Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Computer Science

Disertation Thesis

Solving Imperfect Recall Games

Jiří Čermák

Supervisor: Prof. Dr. Michal Pěchouček MSc.

Study Programme: Electrotechnics and Information Science

Field of Study: Information Science and Computer Engineering

June 1, 2016

Abstract

Imperfect recall games remain an unexplored part of the game theory, even though recent results in abstraction algorithms show that the imperfect recall might be the key to efficiently solving the immense games found in the real world. This thesis aims to answer a set of open questions considering the imperfect recall games, namely (1) what properties allow the existence of Nash equilibrium in behavioral strategies in imperfect recall games and (2) what information are the players allowed to forget in order to still guarantee that the resulting imperfect recall game can be used to solve the original game. Finally, we aim to develop the first algorithm capable of outperforming the state-of-the-art solvers applied directly to perfect recall games, by solving its imperfect recall abstraction.

Contents

Chapter 1

Introduction

Game theory is a mathematical model of strategic interaction between agents. It is a descriptive theory defining conditions for strategies of players to form an equilibrium. The most famous equilibrium, where no rational player wants to deviate from the prescribed strategy, is called a *Nash equilibrium*. Simple one-shot problems are represented as *normal-form games*. In this work, however, we focus on more general situations, where we face finite sequential decisions including partial observability and stochastic environment. For these situations *extensive-form games* are the most suitable representation. Extensive-form games are represented as a game tree with nodes corresponding to the game states and edges representing actions available to players. The partial observability is represented by grouping states indistinguishable for player making decision to informations sets. More specifically, we focus on *imperfect recall games* where the structure of partial information forces players to forget their own moves or the information available to them in the past.

Recent years have seen a significant rise of applications of game theory to real world scenarios, e.g., in security domains [28]. One of the main challenges encountered in applications is the immense size of games needed to be solved. The state-of-the-art approach to handle this issue is the use of abstractions [8, 14, 21]. Basic idea behind abstractions is to shrink the game representation by grouping similar situations in the game. Among the many studied abstraction techniques, the ones which yield the best space savings are those employing imperfect recall in the resulting game [21],

In spite of emerging practical use of imperfect recall [21, 30], many theoretical questions concerning this class of games still remain open. One of the most significant ones is the lack of understanding of the properties the imperfect recall game must posses, to guarantee the existence of Nash equilibrium when representing player's behavior using distribution over action in each decision point, denoted as *behavioral strategy*. This is caused by a different descriptive power of behavioral and *mixed strategy* representation (which represents the behavior using a probability distribution over the whole deterministic assignments of one action to every information set of the player) in imperfect recall games [22]. The concept of mixed strategies is not directly applicable to games with imperfect recall, as it allows players to condition their actions on information hidden by the rules of the game (e.g., see Figure 2.2). For this reason, behavioral strategies are used in imperfect recall games [22]. However, the original proof of existence of Nash equilibrium for finite games considers mixed strategies only [25]. Additionally, from the computational perspective, even to check whether

an imperfect recall game has a Nash equilibrium in behavioral strategies is shown to be NP-hard [12].

Since Nash equilibrium behavioral strategies may not exist, there is an alternative to seek a behavioral strategy that maximizes the worst case expected outcome of a game—*a maxmin strategy*, which is guaranteed to exist in imperfect recall games. Maxmin strategies may require irrational numbers even when the input uses rational numbers [18]. So our results cannot be directly extended to optimal strategies for fundamental reasons, and approximating maxmin strategies is equivalent to solving imperfect recall games. However, also deciding whether a player with imperfect recall can guarantee at least a given value in an imperfect recall game is an NP-hard problem [18].

In this thesis we aim to answer the main open questions concerning imperfect recall games and provide first algorithms for finding maxmin strategies and Nash equilibrium in imperfect recall games. We restrict to games with no *absent-mindedness* where players cannot reach the same decision point more than once during one playthrough. We make such a restriction, since absent-mindedness produces highly unnatural situations, while it greatly complicates the procedure of solving the game [27].

In Chapter 2 we introduce the notation used in this thesis and discuss the difference between mixes strategies and behavioral strategies in imperfect recall games. Furthermore, we introduce subsets of imperfect recall games according to the structure of information lost. In Chapter 3, we introduce the conditions for existence of Nash equilibrium in behavioral strategies in a subset of imperfect recall games called A-loss recall games. In Chapter 4 we exploit these properties and propose an algorithm computing Nash equilibrium in A-loss recall games, or detecting that none exists. In Chapter 5 we suggest a no-regret algorithm for finding Nash equilibrium in A-loss recall games inspired by the Counterfactual regret minimization [32]. Finally, in Chapter 6 we suggest an algorithm finding maxmin strategies in imperfect recall games with no absentmindedness.

1.1 Related Work

The fundamental difficulty of strategy representation and the difference between behavioral and mixed strategies in imperfect recall games was first discussed by Kuhn [22]. An example showing that imperfect recall games need not have Nash equilibrium in behavioral strategies, which builds on the difference of mixed and behavioral strategies described by Kuhn, was provided by Wichardt [31].

Kaneko et. al. [15] study the additional information introduced when using mixed strategies in imperfect recall games. They introduce the notion of A-loss recall games, which form a subset of imperfect recall games, where the imperfect recall is caused solely by forgetting one's own actions. They show that in A-loss recall games the use of mixed strategies completely compensates the information hidden to players due to imperfect recall. In the follow-up work by Kline [16] it is shown that in A-loss recall games any strategy is ex-ante optimal if and only if players do not want to deviate during the actual playthrough. This property is called *time consistency*. Similarly, in parallel work Bonano [4] distinguishes the difference between imperfect recall caused by forgetting information one had previously available and action one has previously played.

The relation of Nash equilibrium, Sequential equilibrium and Perfect equilibrium, which are ordered left to right by increasing restriction on the behavior of players in perfect recall games, is discussed by Halpern and Kline [11, 17] in games of imperfect recall. They show that enforcing sequential rationality described by the Sequential equilibrium is still guaranteed to generate behavior consistent with Nash equilibrium in A-loss recall games, while showing that beyond the A-loss recall games this relationship need not hold.

Another topic discussed in the literature are the games with absent-mindedness, where one decision point can be reached more than once during the playthrough. Piccione and Rubinstein [27] introduce the famous example of the absent-minded driver paradox and discuss the difference of time consistent and ex-ante optimal strategies connected to the interpretation of the information set as either point of decision or the point of strategy execution. In the follow-up paper [26] they summarize and react to the various papers suggesting solutions to the absent-minded driver paradox. Halpern et. al. [10] discuss the possible deviations players consider, when prescribed ex-ante optimal but time inconsistent strategy in both games with and without absentmindedness.

There is only a limited amount of work focused on finding the optimal strategies in imperfect recall games. A restricted classes of imperfect recall games were described, where the Counterfactual regret minimization [32] is guaranteed to converge to Nash equilibrium strategies [23, 21].

1.2 Research Objectives

This thesis aims to explore the theoretical properties of imperfect recall games created via abstraction algorithm from some large perfect recall game. Next, we are interested in introducing algorithms solving the large perfect recall game using abstracted imperfect recall game, outperforming the state-of-the-art algorithms applied directly to the unabstracted perfect recall game. Since the abstraction algorithm can be guided to result in desirable game structure, we are interested in exploring the types of imperfect recall games which we can use to find rational behavior in the original games. The result presented in Chapter 3 shows that A-loss recall games have this property, when some additional conditions are satisfied. Even though we provide an algorithm for solving these games, this algorithm is computationally prohibitive, hence we would like to investigate additional approaches which might yield an efficient algorithm taking advantage of the reduced game size. Next, we would like to explore properties of the imperfect recall games beyond the limitations of the A-loss recall in order to find, whether there are some additional subsets of imperfect recall games which allow us to solve the pre-abstraction games. Finally, in order to reason about the general imperfect recall games, we would like to devise an efficient algorithm capable of solving them.

More specifically the planned contributions can be summarized to following points

- Find properties the imperfect recall game needs to satisfy in order to guarantee the existence of Nash equilibrium in behavioral strategies.
- Detect what information can be removed from some perfect recall game G' by an abstraction algorithm, while still being able to use the resulting abstracted imperfect recall G to find rational strategies in G'.

- Devise an efficient algorithm capable of finding Nash equilibrium of perfect recall game using its imperfect recall abstraction, while outperforming state-of-the-art algorithms solving the original perfect recall game.

1.2.1 Existence of Nash Equilibrium

In Chapter 3 we provide the necessary and sufficient conditions guaranteeing the existence of Nash equilibrium in behavioral strategies for games having A-loss recall. As a future work we plan to extend this result to games having general imperfect recall. This will lead to better understanding of the complications the various types of imperfect recall introduce to decision making.

1.2.2 Correspondence of Nash equilibria of Abstracted and Original Game

In Chapter 3 we show that when there is a Nash equilibrium in behavioral strategies in A-loss recall game, it corresponds to the Nash equilibrium of its coarsest perfect recall refinement. We plan to similarly explore the imperfect recall games without A-loss recall games in order to find as wide range of game classes the abstraction algorithms can output.

1.2.3 Computing Maxmin Strategies in Imperfect Recall Games

In Chapter 6 we introduce an algorithm capable of finding maxmin strategies for player who faces an opponent having A-loss recall and we provide an extension of this algorithm to general imperfect recall games with the only restriction being no absent-mindedness. As a future work we plan to make this algorithm more efficient by using a double oracle approach [5] to iteratively generate columns and rows of the underlying mathematical program.

1.2.4 Computing Nash Eqilibrium in Imperfect Recall Games

In Chapter 4 an algorithm for computing Nash equilibrium in behavioral strategies in A-loss recall games exploiting the properties described in Chapter 3. This algorithm, however, works in double exponential time with respect to the size of the game, which makes it impractical. In Chapter 5 we present an idea of a no-regret algorithm inspired by the counterfactual regret minimization [32] and which benefits from the reduced number of information sets in the imperfect recall game.

Chapter 2

Introduction to Game Theory

In this section we introduce the extensive-form representation of games (EFGs).

A two player EFG G is a tuple $\{\mathcal{P}, \mathcal{H}, \mathcal{Z}, P, u, \mathcal{I}, A, f_c\}$.

- $\mathcal{P} = \{1, 2\}$ denotes the set of players. We use i to denote a player and $-i$ to denote the opponent of i.
- Set $\mathcal{H}$ contains all the states of the game represented as nodes in the tree. 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 all *terminal states* of the game.
- $P : \mathcal{H} \to \mathcal{P} \cup \{c\}$ is the function associating a player from set $\mathcal{P}$ or the nature c with every state of the game. The nature c represents the stochastic environment of the game.
- $u_i : \mathcal{Z} \to \mathbb{R}$ is a utility function assigning to each leaf the value of preference for player i. For zero-sum games it holds that $u_i(z) = -u_{-i}(z), \forall z \in \mathcal{Z}$.
- $\mathcal{I}_i$ is a partitioning of all $\{h \in \mathcal{H} : P(h) = i\}$ into information sets. All states h contained in one information set $I_i \in \mathcal{I}_i$ are indistinguishable to player i.
- $A(h)$ is a function returning the set of actions available to player making decision in h. Since the set of available actions $A(h)$ is the same $\forall h \in I_i$. We overload the notation and use $A(I_i)$ as actions available in I_i.
- f_c is a function assigning to every h where $P(h) = c$ an independent probability measure on $A(h)$.

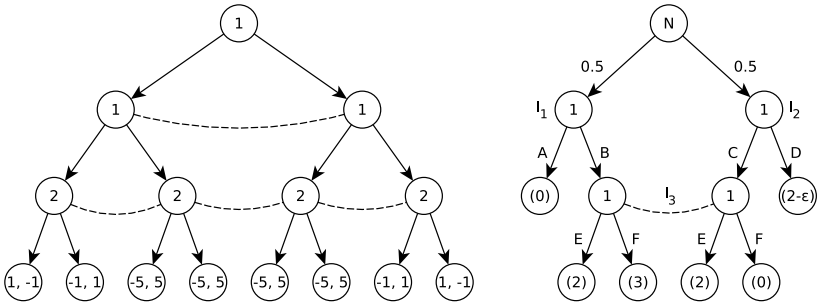

Figure 2.1: (Left) An EFG with A-loss recall where the NE in behavioral strategies does not exist [31]; (Right) An EFG without A-loss recall where a local best response is not the ex ante best response.

2.1 Strategy Representation

A *pure strategy* s_i for player i is a mapping assigning $\forall I_i \in \mathcal{I}_i$ a member of $A(I_i)$. $\mathcal{S}_i$ is a set of all pure strategies for player i. A *mixed strategy* m_i is a probability distribution over $\mathcal{S}_i$, set of all mixed strategies of i is denoted as $\mathcal{M}_i$. A *strategy profile* is a set of strategies, one strategy for each player. In EFGs *behavioral strategies* b_i assign probability distribution over $A(I_i)$ for each I_i. $\mathcal{B}_i$ is a set of all behavioral strategies for i, $\mathcal{B}_i^p \subseteq \mathcal{B}_i$ denotes the set of deterministic behavioral strategies for i. Finally, we can use sequence-form representation for games with perfect recall [19]. A *sequence* σ_i is a list of actions of player i ordered by their occurrence on the path from the root of the game tree to some node. By $seq_i(h)$ we denote the sequence of player i leading to the state h. The strategy can be formulated as a *realization plan* r_i that for a sequence σ_i represents the probability of playing actions in σ_i assuming the other players play such that the actions of σ_i can be executed. Realization plan r_i has to satisfy network flow property; i.e., $r_i(\sigma_i) = \sum_{a \in A(I_i)} r_i(\sigma_i \cdot a)$, where I_i is an information set reached by sequence σ_i and $\sigma_i \cdot a$ stands for σ_i extended by action a. We say that σ_i' is a *continuation* of σ_i if σ_i forms a prefix of σ_i'. $\mathcal{R}_i$ is a set of realization plans for i, $\mathcal{R}_i^p \subseteq \mathcal{R}_i$ denotes the set of pure realization plans for i. We say that a pair of strategies with arbitrary representation is *behaviorally equivalent* if they generate the same probability distribution over all $z \in \mathcal{Z}$. We overload the notation and use u_i also to denote the expected utility of player i when the players play according to pure (mixed, behavioral) strategies or realization plans.

2.2 Perfect, Imperfect and A-Loss Recall

In *perfect recall* all players remember 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. In perfect recall games, behavioral and mixed strategies are equivalent [22] and optimal strategies can be found using the standard sequence-form linear program [29].

If the assumption of perfect recall does not hold in an EFG, we talk about games with *imperfect recall*. In imperfect recall games, mixed and behavioral strategies are not com-

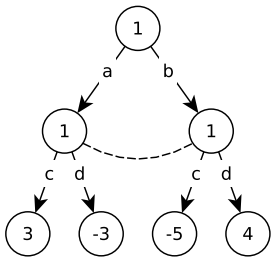

Figure 2.2: Simple imperfect recall game

parable [22]. Let us demonstrate the difference between the described strategy representations on the imperfect recall game from Figure 2.2. This game has 4 pure strategies $\mathcal{S} = \{(a, c), (a, d), (b, c), (b, d)\}$. The set of mixed strategies is created by all the probability distributions over entries of $\mathcal{S}$. As we can see, the structure of pure strategies allows mixed strategies to condition choices of players on the forgotten information, e.g., strategy $m(a, c) = 0.5$, $m(b, d) = 0.5$ allows player 1 to condition playing c and d on the outcome of his non-deterministic choice in the root of the game. This is, however, forbidden by the rules of the game. The behavioral strategy b, on the other hand, assigns for every information set I probability distribution over $A(I)$, and therefore no additional information is disclosed to player. Finally, the sequence-form strategy representation is also unsuitable for imperfect recall games for similar reasons, since for example in the case of the game from Figure 2.2 the sequences correspond to pure strategies, implying that sequence-form also allows conditioning of actions on unavailable information. Additionally, the network flow property can be unsatisfiable, since there can be sequences of i with different realization probabilities leading to some I_i.

In games with *absentmindedness* (AM), players can visit the same information set more than once in the course of the game (see, e.g., [27]). We focus on games without AM and exploit the following proposition:

Lemma 1. *Let G be an imperfect recall game without AM and b_1 strategy of player 1. There exists an* ex ante *pure behavioral best response of player 2.*

Proof. If the strategy of player 1 is fixed, then finding the best response for player 2 is finding an optimum of a function over a closed convex polytope with vertices formed by pure behavioral strategies. Since each action can be chosen at most once in a game without AM, the objective function is multilinear, therefore an optimum must be in one of the vertices of this polytope – that is a pure behavioral strategy. □

Note that in games with AM, the ex ante best response (i.e., when evaluating only the expected value of the strategy) may need randomization (e.g., in the game with absentminded driver [27]).

However, even in imperfect recall games without AM, playing a best action in each information set need not result in an ex ante best response. A strategy that maximizes the expected outcome in each information set given current beliefs over the states in the information set is called a *time consistent strategy* [16]. An equivalence between time consistent

strategies and ex ante best responses was shown to hold exactly for games with so called *A-loss recall* [15, 16].

Informally, a player has A-loss recall if any forgetfulness of the player can be traced back to some loss of memory of his own actions.

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) says that if there exists an information set of player i where two distinct actions a, a' have been taken to reach two different nodes in an information set I then player i has A-loss recall. Figure 2.1 provides two examples of imperfect recall games. Left subfigure shows a game due to [31], where players have A-loss recall, however, NE does not exist in behavioral strategies. Right subfigure shows a game between player 1 and chance. Player 1 does not have A-loss recall in this game – parents of the nodes in information set I_3 are in two distinct information sets I_1, I_2 and their common predecessor is a chance node. The ex ante best response of player 1 in this game is to play B,D,F getting the utility of $\frac{5-\varepsilon}{2}$. Note, however, that when player 1 chooses the best action to play in each information set I separately, given possible beliefs over states in I yields strategy B,C,E with the expected utility of 2.

2.2.1 Coarsest Perfect Recall Refinement

Let us define a partition $H(I_i)$ of states in every information set I_i of some imperfect recall game G to the largest possible subsets, not causing imperfect recall. More formally, let $H(I_i) = \{H_1, ..., H_n\}$ be a disjoint partition of all $h \in I_i$, where $\bigcup_{j=1}^{n} H_j = I_i$ and $\forall H_j \in H(I_i)\ \forall h_k, h_l \in H_j : seq_i(h_k) = seq_i(h_l)$, additionally $\forall H_k, H_l \in H(I_i) : seq_i(H_k) \neq seq_i(H_l)$

Definition 2. *The coarsest perfect recall refinement G' of the imperfect recall game $G = \{\mathcal{P}, \mathcal{H}, \mathcal{Z}, P, u, \mathcal{I}, A\}$ is a tuple $\{\mathcal{P}, \mathcal{H}, \mathcal{Z}, P, u, \mathcal{I}', A'\}$, where $\forall i \in \mathcal{P}\ \forall I_i \in \mathcal{I}_i\ H(I_i)$ defines the information set partition $\mathcal{I}'$. A' is a modification of A, which guarantees that $\forall I \in \mathcal{I}'\ \forall h_k, h_l \in I\ A'(h_k) = A'(h_l)$, while $\forall I^k, I^l \in \mathcal{I}'\ A(I^k) \neq A(I^l)$.*

In the Definition 2 we change the labelling of actions described by A and A', since we modify the structure of the imperfect information $\mathcal{I}$ to $\mathcal{I}'$. We denote by $\Phi : E \to A'$ a function which for an edge $e \in E$, where E is the set of all edges of the game tree of G (and therefore of G'), returns its action label in G', similarly we define $\Psi : E \to A$. In the rest of this section, when we talk about the equivalence of arbitrary strategy representation or sequences in G and G', we talk about the equivalence with respect to Φ and Ψ. Same goes for applying the strategy from G to G' and vice versa.

We can restrict the coarsest perfect recall refinement only for i if we split only information sets of i and information sets of $-i$ remain unchanged.

Lemma 2. *Let G be an imperfect recall game where player 2 has A-loss recall and b_1 is a strategy of player 1, and let G' be the coarsest perfect recall refinement of G for player 2. Let b'_2 be a pure best response in G' and let b_2 be a realization equivalent behavioral strategy in G, then b_2 is a pure best response to b_1 in G.*

Proof. Note that b_1 is a valid strategy in both games since we refine only information sets of player 2. First, we show how b_2 is constructed from b'_2. Consider an information set of player 2 I in G and corresponding information sets $I^1, \ldots, I^j$ in the coarsest perfect refinement G'. b'_2 prescribes which action to play in each information set of $I^1, \ldots, I^j$.

Claim At most one of information sets $I^1, \ldots, I^j$ can eventually be reached when players play according to strategy profile (b_1, b'_2) in G'.

Proof. Due to A-loss recall of player 2, for every pair of nodes h_k, h_l from two different information sets I^k,I^l, there exists an information set I' and two distinct actions $a, a' \in \mathcal{A}_i(I'), a \neq a'$ such that $a \in \mathsf{seq}_i(h_k) \wedge a' \in \mathsf{seq}_i(h_l)$. However, since b'_2 is a pure best response, only one action among pair of actions a, a' can be played with a non-zero probability and consequently, only one information set of pair I^k,I^l can be reached. □

We use this claim to construct b_2. For information set I from G that is divided into information sets $I^1, \ldots, I^j$ in G' we define $b_2(I, a) := 1$ for action $a \in \mathcal{A}(I^k), b'_2(I^k, a) = 1$, where I^k is the reachable set from the claim. If no information set from $I^1, \ldots, I^j$ is reachable, we set b_2 in I arbitrarily. For all information sets $I' \in \mathcal{I}'$ that are not split in the coarsest perfect recall refinement (and therefore are the same as in G), we set $b_2(I') := b'_2(I')$. Due to the construction, the realization equivalence between b'_2 and b_2 follows immediately.

Finally, we show that b_2 is a best response in G. Due to the realization equivalence between b'_2 and b_2, the expected utility is the same $u(b_1, b'_2) = u(b_1, b_2)$. Since b'_2 is the best response in G' and for every pure behavioral strategy $\hat{b_2}$ in G we can find a realization equivalent behavioral strategy $\hat{b_2}'$ similarly to the construction described in the proof of the claim. Now

$$u(b_1, \hat{b_2}) = u(b_1, \hat{b_2}') \leq u(b_1, b'_2) = u(b_1, b_2) \; \forall \hat{b_2} \in \mathcal{B}_2^G$$

concludes the proof. □

2.3 Solution Concepts

Definition 3. *We say that strategy profile $r = \{r_i, r_{-i}\}$ is a Nash equilibrium in realization plans if and only if $\forall i \in P \; \forall r_i^p \in \mathcal{R}_i^p : u_i(r_i, r_{-i}) \geq u_i(r_i^p, r_{-i})$*

Similarly, we can define the Nash equilibrium in behavioral strategies.

Definition 4. *We say that strategy profile $b = \{b_i, b_{-i}\}$ is a Nash equilibrium in behavioral strategies iff $\forall i \in P \; \forall b_i^p \in \mathcal{B}_i^p : u_i(b_i, b_{-i}) \geq u_i(b_i^p, b_{-i})$*

The Nash equilibrium in mixed strategies is defined equally.

Definition 5. *A* maxmin strategy *in behavioral strategies b_i^* is defined as*

$$b_i^* = \arg\max_{b_i \in \mathcal{B}_i} \min_{b_{-i} \in \mathcal{B}_{-i}} u_i(b_i, b_{-i}). \tag{2.1}$$

Note that when a Nash equilibrium in behavioral strategies exists in a two-player zero-sum imperfect recall game then b_i^* is a Nash equilibrium strategy for i.

Chapter 3

Nash Equilibrium Existence in A-Loss Recall Games

In this section we discuss the properties influencing the existence of Nash Equilibrium in behavioral strategies of A-loss recall games.

Lemma 3. *When player i plays according to pure realization plan r_i in A-loss recall game G only states in one $H_k \in H(I_i)$ in every imperfect recall information set I_i of i can have positive probability of occurrence if I_i gets visited.*

Proof. This directly follows from the A-loss recall property. □

Lemma 4. *The sets of pure realization plans $\mathcal{R}^{pG}$ of G and $\mathcal{R}^{pG'}$ of G' are equal.*

Proof. First, we prove that $\forall i \in \mathcal{P}\ \forall {r'}_i \in \mathcal{R}^{pG'}$, ${r'}_i$ forms a pure realization plan of G. This follows from Lemma 3, as each H_k coincides with an information set in G' and i can choose actions independently in every H_k when following pure realization plan. r'_i must therefore be a valid pure realization plan of G.

Next, we show that $\forall i \in \mathcal{P}\ \forall r_i \in \mathcal{R}^{pG}$ it holds that r_i forms a pure realization plan of G'. This is trivially satisfied, as the information structure of G is formed by unification of information sets of G'. □

Lemma 5. *Every pure realization plan of arbitrary imperfect recall game G can be represented as a deterministic behavioral strategy and vice versa.*

Proof. It is straightforward to show that we are able to create a behavioral strategy $b_i^p \in \mathcal{B}_i^{p^G}$ equivalent to any $r_i^p \in \mathcal{R}_i^{p^G}$. Moving from the root of the game tree of G for every I_i, for which $r_i^p(I_i)$ is defined, we create $b(I_i, a) = 1$ for the action prescribed by r_i^p in I_i. This creates equivalent behavior in all I_i.

Creating pure realization plan from deterministic behavioral strategy can be done by the same procedure, ignoring the parts of the tree unreachable when playing this strategy. □

Now we are ready to state the main result of this section.

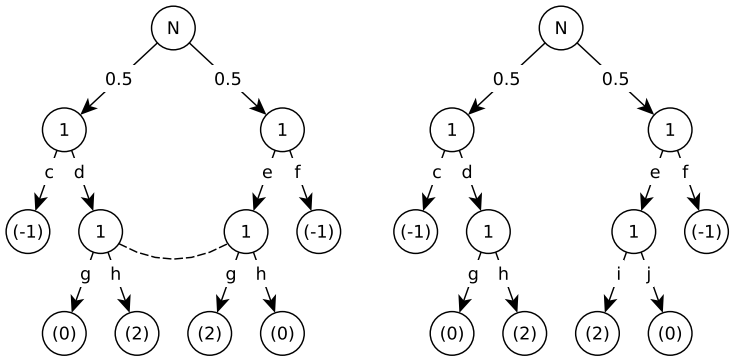

Figure 3.1: (Left) Game without A-loss recall where the only Nash equilibrium in behavioral strategies is not a Nash equilibrium of the coarsest perfect recall refinement. (Right) Its coarsest perfect recall refinement.

Theorem 1. *The A-loss recall game G has a Nash equilibrium in behavioral strategies if and only if there exists a Nash equilibrium $b \in \mathcal{B}^{G'}$ in behavioral strategies of the coarsest perfect recall refinement G' of G, such that $\forall I \in \mathcal{I}^G \; \forall H_k, H_l \in H(I) : b(H_k) = b(H_l)$, where $b(H)$ stands for the behavioral strategy in the information set of G' formed by states in H.*

Proof. First, we prove, that b forms a Nash equilibrium in behavioral strategies of G. Since b is a Nash Equilibrium of G' we know that there exists no incentive for any player to deviate to any pure behavioral strategy in G'. Additionally, from the interchangeability of pure behavioral strategies and pure realization plans in games with perfect recall [22] we know that there exists no pure realization plan to which any of the players wants to deviate. From Lemma 4, it follows that there can exists no pure realization plan in G, and therefore by Lemma 5 no pure behavioral strategy in G to which any of the players wants to deviate. This, in combination with the fact, that b prescribes valid strategy in G implies that b is a Nash Equilibrium in behavioral strategies of G

Second, we prove that there exists no Nash equilibrium b' in behavioral strategies of G which is not a Nash equilibrium of G'. Let us assume that such b' exists. This would imply that there is no pure behavioral strategy in G to which players want to deviate when playing according to b', and therefore no such pure realization plan (Lemma 5). However, then Lemma 4 implies that there is no such pure realization plan and therefore pure behavioral strategy in G' either, implying that b' is a Nash Equilibrium in G'. This contradicts the assumption and completes the proof.

□

Informally, Theorem 1 states that G has a Nash equilibrium in behavioral strategies if and only if there exists a behavioral Nash equilibrium b in G' which prescribes the same behavior in every information set which is connected to some imperfect recall information set of G.

Finally, we provide a counter-example showing that Theorem 1 does not extend to games without A-loss recall games. Consider the game in left subfigure of Figure 3.1. Here the only

Nash equilibrium in behavioral strategies is playing d and e deterministically and mixing uniformly between g, h. The only Nash equilibrium of the coarsest perfect recall refinement (shown in right subfigure of Figure 3.1) is, however, playing d, e, h and i deterministically.

Theorem 1 shows an interesting property, which makes A-loss recall games desirable model to study. If we assume that the A-loss recall game G is an output of some abstraction algorithm working on its coarsest perfect recall refinement G' (this is easily guaranteed by not allowing the abstraction algorithm to merge sets by forgetting anything else than players own actions), then we know that if we find a Nash equilibrium in behavioral strategies of the G, we are sure that this behavioral strategy profile is a Nash equilibrium of G'.

Chapter 4

Computation of Nash Equilibrium in A-loss Recall Games

In this chapter we introduce the algorithm for computing the Nash equilibrium of zero-sum two-player games with A-loss recall without absent mindedness G. The algorithm works in a following way. First, it creates a perfect recall refinement G' of G. It then proceeds to enumerate all the optimal solutions of the sequence-form LP (described in Section 4.1) using the algorithm described in Section 4.3. From this procedure we obtain a finite set of extreme Nash equilibria (described in Section 4.2), which can be used to generate the possibly infinite set of all Nash equilibria. Finally, the algorithm checks, whether there exists a Nash equilibrium of G' prrescribing behavior consistent with the rules of G.

4.1 Sequence-form LP

Here we describe the linear program (LP) for computing the Nash equilibrium that exploits the sequence form due to Koller et al. [19].

$$\max_{r_1, q} \quad f^\top q \tag{4.1}$$

$$s.t. \quad -A^\top r_1 + F^\top q \leq 0 \tag{4.2}$$

$$Er_1 = e \tag{4.3}$$

$$r_1 \geq 0 \tag{4.4}$$

In eqs. (4.1) to (4.4) we present a linear program for solving two player zero-sum EFGs with perfect recall. Matrix A is a utility matrix with rows corresponding to sequences of player 1 and columns to sequences of player 2. Each entry of A corresponds to the sum of utility values of the game states reached by the sequence combination assigned to this entry, weighted by the probability of occurrence of this state considering nature. If all the reached states are non-terminal, or if the sequence combination is incompatible, the entry is 0.

Matrices E and F define the structure of the realization plans for player 1 and 2 respectively. Columns of these matrices are labelled by sequences, rows by information sets. Row

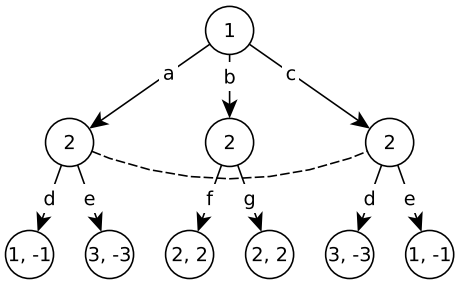

Figure 4.1: Game with two extreme Nash equilibria for each player.

for the information set I contains -1 on the position corresponding to the sequence leading to I, 1 for the sequences leading from I and zeros otherwise. First row, corresponding to the artificial information set has 1 only on position for the empty sequence. Vectors e, f are indexed by information sets of players and consist of zeros, with 1 on the first position. q is a vector of variables representing values in information sets of the opponent. The constraints (4.3) enforce the structure of the realization plan and the constraints (4.2) tighten the upper bound on the value in each of the opponent's information sets I_2 for every sequence leaving I_2.

4.2 Extreme Nash Equilibria

The possibly infinite set of all Nash equilibria of the game G can be completely described by the set of finitely many extreme Nash equilibria $\mathcal{E}$ (see, e.g., [1]). These equilibria have the property, that they cannot be obtained by convex combination of any other Nash equilibria of G. Additionally, all the non-extreme (also called degenerate) Nash equilibria of G are obtained as a convex combination of entries of $\mathcal{E}$. For the case of two-player zero-sum games, we can say that $\mathcal{E} = \mathcal{E}_1 \times \mathcal{E}_2$, where $\mathcal{E}_i$ stands for strategies of i forming part of extreme Nash equilibrium. Additionally, entries of $\mathcal{E}_i$ form the vertices of the convex polyhedron of optimal solutions of the sequence-form LP computing the strategy for i. Finally, the expected value of every player is the same when all players play according to some Nash equilibrium strategy in zero-sum games.

Let us consider the game from Figure 4.1. The only extreme Nash equilibria for player 1 are $\{b_1^1(b) = 1\}$ and $\{b_1^2(a) = 0.5, b_1^2(c) = 0.5\}$. Similarly, for player 2 we obtain two extreme Nash equilibria $\{b_2^1(d) = 0.5, b_2^1(e) = 0.5, b_2^1(f) = 1\}$ and $\{b_2^2(d) = 0.5, b_2^2(e) = 0.5, b_2^2(g) = 1\}$. All the equilibria for both players can be obtained as a convex combination of these extreme equilibria.

4.3 A Pivoting Algorithm for Vertex Enumeration

Here we describe the algorithm for enumeration of all vertices of a convex polyhedron due to Avis et. al. [2]. Informally, the algorithm reverts the steps of the simplex algorithm, making sure that it visits every vertex of the polyhedron exactly once.

4.3.1 Simplex Algorithm.

Simplex algorithm [7] solves linear programming tasks by traversing the vertices of the convex polyhedron formed by the constraints of the linear program, until optimum is found.

Lets assume we are given a LP formulation

$$\min_x \quad c^\top x \tag{4.5}$$

$$s.t. \quad Ax = b \tag{4.6}$$

$$x \geq 0 \tag{4.7}$$

where $A \in \mathbb{R}^{m \times n}$. We say that $B \subseteq \{1, ..., n\}$ is a *basis* of the system in eqs. (4.5) to (4.7) iff $|B| = m$ and the columns indexed by B are linearly independent. We say that x is a *basic solution* with respect to B if $Ax = b$ and $x_j = 0, \forall j \notin B$. x is called *feasible* if $x_j \geq 0, \forall j \in \{1, ..., m\}$. We say that B is *feasible*, if there exists feasible x with respect to B. x is called *degenerate*, if it has less than m non-zero entries. We say that B is a *standard basis*, if the columns corresponding to B form the standard basis of the space of all m-dimensional vectors of real numbers $\mathbb{R}^m$. Two bases are called *neighbouring*, if they differ in exactly one index. Every basis specifies exactly one solution. The solution x can, however, correspond to several bases, this happens, when x is degenerate. The basic solutions form the vertices of the convex polyhedron specified by the constraints (4.6), (4.7), additionally optimal solution to the system in eqs. (4.5) to (4.7) must appear in at least one vertex of the polyhedron. The Simplex algorithm traverses these vertices, until such optimum is found.

For simplicity, let us represent the problem as the simplex table. Let

$$\bar{A} = \begin{pmatrix} c^\top & d \\ A & b \end{pmatrix} \tag{4.8}$$

be the matrix representing the simplex table, let $\bar{a}_{ij}$ be the entry in i^{th} row and j^{th} column of $\bar{A}$. Let us denote the pair $[B, \bar{A}]$ of simplex table $\bar{A}$ and its basis B as a *simplex dictionary*.

To solve the system in eqs. (4.5) to (4.7), the Simplex algorithm traverses the graph, where vertices form dictionaries of the system, with edges connecting dictionaries with neighbouring bases, starting in some given initial dictionary $[B, \bar{A}]$ with standard basis B (if no such basis exists in $\bar{A}$, additional preprocessing step needs to be made, for details see e.g. [6]).

The transition from standard basis B in $[B, \bar{A}]$ to a neighbouring standard basis B' in $[B', \bar{A}']$ is done via pivoting steps. A pivoting step chooses a pair (l, e) where e is the index of the new variable entering the basis and l is the index of the row corresponding to the non-zero coefficient of the variable which will leave the basis in $\bar{A}$. After acquiring this pair, we obtain $\bar{A}'$ from the matrix $\bar{A}$ in the following way, (1) we divide the whole l^{th} row of $\bar{A}$ by $\bar{a}_{l,e}$ and (2) for all $i \neq l$ we subtract $\bar{a}_{i,e}$ times the l^{th} row from the i^{th} row. We do this in order to make sure that B' will form the standard basis of $\bar{A}'$. The pivoting step is repeated, until an optimal solution is found.

To find the pivot (l, e) we use the Bland's rule.

Definition 6. *The Bland's rule [3] performs feasible pivots. Let B be a basis inducing feasible solution.*

1. *Choose the entering basic variable index e such that e is the smallest index with* $c_e < 0$. *If there is no such* c_e *optimum has been reached.*
2. *Choose the index of the row corresponding to the leaving basic variable* $l \in \arg\min_{l|a_{l,e}>0} \frac{b_e}{a_{l,e}}$. *If there are more such indices, choose the smallest one.*

The pivot (l, e) maintains the feasibility of the basis, additionally we are guaranteed to find the optimal solution of the LP given in eqs. (4.5) to (4.7) in finite number of pivoting steps [3].

4.3.2 Enumeration of All Vertices.

Now we are ready to describe the algorithm for enumerating all vertices of a polyhedron P described by linear inequalities

$$Cx \leq b \tag{4.9}$$
$$x \geq 0. \tag{4.10}$$

First, we transform the constraints to the form shown in eqs. (4.6) and (4.7) as follows

$$\min_{x,x_s} \quad -1^\top x \tag{4.11}$$
$$Ix_s + Cx = b \tag{4.12}$$
$$x_s \geq 0,\ x \geq 0, \tag{4.13}$$

where x_s are slack variables used to transfer the inequalities to equalities and I is a unit matrix of corresponding dimension. The objective (4.11) is added to allow handling of degenerate optimal solutions [2]. Next step is to enumerate all the feasible dictionaries of the system in eqs. (4.11) to (4.13). In order to do that, consider again a graph where vertices are the dictionaries. Two vertices are adjacent if the two corresponding bases are neighbouring. There is a unique path consisting of Bland's pivots from any $[B, \bar{A}]$ to some optimal dictionary [3]. The set of all such paths gives us a spanning tree of this graph. We define a reverse Bland's pivot in a following way. Consider some feasible $[B, \bar{A}]$. Let (l, e) be the pivot obtained by applying the Bland's rule to $[B, \bar{A}]$ and let $[B', \bar{A}']$ be the result of applying the pivoting step and the updates of $\bar{A}$. We call (l, e) a reverse Bland's pivot for $[B', \bar{A}']$. Suppose we start at a given optimal $[B, \bar{A}]^*$ and explore reverse Bland's pivots in lexicographic order. This corresponds to the depth-first search of the spanning tree defined above. When moving down the tree, each feasible $[B, \bar{A}]$, corresponding to the vertex of P, is encountered exactly once.

Special care needs to be taken if there exists more than one optimal dictionary $[B, \bar{A}]^*$. This happens when the optimal dictionary is degenerate. Then, instead of a spanning tree we obtain a spanning forest, since the pivot algorithm terminates when any optimal solution is found. Therefore, the procedure described in the previous section must be applied to each optimal dictionary. Fortunately, from any optimal dictionary we can generate all optimal dictionaries. This is done by additional optimization over all $x_j = 0, j \in B$ on a dictionary created by modification of the current optimal dictionary using procedure similar to the one described above.

For more detailed discussion of this algorithm, including variants using different pivots and degeneracy handling, we refer the reader to [2].

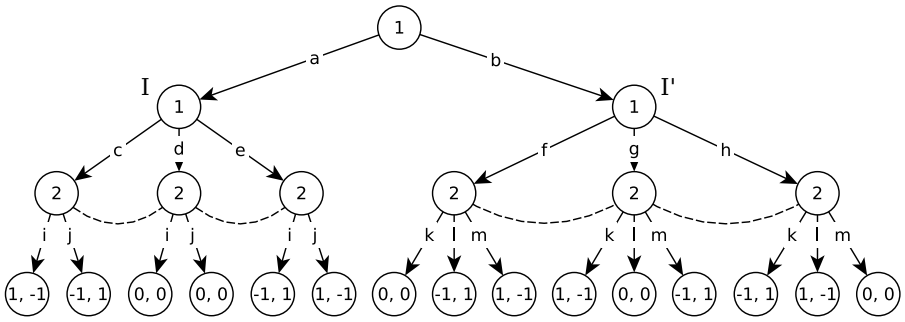

Figure 4.2: A game where extreme Nash equilibrium strategies do not generate the needed behavior.

4.4 Algorithm for Computing Nash Equilibrium in A-loss Recall Games

Theorem 1 allows us to compute the Nash equilibrium behavioral strategy for player i of the two-player zero-sum A-loss recall game G without absent mindedness by following steps.

1. Create the coarsest perfect recall refinement G' of G.
2. Compute all extreme Nash equilibrium strategies for player i of G'.
3. Find a Nash equilibrium strategy prescribing the same behavioral strategy to all $\mathcal{I}_i^{G'}$, which are grouped into an imperfect recall information set in G, if no such equilibrium exists return failure.

The coarsest perfect recall refinement can be found by a traversal over $I_i \in \mathcal{I}_i^G$. For every such I_i we create a new information set in G' according to $H(I_i)$.

The enumeration of all extreme Nash equilibrium strategies for i is done in the following way. (i) we compute the value of the game G' by solving the sequence-form LP, (ii) we fix the value of the game by adding additional constraint to this LP and (iii) we find all the extreme solutions formed by the constraints of this new LP, e.g., using the algorithm due to Avis et. al. [2] described above. By the addition of the constraint in step (ii) we we make sure that all the vertices of the convex polyhedron formed by the constraints correspond to extreme Nash equilibrium strategies of i (using the property that expected value of playing according to any Nash equilibrium is the same for all players in zero-sum games).

From the step 2 we obtain only the set of extreme Nash equilibrium strategies in realization plans $\mathcal{E}_i = \{r_i^1, ..., r_i^n\}$. This is not enough, since the desired behavior can be generated by degenerate equilibria. We therefore need to check if our desired equilibrium strategy is formed by some convex combination of entries of $\mathcal{E}_i$, since the entries of $\mathcal{E}_i$ form the vertices of the convex polyhedron representing all the possible equilibrium strategies for i. This is done by additional mixed integer linear program (MILP), where we assign a coefficient $\lambda_k \in \{\lambda_1, ..., \lambda_n\}$ to every entry in $\mathcal{E}_i$. Then, for every group of information sets $\{I_i^1, ..., I_i^m\}$

in G', unified to some imperfect recall causing I_i in G, we create constraints

$$\sum_{k=1}^{n} \lambda_k b_i^k(I_i^j) = \sum_{k=1}^{n} \lambda_k b_i^k(I_i^{j+1}), \quad \forall j \in \{1, ..., m-1\} \tag{4.14}$$

where $b_i^k(I_i^j)$ stands for the behavioral strategy r_i^k generates in I_i^j. In order to make sure that we compute only convex combinations we add constraints

$$\sum_{k=1}^{n} \lambda_k = 1 \tag{4.15}$$

$$\lambda_k \geq 0, \quad \forall k \in \{1, ..., n\} \tag{4.16}$$

Since the $b_i^k(I_i^j)$ is created from r_i^k, it may happen that $b_i^k(I_i^j)$ is not defined, because r_i^k may not visit I_i^j. To handle this issue, we need to add partial coefficients $\lambda_k^1, ..., \lambda_k^{|A(I_i^j)|}$. These coefficients represent that player i can play arbitrarily in I_i^j according to r_i^k. We assign every λ_k^j to the behavioral strategy, which plays j^{th} action from $A(I_i^j)$ with probability equal to 1. However, we are connecting several strategies together, therefore we need to make sure that player i can play arbitrarily in I_i^j only when I_i^j gets visited with probability 0. Let $\pi_i^k(I_i^j)$ denote the probability that I_i^j will be visited according to r_i^k.

$$\sum_{j=1}^{|A(I_i^j)|} \lambda_k^j \geq \lambda_k - b_{I_i^j}, \quad \sum_{j=1}^{|A(I_i^j)|} \lambda_k^j \leq \lambda_k + b_{I_i^j} \tag{4.17}$$

$$\sum_{j=1}^{|A(I_i^j)|} \lambda_k^j \geq -1 + b_{I_i^j}, \quad \sum_{j=1}^{|A(I_i^j)|} \lambda_k^j \leq 1 - b_{I_i^j} \tag{4.18}$$

$$\lambda_k^j \geq 0, \quad \forall j \in \{1, ..., |A(I_i^j)|\} \tag{4.19}$$

$$b_{I_i^j} \geq \sum_{k=1}^{n} \lambda_k \pi_i^k(I_i^j), \quad b_{I_i^j} \in \{0, 1\} \tag{4.20}$$

In constraints (4.17) to (4.18) we make sure that the coefficients λ_k^j sum to λ_k if I_i^j is visited with zero probability, to 0 otherwise. We do this using binary variable $b_{I_i^j}$ for every information set unified to some imperfect recall information set in G. In constraint (4.20) we make sure that $b_{I_i^j}$ is 1 (and therefore the partial coefficients cannot be used) when I_i^j gets visited with positive probability.

If we construct an infeasible MILP by this approach, we know that there exists no Nash equilibrium in behavioral strategies of G. Otherwise, we obtain coefficients which we can use to construct the Nash equilibrium behavioral strategy for i in G. By running this procedure for every $i \in \mathcal{P}$ we obtain the whole strategy profile forming the Nash equilibrium in behavioral strategies of G.

Finally, to explain the intuition behind the MILP constructing the final solution, let us consider the game G' from Figure 4.2. The extreme equilibria for player 1 in realization plans are the following $\{r_1^1(a, c) = 0.5, r_1^1(a, e) = 0.5\}$, $\{r_1^2(a, d) = 1\}$ and $\{r_1^3(b, f) = r_1^3(b, g) =$

$r_1^3(b,h) = \frac{1}{3}\}$. There are several Nash equilibria in behavioral strategies of A-loss recall game G created from G' by unifying I and I'. Namely, any convex combination of r_1^1 and r_1^2, since action b is played with zero probability. The states reached by this action, therefore, need not be considered since any strategy is optimal in these states. Additionally playing according to r_1^3 also forms a Nash equilibrium in behavioral strategies in G for player 1, because action a is not played. Finally, playing according to r_1^1, r_1^2 and r_1^3 with weights λ_1, λ_2 and λ_3 respectively, where $\frac{\lambda_1}{\lambda_2} = 2$ also creates a Nash equilibrium in behavioral strategies of G for player 1. In the first two equilibria of G, we use the property that player 1 can play arbitrarily in unreachable information sets, represented as the partial coefficients in the MILP. In the last case, however, both I and I' are reached with non-zero probability (if all the coefficients are positive), therefore the partial coefficient are set to zero and we consider only the behavior consistent with Nash equilibrium.

4.5 Complexity Analysis

The complexity of finding behavioral maxmin strategy in zero-sum games with imperfect recall is shown to be NP-hard [18], even finding the pure maxmin strategies in zero sum imperfect recall games is shown to be Σ_2^p complete [18]. Notice that finding behavioral maxmin strategy is equal to finding a Nash equilibrium in behavioral strategies of the zero-sum imperfect recall game if it exists.

Let us now discuss the complexity of the algorithm described in the previous section. The creation of the coarsest perfect recall refinement can be done in $O(n)$ where n is the number of states of the game G. The complexity of finding all the optimal extreme solutions of the convex polyhedron formed by the constraints of the sequence-form LP using the algorithm due to Avis is $O(md(m+d)v)$ [2], where m is the number of constraints, d is the number of variables and v is the number of vertices of the polyhedron. The number of vertices can be at most $\binom{m}{d}$, where $m = |\Sigma_{-i}| + 2|\mathcal{I}_i| + |\Sigma_i|$ and $d = |\Sigma_i| + 2|\mathcal{I}_{-i}|$ for the case of sequence-form LP computing the Nash equilibrium strategy of i.

The MILP finding the final solution has number of continuous variables equal to the number of equilibria found in the previous step, the number of constraints and binary variables is proportional to the number of information sets of G' which are connected to some imperfect recall information set in G. The complexity of solving such MILP is $O(2^{|\mathcal{I}_i|}p(\binom{m}{d}))$ where $p(x)$ is some polynomial function of x, e.g., using branch and bound algorithm [7].

The worst case complexity of the algorithm is $O(2^{|\mathcal{I}_i|}p(\binom{m}{d}) + md(m+d)\binom{m}{d})$, where m and d have the values defined above.

Chapter 5

Regret Minimization

In this chapter we briefly describe the ideas behind external regret and the Counterfactual regret minimization algorithm [32]. Finally, we provide modifications of the tree traversal of the Counterfactual regret minimization algorithm, which guarantee convergence to behavioral Nash equilibrium in A-loss recall games, when such equilibrium exists.

5.1 External Regret

Given a sequence of behavioral strategy profiles $b^1, ..., b^T$, the external regret for player i,

$$R_i^T = \max_{b'_i \in \mathcal{B}_i} \sum_{t=1}^{T} (u_i(b'_i, b^t_{-i}) - u_i(b^t_i, b^t_{-i})), \tag{5.1}$$

is the amount of utility player i could have gained had she played the best single strategy in hindsight for all time steps $t \in \{1, ..., T\}$. An algorithm minimizes regret, or is a no-regret algorithm, for player i if the average positive regret approaches zero; i.e. $lim_{T\to\infty} R_i^{T,+}/T = 0$, where $x^+ = \max(x, 0)$

5.2 Counterfactual Regret Minimization

Counterfactual regret is defined in each iteration t, information set I and action a as $r_i^t(I, a) = v_i(b^t_{I\to a}, I) - v_i(b^t, I)$, where $b^t_{I\to a}$ is the profile b^t except for I, where a is played and

$$v(b, I) = \sum_{z \in Z_I} u_i(z)\pi^b_{-i}(z[I])\pi^b(z[I], z), \tag{5.2}$$

where $\pi^b(h)$ is a probability that h will be reached when players play according to a strategy profile b, with π^b_i and π^b_{-i} being the contribution of player i and all the players except i, respectively. $z[I]$ stands for state h which needs to be visited in I in order to reach leaf z. Finally, $\pi^b(h, h')$ stands for the probability that h' will be reached from h when players play according to b.

Immediate counterfactual regret is defined as $R_i^T(I, a) = \sum_{t=1}^{T} r_i^t(I, a)$.

To update the strategy of players between iterations, regret matching is used.

$$b^{T+1}(I,a) = \frac{R_i^{T,+}(I,a)}{\sum_{a' \in A(I)} R_i^{T,+}(I,a')}. \tag{5.3}$$

Regret matching is a no-regret learner that minimizes the per-information set immediate counterfactual regret,

$$\max_{a \in A(i)} \frac{R_i^T(I,a)}{T} \leq \frac{\Delta_i \sqrt{|A(I)|}}{\sqrt{T}} \tag{5.4}$$

where $\Delta_i = \max_{z.z' \in Z} u_i(z) - u_i(z')$ [32].

In games having perfect recall, minimizing the immediate counterfactual regrets at every information set in turn minimizes the average external regret. This holds because perfect recall implies that the regret is bounded by the sum of the positive parts of the immediate counterfactual regrets [32],

$$R_i^T \leq \sum_{I \in \mathcal{I}_i} \max_{a \in A(I)} R_i^{T,+}(I,a), \tag{5.5}$$

and thus

$$\frac{R_i^T}{T} \leq \frac{\Delta_i |\mathcal{I}_i| \sqrt{|A_i|}}{T}, \tag{5.6}$$

where $|A_i| = \max_{I \in \mathcal{I}_i} |A(I)|$.

While equation (5.4) still holds in IR games, (5.5) and (5.6) are not guaranteed to hold.

5.3 Minimizing External Regret in A-loss Recall Games

In this section we present a no-regret algorithm for A-loss recall two-player zero-sum games. The algorithm is build on following ideas. (1) the regret updates are based on the deviation from the current strategy through the whole tree, (2) approximating the possibly non-convex loss function caused by the imperfect recall by the tangent hyperplane and (3) using a best responding opponent in the coarsest perfect recall refinement when computing strategy for player i (similarly to CFRBR [13]) in order to avoid convergence to exploitable strategies.

Proposition 1. *The inequality* $R_i^T \leq \sum_{I \in \mathcal{I}_i} \max_{a \in A(I)} R_i^{T,+}(I,a)$ *does not hold even for zero-sum two-player games with A-loss recall.*

Proof. In the game in Figure 5.1 the CFR starting with uniform strategy will never perform any strategy update, since no player regrets playing differently in any of her information sets. The external regret, however, is non-zero as player 1 regrets not playing pure behavioral strategy a, h instead of the uniform strategy. □

Proposition 2. *If* $R_i^{T,+}/T = 0$, *then* $\sum_{I \in \mathcal{I}_i} \max_{a \in A(I)} R_i^{T,+}(I,a) = 0$. *The opposite does not hold.*

We present an alternative procedure for minimizing the external regret in two-player zero-sum A-loss recall games which builds on the CFRBR algoirthm [13]. In our approach we assume player i to learn no-regret strategy in A-loss recall game G, while her opponent

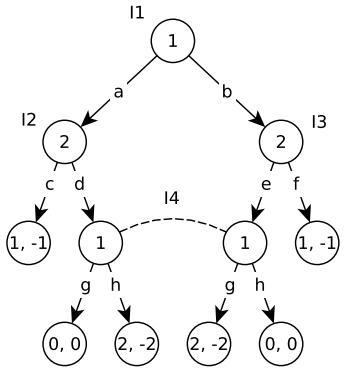

Figure 5.1: A-loss recall game where standard CFR does not converge to zero external regret.

always plays a best response to her current strategy in the coarsest perfect recall refinement of G. When updating the regrets of player i in iteration t of the regret minimizing algorithm, we start by finding a best response b_i^{BR} in G to the best response of $-i$, b_{-i}^t obtained in G' against the current strategy of i from regret matching (note that the best response computation of i must enforce some fixed ordering of actions in every information set and use this ordering as a priority when there are more actions with the same expected outcome). We then proceed to update the regrets only in the information sets visited when playing according to b_i^{BR}. Let us first define the loss $v'_{b_i^{BR}}((b_i, b_{-i}^t), I) = v_i((b_i^{BR}, b_{-i}^t), I) - v_i((b'_i, b_{-i}^t), I)$, where $b'_i = \sum_{a \in A(I)} b_i(I, a) b_i^{a-BR}$ is a behavioral strategy created as a convex combination of the pure behavioral strategies b_i^{a-BR}, with coeficients given by $b_i(I, a)$ and where b_i^{a-BR} is a best response of i to b_{-i}^t when i is forced to play a in I. To update the regrets in every such information set $I \in \mathcal{I}_i$ we use

$$\tilde{r}^t(I, a) = \tilde{v}_{b_i^{BR}}((b_{i,I \to a}^t, b_{-i}^t), I), \tag{5.7}$$

where $\tilde{v}_{b_i^{BR}}((b_i, b_{-i}^t), I) = \frac{\partial v'_{b_i^{BR}}}{\partial b_i(I)}((b_i, b_{-i}^t), I) \cdot (b_i(I) - b_i^{BR}(I))$. We define the loss in terms of $\tilde{v}_{b_i^{BR}}((b_i, b_{-i}^t), I)$ since $v'_{b_i^{BR}}((b_i^t, b_{-i}^t), I)$ is a non-convex function which breaks the assumptions in [9] used to prove the convergence of our algorithm. $\tilde{v}_{b_i^{BR}}((b_i^t, b_{-i}^t), I)$ creates a tangent hyperplane to $v'_{b_i^{BR}}((b_i^t, b_{-i}^t), I)$ in point $b_i^{BR}(I)$, the loss in I ,therefore, becomes linear for a fixed iteration and the regret bounds introduced in [9] can be directly applied.

The immediate total regret is defined equally as in the counterfactual case as $\tilde{R}_i^T(I, a) = \sum_{t=1}^T \tilde{r}_i^t(I, a)$. The regret matching is used to compute the strategy for i in every iteration from immediate total regret in the following way

$$b^{T+1}(I, a) = \begin{cases} \frac{\tilde{R}_i^{T,+}(I,a)}{\sum_{a' \in A(I)} \tilde{R}_i^{T,+}(I,a')} & \text{If } \sum_{a' \in A(I)} \tilde{R}^{T,+}(I, a') \geq 0 \\ \frac{1}{|A(I)|} & \text{otherwise} \end{cases} \tag{5.8}$$

Theorem 2.

$$\frac{1}{T} \sum_{I \in \mathcal{I}_i} \max_{a \in A(I)} \tilde{R}_i^{T,+}(I, a) \leq \frac{\Delta_i |\mathcal{I}_i| \sqrt{|A_i|}}{\sqrt{T}}$$

,where Δ_i is the maximum difference between utilities in the game tree.

Proof. With the modifications to the loss function described above we obtain linear loss function in hypothesis space being the space of behavioral strategies, therefore, the bound directly follows from [9]. □

Furthermore, we argue that when both players use regret matching, they can learn exploitable strategies, which in turn prevents the equilibrial average strategy from becoming zero external regret strategy. Consider as an example again the game in Figure 5.1. When starting from uniform strategy for both players we get oscillating updates where when player 1 increases the probability of playing (a, h), player 2 increases the probability of playing (c, e), which in turn forces player 1 to switch to (b, g) etc. Therefore, the average strategy of both players converges to a uniform strategy profile. This strategy profile yields zero external regret for player 2. Player 1, however, regrets not playing pure behavioral strategy (a, h) or (b, g), even though player 1 plays strategy which forms a part of Nash equilibrium. This is caused by the fact that during the computation player 1's best responses switch in every iteration and his external regret for playing any mix in the root is strictly larger than playing deterministically (a, h) or (b, g). The reason for this is that player 2 plays exploitable strategy, since if he would play (c, f) (which is his only equilibrial strategy) the nonconvexity for player 1 would disappear and her strategy would indeed have zero-regret. We argue that if there is a Nash equilibrium in two-player zero-sum A-loss recall game G, then when using best responding player $-i$ in the coarsest perfect recall refinement G' of G, player i using regret matching always converges to a Nash equilibrium of G, if such equilibrium exists.

Conjecture 1. *When $\frac{1}{T}\sum_{I\in\mathcal{I}_i}\max_{a\in A(I)}\tilde{R}_i^{T,+}(I,a) \leq \epsilon$, obtained from the procedure described above, the average strategy of i forms an $k\epsilon$-Nash equilibrium when paired with some equilibrium strategy of the opponent, for some $k > 0$.*

Chapter 6

Computing Maxmin Strategies in Imperfect Recall Games

In this chapter, we present the first algorithm finding maxmin strategies in imperfect recall games without absentmindedness. Contrary to Nash equilibrium the maxmin strategies are guaranteed to exist in imperfect recall games.

We now state our main mathematical framework that we later exploit to formulate the branch-and-bound search algorithm for computing maxmin strategies. The main idea is to add bilinear constraints into the sequence form LP in order to restrict to imperfect recall strategies. We first show a method for approximating bilinear terms based on Multiparametric Disaggregation Technique (MDT) [20]. We then formulate the bilinear program for the opponent with A-loss recall, approximate the bilinear terms using MDT, and prove the approximation bounds. In appendix, we describe the necessary changes in games where player 2 has general imperfect recall without AM.

6.1 Approximating Bilinear Terms

Here we describe the Multiparametric Disaggregation Technique (MDT) [20] 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 [20].

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 (6.1) ensures that for each position ℓ there is exactly one digit chosen. All digits must sum to b (Constraint (6.3)). 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 (6.6).

$$\sum_{k=0}^{9} w_{k,\ell} = 1 \qquad \ell \in \mathbb{Z} \tag{6.1}$$

$$w_{k,\ell} \in \{0,1\} \tag{6.2}$$

$$\sum_{\ell\in\mathbb{Z}}\sum_{k=0}^{9} 10^{\ell}\cdot k \cdot w_{k,\ell} = b \tag{6.3}$$

$$c^{L}\cdot w_{k,\ell} \le \hat{c}_{k,\ell} \le c^{U}\cdot w_{k,\ell} \qquad \forall \ell \in \mathbb{Z}, \forall k = \{0,\dots,9\} \tag{6.4}$$

$$\sum_{k=0}^{9} \hat{c}_{k,\ell} = c \qquad \forall \ell \in \mathbb{Z} \tag{6.5}$$

$$\sum_{\ell\in\mathbb{Z}}\sum_{k=0}^{9} 10^{\ell}\cdot k \cdot \hat{c}_{k,\ell} = a \tag{6.6}$$

This is an exact formulation that requires infinite sums and infinite number of constraints. However, by restricting the set of all possible positions ℓ to a finite set $\{P_L,\dots,P_U\}$ we get a lower bound approximation. Following the approach in [20] we can extend the lower bound formulation to compute an upper bound:

Constraints (6.1), (6.4), (6.5)

$$\sum_{\ell\in\{P_L,\dots,P_U\}}\sum_{k=0}^{9} 10^{\ell}\cdot k \cdot w_{k,\ell} + \Delta b = b \tag{6.7}$$

$$0 \le \Delta b \le 10^{P_L} \tag{6.8}$$

$$\sum_{\ell\in\{P_L,\dots,P_U\}}\sum_{k=0}^{9} 10^{\ell}\cdot k \cdot \hat{c}_{k,\ell} + \Delta a = a \tag{6.9}$$

$$c^{L}\cdot\Delta b \le \Delta a \le c^{U}\cdot\Delta b \tag{6.10}$$

$$\left(c-c^{U}\right)\cdot 10^{P_L} + c^{U}\cdot\Delta b \le \Delta a \le \left(c-c^{L}\right)\cdot 10^{P_L} + c^{L}\cdot\Delta b \tag{6.11}$$

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 (6.7)–(6.8)). 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 (6.10)–(6.11).

6.2 Bilinear Sequence Form Against A-loss Recall Opponent

$$\max_{x,r,v} \; v(root, \emptyset) \tag{6.12}$$

$$s.t. \quad r(\emptyset) = 1 \tag{6.13}$$

$$0 \leq r(\sigma_1) \leq 1 \qquad \forall \sigma_1 \in \Sigma_1 \tag{6.14}$$

$$\sum_{a \in A(I_1)} r(\sigma_1 a) = r(\sigma_1) \qquad \forall \sigma_1 \in \Sigma_1, \forall I_1 \in \mathsf{inf}_1(\sigma_1) \tag{6.15}$$

$$\sum_{a \in A(I_1)} x(I_1, a) = 1 \qquad \forall I_1 \in \mathcal{I}_1^{IR} \tag{6.16}$$

$$0 \leq x(I_1, a) \leq 1 \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall a \in A(I_1) \tag{6.17}$$

$$r(\sigma_1) \cdot x(I_1, a) = r(\sigma_1 a) \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall \sigma_1 \in \mathsf{seq}_1(I_1), \forall a \in A(I_1) \tag{6.18}$$

$$\sum_{I' \in \mathsf{inf}_2(\sigma_2 a)} v(I', \sigma_2 a) + \sum_{\sigma_1 \in \Sigma_1} g(\sigma_1, \sigma_2 a) r_1(\sigma_1) \leq v(I, \sigma_2) \qquad \forall I \in \mathcal{I}_2, \forall \sigma_2 \in \mathsf{seq}_2(I), \forall a \in A(I) \tag{6.19}$$

In Constraints (6.12)–(6.19) we present a bilinear reformulation of the sequence-form LP due to [29] applied to the information set structure of the imperfect recall game. The objective of player 1 is to find a strategy that maximizes the expected utility of the game. The strategy is represented using variables r that assign probability to sequence $\sigma_1 \in \Sigma_1$. $r(\sigma_1)$ is the probability that σ_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 (6.14)–(6.15). Finally, the strategies of player 1 are constrained by the best-responding opponent that is selecting in each $I \in \mathcal{I}_2$ and for each $\sigma_2 \in \mathsf{seq}_2(I)$ an action that minimizes the expected value in I, when I is reached by σ_2 (Constraint (6.19)) (note that this formulation of the constraints ensures that the opponent plays the best response in the coarsest perfect recall refinement of the game being solved). 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 $g(\sigma_1, \sigma_2) = \sum_{z \in \mathcal{Z} | \mathsf{seq}_1(z) = \sigma_1 \wedge \mathsf{seq}_2(z) = \sigma_2} u(z)\mathcal{C}(z)$.

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 $x(I_i, a)$ for each $a \in A(I_i)$ (Constraints (6.16)–(6.17)). To ensure that the realization probabilities induce the same behavioral strategy in I_i, we add bilinear constraint $r(\sigma_i a) = x(I_i, a) \cdot r(\sigma_i)$ (Constraint (6.18)). Notice that we can use this formulation thanks to Lemma 1.

6.2.1 Upper Bound MILP

The upper bound formulation of the bilinear program follows the MDT example and uses ideas similar to Section 6.1. In accord with the MDT we represent every variable $x(I_1, a)$ 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 6.1 and are used for the digit-wise discretization of $x(I_1, a)$. 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 $x(I_1, a)$ to attain an arbitrary value from $[0, 1]$ interval using a finite number of digits of precision, we add an additional real variable $0 \leq \Delta x(I_1, a) \leq 10^{-P}$ that can span the gap between two adjacent discretization points. Constraints (6.23) and (6.24) describe this loosening. Variables $\Delta x(I_1, a)$ also have to be propagated to bilinear terms $r(\sigma_1) \cdot x(I_1, a)$ involving $x(I_1, a)$. We cannot represent the product $\Delta r(\sigma_1 a) = r(\sigma_1) \cdot \Delta x(I_1, a)$ exactly and therefore we give bounds based on the McCormick envelope (Constraints (6.28)–(6.29)).

$$\max_{x,r,v} v(root, \emptyset) \tag{6.20}$$

$$s.t. \quad \text{Constraints (6.13) - (6.17) , (6.19)}$$

$$w_{k,\ell}^{I_1,a} \in \{0,1\} \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall a \in A(I_1), \quad \forall k \in \{0,\dots,9\}, \forall \ell \in \{-P,\dots,0\} \tag{6.21}$$

$$\sum_{k=0}^{9} w_{k,\ell}^{I_1,a} = 1 \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall a \in A(I_1), \quad \forall \ell \in \{-P,\dots,0\} \tag{6.22}$$

$$\sum_{\ell=-P}^{0} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot w_{k,\ell}^{I_1,a} + \Delta x(I_1, a) = x(I_1, a) \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall a \in A(I_1) \tag{6.23}$$

$$0 \leq \Delta x(I_1, a) \leq 10^{-P} \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall a \in A(I_1) \tag{6.24}$$

$$0 \leq \hat{r}(\sigma_1)_{k,\ell}^{a} \leq w_{k,\ell}^{I_1,a} \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall a \in A(I_1), \quad \forall \sigma_1 \in \mathsf{seq}_1(I_1), \forall \ell \in \{-P,\dots,0\} \tag{6.25}$$

$$\sum_{k=0}^{9} \hat{r}(\sigma_1)_{k,\ell}^{a} = r(\sigma_1) \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall \sigma_1 \in \mathsf{seq}_1(I_1), \quad \forall \ell \in \{-P,\dots,0\} \tag{6.26}$$

$$\sum_{\ell=-P}^{0} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{r}(\sigma_1)_{k,\ell}^{a} + \Delta r(\sigma_1 a) = r(\sigma_1 a) \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall \sigma_1 \in \mathsf{seq}_1(I_1), \quad \forall a \in A(I_1) \tag{6.27}$$

$$0 \leq \Delta r(\sigma_1 a) \leq 10^{-P} \cdot r(\sigma_1) \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall \sigma_1 \in \mathsf{seq}_1(I_1), \quad \forall a \in A(I_1) \tag{6.28}$$

$$0 \leq \Delta r(\sigma_1 a) \leq \Delta x(I_1, a) \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall \sigma_1 \in \mathsf{seq}_1(I_1), \quad \forall a \in A(I_1) \tag{6.29}$$

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. The MILP formulation for obtaining a lower bound is presented in the appendix. 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 $10 \cdot |\mathcal{I}_1^{IR}| \cdot A_1^{max} \cdot P$, where $A_1^{max} = \max_{I \in \mathcal{I}_1} |A_1(I)|$.

6.2.2 Theoretical Analysis of the Upper Bound MILP

The variables $\Delta x(I_1, a)$ 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 $a \in A(I_1)$ 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 $b^1(I_1), \ldots, b^k(I_1)$ be behavioral strategies in the imperfect recall information set $I_1 \in \mathcal{I}_1^{IR}$ that can be reached by sequences $\sigma^1, \ldots, \sigma^k$. These probability distributions can be obtained from the realization plan as $b^j(I_1, a) = r(\sigma^j a)/r(\sigma^j)$ for $\sigma^j \in \mathsf{seq}_1(I_1)$ and $a \in A(I_1)$. We will omit the information set and use $b(a)$ whenever it is clear from the context. If the imperfect recall is violated in I_1, $b^j(a) \neq b^l(a)$ for some j, l and action $a \in A(I_1)$.

Proposition 3. *It is always possible to construct a strategy $b(I_1)$ such that $\|b(I_1) - b^j(I_1)\|_1 \leq |A(I_1)| \cdot 10^{-P}$ for every j.*[1]

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

$$\underline{r}(\sigma_1 a) = \sum_{l=-P}^{0} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{r}(\sigma_1)^a_{k,\ell} \tag{6.30}$$

$$\underline{x}(I_1, a) = \sum_{l=-P}^{0} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot w^{I_1,a}_{k,\ell}. \tag{6.31}$$

Constraints (6.25) and (6.26) 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 $b^1, \ldots, b^k$ is at most 10^{-P}. Taking any of the behavioral strategies $b^1, \ldots, b^k$ as the corrected behavioral strategy $b(I_1)$ therefore satisfies $\|b(I_1) - b^j(I_1)\|_1 \leq \sum_{a \in A(I_1)} 10^{-P} = |A(I_1)| \cdot 10^{-P}$. □

We now connect the distance of a corrected strategy $b(I_1)$ from a set of behavioral strategies $b^1(I_1), \ldots, b^k(I_1)$ in $I_1 \in \mathcal{I}_1^{IR}$ to the expected value of the strategy. First, we bound this error in a single node.

Lemma 6. *Let $h \in I_1$ be a history and b^1, b^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:*

$$|v_{b^1}(h) - v_{b^2}(h)| \leq \frac{v_{max}(h) - v_{min}(h)}{2} \cdot \|b^1(I_1) - b^2(I_1)\|_1,$$

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

where $v_{b^j}(h)$ is the maxmin value $u(b^j, b_2^{BR})$ of strategy b^j of player 1 given the play starts in h.

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

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

We know that $C^+ = C^-$ (as strategies have to be probability distributions). Moreover we know that $\|b^1(I_1) - b^2(I_1)\|_1 = C^+ + C^-$. 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

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

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

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

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

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

Theorem 3. *The error of the Upper Bound MILP is bounded by*

$$\epsilon = 10^{-P} \cdot d \cdot A_1^{max} \cdot \frac{v_{max}(\emptyset) - v_{min}(\emptyset)}{2},$$

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, $A_1^{max} = \max_{I_1 \in \mathcal{I}_1^{IR}} |A(I_1)|$ 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 3 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.

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

where b^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, b^{IR} is its corrected version and $u_h(b)$ is the expected utility of a play starting in history h when player 1 plays according to b 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 $L(h) = \sum_{a \in A(h)} L(h \cdot a) \cdot \mathcal{C}(h \cdot a)/\mathcal{C}(h)$. In the worst case, the chance player selects the child with the highest associated loss, therefore

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

(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, $L(h) \leq \max_{a \in A(n)} L(h \cdot a)$. 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 $L(h) = \sum_{a \in A(h)} b^0(h, a) L(h \cdot a)$. Once again in the worst case player 1's behavioral strategy $b^0(h)$ selects deterministically the child node with the highest associated loss, therefore $L(h) \leq \max_{a \in A(h)} L(h \cdot a)$.

(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 b^{-h} be a behavioral strategy where corrective steps have been taken for successors of h and let us construct a strategy b^h where the strategy was corrected in the whole subtree of h (i.e. including h). Note that ultimately we want to construct strategy $b^\emptyset = b^{IR}$.

We know that values of children have been decreased by at most $\max_{a \in A(h)} L(h \cdot a)$, hence $v_{b^0}(h) - v_{b^{-h}}(h) \leq \max_{a \in A(h)} L(h \cdot a)$. Then we have to take the corrective step at the node h and construct strategy b^h. From Lemma 6 and the observation about the maximum distance of behavioral strategies within a single imperfect recall information set I_1, we get:

$$\begin{aligned} v_{b^{-h}}(h) - v_{b^h}(h) &\leq \frac{v_{max}(h) - v_{min}(h)}{2} \cdot 10^{-P} |A_1(I_1)| \\ &\leq \frac{v_{max}(\emptyset) - v_{min}(\emptyset)}{2} \cdot 10^{-P} A_1^{max} \end{aligned}$$

The loss in the subtree of h is equal to $v_{b^0}(h) - v_{b^{-h}}(h)$ which is bounded by

$$\begin{aligned} L(h) &= v_{b^0}(h) - v_{b^h}(h) = [v_{b^{-h}}(h) - v_{b^h}(h)] + [v_{b^0}(h) - v_{b^{-h}}(h)] \leq \\ &\leq \frac{v_{max}(\emptyset) - v_{min}(\emptyset)}{2} \cdot 10^{-P} A_1^{max} + \max_{a \in A(h)} L(h \cdot a). \end{aligned}$$

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 ($[v_{max}(\emptyset) - v_{min}(\emptyset)] \cdot 10^{-P} A_1^{max}/2$), therefore the bound is $\epsilon = L(\emptyset) \leq [v_{max}(\emptyset) - v_{min}(\emptyset)] \cdot d \cdot 10^{-P} A_1^{max}/2$.

We now know that the expected value of the strategy we have found lies within the interval $[v^* - \epsilon, v^*]$, 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.

□

6.2.3 Lower Bound MILP

We follow the approach from Section 6.1 and apply the lower bound reformulation (Constraints (6.1) to (6.6)) to bilinear terms in Constraint (6.18). In accord with the MDT we represent every variable $x(I_1, a)$ using a finite number of digits. We use $P = -P_L$ digits of precision ($P_U = 0$ since we represent probabilities). Binary variables $w_{k,\ell}^{I_1,a}$ correspond to $w_{k,\ell}$ variables from the example shown in Section 6.1 and are used for the digit-wise discretization of $x(I_1, a)$. Finally, $\hat{r}(\sigma_1)_{k,\ell}^a$ correspond to $\hat{c}_{k,\ell}$ variables used to discretize the bilinear term $r(\sigma_1 a)$.

The main idea of the reformulation is the same as in the MDT example. Note that the number of binary variables is not exponential in the size of the game. There is a $x(I_1, a)$ variable for every action a available in an imperfect recall information set $I_1 \in \mathcal{I}_1^{IR}$. For every such variable, $10(P+1)$ binary variables are created by MDT.

$$\max_{x,r,v} v(root, \emptyset) \tag{6.32}$$

$$s.t. \quad \text{Constraints (6.13) - (6.17), (6.19)}$$

$$w_{k,\ell}^{I_1,a} \in \{0,1\} \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall a \in A(I_1), \forall k \in \{0,\dots,9\}, \forall \ell \in \{-P,\dots,0\} \tag{6.33}$$

$$\sum_{k=0}^{9} w_{k,\ell}^{I_1,a} = 1 \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall a \in A(I_1), \forall \ell \in \{-P,\dots,0\} \tag{6.34}$$

$$\sum_{\ell=-P}^{0} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot w_{k,\ell}^{I_1,a} = x(I_1, a) \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall a \in A(I_1) \tag{6.35}$$

$$0 \leq \hat{r}(\sigma_1)_{k,\ell}^a \leq w_{k,\ell}^{I_1,a} \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall a \in A(I_1), \forall \sigma_1 \in \mathsf{seq}_1(I_1), \forall \ell \in \{-P,\dots,0\} \tag{6.36}$$

$$\sum_{k=0}^{9} \hat{r}(\sigma_1)_{k,\ell}^a = r(\sigma_1) \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall \sigma_1 \in \mathsf{seq}_1(I_1), \forall \ell \in \{-P,\dots,0\} \tag{6.37}$$

$$\sum_{\ell=-P}^{0} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{r}(\sigma_1)_{k,\ell}^a = r(\sigma_1 a) \qquad \forall I_1 \in \mathcal{I}_1^{IR}, \forall \sigma_1 \in \mathsf{seq}_1(I_1), \forall a \in A(I_1) \tag{6.38}$$

6.2.3.1 Theoretical Analysis of the Lower Bound MILP

The reasoning behind derivation of the error bound for the Lower Bound MILP is very similar to the Upper Bound case. In this case however we are not interested in the maximum error induced by the correction of infeasible strategy, instead we will derive the error of a feasible strategy from the Lower Bound MILP by rounding the optimal strategy towards the discretization points.

Let b_1 be the optimal behavioral strategy of player 1. Let $I_1 \in \mathcal{I}_1^{IR}$ be an information set of player 1. The optimal behavioral strategy in the information set $b_1(I_1)$ lies within a $|A(I_1)|$ dimensional hypercube with the edge length 10^{-P} such that discretized strategies represented by variables $x(I_1, a)$ are its vertices. This means that the L1 distance between the optimal strategy $b_1(I_1)$ and the nearest vertex of the hypercube is maximized when the optimal strategy lies exactly in the center of the hypercube. This distance is at most $10^{-P} A_1^{max}/2$.

Lemma 7. *There is a valid imperfect recall strategy in the feasible space of Lower Bound MILP whose minmax value is*

$$v(b_1^*) - \frac{v_{max}(\emptyset) - v_{min}(\emptyset)}{4} \cdot 10^{-P} A_1^{max}$$

where $v(b_1^*)$ is the minmax value of the optimal strategy.

Proof. All nodes except the nodes with imperfect recall are handled identically as in the proof of Theorem 3, we will therefore provide just the difference in handling imperfect recall nodes.

If h is a node contained in come of the imperfect recall information sets, we need to represent its behavioral strategy by variables $x(I_1, a)$ using P digits of precision. We have already discussed that the nearest such strategy is $10^{-P} A_1^{max}/2$ away from it in the Manhattan distance which means that the combined loss of switching to a rounded strategy and the loss from the subtree of n is at most

$$L(h) \leq \frac{v_{max}(\emptyset) - v_{min}(\emptyset)}{4} \cdot 10^{-P} A_1^{max} + \max_{a \in A(h)} L(h \cdot a).$$

By the very same reasoning as in the proof of Theorem 3 this gives us the desired bound. □

6.3 Branch-and-Bound Algorithm

We now introduce a branch-and-bound (BnB) search for approximating minimax strategies, that exploits the observations 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. Finally, we discuss the modifications needed to find the minmax strategies, when the opponent does not have A-loss recall.

The BnB algorithm works on the linear relaxation of the Upper Bound MILP. It searches the BnB tree in a best first search manner (every node n of the BnB tree is evaluated using the upper bound value obtained by solving the LP corresponding to n). In every n the algorithm solves the corresponding LP, 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 $b_1(I, a)$ that a is played in I (as described in the second Observation below). The LP with appropriate restrictions to $b_1(I, a)$ (branching is done using variables $w_{k,l}^{I_1,a}$, using variables corresponding to more significant digits first) is assigned to every successor node. The algorithm terminates when ϵ-optimal strategy is found (using the difference of the current upper bound and the lower bound computed as described in the first Observation below).

Observation. Even if the assignments for binary variables $w_{k,\ell}^{I_1,a}$ are not feasible, the realization plan produced is valid in the perfect recall refinement. We can correct it in the sense of Proposition 3 and use it to estimate the lower bound for the BnB subtree rooted in the current node without assigning all $w_{k,\ell}^{I_1,a}$ variables to 0/1.

Observation. Let $I_1 \in \mathcal{I}_1^{IR}$ and $a \in \mathcal{A}(I_1)$ be an information set and action, which contributes to the current approximation error the most in the BnB node n. When creating successors of a n, branching on I_1 and a is assumed to guide the search in the most promising direction.

More formally, Algorithm 1 takes a LP relaxation of the Upper Bound MILP (relaxing all the binary variables to belong to continuous interval $[0, 1]$) as its input. Initially strategies are represented using 0 digits of precision after the decimal point (i.e. precision $P(I_1, a) = 0$ for every variables $x(I_1, a)$). The algorithm maintains a set of active BnB nodes (fringe) from where the node with the highest upper bound is selected at every iteration (lines 4–5). Subtrees that provably do not contain the optimal solution are disregarded (*bounding* — line 6). Next, we check, whether the current solution increases the lower bound, if yes we replace it (line 8). 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 9). This argument holds, since the 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 11). 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. $x(I_1, a) \leq \lfloor v \rfloor_{-P+1}$ (line 13) and $x(I_1, a) \geq \lceil v \rceil_{-P+1}$ (line 14), where $\lfloor \cdot \rfloor_p$ and $\lceil \cdot \rceil_p$ is flooring, resp. ceiling, of a number towards p digits of precision. This step forces the $x(I_1, a)$ to be lower or higher

than v in the given number of digits. 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 $\lfloor v \rfloor \leq x(I_1, a) \leq \lceil v \rceil$ while increasing the precision used for representing $x(I_1, a)$ (line 16).

The function `CreateNode` computes the upper bound by solving the given LP (line 23) and the lower bound, by using the heuristical construction of a valid strategy b_1 returning some convex combination of strategies found for $\sigma_1^k \in \mathsf{seq}_1(I_1)$ (line 24) and computing the expected value of b_1 against a best response to it.

Note that this algorithm allows us to plug-in any heuristic for the reconstruction of strategies (line 24) and for the action selection (line 11). Below we provide a linear program that can be used for the strategy reconstruction. It also allows us to perform the action selection based on the values of error approximations used in its computation.

6.3.1 LP for Strategy Reconstruction

We provide a linear program that serves as a heuristic to compute a behavioral strategy in I_1, taking into account the realization probabilities $r(\sigma_1^k)$ of sequences $\sigma_1^k \in \mathsf{seq}_1(I_1)$ leading to I_1 as well as errors that can be accumulated in the subtrees of individual histories $h \in I_1$ (b^k is the behavioral strategy obtained for sequence σ_1^k, b is the corrected one):

$$\min_{b,L} \sum_{\sigma_1^k \in \mathsf{seq}(I_1)} r(\sigma_1^k) \cdot L(\sigma_1^k) \tag{6.39}$$

$$\text{s.t.} \quad L(\sigma_1^k) = \sum_{a \in A(I_1)} L(\sigma_1^k, a) \qquad \forall \sigma_1^k \in \mathsf{seq}_1(I_1) \tag{6.40}$$

$$L(\sigma_1^k, a) \geq [b^k(a) - b(a)] \cdot v_{max}(\sigma_1^k \cdot a) \qquad \forall \sigma_1^k \in \mathsf{seq}_1(I_1), \forall a \in A(I_1) \tag{6.41}$$

$$L(\sigma_1^k, a) \geq [b(a) - b^k(a)] \cdot (-v_{min}(\sigma_1^k \cdot a)) \qquad \forall \sigma_1^k \in \mathsf{seq}_1(I_1), \forall a \in A(I_1) \tag{6.42}$$

$$b(a) = \sum_{\sigma_1^k \in \mathsf{seq}_1(I_1)} \alpha(\sigma_1^k) \cdot b^k(a) \qquad \forall a \in A(I_1) \tag{6.43}$$

$$b(a) \geq b^k(a) \qquad \forall \sigma_1^k \in \mathsf{seq}_1(I_1) \tag{6.44}$$

$$b(a) \leq b^k(a) \qquad \forall \sigma_1^k \in \mathsf{seq}_1(I_1) \tag{6.45}$$

$$0 \leq \alpha(\sigma_1^k) \leq 1 \qquad \forall \sigma_1^k \in \mathsf{seq}_1(I_1) \tag{6.46}$$

$$\sum_{\sigma_1^k \in \mathsf{seq}_1(I_1)} \alpha(\sigma_1^k) = 1 \tag{6.47}$$

The LP finds the strategy minimizing the estimated error in the following way. Constraints (6.41), (6.42) compute the maximum cost of changing the probability that action a is played after σ_1^k $L(\sigma_1^k, a)$, assuming that the worst possible outcome in the subtree following playing $\sigma_1^k a$ is reached. Constraint (6.40) computes the estimated errors for every σ_1^k $L(\sigma_1^k)$ by summing all the $L(\sigma_1^k, a)$ for all relevant a, and we minimize the sum of $L(\sigma_1^k)$ weighted by the realization probability of corresponding sequences in the objective. Constraints (6.43) to (6.47) make sure that the result will be a convex combination of all the strategies, with the α variables being the coefficients of the convex combination.

Note that the realization probabilities may change when correcting other information sets. The bound from Theorem 3 on the error of a strategy constructed in this way however

still holds. We have shown that the L1 distance of behavioral strategies b^i in I_1 is at most $10^{-P}|A_1(I_1)|$ — the distance to their convex combination b cannot be larger.

We can use this LP to construct a valid strategy b'_1 from the result of the bilinear program in every imperfect recall information set where the results prescribe inconsistent behavior. We can use b'_1 to compute a lower bound $u_1(b'_1, b_2^{BR})$ where $b_2^{BR} \in BR(b'_1)$ on the overall expected maximin value of player 1. It is a valid lower bound, since this LP uses estimates on the expected loss and b'_1 has therefore no guarantees to be optimal.

6.3.2 Theoretical Properties of the BnB Algorithm

The BnB algorithm takes the error bound ϵ as an input. We will provide a method for setting the $P_{max}(I_1, a)$ parameters appropriately to guarantee ϵ-optimality.

Theorem 4. *Let $P_{max}(I_1, a)$ be the maximum number of digits of precision used for representing variable $x(I_1, a)$ set as*

$$P_{max}(I_1, a) = \left\lceil \max_{h \in I_1} \log_{10} \frac{|A(I_1)| \cdot d \cdot [v_{max}(h) - v_{min}(h)]}{2\epsilon} \right\rceil .$$

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 $x(I_1, a)$. Let p be the current precision used to represent $x(I_1, a)$ and let us consider the first node on the branch where $x(I_1, a)$ 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 16 and interval of size 10^{-p} is handled, every left/right operation (lines 13 and 14) 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 13 and 14) 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 $x(I_1, a)$ 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 3 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 $10^{-P_{max}(I_1,a)} \cdot |A(I_1)|$ (for any $a \in A(I_1)$). This means that the bound on $L(h)$

in h from the proof of Theorem 3 is modified to:

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

Similarly with the reasoning in the proof of Theorem 3, 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 $L(\emptyset) = d \cdot \epsilon / d = \epsilon$. □

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

Proof. We start by proving that there is $N \in O(3^{10|\mathcal{I}_1|\mathcal{A}_1^{max} P_{max}})$ 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 $10P_{max}$ since there is 10 digits and we might require P_{max} number of digits. $10|\mathcal{I}_1|\mathcal{A}_1^{max} P_{max}$ is therefore the maximal depth of the branch-and-bound tree. Finally the branching factor of the branch-and-bound tree is at most 3.

By substituting $\max_{I_1 \in \mathcal{I}_1} \left\lceil \max_{h \in I_1} \log_{10} \frac{|A(I_1)| \cdot d \cdot [v_{max}(h) - v_{min}(h)]}{2\epsilon} \right\rceil$ for P_{max} in the above bound (Theorem 4), we obtain

$$N \in O(3^{10|\mathcal{I}_1|\mathcal{A}_1^{max} \max_{I_1 \in \mathcal{I}_1} \left\lceil \max_{h \in I_1} \log_{10} \frac{|A(I_1)| \cdot d \cdot [v_{max}(h) - v_{min}(h)]}{2\epsilon} \right\rceil}) \tag{6.48}$$

$$\in O(3^{10|\mathcal{I}_1|\mathcal{A}_1^{max} \max_{I_1 \in \mathcal{I}_1} \left\lceil \log_{10} \frac{|A(I_1)| \cdot d \cdot [v_{max}(\emptyset) - v_{min}(\emptyset)]}{2\epsilon} \right\rceil}) \tag{6.49}$$

$$\in O(3^{10S_1 \max_{I_1 \in \mathcal{I}_1} \left\lceil \log_{10} \frac{S_1 \Delta}{2\epsilon} \right\rceil}), S_1 = |\mathcal{I}_1|\mathcal{A}_1^{max}, \Delta = v_{max}(\emptyset) - v_{min}(\emptyset) \tag{6.50}$$

$$\in O(3^{10S_1 \left\lceil \log_{10} \frac{S_1 \cdot \Delta}{2\epsilon} \right\rceil}) \tag{6.51}$$

$$\in O(3^{10S_1 (\log_{10} \frac{S_1 \cdot \Delta}{2\epsilon} + 1)}) \tag{6.52}$$

$$\in O(3^{10S_1 (\log_{10}(S_1 \cdot \Delta) - \log_{10}(2\epsilon) + 1)}) \tag{6.53}$$

$$\in O(3^{10S_1 (\log_{10} S_1 \cdot \Delta) + 1)} 3^{-10S_1 \log_{10}(2\epsilon)}) \tag{6.54}$$

$$\in O(3^{10S_1 (\log_{10} S_1 \cdot \Delta) + 1)} 3^{-10S_1 \frac{\log_3(2\epsilon)}{\log_3(10)}}) \tag{6.55}$$

$$\in O(3^{10S_1 (\log_{10} S_1 \cdot \Delta) + 1)} (2\epsilon)^{\frac{-10S_1}{\log_3(10)}}) \tag{6.56}$$

$$\in O(3^{10S_1 (\log_{10}(S_1 \cdot \Delta) + 1)} (2\epsilon)^{-5S_1}) \tag{6.57}$$

□

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

fringe $\leftarrow$ {`CreateNode`(LP_0)}
opt $\leftarrow$ (nil, $-\infty, \infty$)
while fringe $\neq \varnothing$ **do**
  $(LP, lb, ub) \leftarrow \arg\max_{n \in \text{fringe}} n.\text{ub}$
  fringe $\leftarrow$ fringe $\setminus (LP, lb, ub)$
  if opt.lb $\geq n$.ub **then** discard n
  else
    if opt.lb $< n$.lb **then** opt $\leftarrow n$
    if n.ub $- n$.lb $\leq \epsilon$ **then return** `ReconstructStrategy`(opt)
    else
      $(I_1, a) \leftarrow$ `SelectAction`(n)
      $P \leftarrow$ number of digits of precision representing $x(I_1, a)$ in LP
      fringe $\leftarrow$ fringe $\cup$ {`CreateNode`($LP \cup \{\sum_{k=0}^{LP.x(I_1,a)_{-P}-1} w_{k,P}^{I_1,a} = 1\}$)}
      fringe $\leftarrow$ fringe $\cup$ {`CreateNode`($LP \cup \{\sum_{k=LP.x(I_1,a)_{-P}+1}^{9} w_{k,P}^{I_1,a} = 1\}$)}
      if $P < P_{max}(I_1, a)$ **then**
        fringe $\leftarrow$ fringe $\cup$ {`CreateNode`($LP \cup \{w_{LP.x(I_1,a)_{-P},P}^{I_1,a} = 1$, *introduce vars* $w_{0,P+1}^{I_1,a}, \ldots, w_{9,P+1}^{I_1,a}$ *and corresponding constraints from MDT* $\}$)}
      end
    end
  end
end
return `ReconstructStrategy`(opt)
function `CreateNode`(LP)
  $ub \leftarrow$ `Solve`(LP)
  $b_1 \leftarrow$ `ReconstructStrategy`(LP)
  $lb \leftarrow u_1(b_1, \texttt{BestResponse}(b1))$
  return (LP, lb, ub)

Algorithm 1: BnB algorithm

6.3.3 Opponent without A-Loss Recall

If player 2 does not have A-loss recall, we currently need to add each pure best response as a constraint causing an exponential number of constraints in MILP. To reduce the impact of the exponential number of constraints, we can exploit constraint generation techniques (or single/double oracle algorithms [24]).

In Algorithm 2 we present the pseudocode of the single oracle approach. We start with a bilinear sequence form program with no constraints related to player 2 best responses in the first iteration on line 5. After solving the program (line 6), we compute the best response to the strategy of player 1 that is currently thought to be optimal (line 7). If we find out that the current best response was not reflected in the constraints of the MILP (line 9) we add a new constraint into the program representing this strategy and solve it again (lines 5, 6). Whenever the current best response coincides with some strategy represented in the program, we know that we have found all the best responses for current precision. We can either increase the precision (line 11) and continue with the computation, or terminate if the desired precision was reached.

```
   input     : EFG with imperfect recall for both players
   output    : Player 1 strategy b_1
   parameter: precision limit P_max
 1 P ← 1
 2 BR ← ∅
 3 b_1 ← nil
 4 while P < P_max do
 5 |   MP ← bilinear sequence form with best response constraint for every π_2 ∈ BR
 6 |   b_1 ← Solve(MP)
 7 |   π_2^BR ← BestResponse(b_1)
 8 |   if π_2^BR ∉ BR then
 9 |   |   BR ← BR ∪ {π_2^BR}
10 |   else
11 |   |   P ← P + 1
12 |   end
13 end
14 return b_1
```

Algorithm 2: Bilinear sequence form based algorithm for solving imperfect recall games

Chapter 7

Conclusion

This thesis aims to provides a thorough theoretical description of the effect of imperfect recall on decision making and first algorithms capable of tackling the problem of finding Nash equilibrium and maxmin strategies in imperfect recall games.

In this thesis proposal, we introduce sufficient and necessary conditions for existence of Nash equilibrium in behavioral strategies of imperfect recall games where the loss of recall can be traced back to a loss of information about one's own actions, called A-loss and we show that A-loss recall games satisfying additional properties are a good candidate for the output of an abstraction algorithm as they can be used to find rational strategies in the original game. Furthermore, we provide the first algorithm capable of finding Nash equilibrium in two-player zero-sum A-loss recall games. This algorithm is, however, computationally prohibitive. Therefore, we suggest a no-regret learning algorithm for this class of imperfect recall games, which is expected to take advantage of the reduced number of information sets. Finally, we provide an algorithm for finding maxmin strategies in imperfect recall games.

7.1 Future Work

The main future goal of this thesis is to create an algorithm capable of solving the large perfect recall games using their imperfect recall abstractions, while outperforming the state-of-the-art algorithms applied directly to the unabstracted perfect recall game. Since the abstraction algorithm can be guided to result in desirable game structure, we are interested in exploring additional types of imperfect recall games which we can use to find rational behavior in the original games, building on results in A-loss recall games presented in Chapter 3. Next step is to develop an efficient algorithm taking advantage of the reduced game size caused by the imperfect recall abstractions. Here, there are two possible directions. (1) In Chapter 5 we present an outline of a no-regret algorithm which should take advantage of the reduced size of the abstracted games. As a future work we need to devise a bound on the convergence of this algorithm and experimentally evaluate its performance. (2) Creation of a more effective variant of the algorithm presented in Chapter 6 using double-oracle framework.

Bibliography

[1] Charles Audet, Slim Belhaiza, and Pierre Hansen. A New Sequence Form Approach for the Enumeration and Refinement of all Extreme Nash Equilibria for Extensive Form Games. *International Game Theory Review 11*, 2009.

[2] David Avis and Komei Fukuda. A Pivoting Algorithm for Convex Hulls and Vertex Enumeration of Arrangements and Polyhedra. *Discrete & Computational Geometry*, 8(1):295–313, 1992.

[3] Robert G Bland. New Finite Pivoting Rules for the Simplex Method. *Mathematics of Operations Research*, 2(2):103–107, 1977.

[4] Giacomo Bonanno. Memory and perfect recall in extensive games. *Games and Economic Behavior*, 47(2):237–256, 2004.

[5] Branislav Bosansky, Christopher Kiekintveld, Viliam Lisy, and Michal Pechoucek. An exact double-oracle algorithm for zero-sum extensive-form games with imperfect information. *Journal of Artificial Intelligence Research*, pages 829–866, 2014.

[6] Vasek Chvatal. *Linear Programming*. Macmillan, 1983.

[7] George Bernard Dantzig. *Linear Programming and Extensions*. Princeton University Press, 1998.

[8] Andrew Gilpin and Tuomas Sandholm. Lossless Abstraction of Imperfect Information Games. *Journal of the ACM (JACM)*, 54(5):25, 2007.

[9] Geoffrey J Gordon. No-regret algorithms for online convex programs. In *Advances in Neural Information Processing Systems*, pages 489–496, 2006.

[10] Adam J Grove and Joseph Y Halpern. On the expected value of games with absentmindedness. *Games and Economic Behavior*, 20(1):51–65, 1997.

[11] Joseph Y Halpern and Rafael Pass. Sequential equilibrium and perfect equilibrium in games of imperfect recall. *Unpublished manuscript*, 2009.

[12] Kristoffer Arnsfelt Hansen, Peter Bro Miltersen, and Troels Bjerre Sørensen. Finding Equilibria in Games of no Chance. In *Computing and Combinatorics*, pages 274–284. Springer, 2007.

[13] Michael Johanson, Nolan Bard, Neil Burch, and Michael Bowling. Finding optimal abstract strategies in extensive-form games. In *AAAI*, 2012.

[14] Michael Johanson, Neil Burch, Richard Valenzano, and Michael Bowling. Evaluating state-space abstractions in extensive-form games. In *AAMAS*, pages 271–278, 2013.

[15] Mamoru Kaneko and J. Jude Kline. Behavior Strategies, Mixed Strategies and Perfect Recall. *International Journal of Game Theory*, 24:127–145, 1995.

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

[17] Jeffrey Jude Kline. Imperfect recall and the relationships between solution concepts in extensive games. *Economic Theory*, 25(3):703–710, 2005.

[18] Daphne Koller and Nimrod Megiddo. The Complexity of Two-person Zero-sum Games in Extensive Form. *Games and Economic Behavior*, 4:528–552, 1992.

[19] Daphne Koller, Nimrod Megiddo, and Berhard von Stengel. Fast Algorithms for Finding Randomized Strategies in Game Trees. In *26th annual ACM symposium on Theory of computing*, 1994.

[20] Scott Kolodziej, Pedro M. Castro, and Ignacio E. Grossmann. Global optimization of bilinear programs with a multiparametric disaggregation technique. *Journal of Global Optimization*, 57(4):1039–1063, 2013.

[21] Christian Kroer and Tuomas Sandholm. Extensive-Form Game Imperfect-Recall Abstractions With Bounds. *arXiv preprint arXiv:1409.3302*, 2014.

[22] Harold W. Kuhn. Extensive Games and the Problem of Information. *Annals of Mathematics Studies*, 1953.

[23] Marc Lanctot, Richard Gibson, Neil Burch, Martin Zinkevich, and Michael Bowling. No-regret Learning in Extensive-form Games with Imperfect Recall. *arXiv preprint arXiv:1205.0622*, 2012.

[24] H. Brendan McMahan, Geoffrey J. Gordon, and Avrim Blum. Planning in the Presence of Cost Functions Controlled by an Adversary. In *Proceedings of the International Conference on Machine Learning*, pages 536–543, 2003.

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

[26] Michele Piccione and Ariel Rubinstein. The absent-minded driver's paradox: synthesis and responses. *Games and Economic Behavior*, 20(1):121–130, 1997.

[27] Michele Piccione and Ariel Rubinstein. On the interpretation of decision problems with imperfect recall. *Games and Economic Behavior*, 20(1):3–24, 1997.

[28] Milind Tambe. *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned*. Cambridge University Press, 2011.

[29] Bernhard von Stengel. Efficient Computation of Behavior Strategies. *Games and Economic Behavior*, 14:220–246, 1996.

[30] Kevin Waugh, Martin Zinkevich, Michael Johanson, Morgan Kan, David Schnizlein, and Michael H Bowling. A Practical Use of Imperfect Recall. In *SARA*. Citeseer, 2009.

[31] Philipp C Wichardt. Existence of Nash Equilibria in Finite Extensive Form Games with Imperfect Recall: A Counterexample. *Games and Economic Behavior*, 63(1):366–369, 2008.

[32] Martin Zinkevich, Michael Bowling, and Neil Burch. A New Algorithm for Generating Equilibria in Massive Zero-Sum Games. In *AAAI*, 2007.