Computing Maxmin Strategies in Extensive-form Zero-sum Games with Imperfect Recall

Branislav Bošanský, Jiří Čermák, Karel Horák and Michal Pěchouček
Department of Computer Science, Czech Technical University in Prague, Prague, Czech Republic

Keywords: Game Theory, Imperfect Recall, Maxmin Strategies.

Abstract: Extensive-form games with imperfect recall are an important game-theoretic model that allows a compact representation of strategies in dynamic strategic interactions. Practical use of imperfect recall games is limited due to negative theoretical results: a Nash equilibrium does not have to exist, computing maxmin strategies is NP-hard, and they may require irrational numbers. We present the first algorithm for approximating maxmin strategies in two-player zero-sum imperfect recall games without absentmindedness. We modify the well-known sequence-form linear program to model strategies in imperfect recall games resulting in a bilinear program and use a recent technique to approximate the bilinear terms. Our main algorithm is a branch-and-bound search that provably reaches the desired approximation after an exponential number of steps in the size of the game. Experimental evaluation shows that the proposed algorithm can approximate maxmin strategies of randomly generated imperfect recall games of sizes beyond toy-problems within few minutes.

1 INTRODUCTION

The extensive form is a well-known representation of dynamic strategic interactions that evolve in time. Games in the extensive form (extensive-form games; EFGs) are visualized as game trees, where nodes correspond to states of the game and edges to actions executed by players. This representation is general enough to model stochastic events and imperfect information when players are unable to distinguish among several states. Recent years have seen advancements in algorithms for computing solution concepts in large zero-sum extensive-form games (e.g., solving heads-up limit texas hold'em poker (Bowling et al., 2015)).

Most of the algorithms for finding optimal strategies in EFGs assume that players remember all information gained during the course of the game (Zinkevich et al., 2008; Hoda et al., 2010; Bošanský et al., 2014). This assumption is known as *perfect recall* and has a significant impact on theoretical properties of finding optimal strategies in EFGs. Namely, there is an equivalence between two types of strategies in perfect recall games – *mixed strategies* (probability distributions over pure strategies[1]) and *behavioral strategies* (probability distributions over actions in each decision point) (Kuhn, 1953). This equivalence guarantees that a Nash equilibrium (NE) exists in behavioral strategies in perfect recall games (the proof of the existence of NE deals with mixed strategies only (Nash, 1950)) and it is exploited by algorithms for computing a NE in zero-sum EFGs with perfect recall – the well-known sequence-form linear program (Koller et al., 1996; von Stengel, 1996).

The caveat of perfect recall is that remembering all information increases the number of decision points (and consequently the size of a behavioral strategy) exponentially with the number of moves in the game. One possibility for tackling the size of perfect recall EFGs is to create an abstracted game where certain decision points are merged together, solve this abstracted game, and then translate the strategy from the abstracted game into the original game (e.g., see (Gilpin and Sandholm, 2007; Kroer and Sandholm, 2014; Kroer and Sandholm, 2016)). However, devising abstracted games that have *imperfect recall* is desirable due to the reduced size. One then must compute behavioral strategies in order to exploit the reduced size (mixed strategies already operate over an exponentially large set of pure strategies).

Solving imperfect recall games has several fundamental problems. The best known game-theoretic solution concept, a Nash equilibrium (NE), does not

[1] A pure strategy in an EFG is an assignment of an action to play in each decision point.

Bosansky B., Cermak J., Horak K. and Pechoucek M.
Computing Maxmin Strategies in Extensive-form Zero-sum Games with Imperfect Recall.
DOI: 10.5220/0006121200630074
In *Proceedings of the 9th International Conference on Agents and Artificial Intelligence (ICAART 2017)*, pages 63-74
ISBN: 978-989-758-220-2
Copyright © 2017 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

have to exist even in zero-sum games (see (Wichardt, 2008) for a simple example) and standard algorithms (e.g., a Counterfactual Regret Minimization (CFR) (Zinkevich et al., 2008)) can converge to incorrect strategies (see Example 1). Therefore, we focus on finding a strategy that guarantees the best possible expected outcome for a player – *a maxmin strategy*. However, computing a maxmin strategy is NP-hard and such strategies may require irrational numbers even when the input uses only rational numbers (Koller and Megiddo, 1992).

Existing works avoid these negative results by creating very specific abstracted games so that perfect recall algorithms are still applicable. One example is a subset of imperfect recall games called *(skewed) well-formed games*, motivated by the poker domain, in which the standard perfect-recall algorithms (e.g., CFR) are still guaranteed to find an approximate Nash behavioral strategy (Lanctot et al., 2012; Kroer and Sandholm, 2016). The restrictions on games to form (skewed) well-formed games are, however, rather strict and can prevent us from creating sufficiently small abstracted games. To fully explore the possibilities of exploiting the concept of abstractions and/or other compactly represented dynamic games (e.g., Multi-Agent Influence Diagrams (Koller and Milch, 2003)), a new algorithm for solving imperfect recall games is required.

1.1 Our Contribution

We advance the state of the art and provide the first approximate algorithm for computing maxmin strategies in imperfect recall games (since maxmin strategies might require irrational numbers (Koller and Megiddo, 1992), finding exact maxmin has fundamental difficulties). We assume imperfect recall games with no *absentmindedness*, which means that each decision point in the game can be visited at most once during the course of the game and it is arguably a natural assumption in finite games (see, e.g., (Piccione and Rubinstein, 1997) for a detailed discussion). The main goal of our approach is to find behavioral strategies that maximize the expected outcome of player 1 against an opponent that minimizes the outcome. We base our formulation on the sequence-form linear program for perfect recall games (Koller et al., 1996; von Stengel, 1996) and we extend it with bilinear constraints necessary for the correct representation of strategies of player 1 in imperfect recall games. We approximate the bilinear terms using recent Multiparametric Disaggregation Technique (MDT) (Kolodziej et al., 2013) and provide a mixed-integer linear program (MILP) for approximating maxmin strategies. Finally, we consider a linear relaxation of the MILP and propose a branch-and-bound algorithm that (1) repeatedly solves this linear relaxation and (2) tightens the constraints that approximate bilinear terms as well as relaxed binary variables from the MILP. We show that the branch-and-bound algorithm ends after exponentially many steps while guaranteeing the desired precision.

Our algorithm approximates maxmin strategies for player 1 having generic imperfect recall without absentmindedness and we give two variants of the algorithm depending on the type of imperfect recall of the opponent. If the opponent, player 2, has either a perfect recall or so-called *A-loss recall* (Kaneko and Kline, 1995; Kline, 2002), the linear program solved by the branch-and-bound algorithm has a polynomial size in the size of the game. If player 2 has a generic imperfect recall without absentmindedness, the linear program solved by the branch-and-bound algorithm can be exponentially large.

We provide a short experimental evaluation to demonstrate that our algorithm can solve games far beyond the size of toy problems. Randomly generated imperfect recall games with up to $5 \cdot 10^3$ states can be typically solved within few minutes.

All the technical proofs can be found in the appendix or in the full version of this paper.

2 TECHNICAL PRELIMINARIES

Before describing our algorithm we define extensive-form games, different types of recall, and describe the approximation technique for the bilinear terms.

A two-player extensive-form game (EFG) is a tuple $G = (\mathcal{N}, \mathcal{H}, \mathcal{Z}, \mathcal{A}, u, \mathcal{C}, \mathcal{I})$. $\mathcal{N} = \{1,2\}$ is a set of players, by i we refer to one of the players, and by $-i$ to his opponent. $\mathcal{H}$ denotes a finite set of *histories* of actions taken by all players and chance from the root of the game. Each history corresponds to a *node* in the game tree; hence, we use terms history and node interchangeably. We say that h is a *prefix* of h' ($h \sqsubseteq h'$) if h lies on a path from the root of the game tree to h'. $\mathcal{Z} \subseteq \mathcal{H}$ is the set of *terminal states* of the game. $\mathcal{A}$ denotes the set of all actions. An ordered list of all actions of player i from root to h is referred to as a *sequence*, $\sigma_i = \mathsf{seq}_i(h)$, Σ_i is a set of all sequences of i. For each $z \in \mathcal{Z}$ we define a *utility function* $u_i : \mathcal{Z} \to \mathbb{R}$ for each player i ($u_i(z) = -u_{-i}(z)$ in zero-sum games). The chance player selects actions based on a fixed probability distribution known to all players. Function $\mathcal{C} : \mathcal{H} \to [0,1]$ is the probability of reaching h due to chance.

Imperfect observation of player i is modeled via *information sets* $\mathcal{I}_i$ that form a partition over $h \in \mathcal{H}$ where i takes action. Player i cannot distinguish between nodes in any $I_i \in \mathcal{I}_i$. $\mathcal{A}(I_i)$ denotes actions available in each $h \in I_i$. The action a uniquely identifies the information set where it is available. We use $\mathsf{seq}_i(I_i)$ as a set of all sequences of player i leading to I_i. Finally, we use $\mathsf{inf}_i(\sigma_i)$ to be a set of all information sets to which sequence σ_i leads.

A *behavioral strategy* $\beta_i \in \mathcal{B}_i$ is a probability distribution over actions in each information set $I \in \mathcal{I}_i$. We use $u_i(\beta) = u_i(\beta_i, \beta_{-i})$ for the expected outcome of the game for i when players follow β. A *best response* of player i against β_{-i} is a strategy $\beta_i^{BR} \in BR_i(\beta_{-i})$, where $u_i(\beta_i^{BR}, \beta_{-i}) \geq u_i(\beta_i', \beta_{-i})$ for all $\beta_i' \in \mathcal{B}_i$. $\beta_i(I,a)$ is the probability of playing a in I, $\beta(h)$ denotes the probability that h is reached when both players play according to β and due to chance.

We say that β_i and β_i' are *realization equivalent* if for any β_{-i}, $\forall z \in \mathcal{Z}\ \beta(z) = \beta'(z)$, where $\beta = (\beta_i, \beta_{-i})$ and $\beta' = (\beta_i', \beta_{-i})$.

A *maxmin strategy* β_i^* is defined as $\beta_i^* = \arg\max_{\beta_i \in \mathcal{B}_i} \min_{\beta_{-i} \in \mathcal{B}_{-i}} u_i(\beta_i, \beta_{-i})$. Note that when a Nash equilibrium in behavioral strategies exists in a two-player zero-sum imperfect recall game then β_i^* is a Nash equilibrium strategy for i.

2.1 Types of Recall

We now briefly define types of recall in EFGs and state several lemmas and observations about characteristics of strategies in imperfect recall EFGs that are later exploited by our algorithm.

In *perfect recall*, all players remember the history of their own actions and all information gained during the course of the game. As a consequence, all nodes in any information set I_i have the same sequence for player i. If the assumption of perfect recall does not hold, we talk about games with *imperfect recall*. In imperfect recall games, mixed and behavioral strategies are not comparable (Kuhn, 1953). However, in games without *absentmindedness* (AM) where each information set is encountered at most once during the course of the game, the following observation allow us to consider only pure best responses of the opponent when computing maxmin strategies:

Lemma 1. *Let G be an imperfect recall game without AM and β_1 strategy of player 1. There exists an* ex ante *(i.e., when evaluating only the expected value of the strategy) pure behavioral best response of player 2.*

The proof is in the full version of the paper.

This lemma is applied when a mathematical program for computing maxmin strategies is formulated – strategies of player 2 can be considered as constraints using pure best responses. Note that this is not true in general imperfect recall games – in games with AM, an ex ante best response may need to be randomized (e.g., in the game with absentminded driver (Piccione and Rubinstein, 1997)).

A disadvantage of using pure best responses as constraints for the minimizing player is that there are exponentially many pure best responses in the size of the game. In perfect recall games, this can be avoided by formulating best-response constraints such that the opponent is playing the best action in each information set. However, this type of response, termed *time consistent strategy* (Kline, 2002), does not have to be an ex ante best response in general imperfect recall games (see (Kline, 2002) for an example). A class of imperfect recall games where it is sufficient to consider only time consistent strategies when computing best responses was termed as *A-loss* recall games (Kaneko and Kline, 1995; Kline, 2002).

Definition 1. *Player i has* A-loss recall *if and only if for every $I \in \mathcal{I}_i$ and nodes $h, h' \in I$ it holds either (1) $\mathsf{seq}_i(h) = \mathsf{seq}_i(h')$, or (2) $\exists I' \in \mathcal{I}_i$ and two distinct actions $a, a' \in \mathcal{A}_i(I'), a \neq a'$ such that $a \in \mathsf{seq}_i(h) \wedge a' \in \mathsf{seq}_i(h')$.*

Condition (1) in the definition says that if player i has perfect recall then she also has A-loss recall. Condition (2) requires that each loss of memory of A-loss recall player can be traced back to some loss of memory of the player's own previous actions.

The equivalence between time consistent strategies and ex ante best responses allows us to simplify the best responses of player 2 in case she has A-loss recall. Formally, it is sufficient to consider best responses that correspond to the best response in a coarsest perfect-recall refinement of the imperfect recall game when computing best response for a player with A-loss recall. By a *coarsest perfect recall refinement* of an imperfect recall game G we define a perfect recall game G' where we split the imperfect recall information sets to biggest subsets still fulfilling the perfect recall.

Definition 2. The coarsest perfect recall refinement G' *of the imperfect recall game $G = \{\mathcal{N}, \mathcal{H}, \mathcal{Z}, \mathcal{A}, u, \mathcal{C}, \mathcal{I}\}$ is a tuple $\{\mathcal{N}, \mathcal{H}, \mathcal{Z}, \mathcal{A}', u, \mathcal{C}, \mathcal{I}'\}$, where $\forall i \in \mathcal{N}\ \forall I_i \in \mathcal{I}_i$ $H(I_i)$ partitions information set I_i such that $H(I_i) = \{H_1, ..., H_n\}$ is a disjoint partition of all $h \in I_i$, where $\bigcup_{j=1}^{n} H_j = I_i$ and $\forall H_j \in H(I_i)\ \forall h_k, h_l \in H_j : \mathsf{seq}_i(h_k) = \mathsf{seq}_i(h_l)$ and $\forall h_k \in H_k, h_l \in H_l : H_k \cap H_l = \emptyset \Rightarrow \mathsf{seq}_i(h_k) \neq \mathsf{seq}_i(h_l)$. Each set from $H(I_i)$ corresponds to an information set $I_i' \in \mathcal{I}_i'$. Moreover, $\mathcal{A}'$ is a modification of $\mathcal{A}$*

guaranteeing $\forall I \in \mathcal{I}' \ \forall h_k, h_l \in I \ \mathcal{A}'(h_k) = \mathcal{A}'(h_l)$, *while for all distinct* $I^k, I^l \in \mathcal{I}' \ \mathcal{A}(I^k) \neq \mathcal{A}(I^l)$.

Note that we can restrict the coarsest perfect recall refinement only for i by splitting only information sets of i (information sets of $-i$ remain unchanged). Finally, we assume that there is a mapping between actions from the coarsest perfect recall refinement $\mathcal{A}'$ and actions in the original game $\mathcal{A}$ so that we can identify to which actions from $\mathcal{A}'$ an original action $a \in \mathcal{A}$ maps. We assume this mapping to be implicit since it is clear from the context.

Lemma 2. *Let G be an imperfect recall game where player 2 has A-loss recall and* β_1 *is a strategy of player 1, and let* G' *be the coarsest perfect recall refinement of G for player 2. Let* β_2' *be a pure best response in* G' *and let* β_2 *be a realization equivalent behavioral strategy in G, then* β_2 *is a pure best response to* β_1 *in G.*

The proof is in the full version of the paper.

Note that the *NP*-hardness proof of computing maxmin strategies due to Koller (Koller and Megiddo, 1992) still applies, since we assume the maximizing player to have generic imperfect recall and the reduction provided by Koller results in a game where the maximizing player has generic imperfect recall while the minimizing player has perfect recall, which is a special case of both settings assumed in our paper.

Finally, let us show that CFR cannot be applied in these settings. This is caused by the fact that CFR iteratively minimizes per information set regret terms (counterfactual regrets). Since in perfect recall games the sum of counterfactual regrets provides an upper bound on the external regret, this minimization is guaranteed to converge to a strategy profile with 0 external regret. In imperfect recall games, however, the sum of counterfactual regrets no longer forms an upper bound on the external regret (Lanctot et al., 2012), and the minimization of these regret terms can, therefore, lead to a strategy profile with a non-zero external regret.

Example 1: Consider the A-loss recall game in Figure 1. When setting the $x > 2$, one of the strategy profiles with zero counterfactual regret (and therefore a profile to which CFR can converge) is mixing uniformly between both a, b and g, h, while player 2 plays d, e deterministically. By setting the utility x as some large number, this strategy profile can have expected utility arbitrarily worse than the maxmin value -1. The reason is the presence of the conflicting outcomes for some action in an imperfect recall information set that cannot be generally avoided, or easily detected in imperfect recall games.

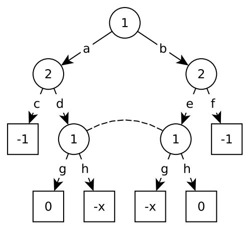

Figure 1: An A-loss recall game where CFR finds a strategy with the expected utility arbitrarily distant from the maxmin value.

2.2 Approximating Bilinear Terms

The final technical tool that we use in our algorithm is the approximation of bilinear terms by Multiparametric Disaggregation Technique (MDT) (Kolodziej et al., 2013) for approximating bilinear constraints. The main idea of the approximation is to use a digit-wise discretization of one of the variables from a bilinear term. The main advantage of this approximation is a low number of newly introduced integer variables and an experimentally confirmed speed-up over the standard technique of piecewise McCormick envelopes (Kolodziej et al., 2013).

$$\sum_{k=0}^{9} w_{k,\ell} = 1 \qquad \ell \in \mathbb{Z} \tag{1a}$$

$$w_{k,\ell} \in \{0,1\} \tag{1b}$$

$$\sum_{\ell \in \mathbb{Z}} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot w_{k,\ell} = b \tag{1c}$$

$$c^L \cdot w_{k,\ell} \leq \hat{c}_{k,\ell} \leq c^U \cdot w_{k,\ell} \qquad \forall \ell \in \mathbb{Z}, \forall k \in 0..9 \tag{1d}$$

$$\sum_{k=0}^{9} \hat{c}_{k,\ell} = c \qquad \forall \ell \in \mathbb{Z} \tag{1e}$$

$$\sum_{\ell \in \mathbb{Z}} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{c}_{k,\ell} = a \tag{1f}$$

Let $a = bc$ be a bilinear term. MDT discretizes variable b and introduces new binary variables $w_{k,l}$ that indicate whether the digit on ℓ-th position is k. Constraint (1a) ensures that for each position ℓ there is exactly one digit chosen. All digits must sum to b (Constraint (1c)). Next, we introduce variables $\hat{c}_{k,\ell}$ that are equal to c for such k and ℓ where $w_{k,l} = 1$, and $\hat{c}_{k,\ell} = 0$ otherwise. c^L and c^U are bounds on the value of variable c. The value of a is given by Constraint (1f).

This is an exact formulation that requires infinite sums and an infinite number of constraints. However, by restricting the set of all possible positions ℓ to a finite set $\{P_L, \ldots, P_U\}$ we get a lower bound approximation. Following the approach in (Kolodziej et al., 2013) we can extend the lower bound formulation to compute an upper bound:

$$\text{Constraints } (1a), (1d), (1e)$$

$$\sum_{\ell \in \{P_L,\dots,P_U\}} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot w_{k,\ell} + \Delta b = b \tag{2a}$$

$$0 \le \Delta b \le 10^{P_L} \tag{2b}$$

$$\sum_{\ell \in \{P_L,\dots,P_U\}} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{c}_{k,\ell} + \Delta a = a \tag{2c}$$

$$c^L \cdot \Delta b \le \Delta a \le c^U \cdot \Delta b \tag{2d}$$

$$\left(c - c^U\right) \cdot 10^{P_L} + c^U \cdot \Delta b \le \Delta a \tag{2e}$$

$$\left(c - c^L\right) \cdot 10^{P_L} + c^L \cdot \Delta b \ge \Delta a \tag{2f}$$

Here, Δb is assigned to every discretized variable b allowing it to take up the value between two discretization points created due to the minimal value of ℓ (Constraints (2a)–(2b)). Similarly, we allow the product variable a to be increased with variable $\Delta a = \Delta b \cdot c$. To approximate the product of the delta variables, we use the McCormick envelope defined by Constraints (2d)–(2f).

3 MATHEMATICAL PROGRAMS FOR APPROXIMATING MAXMIN STRATEGIES

We now state the mathematical programs for approximating maxmin strategies. The main idea is to add bilinear constraints into the sequence form LP to restrict to imperfect recall strategies. We formulate an exact bilinear program, followed by the approximation of bilinear terms using MDT.

3.1 Exact Bilinear Sequence Form Against A-loss Recall Opponent

$$\max_{x,r,v} v(root, \emptyset) \tag{3a}$$

$$s.t. \quad r(\emptyset) = 1 \tag{3b}$$

$$0 \le r(\sigma) \le 1 \qquad \forall \sigma \in \Sigma_1 \tag{3c}$$

$$\sum_{a \in \mathcal{A}(I)} r(\sigma a) = r(\sigma) \qquad \forall \sigma \in \Sigma_1, \forall I \in \mathsf{inf}_1(\sigma_1) \tag{3d}$$

$$\sum_{a \in \mathcal{A}(I)} x(a) = 1 \qquad \forall I \in \mathcal{I}_1^{IR} \tag{3e}$$

$$0 \le x(a) \le 1 \qquad \forall I \in \mathcal{I}_1^{IR}, \forall a \in \mathcal{A}(I) \tag{3f}$$

$$r(\sigma) \cdot x(a) = r(\sigma a) \qquad \forall I \in \mathcal{I}_1^{IR}, \forall a \in \mathcal{A}(I), \forall \sigma \in \mathsf{seq}_1(I) \tag{3g}$$

$$\sum_{\sigma_1 \in \Sigma_1} g(\sigma_1, \sigma_2 a) r_1(\sigma_1) + \sum_{I' \in \mathsf{inf}_2(\sigma_2 a)} v(I', \sigma_2 a) \ge v(I, \sigma_2) \qquad \forall I \in \mathcal{I}_2, \forall \sigma_2 \in \mathsf{seq}_2(I), \forall a \in \mathcal{A}(I) \tag{3h}$$

Constraints (3a)–(3h) represent a bilinear reformulation of the sequence-form LP due to (von Stengel, 1996) applied to the information set structure of an imperfect recall game G. The objective of player 1 is to find a strategy that maximizes the expected utility of the game. The strategy is represented by variables r that assign probability to a sequence: $r(\sigma_1)$ is the probability that $\sigma_1 \in \Sigma_1$ will be played assuming that information sets, in which actions of sequence σ_1 are applicable, are reached due to player 2. Probabilities r must satisfy so-called network flow Constraints (3c)–(3d). Finally, a strategy of player 1 is constrained by the best-responding opponent that selects an action minimizing the expected value in each $I \in \mathcal{I}_2$ and for each $\sigma_2 \in \mathsf{seq}_2(I)$ that was used to reach I (Constraint (3h)). These constraints ensure that the opponent plays the best response in the coarsest perfect recall refinement of G and thus also in G due to Lemma 2. The expected utility for each action is a sum of the expected utility values from immediately reachable information sets I' and from immediately reachable leafs. For the latter we use generalized utility function $g : \Sigma_1 \times \Sigma_2 \to \mathbb{R}$ defined as $g(\sigma_1, \sigma_2) = \sum_{z \in Z | \mathsf{seq}_1(z) = \sigma_1 \wedge \mathsf{seq}_2(z) = \sigma_2} u(z) \mathcal{C}(z)$.

In imperfect recall games multiple σ_i can lead to some imperfect recall information set $I_i \in \mathcal{I}_i^{IR} \subseteq \mathcal{I}_i$; hence, realization plans over sequences do not have to induce the same behavioral strategy for I_i. Therefore, for each $I_i \in \mathcal{I}_i^{IR}$ we define behavioral strategy $x(a)$ for each $a \in \mathcal{A}(I_i)$ (Constraints (3e)–(3f)). To ensure that the realization probabilities induce the same behavioral strategy in I_i, we add bilinear constraint $r(\sigma_i a) = x(a) \cdot r(\sigma_i)$ (Constraint (3g)).

3.1.1 Player 2 without A-Loss Recall

If player 2 does not have A-loss recall, the mathematical program must use each pure best response of player 2 $\pi_2 \in \Pi_2$ as a constraint as follows:

$$\max_{x,r,v} v(root) \tag{4a}$$

$$\text{Constraints (3b)–(3f)}$$

$$\sum_{z \in Z \,|\, \pi_2(z) = 1} u(z) \mathcal{C}(z) r(\mathsf{seq}_1(z)) \ge v(root) \qquad \forall \pi_2 \in \Pi_2 \tag{4b}$$

Since the modification does not change the parts of the program related to the approximation of strategies of player 1, all the following approximation methods, theorems, and the branch-and-bound algorithm are applicable for general imperfect recall games without absentminded players.

3.2 Upper Bound MILP Approximation

The upper bound formulation of the bilinear program follows the MDT example and uses ideas similar to Section 2.2. In accord with the MDT, we represent every variable $x(a)$ using a finite number of digits. Binary variables $w_{k,\ell}^{I_1,a}$ correspond to $w_{k,\ell}$ variables from the example shown in Section 2.2 and are used for the digit-wise discretization of $x(a)$. Finally, $\hat{r}(\sigma_1)_{k,\ell}^{a}$ correspond to $\hat{c}_{k,\ell}$ variables used to discretize the bilinear term $r(\sigma_1 a)$. In order to allow variable $x(a)$ to attain an arbitrary value from $[0,1]$ interval using a finite number of digits of precision, we add an additional real variable $0 \leq \Delta x(a) \leq 10^{-P}$ that can span the gap between two adjacent discretization points. Constraints (5d) and (5e) describe this loosening. Variables $\Delta x(a)$ also have to be propagated to bilinear terms $r(\sigma_1) \cdot x(a)$ involving $x(a)$. We cannot represent the product $\Delta r(\sigma_1 a) = r(\sigma_1) \cdot \Delta x(a)$ exactly and therefore we give bounds based on the McCormick envelope (Constraints (5i)–(5j)).

$$\max_{x,r,v} v(root, \emptyset) \tag{5a}$$

s.t. Constraints (3b) - (3f) , (3h)

$$w_{k,\ell}^{I,a} \in \{0,1\} \qquad \forall I \in \mathcal{I}_1^{IR}, \forall a \in \mathcal{A}(I), \forall k \in 0..9, \forall \ell \in -P..0 \tag{5b}$$

$$\sum_{k=0}^{9} w_{k,\ell}^{I,a} = 1 \qquad \forall I \in \mathcal{I}_1^{IR}, \forall a \in \mathcal{A}(I), \forall \ell \in -P..0 \tag{5c}$$

$$\sum_{\ell=-P}^{0} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot w_{k,\ell}^{I,a} + \Delta x(a) = x(a) \qquad \forall I \in \mathcal{I}_1^{IR}, \forall a \in \mathcal{A}(I) \tag{5d}$$

$$0 \leq \Delta x(a) \leq 10^{-P} \qquad \forall I \in \mathcal{I}_1^{IR}, \forall a \in \mathcal{A}(I) \tag{5e}$$

$$0 \leq \hat{r}(\sigma)_{k,\ell}^{a} \leq w_{k,\ell}^{I,a} \qquad \forall I \in \mathcal{I}_1^{IR}, \forall a \in \mathcal{A}(I), \forall \sigma \in \mathsf{seq}_1(I), \forall \ell \in -P..0 \tag{5f}$$

$$\sum_{k=0}^{9} \hat{r}(\sigma)_{k,\ell}^{a} = r(\sigma) \qquad \forall I \in \mathcal{I}_1^{IR}, \forall \sigma \in \mathsf{seq}_1(I), \forall \ell \in -P..0 \tag{5g}$$

$$\sum_{\ell=-P}^{0} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{r}(\sigma)_{k,\ell}^{a} + \Delta r(\sigma a) = r(\sigma a) \qquad \forall I \in \mathcal{I}_1^{IR}, \forall a \in \mathcal{A}(I), \forall \sigma \in \mathsf{seq}_1(I) \tag{5h}$$

$$(r(\sigma) - 1) \cdot 10^{-P} + \Delta x(a) \leq \Delta r(\sigma a) \leq 10^{-P} \cdot r(\sigma) \qquad \forall I \in \mathcal{I}_1^{IR}, \forall a \in \mathcal{A}(I), \forall \sigma \in \mathsf{seq}_1(I) \tag{5i}$$

$$0 \leq \Delta r(\sigma a) \leq \Delta x(a) \qquad \forall I \in \mathcal{I}_1^{IR}, \forall \sigma \in \mathsf{seq}_1(I), \forall a \in \mathcal{A}(I) \tag{5j}$$

Due to this loose representation of $\Delta r(\sigma_1 a)$, the reformulation of bilinear terms is no longer exact and this MILP therefore yields an upper bound of the bilinear sequence form program (3). Note that the MILP has both the number of variables and the number of constraints bounded by $O(|\mathcal{I}| \cdot |\Sigma| \cdot P)$, where $|\Sigma|$ is the number of sequences of both players. The number of binary variables is equal to $10 \cdot |\mathcal{I}_1^{IR}| \cdot \mathcal{A}_1^{max} \cdot P$, where $\mathcal{A}_1^{max} = \max_{I \in \mathcal{I}_1} |\mathcal{A}_1(I)|$.

3.3 Theoretical Analysis of the Upper Bound MILP

The variables $\Delta x(a)$ and $\Delta r(\sigma)$ ensure that the optimal value of the MILP is an upper bound on the value of the bilinear program. The drawback is that the realization probabilities do not have to induce a valid strategy in the imperfect recall game G, i.e. if σ_1, σ_2 are two sequences leading to an imperfect recall information set $I_1 \in \mathcal{I}_1^{IR}$ where action $a \in \mathcal{A}(I_1)$ can be played, $r(\sigma_1 a)/r(\sigma_1)$ need not equal $r(\sigma_2 a)/r(\sigma_2)$. We will show that it is possible to create a valid strategy in G which decreases the value by at most ε, while deriving bound on this ε.

Let $\beta^1(I_1), \ldots, \beta^k(I_1)$ be behavioral strategies in the imperfect recall information set $I_1 \in \mathcal{I}_1^{IR}$ corresponding to realization probabilities of continuations of sequences $\sigma^1, \ldots, \sigma^k$ leading to I_1. These probability distributions can be obtained from the realization plan as $\beta^j(I_1, a) = r(\sigma^j a)/r(\sigma^j)$ for $\sigma^j \in \mathsf{seq}_1(I_1)$ and $a \in \mathcal{A}(I_1)$. We will omit the information set and use $\beta(a)$ whenever it is clear from the context. If the imperfect recall is violated in I_1, $\beta^j(a)$ may not be equal to $\beta^l(a)$ for some j, l and action $a \in \mathcal{A}(I_1)$.

Proposition 1. *It is always possible to construct a strategy $\beta(I_1)$ such that $\|\beta(I_1) - \beta^j(I_1)\|_1 \leq |\mathcal{A}(I_1)| \cdot 10^{-P}$ for every j.*[2]

We now connect the distance of a corrected strategy $\beta(I_1)$ from a set of behavioral strategies $\beta^1(I_1), \ldots, \beta^k(I_1)$ in $I_1 \in \mathcal{I}_1^{IR}$ to the expected value of the strategy.

Theorem 1. *The error of the Upper Bound MILP is bounded by*

$$\varepsilon = 10^{-P} \cdot d \cdot \mathcal{A}_1^{max} \cdot \frac{v_{max}(\emptyset) - v_{min}(\emptyset)}{2},$$

where d is the maximum number of player 1's imperfect recall information sets encountered on a path from the root to a terminal node, $\mathcal{A}_1^{max} = \max_{I_1 \in \mathcal{I}_1^{IR}} |\mathcal{A}(I_1)|$ is the branching factor and $v_{min}(\emptyset)$, $v_{max}(\emptyset)$ are the lowest and highest utilities for player 1 in the whole game, respectively.

The idea of the proof is to bound the error in every $I_1 \in \mathcal{I}_1^{IR}$ and propagate the error in a bottom-up fashion.

[2]The L1 norm is taken as $\|x_1 - x_2\|_1 = \sum_{a \in \mathcal{A}(I_1)} |x_1(a) - x_2(a)|$

The error of Upper Bound MILP is bounded if the precision of all approximations of bilinear terms is P. However, we can increase the precision for each term separately and thus design the following iterative algorithm (termed simply as MILP in the experiments): (1) start with the precision set to 0 for all bilinear terms, (2) for each approximation of a bilinear term calculate the current error contribution (the difference between $\Delta r(\sigma_1 a)$ and $r(\sigma_1)\Delta x(a)$ multiplied by the expected utility) and increase the precision only for the term that contributes to the overall error the most. Once the term with maximal error has already reached maximal precision P, Theorem 1 guarantees us that we are ε close to the optimal solution. Our algorithm in the following section simultaneously increases the precision for approximating bilinear terms together with searching for optimal values for binary variables.

4 BRANCH-AND-BOUND ALGORITHM

We now introduce a branch-and-bound (BNB) search for approximating maxmin strategies, that exploits the observation below and thus improves the performance compared to the previous MILP formulations. Additionally, we provide bounds on the overall runtime as a function of the desired precision.

The BNB algorithm works on the linear relaxation of the Upper Bound MILP and searches the BNB tree in the best first search manner. In every node n, the algorithm solves the relaxed LP corresponding to node n, heuristically selects the information set I and action a contributing to the current approximation error the most, and creates successors of n by restricting the probability $\beta_1(I,a)$ that a is played in I. The algorithm adds new constraints to LP depending on the value of $\beta_1(I,a)$ by constraining (and/or introducing new) variables $w_{k,l}^{I_1,a}$ and creating successors of the BNB node in the search tree. Note that $w_{k,l}^{I_1,a}$ variables correspond to binary variables in the MILP formulation. This way, the algorithm simultaneously searches for the optimal approximation of bilinear terms as well as the assignment for binary variables. The algorithm terminates when ε-optimal strategy is found (using the difference of the global upper bound and the lower bound computed as described in Observation 1 below).

Observation 1. Even if the current assignment to variables $w_{k,\ell}^{I_1,a}$ is not feasible (they are not set to binary values), the realization plan produced is valid in the perfect recall refinement. We can fix it in the sense of Proposition 1 and use it to estimate the lower bound for the BNB subtree rooted in the current node without a complete assignment of all $w_{k,\ell}^{I_1,a}$ variables to either 0 or 1.

Algorithm 1: BNB algorithm.

```
input      : Initial LP relaxation $LP_0$ of Upper
             Bound MILP using a $P = 0$
             discretization
output     : $\varepsilon$-optimal strategy for a player having
             imperfect recall
parameters: Bound on maximum error $\varepsilon$, precision
             bounds for $x(a)$ variables $P_{max}(I_1,a)$

1  fringe ← {CreateNode($LP_0$)}
2  opt ← (nil, $-\infty$, $\infty$)
3  while fringe $\neq \emptyset$ do
4      $(LP, lb, ub) \leftarrow \arg\max_{n \in \text{fringe}} n$.ub
5      fringe ← fringe $\setminus (LP, lb, ub)$
6      if opt.lb $\geq n$.ub then
7          return ReconstructStrategy(opt)
8      if opt.lb $< n$.lb then
9          opt ← $n$
10     if $n$.ub $-$ $n$.lb $\leq \varepsilon$ then
11         return ReconstructStrategy(opt)
12     else
13         $(I_1, a) \leftarrow$ SelectAction($n$)
14         $P \leftarrow$ number of digits of precision
             representing $x(a)$ in $LP$
15         fringe ← fringe ∪ {CreateNode($LP \cup$
             $\{\sum_{k=0}^{\lfloor \frac{a_{ub}+a_{lb}}{2} \rfloor_{-P}} w_{k,P}^{I_1,a} = 1\}$)}
16         fringe ← fringe ∪ {CreateNode($LP \cup$
             $\{\sum_{k=\lfloor \frac{a_{ub}+a_{lb}}{2} \rfloor_{-P}}^{9} w_{k,P}^{I_1,a} = 1\}$)}
17         if $P < P_{max}(I_1,a)$ then
18             fringe ← fringe ∪ {CreateNode($LP \cup$
                 $\{w_{LP.x(a)_{-P},P}^{I_1,a} = 1$, introduce vars
                 $w_{0,P+1}^{I_1,a}, \dots, w_{9,P+1}^{I_1,a}$ and corresponding
                 constraints from MDT$\}$)}
19 return ReconstructStrategy(opt)

20 function CreateNode($LP$)
21     $ub \leftarrow$ Solve($LP$)
22     $\beta_1 \leftarrow$ ReconstructStrategy($LP$)
23     $lb \leftarrow u_1(\beta_1$, BestResponse($\beta1$))
24     return $(LP, lb, ub)$
```

Algorithm 1 depicts the complete BNB algorithm. It takes an LP relaxation of the Upper Bound MILP as its input. Initially, the maxmin strategy is approximated using 0 digits of precision after the decimal point (i.e. precision $P(I_1,a) = 0$ for every variable $x(a)$). The algorithm maintains a set of active BNB nodes (fringe) and a candidate with the best guaranteed value opt. The algorithm selects the node with the highest upper bound from fringe at each iteration (lines 4–5). If there is no potential for improvement in the unexplored parts of the branch and bound tree, the current best solution is returned (line 7) (up-

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

per bounds of the nodes added to the fringe in the future will never be higher than the current upper bound). Next, we check, whether the current solution has better lower bound than the current best, if yes we replace it (line 9). Since we always select the most promising node with respect to the upper bound, we are sure that if the lower bound and upper bound have distance at most ε, we have found an ε-optimal solution and we can terminate (line 11) (upper bounds of the nodes added to the fringe in the future will never be higher than the current upper bound). Otherwise, we heuristically select an action having the highest effect on the gap between the current upper and lower bound (line 13). We obtain the precision used to represent behavioral probability of this action. By default we add two successors of the current BNB node, each with one of the following constraints. $x(a) \leq \lfloor \frac{a_{ub}+a_{lb}}{2} \rfloor_{-P}$ (line 15) and $x(a) \geq \lfloor \frac{a_{ub}+a_{lb}}{2} \rfloor_{-P}$ (line 16), where $\lfloor \cdot \rfloor_p$ is flooring of a number towards p digits of precision and a_{ub} and a_{lb} are the lowest and highest allowed values of playing $x(a)$. This step performs binary halving restricting allowed values of $x(a)$ in current precision. Additionally, if the current precision is lower than the maximal precision $P_{max}(I_1,a)$ the gap between bounds may be caused by the lack of discretization points; hence, we add one more successor $\lfloor v \rfloor \leq x(a) \leq \lceil v \rceil$, where v is the current probability of playing a, while increasing the precision used for representing $x(a)$ (line 18) (all the restriction to $x(a)$ in all 3 cases are done via $w_{k,l}^{I_1,a}$ variables).

The function `CreateNode` computes the upper bound by solving the given LP (line 21) and the lower bound, by using the heuristical construction of a valid strategy β_1 returning some convex combination of strategies found for $\sigma_1^k \in \mathsf{seq}_1(I_1)$ (line 22) and computing the expected value of β_1 against a best response to it.

Note that this algorithm allows us to plug-in custom heuristic for the reconstruction of strategies (line 22) and for the action selection (line 13).

4.1 Theoretical Properties of the BNB Algorithm

The BNB algorithm takes the error bound ε as an input. We provide a method for setting the $P_{max}(I_1,a)$ parameters appropriately to guarantee ε-optimality. Finally, we provide a bound on the number of steps the algorithm needs to terminate.

Theorem 2. *Let $P_{max}(I_1,a)$ be the maximum number of digits of precision used for representing variable $x(a)$ set as*

$$P_{max}(I_1,a) = \left\lceil \max_{h\in I_1} \log_{10} \frac{|\mathcal{A}(I_1)| \cdot d \cdot v_{diff}(h)}{2\varepsilon} \right\rceil,$$

where $v_{diff}(h) = v_{max}(h) - v_{min}(h)$. With this setting Algorithm 1 terminates and it is guaranteed to return an ε-optimal strategy for player 1.

The proof provides a bound on the error per node in every $I \in \mathcal{I}_i^{IR}$ and propagates this bound through the game tree in a bottom up fashion.

Theorem 3. *When using $P_{max}(I_1,a)$ from Theorem 2 for all $I_1 \in \mathcal{I}_1$ and all $a \in \mathcal{A}(I_1)$, the number of iterations of the* BNB *algorithm needed to find an ε-optimal solution is in $O(3^{4S_1(\log_{10}(S_1 \cdot v_{diff}(\emptyset))+1)}2^{-5S_1}\varepsilon^{-5S_1})$, where $S_1 = |\mathcal{I}_1||\mathcal{A}_1^{max}|$.*

The proof derives a bound on the number of nodes in the BNB tree dependent on $\max_{I_1\in\mathcal{I}_i, a\in\mathcal{A}_1(I_1)} P_{max}(I_1,a)$ and uses the formula from Theorem 2 to transform the bound to a function of ε.

5 EXPERIMENTS

We now demonstrate the practical aspects of our main branch-and-bound algorithm (BNB) described in Section 4 and the iterative MILP variant described in Section 3. We compare the algorithms on a set of random games where player 2 has A-loss recall. Both algorithms were implemented in Java, each algorithm uses a single thread, 8 GB memory limit. We use IBM ILOG CPLEX 12.6 to solve all LPs/MILPs.

5.1 Random Games

Since there is no standardized collection of benchmark EFGs, we use randomly generated games in order to obtain statistically significant results. We randomly generate a perfect recall game with varying branching factor and fixed depth of 6. To control the information set structure, we use observations assigned to every action – for player i, nodes h with the same observations generated by all actions in history belong to the same information set. In order to obtain imperfect recall games with a non-trivial information set structure, we run a random abstraction algorithm which randomly merges information sets with the same action count, which do not cause absentmindedness. We generate a set of experimental instances by varying the branching factor. Such games are rather difficult to solve since (1) information sets can span multiple levels of the game tree (i.e., the nodes in an information set often have histories

Table 1: Average runtime and standard error in seconds needed to solve at least 100 different random games in every setting, while increasing the b.f.with fixed depth of 6.

Algs $\setminus$ $b.f.$	3	4
MILP	157.87 ± 61.47	459.09 ± 75.95
BNB	9.52 ± 3.78	184.50 ± 48.22

with differing sizes) and (2) actions can easily lead to leafs with very differing utility values. We always devise an abstraction which results to A-loss recall for the minimizing player.

5.2 Results

Table 1 reports the average runtime and its standard error in seconds for both algorithms over at least 100 different random games with increasing branching factor and fixed depth of 6. We limited the runtime for 1 instance to 2 hours (in the Table we report results only for instances where both algorithms finished under 2 hours). The BNB algorithm was terminated 17 and 30 times after 2 hours for the reported settings respectively, while MILP algorithm was terminated 19 and 34 times. Note that the random games form an unfavorable scenario for both algorithms since the construction of the abstraction is completely random, which makes conflicting behavior in merged information sets common. As we can see, however, even in these scenarios we are typically able to solve games with approximately $5 \cdot 10^3$ states in several minutes.

6 CONCLUSIONS

We provide the first algorithm for approximating maxmin strategies in imperfect recall zero-sum extensive-form games without absentmindedness. We give a novel mathematical formulation for computing approximate strategies by means of a mixed-integer linear program (MILP) that uses recent methods in the approximation of bilinear terms. Next, we use a linear relaxation of this MILP and introduce a branch-and-bound search (BNB) that simultaneously looks for the correct solution of binary variables and increases the precision for the approximation of bilinear terms. We provide guarantees that both MILP and BNB find an approximate optimal solution. Finally, we show that the algorithms are capable of solving games of sizes far beyond toy problems (up to $5 \cdot 10^3$ states) typically within few minutes in practice.

Results presented in this paper provide the first baseline algorithms for the class of imperfect recall games that are of a great importance in solving large extensive-form games with perfect recall. As such, our algorithms can be further extended to improve the current scalability, e.g., by employing incremental strategy generation methods.

ACKNOWLEDGEMENTS

This research was supported by the Czech Science Foundation (grant no. 15-23235S) and by the Grant Agency of the Czech Technical University in Prague, grant No. SGS16/235/OHK3/3T/13. Computational resources were provided by the CESNET LM2015042 and the CERIT Scientific Cloud LM2015085, provided under the programme ”Projects of Large Research, Development, and Innovations Infrastructures”.

REFERENCES

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*, 51:829–866.

Bowling, M., Burch, N., Johanson, M., and Tammelin, O. (2015). Heads-up limit hold’em poker is solved. *Science*, 347(6218):145–149.

Gilpin, A. and Sandholm, T. (2007). Lossless Abstraction of Imperfect Information Games. *Journal of the ACM*, 54(5).

Hoda, S., Gilpin, A., Peña, J., and Sandholm, T. (2010). Smoothing Techniques for Computing Nash Equilibria of Sequential Games. *Mathematics of Operations Research*, 35(2):494–512.

Kaneko, M. and Kline, J. J. (1995). Behavior Strategies, Mixed Strategies and Perfect Recall. *International Journal of Game Theory*, 24:127–145.

Kline, J. J. (2002). Minimum Memory for Equivalence between Ex Ante Optimality and Time-Consistency. *Games and Economic Behavior*, 38:278–305.

Koller, D. and Megiddo, N. (1992). The Complexity of Two-Person Zero-Sum Games in Extensive Form. *Games and Economic Behavior*, 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.

Koller, D. and Milch, B. (2003). Multi-agent influence diagrams for representing and solving games. *Games and Economic Behavior*, 45(1):181–221.

Kolodziej, S., Castro, P. M., and Grossmann, I. E. (2013). Global optimization of bilinear programs with a multiparametric disaggregation technique. *Journal of Global Optimization*, 57(4):1039–1063.

Kroer, C. and Sandholm, T. (2014). Extensive-Form Game Abstraction with Bounds. In *ACM conference on Economics and computation*.

Kroer, C. and Sandholm, T. (2016). Imperfect-Recall Abstractions with Bounds in Games. In *EC*.

Kuhn, H. W. (1953). Extensive Games and the Problem of Information. *Contributions to the Theory of Games*, II:193–216.

Lanctot, M., Gibson, R., Burch, N., Zinkevich, M., and Bowling, M. (2012). No-Regret Learning in Extensive-Form Games with Imperfect Recall. In *ICML*.

Nash, J. F. (1950). Equilibrium Points in n-person Games. *Proc. Nat. Acad. Sci. USA*, 36(1):48–49.

Piccione, M. and Rubinstein, A. (1997). On the Interpretation of Decision Problems with Imperfect Recall. *Games and Economic Behavior*, 20:3–24.

von Stengel, B. (1996). Efficient Computation of Behavior Strategies. *Games and Economic Behavior*, 14:220–246.

Wichardt, P. C. (2008). Existence of nash equilibria in finite extensive form games with imperfect recall: A counterexample. *Games and Economic Behavior*, 63(1):366–369.

Zinkevich, M., Johanson, M., Bowling, M., and Piccione, C. (2008). Regret Minimization in Games with Incomplete Information. In *NIPS*.

APPENDIX

Proposition 1. *It is always possible to construct a strategy $\beta(I_1)$ such that $\|\beta(I_1)-\beta^j(I_1)\|_1 \leq |\mathcal{A}(I_1)| \cdot 10^{-P}$ for every j.*

Proof. Probabilities of playing action a in $\beta^1,\dots,\beta^k$ can differ by at most 10^{-P}, i.e. $|\beta^j(a)-\beta^l(a)| \leq 10^{-P}$ for every j,l and action $a \in \mathcal{A}(I_1)$. This is based on the MDT we used to discretize the bilinear program. Let us denote

$$\underline{r}(\sigma_1 a) \quad = \quad \sum_{l=-P}^{0}\sum_{k=0}^{9} 10^{\ell}\cdot k \cdot \hat{r}(\sigma_1)^{a}_{k,\ell} \tag{6}$$

$$\underline{x}(I_1,a) \quad = \quad \sum_{l=-P}^{0}\sum_{k=0}^{9} 10^{\ell}\cdot k \cdot w^{I_1,a}_{k,\ell}. \tag{7}$$

Constraints (5f) and (5g) ensure that $\underline{r}(\sigma_1 a) = r(\sigma_1)\cdot \underline{x}(I_1,a)$. The only way how the imperfect recall can be violated is thus in the usage of $\Delta r(\sigma_1 a)$. We know however that $\Delta r(\sigma_1 a) \leq 10^{-P}\cdot r(\sigma_1)$ which ensures that the amount of imbalance in $\beta^1,\dots,\beta^k$ is at most 10^{-P}. Taking any of the behavioral strategies $\beta^1,\dots,\beta^k$ as the corrected behavioral strategy $\beta(I_1)$ therefore satisfies $\|\beta(I_1)-\beta^j(I_1)\|_1 \leq \sum_{a\in\mathcal{A}(I_1)} 10^{-P} = |\mathcal{A}(I_1)|\cdot 10^{-P}$. $\square$

We now provide the technical proof of Theorem 1. First, we connect the distance of a corrected strategy $\beta(I_1)$ from a set of behavioral strategies $\beta^1(I_1),\dots,\beta^k(I_1)$ in $I_1 \in \mathcal{I}_1^{IR}$ to the expected value of the strategy. We start with bounding this error in a single node.

Lemma 3. *Let $h \in I_1$ be a history and β^1, β^2 be behavioral strategies (possibly prescribing different behavior in I_1) prescribing the same distribution over actions for all subsequent histories $h' \sqsupset h$. Let $v_{max}(h)$ and $v_{min}(h)$ be maximal and minimal utilities of player 1 in the subtree of h, respectively. Then the following holds:*

$$|v_{\beta^1}(h)-v_{\beta^2}(h)| \leq \frac{v_{diff}(h)}{2}\cdot\|\beta^1(I_1)-\beta^2(I_1)\|_1,$$

where $v_{\beta^j}(h)$ is the maxmin value $u(\beta^j,\beta_2^{BR})$ of strategy β^j of player 1 given the play starts in h and $v_{diff}(h) = v_{max}(h)-v_{min}(h)$.

Proof. Let us study strategies β^1 and β^2 in node h. Let us take $\beta^1(I_1)$ as a baseline and transform it towards $\beta^2(I_1)$. We can identify two subsets of $\mathcal{A}(I_1)$ — a set of actions A^+ where the probability of playing the action in β^2 was increased and A^- where the probability was decreased. Let us denote

$$C^\circ = \sum_{a\in A^\circ} |\beta^1(I_1,a)-\beta^2(I_1,a)| \quad \forall \circ \in \{+,-\}.$$

We know that $C^+ = C^-$ (as strategies have to be probability distributions). Moreover we know that $\|\beta^1(I_1)-\beta^2(I_1)\|_1 = C^+ + C^-$. In the worst case, decreasing the probability of playing action $a \in A^-$ risks losing quantity proportional to the amount of this decrease multiplied by the highest utility in the subtree $v_{max}(h)$. For all actions $a \in A^-$ this loss is equal to

$$v_{max}(h)\cdot\sum_{a\in A^-}|\beta^1(I_1,a)-\beta^2(I_1,a)| = v_{max}(h)\cdot C^-.$$

Similarly the increase of the probabilities of actions in A^+ can add in the worst case $v_{min}(h)\cdot C^+$ to the value of the strategy. This combined together yields

$$\begin{aligned} v_{\beta^2}(h)-v_{\beta^1}(h) &\geq -v_{max}(h)\cdot C^- + v_{min}(h)\cdot C^+ \\ &= [-v_{max}(h)+v_{min}(h)]\cdot C^+ \\ &= \frac{-v_{max}(h)+v_{min}(h)}{2}\cdot 2C^+ \\ &= \frac{-v_{max}(h)+v_{min}(h)}{2}\cdot\|\beta^1(I_1)-\beta^2(I_1)\|_1. \end{aligned}$$

The strategies β^1, β^2 are interchangeable which results in the final bound on the difference of $v_{\beta^2}(h)$, $v_{\beta^1}(h)$. $\square$

Now we are ready to bound the error in the whole game tree.

Theorem 1. *The error of the Upper Bound MILP is bounded by*

$$\varepsilon = 10^{-P} \cdot d \cdot \mathcal{A}_1^{max} \cdot \frac{v_{max}(\emptyset) - v_{min}(\emptyset)}{2},$$

where d is the maximum number of player 1's imperfect recall information sets encountered on a path from the root to a terminal node, $\mathcal{A}_1^{max} = \max_{I_1 \in \mathcal{I}_1^{IR}} |\mathcal{A}(I_1)|$ is the branching factor and $v_{min}(\emptyset)$, $v_{max}(\emptyset)$ are the lowest and highest utilities for player 1 in the whole game, respectively.

Proof. We show an inductive way to compute the bound on the error and we show that the bound from Theorem 1 is its upper bound. Throughout the derivation we assume that the opponent plays to maximize the error bound. We proceed in a bottom-up fashion over the nodes in the game tree, computing the maximum loss $L(h)$ player 1 could have accumulated by correcting his behavioral strategy in the subtree of h, i.e.

$$L(h) \geq u_h(\beta^0) - u_h(\beta^{IR}),$$

where β^0 is the (incorrect) behavioral strategy of player 1 acting according to the realization probabilities $r(\sigma)$ from the solution of the Upper Bound MILP, β^{IR} is its corrected version and $u_h(\beta)$ is the expected utility of a play starting in history h when player 1 plays according to β and his opponent best responds (without knowing that the play starts in h). The proof follows in case to case manner.

(1) No corrections are made in subtrees of leafs h, thus the loss $L(h) = 0$.

(2) The chance player selects one of the successor nodes based on the fixed probability distribution. The loss is then the expected loss over all child nodes $L(h) = \sum_{a \in \mathcal{A}(h)} L(h \cdot a) \cdot \mathcal{C}(h \cdot a)/\mathcal{C}(h)$. In the worst case, the chance player selects the child with the highest associated loss, therefore

$$L(h) \leq \max_{a \in \mathcal{A}(h)} L(h \cdot a).$$

(3) Player 2 wants to maximize player 1's loss. Therefore she selects such an action in her node h that leads to a node with the highest loss, $L(h) \leq \max_{a \in \mathcal{A}(n)} L(h \cdot a)$. This is a pessimistic estimate of the loss as she may not be able to pick the maximizing action in every state because of the imperfection of her information.

(4) If player 1's node h is not a part of an imperfect recall information set, no corrective steps need to be taken. The expected loss at node h is therefore $L(h) = \sum_{a \in \mathcal{A}(h)} \beta^0(h,a) L(h \cdot a)$. Once again in the worst case player 1's behavioral strategy $\beta^0(h)$ selects deterministically the child node with the highest associated loss, therefore $L(h) \leq \max_{a \in \mathcal{A}(h)} L(h \cdot a)$.

(5) So far we have considered cases that only aggregate losses from child nodes. If player 1's node h is part of an imperfect recall information set, the correction step may have to be taken. Let β^{-h} be a behavioral strategy where corrective steps have been taken for successors of h and let us construct a strategy β^h where the strategy was corrected in the whole subtree of h (i.e. including h). Note that ultimately we want to construct strategy $\beta^\emptyset = \beta^{IR}$.
We know that values of children have been decreased by at most $\max_{a \in \mathcal{A}(h)} L(h \cdot a)$, hence $v_{\beta^0}(h) - v_{\beta^{-h}}(h) \leq \max_{a \in \mathcal{A}(h)} L(h \cdot a)$. Then we have to take the corrective step at the node h and construct strategy β^h. From Lemma 3 and the observation about the maximum distance of behavioral strategies within a single imperfect recall information set I_1, we get:

$$\begin{aligned} v_{\beta^{-h}}(h) - v_{\beta^h}(h) &\leq \frac{v_{diff}(h)}{2} \cdot 10^{-P} |\mathcal{A}_1(I_1)| \\ &\leq \frac{v_{diff}(\emptyset)}{2} \cdot 10^{-P} \mathcal{A}_1^{max} \end{aligned}$$

The loss in the subtree of h is equal to $v_{\beta^0}(h) - v_{\beta^{-h}}(h)$ which is bounded by player 1 acting according to the realization

$$\begin{aligned} L(h) &= v_{\beta^0}(h) - v_{\beta^h}(h) \\ &= \left[v_{\beta^{-h}}(h) - v_{\beta^h}(h)\right] + \left[v_{\beta^0}(h) - v_{\beta^{-h}}(h)\right] \\ &\leq \frac{v_{diff}(\emptyset)}{2} \cdot 10^{-P} \mathcal{A}_1^{max} + \max_{a \in \mathcal{A}(h)} L(h \cdot a). \end{aligned}$$

We will now provide an explicit bound on the loss in the root node $L(\emptyset)$. We have shown that in order to prove the worst case bound it suffices to consider deterministic choice of action at every node — this means that a single path in the game tree is pursued during propagation of loss. The loss is increased exclusively in imperfect recall nodes and we can encounter at most d such nodes on any path from the root. The increase in such nodes is constant ($[v_{max}(\emptyset) - v_{min}(\emptyset)] \cdot 10^{-P} \mathcal{A}_1^{max}/2$), therefore the bound is $\varepsilon = L(\emptyset) \leq [v_{max}(\emptyset) - v_{min}(\emptyset)] \cdot d \cdot 10^{-P} \mathcal{A}_1^{max}/2$.

We now know that the expected value of the strategy we have found lies within the interval $[v^* - \varepsilon, v^*]$, where v^* is the optimal value of the Upper Bound MILP. As v^* is an upper bound on the solution of the original bilinear program, no strategy can be better than v^* — which means that the strategy we found is ε-optimal. □

Theorem 2. *Let $P_{max}(I_1,a)$ be the maximum number of digits of precision used for representing variable $x(a)$ set as*

$$P_{max}(I_1,a) = \left\lceil \max_{h \in I_1} \log_{10} \frac{|\mathcal{A}(I_1)| \cdot d \cdot v_{diff}(h)}{2\varepsilon} \right\rceil,$$

where $v_{diff}(h) = v_{max}(h) - v_{min}(h)$. With this setting, Algorithm 1 terminates and it is guaranteed to return an ε-optimal strategy for player 1.

Proof. We start by proving that Algorithm 1 with this choice of $P_{max}(I_1,a)$ terminates. We will show that every branch of the branch-and-bound search tree is finite. This together with the fact that every node is visited at most once and the branching factor of the search tree is finite (every node of the search tree has at most 3 child nodes) ensures that the algorithm terminates.

Every node of the search tree is tied to branching on some variable $x(a)$. Let p be the current precision used to represent $x(a)$ and let us consider the first node on the branch where $x(a)$ is represented with such precision. At such point, $p-1$ digits are fixed and thus $x \in [c, c+10^{-(p-1)}]$ for some $c \in [0,1]$. On line 18 an interval of size 10^{-p} is handled, every left/right operation (lines 15 and 16) may thus handle an interval whose size is reduced at least by 10^{-p}. We can conduct at most 9 left/right branching operations (lines 15 and 16) before the size of the interval drops below 10^{-p}, which forces us to increase p. At most 10 operations can be performed on every $x(a)$ for every precision p, the limit on p is finite for every such variable and the number of variables is finite as well, the branch has therefore to terminate.

Let us now show that these limits on the number of refinements $P_{max}(I_1,a)$ are enough to guarantee ε-optimality. We will refer the reader to the proof of Theorem 1 for details while we focus exclusively on the behavior in nodes from imperfect recall information sets.

Let $I_1 \in \mathcal{I}_1^{IR}$ and $h \in I_1$. We know that the L1 distance between behavioral strategies in I_1 is at most $10^{-P_{max}(I_1,a)} \cdot |\mathcal{A}(I_1)|$ (for any $a \in \mathcal{A}(I_1)$). This means that the bound on $L(h)$ in h from the proof of Theorem 1 is modified to:

$$\begin{aligned}
L(h) &= v_{\beta^0}(h) - v_{\beta^h}(h) \\
&= \left[v_{\beta^{-h}}(h) - v_{\beta^h}(h)\right] + \left[v_{\beta^0}(h) - v_{\beta^{-h}}(h)\right] \\
&\leq \frac{v_{diff}(h)}{2} \cdot 10^{-P_{max}(I_1,a)} \cdot |\mathcal{A}(I_1)| + \max_{a \in \mathcal{A}(h)} L(h \cdot a) \\
&\leq \frac{v_{diff}(h)}{2} \cdot \frac{|\mathcal{A}(I_1)| \cdot 2\varepsilon}{|\mathcal{A}(I_1)| \cdot d \cdot [v_{diff}(h)]} + \max_{a \in \mathcal{A}(h)} L(h \cdot a) \\
&= \frac{\varepsilon}{d} + \max_{a \in \mathcal{A}(h)} L(h \cdot a).
\end{aligned}$$

Similarly with the reasoning in the proof of Theorem 1, it suffices to assume players choosing action at every node in a deterministic way. The path induced by these choices contains at most d imperfect recall nodes, thus $L(\emptyset) = d \cdot \varepsilon / d = \varepsilon$. □

Theorem 3. *When using $P_{max}(I_1,a)$ from Theorem 2 for all $I_1 \in \mathcal{I}_1$ and all $a \in \mathcal{A}(I_1)$, the number of iterations of the* BnB *algorithm needed to find an ε-optimal solution is in $O(3^{4S_1(\log_{10}(S_1 \cdot v_{diff}(\emptyset))+1)} 2^{-5S_1} \varepsilon^{-5S_1})$, where $S_1 = |\mathcal{I}_1|\mathcal{A}_1^{max}$.*

Proof. We start by proving that there is $N \in O(3^{4|\mathcal{I}_1|\mathcal{A}_1^{max} P_{max}})$ nodes in the BnB tree, where $\mathcal{A}_1^{max} = \max_{I \in \mathcal{I}_1} |\mathcal{A}(I)|$ and $P_{max} = \max_{I \in \mathcal{I}_1, a \in A(I)} P_{max}(I,a)$. This holds since in the worst case we branch for every action in every information set (hence $|\mathcal{I}_1|\mathcal{A}_1$). We can bound the number of branchings for a fixed action by $4P_{max}$, since there are 10 digits (we branch at most 4 times using binary halving) and we might require P_{max} number of digits of precision. $4|\mathcal{I}_1|\mathcal{A}_1^{max} P_{max}$ is therefore the maximum depth of the branch-and-bound tree. Finally the branching factor of the branch-and-bound tree is at most 3.

By substituting

$$\max_{I_1 \in \mathcal{I}_1} \left\lceil \max_{h \in I_1} \log_{10} \frac{|\mathcal{A}(I_1)| \cdot d \cdot v_{diff}(h)}{2\varepsilon} \right\rceil$$

for P_{max} in the above bound (Theorem 2), we obtain

$$\begin{aligned}
N &\in O(3^{4S_1 \max_{I_1 \in \mathcal{I}_1} \left\lceil \max_{h \in I_1} \log_{10} \frac{|\mathcal{A}(I_1)| \cdot d \cdot v_{diff}(h)}{2\varepsilon} \right\rceil}), \\
&\qquad \text{where } S_1 = |\mathcal{I}_1|\mathcal{A}_1^{max} \\
&\in O(3^{4S_1 \max_{I_1 \in \mathcal{I}_1} \left\lceil \log_{10} \frac{|\mathcal{A}(I_1)| \cdot d \cdot v_{diff}(\emptyset)}{2\varepsilon} \right\rceil}) \\
&\in O(3^{4S_1 \max_{I_1 \in \mathcal{I}_1} \left\lceil \log_{10} \frac{S_1 v_{diff}(\emptyset)}{2\varepsilon} \right\rceil}) \\
&\in O(3^{4S_1 \left\lceil \log_{10} \frac{S_1 \cdot v_{diff}(\emptyset)}{2\varepsilon} \right\rceil}) \\
&\in O(3^{4S_1 (\log_{10} \frac{S_1 \cdot v_{diff}(\emptyset)}{2\varepsilon} + 1)}) \\
&\in O(3^{4S_1 (\log_{10}(S_1 \cdot v_{diff}(\emptyset)) - \log_{10}(2\varepsilon) + 1)}) \\
&\in O(3^{4S_1 (\log_{10} S_1 \cdot v_{diff}(\emptyset)) + 1)} 3^{-10S_1 \log_{10}(2\varepsilon)}) \\
&\in O(3^{4S_1 (\log_{10} S_1 \cdot v_{diff}(\emptyset)) + 1)} 3^{-10S_1 \frac{\log_3(2\varepsilon)}{\log_3(10)}}) \\
&\in O(3^{4S_1 (\log_{10} S_1 \cdot v_{diff}(\emptyset)) + 1)} (2\varepsilon)^{\frac{-10S_1}{\log_3(10)}}) \\
&\in O(3^{4S_1 (\log_{10}(S_1 \cdot v_{diff}(\emptyset)) + 1)} (2\varepsilon)^{-5S_1})
\end{aligned}$$

□