{R}$ and $\vec{v} \in \mathbb{R}^t$, where t is the number of types of MIN. We call such a pair an *annotated vector*; it implicitly represents a strategy for MAX for which the payoff against ω_- is s, and the worst payoff against unpredictable behaviour is given by $\vec{v}$. We also maintain an NBS $\overrightarrow{\mathsf{nbs}}(n)$ for each node n, over MIN's types, as in Section 5.

Proposition 6. *For a CGII with root r, t types of MIN with common prior $\vec{q}$, opponent model ω_-, and probability p^∞ that MIN does not behave according to ω_-, consider the instantiation of Algorithm 1 where: situational values are finite sets of annotated vectors; for all terminal nodes n,* $\mathsf{eval}(n) :- \{\langle \overrightarrow{\mathsf{nbs}}(n) \cdot \vec{u}(n), \vec{u}(n)\rangle\}$*; MAX's operator is set union $\cup$; MIN's operator is $\sqcap'$, where, for all situational values f and g, $f \sqcap' g$ is defined to be*

$$\{\langle s + s', (\min(v_i, v_i'))_{1\leq i\leq t}\rangle \mid \langle s, \vec{v}\rangle \in f, \langle s', \vec{v}\,'\rangle \in g\}.$$

Then the modified pure maxmin value satisfies

$$\underline{v}_+ = \max_{\langle s,\vec{v}\rangle \in \mathsf{val}(r)} \big((1 - p^\infty)s + p^\infty(\vec{q} \cdot \vec{v})\big).$$

Notice that when combining two annotated vectors at a MIN's node, the scalar part is additive; this reflects the fact that when following the (single) OM, MIN has no agency, just as in the case without uncertainty.

Example. *Using the algorithm above for the game in Figure 1 with the aforementioned OM ω_-, we find that MAX's optimal strategy is* (l, R) *for $p^\infty \leq 5/7$, otherwise* (l, L) *or* (r, R)*. Again, this shows that it may be worth deviating from maxmin strategies in order to exploit an OM.*

6.0.3 Reduction of situational values

The algorithm in Proposition 6 generalises the one in Proposition 1, which can be regarded as the case $p^\infty = 1$. We have seen that, in the latter case, the only sound reduction of situational values is the elimination of weakly dominated strategies. Interestingly, when an OM is available, further reductions become sound.

Let n be a node, and $\langle s, \vec{v}\rangle, \langle s', \vec{v}\,'\rangle \in \mathsf{val}(n)$ be two annotated vectors. Discarding $\langle s', \vec{v}\,'\rangle$ because of $\langle s, \vec{v}\rangle$ is sound if MAX is never worse-off in the whole game if they choose $\langle s, \vec{v}\rangle$ instead of $\langle s', \vec{v}\,'\rangle$ at n.

Since scalar parts are summed up, if $s > s'$ holds, then $\langle s, \vec{v} \rangle$ has an advantage $s - s'$ over $\langle s', \vec{v}\,' \rangle$ in terms of contribution to the final value at the root. Contrastingly, for the vectorial part, components for which $\vec{v}$ is larger than $\vec{v}\,'$ might be erased by the combination (via component-wise $\min$) of vectors at an ancestor of n. In other words, $\vec{v}$'s advantage with respect to $\vec{v}\,'$ can be annihilated at the root. On the other hand, the components for which $\vec{v}$ is smaller than $\vec{v}\,'$ may never get erased so that $\vec{v}$'s disadvantage with respect to $\vec{v}$ can survive intact at the root.

Hence, in the worst case, $\vec{v}\,'$ can keep all advantages it has compared to $\vec{v}$, while $\vec{v}$ can lose all its advantages. Hence, to safely discard $\langle s', \vec{v}\,' \rangle$, the advantage of $\vec{v}\,'$ over $\vec{v}$ must be no larger than the advantage of s over s'. More formally, we can safely discard $\langle s', \vec{v}\,' \rangle$ when the following holds:

$$(1 - p^{\infty})(s - s') \geq p^{\infty} \sum_{1 \leq i \leq t} \big(q_i \max(v'_i - v_i, 0) \big), \tag{3}$$

Notice that without the scalar part (e.g. when $p^{\infty} = 1$), the pruning condition (3) reduces to $v_i \geq v'_i$ for all i, which is exactly the pruning condition shown in Section 4.

7 Application to recursive opponent models

We now propose an application of the algorithms presented before to the computation of optimal strategies with recursive opponent models. We formulate a quite general setting, where various types of opponent models naturally arise.

7.0.1 Limitations of the best-defence model

In general, in a game with incomplete information, both players have incomplete information, rather than just MAX. As a result, the best-defence model usually gives MIN too much power.

Example. *Consider the game in Figure 2, where MAX has* 3 *types and MIN has only* 1 *(hence MIN has incomplete information). If MAX reasons according to the best-defence model, then both actions* a *and* b *have a value of* 0*: MAX of type* i *reasons that MIN will play* a_i *at node* A*, and* b_j *at node* B *for some* $j \neq i$*. The culprit is that under the best-defence model, MAX assumes MIN is aware of MAX's type, therefore can adapt their strategy to it. However, if MAX realises MIN is unaware of their type, then MAX will prefer* a *since under uniform common prior over MAX's types,* a *yields an expected payoff of* $2/3$*, compared to* b*'s* $1/3$*.*

On the other hand, computing maxmin strategies for the original game tree without using the best-defence model is not ideal either, for these strategies fail to exploit any assumption one may have about their adversary, such as that they have limited computational power or reasoning depth, or that they have a predictable behaviour pattern. Such assumptions make sense in particular when playing against humans (Iida et al., 1993; Stahl and Wilson, 1995).

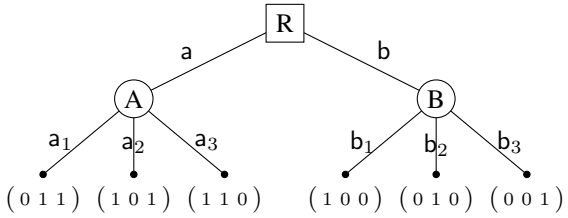


Figure 2: A CGII with 3 possible types of MAX.

7.0.2 Proposed framework

The framework which we propose can be seen as a generalisation of the cognitive hierarchy model (Camerer, Ho, and Chong, 2004) and at the same time as a counterpart of interactive POMDPs (Doshi, Gmytrasiewicz, and Durfee, 2020) for competitive games. The general idea is to define *level-k* strategies to be the optimal strategies against an adversary of level $k-1$, and recursively down to level-0 strategies. We however give a general and parametrizable definition about (1) how level-0 strategies are defined, (2) how optimal strategies at a given level are aggregated, and (3) how strategies of various levels are aggregated. Moreover, using our results in Section 6, the framework leaves the possibility for players to assign a non-zero probability to the event that their opponent has an unknown strategy/level.

As a consequence, this framework serves as a compromise between the best-defence model and the full game, and can be used to find better strategies against non-omnipotent and non-omniscient players; in particular, it generalises the best-defence model. Moreover, this framework can be used to explain real-life human psychological gameplay in games with incomplete information such as bridge, as we illustrate at the end of this section.

For the formal definition, consider a two-player zero-sum game. Let Σ^0_+, Σ^0_- be non-empty sets of strategies of MAX and MIN, respectively. Moreover, let $\oplus : 2^\Sigma \to \Sigma$ be a function that maps any set of (pure or mixed) strategies to a single (pure or mixed) strategy, and $\mathrm{BR} : \Sigma^* \to 2^\Sigma$ be a function which maps any tuple of strategies to a set of strategies; $\oplus$ will be used to aggregate strategies of a player at a given level, and BR to compute the set of optimal strategies given a tuple of opponent models (one per lower level).[6] Finally, for a player $i \in \{+,-\}$, we write $-i$ for another player.

Definition 7 (level-k strategies)**.** *Let Σ^0_+, Σ^0_-, $\oplus$, BR be defined as above, and let $i \in \{+,-\}$. The set of* level-0 *strategies for player i is defined to be Σ^0_i. For $k \geq 1$, the set of* level-k strategies *for i, written as Σ^k_i, is defined to be* $\mathrm{BR}\left(\oplus(\Sigma^{k-1}_{-i}), \oplus(\Sigma^{k-2}_{-i}), \ldots, \oplus(\Sigma^0_{-i})\right)$.

In short, the level-k strategies of player i are the best responses (computed by BR, the *best-response function*) against an opponent using the strategy $\oplus(\Sigma^{k'}_{-i})$ (computed by $\oplus$, the *intra-level aggregation*) at all levels $k' \leq k$. The boundary conditions, i.e. the level-0 strategies, are given by Σ^0_+ and Σ^0_-, which can come from maxmin strategies

[6]The framework could be easily adapted to more general functions, e.g. an aggregation of the strategies at the same level into a set or a tuple of strategies. It could also be easily adapted to general games, beyond the two-player and zero-sum assumptions.

under the best-defence model, randomly chosen strategies (McMahan, Gordon, and Blum, 2003), modelling assumptions for human players (Wright and Leyton-Brown, 2019), etc.

Example. *The Poisson-CH model in Camerer, Ho, and Chong (2004) is captured by choosing Σ_+^0 and Σ_-^0 to be the set of all pure strategies of MAX and MIN, the intra-level aggregation $\oplus$ to map any set of strategies to the uniform mixture of the set, and the best-response function* BR *to map a tuple of strategies $(\sigma_{-i}^{k-1}, \ldots, \sigma_{-i}^0)$ to the set of all pure best responses to the mixed strategy $p_{k-1}\sigma_{-i}^{k-1} + \cdots + p_0\sigma_{-i}^0$, where $p_{k-1}, \ldots, p_0$ follow some Poisson distribution.*

An interesting choice for the intra-level aggregation $\oplus : 2^\Sigma \to \Sigma$ is given by the uniform mixture, as in the example above. Under this $\oplus$, many other situations can be modelled by using different best-response functions BR for the inter-level aggregation. For games with incomplete information, if a function BR computes the best responses per type of player[7], then such BR can be implemented by the algorithms presented in the last sections. Some examples follow:

Probabilistic model If each player i at level k has a subjective probability over their opponent's reasoning levels in the form of a vector $(p_{i,k}^{k-1}, p_{i,k}^{k-2}, \ldots p_{i,k}^0)$, then we can define BR to compute, for each player i and level k, the best responses against the mixture $p_{i,k}^{k-1} \oplus (\Sigma_{-i}^{k-1}) + \cdots + p_{i,k}^0 \oplus (\Sigma_{-i}^0)$ (which can be implemented by the algorithm in Proposition 3). This model amounts to assuming that a player at level k reasons as if their opponent places themselves at a reasoning level drawn from the above distribution; such a distribution can be obtained by empirical studies, for instance by fitting a model against a population of possible opponents in an open tournament.

Iterative model By setting $p_{i,k}^{k-1} = 1$ for all i and k in the previous model, we can model situations where each player at level k assumes their opponent reason at exactly level $k-1$, which corresponds to Proposition 2.

Lexicographic model BR can also be defined to compute the best responses against the tuple of opponent models $(\oplus(\Sigma_{-i}^{k-1}), \ldots, \oplus(\Sigma_{-i}^0))$ under the lexicographic interpretation (which can be implemented by the algorithm in Proposition 4); this amounts to consider the opponent to reason at level $k-1$, and to tie-break equivalent strategies by level $k-2$, and so on.

Nondeterministic model With a BR similar to those in Proposition 5, we can model situations where each player at level k assumes the opponent reason at a level lower than k but without assuming a distribution over their levels. In such cases, the incomplete information about the opponent's types is transformed into the one about their reasoning levels, which are in general much fewer.

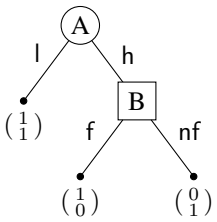
Partially unknown opponent model if in addition to the probabilistic or the lexicographic model above, we consider probabilities $p_{i,k}^\infty$ that the opponent of player i at level k is not reasoning at any level lower than k, then we can use the approaches under uncertainty from Section 6.

[7] In other words, for each player, BR computes the best strategies for each type of this player.

Let us also emphasise that the straightforward generalisation of this framework to general games allows, for instance, to take into account one's partner's incomplete information in multiplayer games, akin to interactive POMDPs.

7.0.3 A real-life example

We now give an example application of our formalism, which captures the psychological strategies of a contract bridge deal played in a bridge tournament. We present the abstract version of the game on Figure 3 (left); for the bridge deal itself, see Karpin (1977, p.266).



k	MAX	MIN 1	MIN 2
0	$\{\mathsf{nf}\}$	$\{\mathsf{l},\mathsf{h}\}$	$\{\mathsf{l},\mathsf{h}\}$
1	$\{\mathsf{nf}\}$	$\{\mathsf{h}\}$	$\{\mathsf{l},\mathsf{h}\}$
2	$\{\mathsf{f}\}$	$\{\mathsf{h}\}$	$\{\mathsf{l},\mathsf{h}\}$
3	$\{\mathsf{f}\}$	$\{\mathsf{h}\}$	$\{\mathsf{h}\}$

Figure 3: Recursive reasoning in a CGII with 2 types of MIN. For each $k \leq 3$, the set of level-k strategies (as defined in Definition 7) for MAX, MIN of type 1, and MIN of type 2 are given in the table.

In this game, the common prior about MIN's types is given by $p_1 = 0.4$ and $p_2 = 0.6$. For the recursive reasoning, $\oplus$ is given the uniform mixture, BR is given by the lexicographic model, and the level-0 strategies for both players are their pure maxmin strategies.

The first few levels of the recursive reasoning proceed as in Figure 3 (right). In the following, we write $\sigma^1_-|\sigma^2_-$ for MIN's strategy if type-1 MIN plays σ^1_- and type-2 MIN plays σ^2_-; and $\frac{\mathsf{l}+\mathsf{h}}{2}$ for the uniform mixed strategy $\frac{1}{2}\mathsf{l} + \frac{1}{2}\mathsf{h}$.

$\mathbf{k = 0}$: MAX prefers nf, which achieves a maxmin value of 0.6, against 0.4 for f; both types of MIN are indifferent between l and h since both yield a minmax value of 1.

$\mathbf{k = 1}$: Against $(\oplus(\Sigma^0_-)) = (\frac{\mathsf{l}+\mathsf{h}}{2}|\frac{\mathsf{l}+\mathsf{h}}{2})$, MAX's best strategy is still nf; however, against $(\oplus(\Sigma^0_+)) = (\mathsf{nf})$, type-1 MIN prefers h which yields a value of 0.

$\mathbf{k = 2}$: Against $(\oplus(\Sigma^1_-), \oplus(\Sigma^0_-))$, MAX now prefers f, which is strictly better than nf against $\oplus(\Sigma^1_-) = \mathsf{h}|\frac{\mathsf{l}+\mathsf{h}}{2}$ since the NBS of MAX at node B judges MIN is more likely to be of type 1 than of type 2 if MIN plays $\mathsf{h}|\frac{\mathsf{l}+\mathsf{h}}{2}$;

$\mathbf{k = 3}$: At level-3, type-1 MIN still prefers h: h and l are equivalent against $\oplus(\Sigma^2_+) = \mathsf{f}$, but h is preferred against $\oplus(\Sigma^1_+) = \mathsf{nf}$; but now type-2 MIN also prefers h!

As it turns out, this recursive reasoning perfectly captures what happened during the bridge deal, where MAX was at level 2 and therefore chose f (rather than the maxmin

strategy nf) while MIN, being of type 2, reasoned at level 3 and used strategy h to defeat MAX.

Admittedly, in the game of Figure 3, MIN's strategy h weakly dominates l, and therefore MIN should never play l. However, this game is an extreme abstraction of the real game, which has a huge number of strategies; it is not at all obvious that h is weakly dominant. In addition, many other real-life examples of recursive reasoning, which we cannot give here for space reasons, yield risky strategies, i.e. those that are neither maxmin nor dominant and as a result could perform worse if the opponent's reasoning level is incorrectly estimated. Indeed, in our example, this is the case for MAX's level-2 strategy f: against MIN of level 1, it indeed performs better (0.7) than the maxmin strategy nf (0.6), but against MIN of level 3 it performs worse (0.4).

8 Conclusion

We have proposed a number of ways to take into account opponent models in games with incomplete information. For each type of opponent model, we have formally defined the maxmin value and proposed an algorithm to compute it. We have also considered the case where the opponent, with some probability, does not follow any model, and the goal is to be robust against any possible adversarial strategy while maximally exploiting the knowledge of opponent models.

As an application, we have proposed a general framework of recursive opponent models. This parametrizable framework can model, by using appropriate intra-level aggregations and best-response functions, a wide range of situations of recursive reasoning, including the possibility that an opponent does not follow any model. Illustrated by an example from the game of Bridge, we have shown how this framework captures real-life strategic reasoning, and how our algorithms can be used for models defined in the literature of economy (Camerer, Ho, and Chong, 2004, for instance).

Two main directions are worth pursuing for future work: To consider games represented compactly (e.g. by game rules) instead of explicitly by their game tree; and to formally define our recursive framework in doxastic logic, which is similar to the notion of rationalizability in epistemic game theory but allows false beliefs (about the others' level, for instance). For the latter direction, the intuition is that level-k strategies can be seen as strategies optimal for an agent with a depth of knowledge of k in the Kripke structure over the players' types. For instance, assuming the players' distributed knowledge of the actual combination of types, it can be shown that under this definition, level-0 strategies are optimal strategies against the best-defence model.

References

Albrecht, S. V., and Stone, P. 2018. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artif. Intell.* 258:66–95.

Bonanno, G. 2018. *Game Theory*. Kindle Direct Publishing.

Bošanský, B.; Kiekintveld, C.; Lisý, V.; and Pěchouček, M. 2014. An exact double-oracle algorithm for zero-sum extensive-form games with imperfect information. *Journal of Artificial Intelligence Research* 829–866.

Camerer, C. F.; Ho, T.-H.; and Chong, J.-K. 2004. A cognitive hierarchy model of games. *The Quarterly Journal of Economics* 119(3):861–898.

de Weerd, H.; Verbrugge, R.; and Verheij, B. 2013. How much does it help to know what she knows you know? an agent-based simulation study. *Artificial Intelligence* 199–200:67–92.

Doshi, P.; Gmytrasiewicz, P. J.; and Durfee, E. H. 2020. Recursively modeling other agents for decision making: A research perspective. *Artif. Intell.* 279.

Frank, I., and Basin, D. A. 2001. A theoretical and empirical investigation of search in imperfect information games. *Theor. Comput. Sci.* 252(1-2):217–256.

Ginsberg, M. L. 2001. GIB: imperfect information in a computationally challenging game. *J. Artif. Intell. Res.* 14:303–358.

Gmytrasiewicz, P. J., and Doshi, P. 2005. A framework for sequential planning in multi-agent settings. *J. Artif. Intell. Res.* 24:49–79.

Iida, H.; Uiterwijk, J. W. H. M.; van den Herik, H. J.; and Herschberg, I. S. 1993. Potential applications of opponent-model search, part 1: The domain of applicability. *J. Int. Comput. Games Assoc.* 16(4):201–208.

Iida, H.; Uiterwijk, J. W. H. M.; van den Herik, H. J.; and Herschberg, I. S. 1994. Potential applications of opponent-model search, part 2: Risks and strategies. *J. Int. Comput. Games Assoc.* 17(1):10–14.

Karpin, F. L. 1977. *Psychological strategy in contract bridge: The techniques of deception and harassment in bidding and play*. Dover Publications.

Koller, D., and Megiddo, N. 1992. The complexity of two-person zero-sum games in extensive form. *Games and Economic Behavior* 4(4):528–552.

Koller, D.; Megiddo, N.; and von Stengel, B. 1996. Efficient computation of equilibria for extensive two-person games. *Games and Economic Behavior* 14(2):247–259.

Kuhn, H. W. 1953. *11. Extensive Games and the Problem of Information*. Princeton: Princeton University Press. 193–216.

Li, J.; Zanuttini, B.; Cazenave, T.; and Ventos, V. 2022. Generalisation of alpha-beta search for AND-OR graphs with partially ordered values. In Raedt, L. D., ed., *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, 4769–4775. ijcai.org.

Maschler, M.; Solan, E.; and Zamir, S. 2020. *Game Theory*. Cambridge University Press, 2 edition.

McMahan, H. B.; Gordon, G. J.; and Blum, A. 2003. Planning in the presence of cost functions controlled by an adversary. In Fawcett, T., and Mishra, N., eds., *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA*, 536–543. AAAI Press.

Nashed, S. B., and Zilberstein, S. 2022. A survey of opponent modeling in adversarial domains. *J. Artif. Intell. Res.* 73:277–327.

Perea, A. 2012. *Epistemic Game Theory: Reasoning and Choice*. Cambridge University Press.

Stahl, D. O., and Wilson, P. W. 1995. On players' models of other players: Theory and experimental evidence. *Games and Economic Behavior* 10(1):218–254.

van Damme, E. 1991. *Stability and Perfection of Nash Equilibria*. Springer Berlin Heidelberg.

von Stengel, B. 1996. Efficient computation of behavior strategies. *Games and Economic Behavior* 14(2):220–246.

Wright, J. R., and Leyton-Brown, K. 2019. Level-0 models for predicting human behavior in games. *J. Artif. Intell. Res.* 64:357–383.

A Proofs

For the following proofs, we need some additional notations. Let ς_+ and ς_- be a arbitrary (pure/mixed/behaviour) strategy of MAX and MIN, respectively. We denote by $u(n, \varsigma_+, \varsigma_-, i)$ the expected value of the leaves reached by ς_+ and ς_- if the game starts at n, MAX and MIN respectively plays the strategies ς_+ and ς_-, and MIN's type is i. In addition, we write

$$\vec{u}(n, \varsigma_+, \varsigma_-) :- \big(u(n, \varsigma_+, \varsigma_-, 1), \ldots, u(n, \varsigma_+, \varsigma_-, t)\big),$$

where t is the number of MIN's types. In particular,

$$u(\varsigma_+, \varsigma_-) = \vec{q} \cdot \vec{u}(r, \varsigma_+, \varsigma_-),$$

where r is the root and $\vec{q}$ is the common prior on MIN's types.

Proposition 2. *For a CGII with root r and a single opponent model ω_-, consider the instantiation of Algorithm 1 where: $V = \mathbb{R}$; for all terminal nodes n, $\mathsf{eval}(n) :- \overrightarrow{\mathsf{nbs}}(n) \cdot \vec{u}(n)$; MAX's operator is $\max$; MIN's operator is $+$. Then it holds that $\underline{v}_+ = \mathsf{val}(r)$, and the algorithm is polynomial-time.*

Proof. First, recall that

$$\underline{v}_+ :- \max_{s_+ \in \Sigma_+^{\mathrm{P}}} u(s_+, \omega_-) = \max_{\sigma_+ \in \Sigma_+^{\mathrm{M}}} u(\sigma_+, \omega_-),$$

where the last equality is due to th linearity of u. Therefore, we only consider pure strategies in the following.

We first observe that by the construction of the NBS, for all nodes n in the game tree and all types i for MIN, the i-th component of $\overrightarrow{\mathsf{nbs}}(n)$ is the probability that n is reached if MIN's type is i, given that MAX plays their unique sequence of moves leading to n and MIN plays according to the OM ω_-. In particular, $\overrightarrow{\mathsf{nbs}}(r) = \vec{q}$, where $\vec{q}$ is the common prior over MIN's types.

In the following, we will establish by induction on the game tree that for all nodes n, its situational value $\mathsf{val}(n)$ satisfies

$$\mathsf{val}(n) = \max_{s_+ \in \Sigma_+^{\mathrm{P}}} \overrightarrow{\mathsf{nbs}}(n) \cdot \vec{u}(n, s_+, \omega_-). \tag{4}$$

First, consider a leaf n. Then

$$\mathsf{val}(n) = \mathsf{eval}(n) = \overrightarrow{\mathsf{nbs}}(n) \cdot \vec{u}(n).$$

Since n is a leaf, by definition of $\vec{u}(n, s_+, \omega_-)$, it trivially holds that $\vec{u}(n, s_+, \omega_-) = \vec{u}(n)$ for all $s_+ \in \Sigma_+^{\mathrm{P}}$. Hence, (4) holds for n.

Now consider an internal node n. Let $C(n)$ be the set of n's children, which is not empty since n is not a leaf. If n is MAX's decision node, then $\mathsf{val}(n) = \max_{n' \in C(n)} \mathsf{val}(n')$. By the induction hypothesis,

$$\mathsf{val}(n) = \max_{n' \in C(n)} \max_{s_+ \in \Sigma_+^{\mathrm{P}}} \overrightarrow{\mathsf{nbs}}(n') \cdot \vec{u}(n', s_+, \omega_-),$$

from which we deduce that (4) holds for n since $\overrightarrow{\mathsf{nbs}}(n') = \overrightarrow{\mathsf{nbs}}(n)$ for $n' \in C(n)$, and

$$\max_{s_+ \in \Sigma_+^{\mathrm{P}}} \vec{u}(n, s_+, \omega_-) = \max_{n' \in C(n)} \max_{s_+ \in \Sigma_+^{\mathrm{P}}} \vec{u}(n', s_+, \omega_-).$$

If n is MIN's decision node, then we have $\mathsf{val}(n) = \sum_{n' \in C(n)} v(n')$. By the induction hypothesis,

$$\mathsf{val}(n) = \sum_{n' \in C(n)} \max_{s_+ \in \Sigma_+^{\mathrm{P}}} \overrightarrow{\mathsf{nbs}}(n') \cdot \vec{u}(n', s_+, \omega_-).$$

Now since the n's are different nodes in the game tree, MAX can apply any combination of strategies in the subtree rooted at these nodes. Hence, the sum over n' can be exchanged with the maximum, which yields

$$\mathsf{val}(n) = \max_{s_+ \in \Sigma_+^{\mathrm{P}}} \sum_{n' \in C(n)} \overrightarrow{\mathsf{nbs}}(n') \cdot \vec{u}(n', s_+, \omega_-).$$

Finally, by definition of u and $\overrightarrow{\mathsf{nbs}}$,

$$\sum_{n' \in C(n)} \overrightarrow{\mathsf{nbs}}(n') \cdot \vec{u}(n', s_+, \omega_-) = \overrightarrow{\mathsf{nbs}}(n) \cdot \vec{u}(n, s_+, \omega_-),$$

since n is MIN's node, and MIN chooses n' according to the strategy ω_-. Therefore, (4) holds for n, which concludes the induction.

Applying (4) to the root, we have

$$\mathsf{val}(r) = \max_{s_+ \in \Sigma_+^{\mathrm{P}}} \overrightarrow{\mathsf{nbs}}(r) \cdot \vec{u}(r, s_+, \omega_-) = \max_{s_+ \in \Sigma_+^{\mathrm{P}}} u(s_+, \omega_-),$$

since $\overrightarrow{\mathsf{nbs}}(r) = \vec{q}$ and $u(s_+, \omega_-) = \vec{q} \cdot \vec{u}(r, s_+, \omega_-)$. Therefore, the situational value of the root is the maxmin value. □