Large-Scale Sequential Imperfect-Information Game Solving: Theoretical Foundations and Practical Algorithms with Guarantees

Christian Kroer

Thursday 12th January, 2017

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Tuomas Sandholm, Chair
Geoffrey J. Gordon
Fatma Kılınç-Karzan
Vince Conitzer, Duke University
Yurii Nesterov, Universitè catholique de Louvain

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Keywords: equilibrium finding, extensive-form games, sequential games, imperfect-information games, Nash equilibrium, abstraction, convex optimization, first-order methods, limited lookahead

Abstract

Game-theoretic equilibrium concepts provide a sound definition of how rational agents should act in multiagent settings. To operationalize them, they have to be accompanied by techniques to compute equilibria. We study the computation of equilibria for extensive-form games, a broad game class that can model sequential interaction, imperfect information, and outcome uncertainty. Practical equilibrium computation in extensive-form games relies on two complementary methods: abstraction methods and sparse iterative equilibrium-finding algorithms. These methods are necessary in order to handle the size of many real-world games.

We present new algorithmic and structural results for both parts of extensive-form game solving. We introduce state-of-the-art theoretical guarantees on the performance of our algorithms, and first-of-their-kind guarantees on the solution quality of abstractions for large games.

For abstraction, we develop new theoretical guarantees on the solution quality of equilibria computed in abstractions. We develop new results for several types of games and abstractions: discrete and continuous extensive-form games, and perfect and imperfect-recall abstractions. For all settings, our results are the first algorithm-agnostic solution-quality guarantees. Additionally, even compared to algorithm-specific results, our approach leads to exponentially stronger bounds than prior results, and extend to more general games and abstractions.

For equilibrium computation, we focus on the formulation of an extensive-form two-player zero-sum Nash equilibrium as a bilinear saddle-point problem. This allows us to leverage methods from the convex optimization literature. We consider a smoothing method based on a dilated entropy function. We prove bounds on the strong convexity and polytope diameter associated with this function that are significantly stronger than bounds for prior smoothing methods. This leads to the state-of-the-art in convergence rate for sparse iterative methods for computing a Nash equilibrium. Our results can also be viewed more generally as strong convexity results for a class of convex polytopes called treeplexes. These results could be of independent interest in other sequential decision-making settings.

Finally, we develop new solution concepts and associated algorithmic results for games where opponents have limited lookahead.

Contents

Chapter 1

Introduction

Game-theoretic solution concepts provide a sound notion of rational behavior in multiagent settings. To operationalize equilibria, they have to be accompanied by computational techniques for identifying them. Thus, equilibrium computation has emerged as a central topic in economics and computation [Gilpin and Sandholm, 2007b, Jiang and Leyton-Brown, 2011, Lipton et al., 2003, Littman and Stone, 2003, Zinkevich et al., 2007, Koller et al., 1996, Daskalakis et al., 2015, von Stengel, 1996]. In this thesis we focus mainly on *extensive-form games*. Extensive-form games are a broad class of games that can model sequential and simultaneous moves, outcome uncertainty, and imperfect information. This includes real-world settings such as negotiation, sequential auctions, security games, cybersecurity games, recreational games such as poker and billiards, and certain medical treatment settings [Archibald and Shoham, 2009, Lisy et al., 2016, Munoz de Cote et al., 2013, Sandholm, 2010, DeBruhl et al., 2014, Chen and Bowling, 2012].

The primary benchmark for large-scale equilibrium-finding is the *annual computer poker competition* (ACPC). Each year, research labs and individual hobbyists submit poker bots which are faced off in competition. For many years, all of the most successful bots have been based on equilibrium-finding techniques [ACPC, 2016, Sandholm, 2010, Brown et al., 2015]. More specifically, these bots employ the following technique: First, the game is abstracted to generate a smaller game. Then the abstract game is solved for (near-)equilibrium. Then, the strategy from the abstract game is mapped back to the original game. This process is shown pictorally in Figure 1.1. The figure suggests that an ϵ-Nash equilibrium is obtained in the full game. Current practical methods have no such guarantee on the quality of the equilibrium in the full game. Furthermore, practical abstractions are so large that exact Nash equilibria can't be computed, even in the abstraction. Instead, sparse iterative methods are applied, which converge to a Nash equilibrium in the limit.

This thesis proposal explores and develops theoretical foundations for all steps of the solution process described above. Secion 3 develops state-of-the-art convergence rate bounds for sparse iterative solvers. Section 4 ameliorates the lack of guarantees on abstraction solution quality by developing some of the first, and by far the strongest, bounds on solution quality. Section 5 develops new solution concepts for settings where a rational player is facing one or more myopic adversaries. In such a setting rationality may be an overly pessimistic assumption, and instead we present a model of limited lookahead in EFGs.

The next three sections will give a brief overview of the results obtained so far on each of the equilibrium-finding dimensions, as well as briefly mention the proposed extensions.

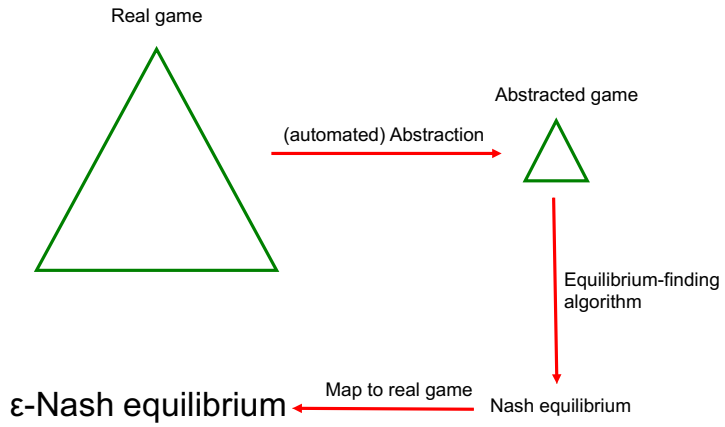


Figure 1.1: An overview of the general abstraction approach (figure borrowed from Ganzfried and Sandholm [2012]).

1.1 Algorithms for computing equilibria (Chapter 3)

This chapter focuses on sparse iterative methods for large-scale equilibrium computation. In particular, we investigate the application of first-order methods (FOMs) to the computation of equilibria in two-player zero-sum games. The application of FOMs relies on three core ingredients: (1) the sequence-form transformation [von Stengel, 1996] for obtaining a bilinear saddle-point formulation, (2) convex minimization algorithms such as the excessive gap technique (EGT) [Nesterov, 2005a] or mirror prox [Nemirovski, 2004], and (3) smoothing techniques for the strategy spaces of the two players. The third part is currently not well understood.

1.1.1 Completed work

Hoda et al. [2010] introduced a class of smoothing techniques that can be applied to EFGs. However, the convergence rate bounds obtained from the results of Hoda et al. [2010] have a significantly worse dependence on the EFG size than CFR. We show that a much better dependence on game parameters can be achieved by focusing on the entropy function as a smoothing technique for each information set. This is done by carefully choosing information-set weights such that a desirable smoothing function for the game tree is obtained by summing over the weighted entropy functions for each information sets. Our result leads to the strongest known convergence rate for sparse iterative methods, achieving a dependence on game constants that is much stronger than that of Hoda et al. [2010], while maintaining a $1/T$ convergence rate. In particular, our strong convexity result is the first to achieve no dependence on the branching branching factor associated with a player. Furthermore, we introduce a new class of gradient estimators that allow instantiation of stochastic FOMs such as stochastic mirror prox [Juditsky et al., 2011]. Finally, we show experimentally that EGT and mirror prox instantiated with our smoothing technique leads to better practical convergence rates than CFR for medium-to-high accuracy solutions on medium-sized games.

Part of this work was published at EC-2015 [Kroer et al., 2015]. A follow-up manuscript is available at (currently unpublished, cite arXiv .

1.1.2 Proposed work (Section 3.3)

While our current results are state-of-the-art for theoretical convergence rates for sparse iterative methods, they are not the practical state-of-the-art. In practice, CFR algorithms are still preferred. We propose extending our theoretical work by experimentally findings ways to scale up FOMs to be competitive with CFR algorithms. Our two most promising ideas for achieving this are 1) heuristic aggressive stepsizing for EGT coupled with checks of the excessive gap condition, and 2) better initialization of EGT (preliminary experimental results suggest that even simple tweaks to the starting point can have several orders of magnitude impact on convergence rate). The success of this approach will be measured by showing that FOMs can outperform the leading CFR-like algorithm on real-world games. Specifically, we will consider one or both of the following games:

- Large texas holdem poker endgames. An endgame is, at a high level, an extensive-form game constructed ad-hoc to represent some subset of the game. Endgames are currently solved with CFR+, and represent the smallest class of games where FOMs can have practical impact.
- No limit Texas hold'em (NLTHE) game abstractions. NLTHE is the current frontier of large-scale EFG solving, where many state-of-the-art contemporary bots are more exploitable than always folding (i.e. laying down your hand immediately) [Lisy and Bowling, 2017].

Theoretically, we propose extending our results by developing corresponding lower bounds on the convergence rates achievable with the class of entropy-based smoothing functions we consider. Furthermore, we propose showing upper and/or lower bounds on the convergence rate achievable with l_2 smoothing. Ideally, this will allow us to show that entropy-based smoothing is theoretically superior to l_2 smoothing for EFGs. This would complement experimental results from Hoda et al. [2010], which found entropy smoothing to be superior.

Finally, we propose the development of sparse iterative methods for computing Nash-equilibrium refinements. Polynomial-time algorithms are known for refinements such as normal-form proper and quasi-perfect equilibria [Miltersen and Sørensen, 2008, 2010]. However, these algorithms rely on solving one or more linear programs, usually relying on modifications to the sequence-form LP of von Stengel [1996]. As with Nash equilibria, such LP formulations are unlikely to be practical for large-scale games. Instead, we propose leveraging a perturbed sequence-form polytope in the standard bilinear saddle-point formulation, and then focus on solving this saddle-point problem directly. For small-enough perturbations, this should lead to an approximate Nash equilibrium refinement.

1.2 Abstraction for large games (Chapter 4)

As mentioned previously, practical equilibrium computation usually relies on dimensionality reduction through creating an abstracted game first. This chapter focuses on abstraction methods for reducing the dimensionality of extremely large games. Practical abstraction algorithms for EFGs have all been without any solution quality bounds [Gilpin and Sandholm, 2006, 2007a, Gilpin et al., 2007, Gilpin and Sandholm, 2008, Ganzfried and Sandholm, 2014]. This chapter ameliorates this fact by developing some of the first theoretical results on the quality of equilibria computed in abstractions of EFGs.

1.2.1 Completed work

We have developed theoretical guarantees on solution quality of equilibria in several settings.

Perfect-recall abstraction. We develop the first algorithm-agnostic bounds on solution quality for equilibria computed in perfect-recall abstractions. This work also improves the best prior bounds, which were specific to CFR, by an exponential amount. In particular, the payoff error in prior bounds had a linear dependence on the number of information sets, whereas the payoff error in our bounds has no dependence on the number of information sets. Our result is obtained by introducing a new method for mapping abstract strategies to the full game, and leverages a new equilibrium refinement in the analysis. Using this framework, we develop the first general lossy extensive-form game abstraction method with bounds. Experiments show that it finds a lossless abstraction when one is available and lossy abstractions when smaller abstractions are desired.

We also develop the first complexity results on computing good abstractions of EFGs. Prior abstraction algorithms typically operate level by level in the game tree. We introduce the extensive-form game tree isomorphism and action subset selection problems, both important subproblems for computing abstractions on a level-by-level basis. We show that the former is graph isomorphism complete, and the latter NP-complete. This suggests that level-by-level abstraction is, in general, a computationally difficult problem. We also prove that level-by-level abstraction can be too myopic and thus fail to find even obvious lossless abstractions.

This work was published at EC-2014 [Kroer and Sandholm, 2014]

Imperfect-recall abstraction. Imperfect-recall abstraction has emerged as the leading paradigm for practical large-scale equilibrium computation in imperfect-information games. However, imperfect-recall abstractions are poorly understood, and only weak algorithm-specific guarantees on solution quality are known. We develop the first general, algorithm-agnostic, solution quality guarantees for Nash equilibria and approximate self-trembling equilibria (a new solution concept that we introduce) computed in imperfect-recall abstractions, when implemented in the original (perfect-recall) game. Our results are for a class of games that generalizes the only previously known class of imperfect-recall abstractions for which any such results have been obtained. Further, our analysis is tighter in two ways, each of which can lead to an exponential reduction in the solution quality error bound.

We then show that for extensive-form games that satisfy certain properties, the problem of computing a bound-minimizing abstraction for a single level of the game reduces to a clustering problem, where the increase in our bound is the distance function. This reduction leads to the first imperfect-recall abstraction algorithm with solution quality bounds. We proceed to show a divide in the class of abstraction problems. If payoffs are at the same scale at all information sets considered for abstraction, the input forms a metric space, and this immediately yields a 2-approximation algorithm for abstraction. Conversely, if this condition is not satisfied, we show that the input does not form a metric space. Finally, we provide computational experiments to evaluate the practical usefulness of the abstraction techniques. They show that running counterfactual regret minimization on such abstractions leads to good strategies in the original games.

This work was published at EC-2016 [Kroer and Sandholm, 2016a]

Discretization of continuous games. While EFGs are a very general class of games, most solution algorithms require discrete, finite games. In contrast, many real-world domains require modeling with continuous action spaces. This is usually handled by heuristically discretizing the continuous action space without solution quality bounds. Leveraging our results on abstraction solution quality, we develop the first framework for providing bounds on solution quality for discretization of continuous action spaces in extensive-form games. For games where the error is Lipschitz-continuous in the distance of a continuous point to its nearest discrete point, we show that a uniform discretization of the

space is optimal. When the error is monotonically increasing in distance to nearest discrete point, we develop an integer program for finding the optimal discretization when the error is described by piecewise linear functions. This result can further be used to approximate optimal solutions to general monotonic error functions.

This work was published at AAMAS-2015 [Kroer and Sandholm, 2015b]

1.2.2 Proposed work (Section 4.4)

While the above lines of work have provided much-needed theoretical foundations for the area of abstraction in EFGs, there is still a lot of work to do in order to bridge the gap to practical abstraction work. All abstraction algorithms mentioned in the preceding section require either 1) solving an integer program whose size is on the order of magnitude of the game tree (this can be improved to just the signal tree for games of ordered signals) [Kroer and Sandholm, 2014], or 2) solving for abstractions level-by-level with prohibitively strong constraints on the type of abstraction allowed [Kroer and Sandholm, 2016a].

We propose developing practical methods for computing abstractions, while retaining bounds on the solution quality. One way to do this is by extending our framework for bounding abstraction quality to allow additional worst-case assumptions about paths in the game tree that our current framework cannot provide bounds for. This will greatly expand the set of feasible abstractions, thereby making the computational problem easier. Hopefully, it will also allow analysis of a general level-by-level abstraction algorithm. An alternative approach would be to make additional assumptions about the game structure, potentially assuming structure that is similar to that of *games of ordered signals* [Gilpin and Sandholm, 2007b].

We will attempt one or more of the following goals:

- Compute the first strategy with a meaningful bound on regret in NLTHE (through solving an abstract game).
- Show that existing practical abstraction algorithms have bounded solution quality.
- Show an impossibility result stating that the kind of abstraction employed in practice cannot lead to meaningful ex-ante bounds on solution quality.

1.3 Limited lookead (Chapter 5)

In some application domains, the assumption of perfect rationality might not be practical, or even useful, as opponents might display exploitable behavior that the perfect rationality assumption does not allow exploitation of. In order to model such settings, we initiate the game-theoretic study of limited lookahead in imperfect-information games.

This model has applications such as biological games, where the goal is to steer an evolutionary or adaptation process (which typically acts myopically with lookahead 1) [Sandholm, 2015], and security games where opponents are often assumed to be myopic (as makes sense when the number of adversaries is large [Yin et al., 2012]). Furthermore, investigating how well a rational player can exploit a limited-lookahead player lends insight into the limitations and risks of using limited-lookahead algorithms in multiagent decision making.

1.3.1 Completed work

We generalize limited-lookahead research to imperfect-information games and a game-theoretic approach. We study the question of how one should act when facing an opponent whose lookahead is limited along

multiple axes: lookahead depth, whether the opponent(s), too, have imperfect information, and how they break ties. We characterize the hardness of finding a Nash equilibrium or an optimal commitment strategy for either player, showing that in some of these variations the problem can be solved in polynomial time while in others it is PPAD-hard or NP-hard. We proceed to design algorithms for computing optimal commitment strategies for when the opponent breaks ties 1) favorably, 2) according to a fixed rule, or 3) adversarially. The impact of limited lookahead is then investigated experimentally. The limited-lookahead player often obtains the value of the game if she knows the expected values of nodes in the game tree for some equilibrium, but we prove this is not sufficient in general. Finally, we study the impact of noise in those estimates and different lookahead depths. This uncovers a lookahead pathology.

This work was published at IJCAI-2015 [Kroer and Sandholm, 2015c]

1.3.2 Proposed work (Section 5.7)

While our current model of limited lookahead can model certain myopic adversaries, it has one important weakness: it does not allow for modeling of uncertainty over the exact myopic behavior of the opponents. We propose extending our current model of limited lookahead to include a distribution over limited-lookahead opponent models. This will allow much more general modeling of myopic behavior, as well as allow for computing robust strategies for exploiting a myopic opponent. Depending on the specific assumptions made, this can make the computation of the associated solution concepts significantly harder. We propose managing this increased complexity by viewing the problem through a robust optimization lends.

1.4 Structure of this document

This chapter gave a high-level description of each of the dimensions of sequential game solving considered in this thesis. The following chapters will go into more depth on our results and proposed research directions for each topic. Chapter 2 develops some notation necesary in order to read the technical sections of the following chapters. Chapter 2 should not be neccesary in order to understand the introductory and proposed work sections of each chapter. Chapters 3, 4 and 5 present my current and proposed work on first-order methods to compute equilibria, abstraction methods, and new models of limited-lookahead adversaries. These chapters include fairly detailed technical sections, but the gist of the chapters can be gleaned from the introductory sections as well as proposed work sections.

Finally, Chapter 7 will briefly discuss some other research work that I've done during my doctoral studies.

Chapter 2

Notation

This section introduces some general notation for EFGs that we will use throughout most of this proposal. Chapter 3 is largely independent of this notation, and so can safely be read in isolation. Chapters 4 and 5 rely more heavily on the notation described here for formally describing the obtained results.

2.1 Extensive-form games

An *extensive-form game* (EFG) Γ is a tuple $\langle N, A, S, Z, \mathcal{H}, \sigma_0, u, \mathcal{I} \rangle$. N is the set of players. A is the set of all actions. S is a set of nodes corresponding to sequences of actions. They describe a tree with root node $r \in S$. At each node s, some Player i is active with actions A_s, and each branch at s denotes a different choice in A_s. Let t_a^s be the node transitioned to by performing action $a \in A_s$ at node s. The set of all nodes where Player i is active is called S_i. The set of leaf nodes is denoted by $Z \subset S$. For each leaf node z, Player i receives a reward of $u_i(z)$. We assume, WLOG., that all utilities are non-negative. Z_s is the subset of leaf nodes reachable from a node s. $\mathcal{H}_i \subseteq \mathcal{H}$ is the set of heights in the game tree where Player i acts. $\mathcal{H}_0$ is the set of heights where nature acts. σ_0 specifies the probability distribution for nature, with $\sigma_0(s, a)$ denoting the probability of nature choosing outcome a at node s. $\mathcal{I}_i \subseteq \mathcal{I}$ is the set of information sets where Player i acts. $\mathcal{I}_i$ partitions S_i. For any two nodes $s_1, s_2 \in I \in \mathcal{I}_i$, Player i cannot distinguish among them, and $A_{s_1} = A_{s_2}$. We let $X(s)$ denote the set of information set and action pairs I, a in the sequence leading to a node s, including nature. We let $X_{-i}(s), X_i(s) \subseteq X(s)$ be the subset of this sequence such that actions by the subscripted player(s) are excluded or exclusively chosen. We let $X^b(s)$ be the set of possible sequences of actions players can take in the subtree below s, with $X^b_{-i}(s), X^b_i(s)$ being the set of future sequences excluding or limited to Player i, respectively. We denote elements in these sets as $\vec{a}$. $X^b(s, \vec{a}), X^b_{-i}(s, \vec{a}), X^b_i(s, \vec{a})$ are the analogous sets limited to sequences that are consistent with the sequence of actions $\vec{a}$. We let the set of leaf nodes reachable from s for a particular sequence of actions $\vec{a} \in X^b(s)$ be $Z_s^{\vec{a}}$ For an information set I on the path to a leaf node z, $z[I]$ denotes the predecessor $s \in I$ of z.

Perfect recall means that no player forgets anything that the player observed in the past. Formally, for every Player $i \in N$, information set $I \in \mathcal{I}_i$, and nodes $s_1, s_2 \in I : X_i(s_1) = X_i(s_2)$. Otherwise, the game has *imperfect recall*. The most important consequence of imperfect recall is that a player can affect the distribution over nodes in their own information sets, as nodes in an information set may originate from different past information sets of the player.

We denote by σ_i a *behavioral strategy* for Player i. For each information set I where it is the player's turn to move, it assigns a probability distribution over A_I, the actions at the information set. $\sigma_i(I, a)$ is

the probability of playing action a. A *strategy profile* $\sigma = (\sigma_0, \ldots, \sigma_n)$ consists of a behavioral strategy for each player. We will often use $\sigma(I, a)$ to mean $\sigma_i(I, a)$, since the information set uniquely specifies which Player i is active. As described above, randomness external to the players is captured by the nature outcomes σ_0. We let $\sigma_{I \to a}$ denote the strategy profile obtained from σ by having Player i deviate to taking action a at $I \in \mathcal{I}_i$. Let the probability of going from node s to a descendant $\hat{s}$ under strategy profile σ be $\pi^\sigma(s, \hat{s}) = \Pi_{\langle \bar{s}, \bar{a} \rangle \in X_{s,\hat{s}}} \sigma(\bar{s}, \bar{a})$ where $X_{s,\hat{s}}$ is the nonempty set of pairs of nodes and actions on the path from s to $\hat{s}$. We let the probability of reaching node s be $\pi^\sigma(s) = \pi^\sigma(r, s)$, the probability of going from the root node r to s. Let $\pi^\sigma(I) = \sum_{s \in I} \pi^\sigma(s)$ be the probability of reaching any node in I. For probabilities over nature, $\pi_0^\sigma(s) = \pi_0^{\bar{\sigma}}(s)$ for all $\sigma, \bar{\sigma}, s \in S_0$, so we can ignore the superscript and write π_0.

For all definitions, the subscripts $i, -i$ refer to the same definition, but exclusively over or excluding Player i in the product of probabilities, respectively.

For information set I and action $a \in A_I$ at level $k \in \mathcal{H}_i$, we let $\mathcal{D}_I^a$ be the set of information sets at the next level in $\mathcal{H}_i$ reachable from I when taking action a. Similarly, we let $\mathcal{D}_I^l$ be the set of descendant information sets at height $l \leq k$, where $\mathcal{D}_I^k = \{I\}$. Finally, we let $\mathcal{D}_s^{\vec{a},j}$ be the set of information sets reachable from node s when action-vector $\vec{a}$ is played with probability one.

2.2 Value functions for an extensive-form game

We define value functions both for individual nodes and for information sets.

Definition 1. *The value for Player i of a given node s under strategy profile σ is*

$$V_i^\sigma(s) = \sum_{z \in Z_s} \pi^\sigma(s, z) u_i(z).$$

We use the definition of counterfactual value of an information set, introduced by Zinkevich et al. [2007], to reason about the value of an information set under a given strategy profile. The counterfactual value of an information set I is the expected utility of the information set, assuming that all players follow strategy profile σ, except that Player i plays to reach I.

Definition 2. *The counterfactual value for Player i of a given information set I under strategy profile σ is*

$$V_i^\sigma(I) = \begin{cases} \sum_{s \in I} \frac{\pi_{-i}^\sigma(s)}{\pi_{-i}^\sigma(I)} \sum_{z \in Z_s} \pi^\sigma(s, z) u_i(z) & \text{if } \pi_{-i}^\sigma(I) > 0 \\ 0 & \text{if } \pi_{-i}^\sigma(I) = 0 \end{cases}.$$

For the information set I_r that contains just the root node r, we have that $V_i^\sigma(I_r) = V_i^\sigma(r)$, which is the value of playing the game with strategy profile σ. We assume that at the root node it is not nature's turn to move. This is without loss of generality since we can insert dummy player nodes above it.

We show that for information set I at height $k \in \mathcal{H}_i$, $V_i^\sigma(I)$ can be written as a sum over descendant information sets at height $\hat{k} \in \mathcal{H}_i$, where $\hat{k}$ is the next level below k that belongs to Player i.

Proposition 1 (Kroer and Sandholm [2014])**.** *Let I be an information set at height $k \in \mathcal{H}_i$ such that the there is some $\hat{k} \in \mathcal{H}_i, \hat{k} < k$. For such I, $V_i^\sigma(I)$ can be written as a weighted sum over descendant information sets at height $\hat{k}$:*

$$V_i^\sigma(I) = \sum_{a \in A_I} \sigma(I, a) \sum_{\hat{I} \in \mathcal{D}_I^a} \frac{\pi_{-i}^\sigma(\hat{I})}{\pi_{-i}^\sigma(I)} V_i^\sigma(\hat{I}).$$

2.3 Equilibrium concepts

In this section we define the equilibrium concepts we use. We start with two classics.

Definition 3 (ϵ-Nash and Nash equilibria)**.** *An ϵ-Nash equilibrium is a strategy profile σ such that for all i, $\bar{\sigma}_i$: $V_i^{\sigma}(r) + \epsilon \geq V_i^{\sigma_{-i},\bar{\sigma}_i}(r)$. A Nash equilibrium is an ϵ-Nash equilibrium where $\epsilon = 0$.*

We will also introduce the concept of a *self-trembling equilibrium* [Kroer and Sandholm, 2014]. It is a Nash equilibrium where the player assumes that opponents make no mistakes, but she might herself make mistakes, and thus her strategy must be optimal for all information sets that she could mistakenly reach by her own fault.

Definition 4 (Self-trembling equilibrium)**.** *For a game Γ, a strategy profile σ is a self-trembling equilibrium if it satisfies two conditions. First, it must be a Nash equilibrium. Second, for any information set $I \in \mathcal{I}_i$ such that $\pi_{-i}^{\sigma}(I) > 0$, and for all alternative strategies $\bar{\sigma}_i$, $V_i^{\sigma}(I) \geq V_i^{\sigma_{-i},\bar{\sigma}_i}(I)$. We call this second condition the* self-trembling *property.*

An ϵ-self-trembling equilibrium is defined analogously, for each information set $I \in \mathcal{I}_i$, we require $V_i^{\sigma}(I) \geq V_i^{\sigma_{-i},\bar{\sigma}_i}(I) - \epsilon$. For imperfect-recall games, the property $\pi_{-i}^{\sigma}(I') > 0$ does not give a probability distribution over the nodes in an information set I', since Player i can affect the distribution over the nodes. For such information sets, it will be sufficient for our purposes to assume that σ_i is (approximately) utility maximizing for some (arbitrary) distribution over $\mathcal{P}(I')$; our bounds are the same for any such distribution.

We introduce this Nash equilibrium refinement because it turns out to be a useful tool for reasoning inductively about quality of Nash equilibria. For perfect-recall games, self-trembling equilibria are a strict superset of perfect Bayesian equilibria. A Perfect Bayesian equilibrium is a strategy profile and a belief system such that the strategies are sequentially rational given the belief system, and the belief system is *consistent*. A belief system is consistent if Bayes' rule is used to compute the belief for any information set where it is applicable. In a perfect recall game, the information sets where Bayes' rule is applicable are exactly the ones where $\sigma_{-i}(I) > 0$. These are exactly the information sets where self-trembling equilibria are sequentially rational. Perfect Bayesian equilibria are a strict subset because there might not exist a belief system for which a self-trembling equilibrium is rational on information sets where Bayes' rule does not apply. In particular, for a given information set where Bayes' rule does not apply, there might be an action that is strictly worse than another action for every node in the information set. Since Bayes' rule does not apply, the self-trembling property places no restrictions on the information set, and thus may play the strictly dominated action with probability one. This is not sequentially rational for *any* belief system.

Chapter 3

Algorithms for computing equilibria

As mentioned above, the abstractions computed in practical sequential game settings are so large that exact approaches are impractical. This holds even for two-player zero-sum games, despite the fact that exact equilibria can be computed with a linear program (LP) that has size linear in the size of the game tree [von Stengel, 1996]. This has led to extensive research into sparse iterative methods that converge to a Nash equilibrium in the limit, but have relatively cheap iterations and low memory requirements [Zinkevich et al., 2007, Lanctot et al., 2009, Hoda et al., 2010, Gibson et al., 2012, Johanson et al., 2012, Brown and Sandholm, 2014, 2015]. Recently, a sparse iterative solver, CFR+ [Tammelin et al., 2015], was used to practically solve limite texas holdem [Bowling et al., 2015], a poker variant with fixed betsizes. Notably, this game was unabstracted, and high accuracy was desired, and yet a sparse iterative method was employed, rather than the LP approach.

Sparse iterative methods can be coarsely categorized into first-order methods (FOMs) and counterfactual regret minimization (CFR) variants [Zinkevich et al., 2007]. [1] CFR variants have dominated the ACPC in recent years, and as mentioned a CFR variant was used to (almost) solve limit texas holdem. CFR variants all have a convergence rate on the order of $1/\sqrt{T}$, where T is the number of iterations [Zinkevich et al., 2007, Lanctot et al., 2009, Johanson et al., 2012, Tammelin et al., 2015]. In contrast to this, FOMs such as the excessive gap technique (EGT) [Nesterov, 2005a] and mirror prox [Nemirovski, 2004] achieve a convergence rate of $1/T$, with an iteration cost that is only 2-3 times that of CFR, when instantiated with an appropriate smoothing technique for EFGs [Hoda et al., 2010]. Given this seeming disparity in convergence rate, it is surprising that CFR variants are preferred in practice. This chapter will largely focus on the practical and theoretical acceleration of FOMs with a $1/T$ convergence rate. The current main results of this section are of theoretical nature, showing that proper choice of Bregman divergence as well as careful analysis leads to a significant improvement of the dependence on game dimension for the convergence rate of FOMs for EFGs. The main future research proposal for this chapter is to make such FOMs better than, or at lest competitive with, CFR for practical game solving.

3.1 Related work

Nash equilibrium computation is a topic that has received much attention in the literature [Littman and Stone, 2003, Lipton et al., 2003, Zinkevich et al., 2007, Jiang and Leyton-Brown, 2011, Kroer and Sandholm, 2014,

[1] Waugh and Bagnell [2015] showed how CFR can be interpreted as a FOM. Nonetheless, the distinction between FOMs and CFR variants will be useful and sufficient for our purposes.

Daskalakis et al., 2015]. The equilibrium-finding problems vary quite a bit based on their characteristics; here we restrict our attention to two-player zero-sum sequential games.

Koller et al. [1996] present an LP whose size is linear in the size of the game tree. This approach, coupled with lossless abstraction techniques, was used to solve Rhode-Island hold'em [Shi and Littman, 2002, Gilpin and Sandholm, 2007c], a game with 3.1 billion nodes (roughly size $5 \cdot 10^7$ after lossless abstraction). However, for games larger than this, the resuting LPs tend to not fit in the computer memory thus requiring approximate solution techniques. These techniques fall into two categories: iterative ϵ-Nash equilibrium-finding algorithms and game abstraction techniques [Sandholm, 2010].

The most popular iterative Nash equilibrium algorithm is undoubtedly the counterfactual regret minimization (CFR) algorithm [Zinkevich et al., 2007] and its sampling-based variant monte-carlo CFR (MCCFR) [Lanctot et al., 2009]. Both of these regret-minimization algorithms perform local regret-based updates at each information set. Despite their slow convergence rate of $O(\frac{1}{\epsilon^2})$, they perform very well in pratice, likely owing to their very cheap iterations. As a stochastic algorithm MCCFR touches only the sampled part of the game tree. Also, as experimentally shown by Lanctot et al. [2009], even CFR can prune large parts of the game tree due to actions with probability zero.

Hoda et al. [2010] has suggested the only other FOM, a customization of EGT with a convergence rate of $O(\frac{1}{\epsilon})$, that scales to solve large games. Their work forms the basis of our approach, and we will compare to this work extensively in the following sections. Gilpin et al. [2012] give an algorithm with convergence rate $O(\ln(\frac{1}{\epsilon}))$. But their bound has a dependence on a certain condition number of the sequence-form payoff matrix, which is difficult to estimate; in fact, estimating it may be as hard as solving the Nash equilibrium problem itself [Mordukhovich et al., 2010]. This parameter is also not guaranteed to be polynomial in the size of the game. As a result they show a $O(\frac{1}{\epsilon})$ bound in the worst case.

Finally, Bosansky et al. [2014] develop an iterative double-oracle algorithm for exact equilibrium computation. However, this algorithm only scales for games where it can identify an equilibrium of small support, and thus suffers from the same performance issues as the general LP approach for large EFGs.

3.2 Accelerating first-order methods through better smoothing

3.2.1 Introduction

Nash equilibrium computation of a two-player zero-sum EFG with perfect recall admits a Bilinear Saddle Point Problem (BSPP) formulation where the domains are given by the polytopes that encode strategy spaces of the players. The classical FOMs to solve BSPPs such as *mirror prox* (MP) [Nemirovski, 2004] or the *excessive gap technique* (EGT) [Nesterov, 2005a] utilize *distance-generating functions* (DGFs) to measure appropriate notions of distances over the domains. Then the convergence rate of these FOMs relies on the DGFs and their relation to the domains in two critical ways: Through the strong convexity parameters of the DGFs, and set widths of the domains as measured by the DGFs.

Hoda et al. [2010] introduced a general framework for constructing DGFs for *treeplexes*—a class of convex polytopes that generalize the domains associated with the sequence-form strategy spaces. While they also established strong convexity bounds for their DGFs in some special cases, these lead to very weak bounds and result in slow convergence rates. For more general treeplexes, in our preliminary conference paper Kroer et al. [2015], we developed explicit strong convexity bounds for entropy-based DGFs (a particular subclass of DGFs) for the first time. These bounds from Kroer et al. [2015] generate the current state-of-the-art parameters associated with the convergence rate for FOMs with $O(\frac{1}{\epsilon})$ convergence. The results presented in this section are a combination of the results from Kroer et al. [2015] and currently unpublished results.

In this section we construct a new weighting scheme for such entropy-based DGFs. This weighting scheme leads to new and improved bounds on the strong convexity parameter associated with general treeplex domains. In particular, our new bounds are first-of-their kind as they have no dependence on the branching operation of the treeplex. Our strong convexity result allows us to improve the convergence rate of FOMs by a factor of $O(b^d d)$ (where b is the average branching factor for a player and d is the depth of the EFG) compared to the prior state-of-the-art results from Kroer et al. [2015]. Our bounds parallel the simplex case for matrix games where the entropy function also achieves a logarithmic dependence on the dimension of the simplex domain. From a practical perspective, we emphasize that such an improvement of the strong convexity parameter is especially critical for stochastic FOMs, where it is not possible to speed up the search through line search techniques.

The top poker bots at the Annual Computer Poker Competition are created with the monte-carlo counterfactual regret minimization (MCCFR) algorithm [Lanctot et al., 2009], which is a sampling variant of the counterfactual regret minimization algorithm (CFR) [Zinkevich et al., 2007, Brown et al., 2015]. Inspired by this success, we also describe a family of sampling schemes that lead to unbiased gradient estimators for EFGs. Using these estimators and our DGF, we instantate the first stochastic FOM for EFGs.

Finally, we complement our theoretical results with numerical experiments to investigate the speed up of FOMs with convergence rate $O(\frac{1}{\epsilon})$ and compare the performance of these algorithms with the premier regret-based methods CFR and MCCFR. Our experiments show that FOMs are substantially faster than CFR algorithms for medium-to-high-accuracy solutions. We also test the impact of stronger bounds on the strong convexity parameter: we instantiate MP and EGT with the parameters developed in this paper, and compare the performance to the parameters developed in our preliminary conference version. These experiments illustrate that, for both MP and EGT algorithms, the tighter parameters developed here lead to better practical convergence rate.

The rest of the section is organized as follows. We present the general class of problems that we address—bilinear saddle-point problems—and describes how they relate to EFGs in Section 3.2.2. Then Section 3.2.3 describes our optimization setup. We introduce treeplexes, the class of convex polytopes that define our domains of the optimization problems in Section 3.2.4. Our focus is on dilated entropy based DGFs; we introduce these in Section 3.2.5 and present our main results—bounds on the associated strong convexity parameter and treeplex diameter. In Section 3.2.6 we demonstrate the use of our results on instantiating MP algorithm. Section 3.2.7 introduces gradient estimators that enable the use of stochastic MP. We compare our approach with the current state-of-art in EFG solving and discuss the extent of theoretical improvements achievable via our approach in Section 3.2.8. Section 3.2.9 presents numerical experiments testing the effect of various parameters on the performance of our approach as well as comparing the performance of our approach to CFR and MCCFR. We close with a summary of our results and a few compelling further research directions in Section 3.2.10.

3.2.2 Problem setup

Computing a Nash equilibrium in a two-player zero-sum EFG with perfect recall can be formulated as a Bilinear Saddle Point Problem (BSPP):

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} \langle x, Ay \rangle = \max_{y \in \mathcal{Y}} \min_{x \in \mathcal{X}} \langle x, Ay \rangle. \tag{3.1}$$

This is known as the *sequence-form* formulation [Romanovskii, 1962, Koller et al., 1996, von Stengel, 1996]. In this formulation, x and y correspond to the nonnegative strategy vectors for players 1 and 2 and the sets $\mathcal{X}, \mathcal{Y}$ are convex polyhedral reformulations of the sequential strategy space of these players. Here $\mathcal{X}, \mathcal{Y}$

are defined by the constraints $Ex = e, Fy = f$, where each row of E, F encodes part of the sequential nature of the strategy vectors, the right hand-side vectors e, f are $|\mathcal{I}_1|, |\mathcal{I}_2|$-dimensional vectors, and $\mathcal{I}_i$ is the information sets for player i. For a complete treatment of this formulation, see von Stengel [1996].

Our theoretical developments mainly exploit the treeplex domain structure and are independent of other structural assumptions resulting from EFGs. Therefore, we describe our results for the general BSPPs. We follow the presentation and notation of Juditsky and Nemirovski [2011a,b] for BSPPs. For notation and presentation of treeplex structure, we follow Hoda et al. [2010], Kroer et al. [2015].

Basic notation

We let $\langle x, y\rangle$ denote the standard inner product of vectors x, y. Given a vector $x \in \mathbb{R}^n$, we let $\|x\|_p$ denote its ℓ_p norm given by $\|x\|_p := (\sum_{i=1}^n |x_i|^p)^{1/p}$ for $p \in [1, \infty)$ and $\|x\|_\infty := \max_{i\in[n]} |x_i|$ for $p = \infty$. Throughout this paper, we use Matlab notation to denote vector and matrices, i.e., $[x; y]$ denotes the concatenation of two column vectors x, y. For a given set Q, we let $\mathrm{ri}\,(Q)$ denote its relative interior. Given $n \in \mathbb{N}$, we denote the simplex $\Delta_n := \{x \in \mathbb{R}^n_+ : \sum_{i=1}^n x_i = 1\}$.

3.2.3 Optimization setup

In its most general form a BSPP is defined as

$$\mathrm{Opt} := \max_{y\in\mathcal{Y}} \min_{x\in\mathcal{X}} \phi(x, y), \tag{$\mathcal{S}$}$$

where $\mathcal{X}, \mathcal{Y}$ are nonempty convex compact sets in Euclidean spaces $\mathbf{E}_x, \mathbf{E}_y$ and $\phi(x, y) = \upsilon + \langle a_1, x\rangle + \langle a_2, y\rangle + \langle y, Ax\rangle$. We let $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$; so $\phi(x, y) : \mathcal{Z} \to \mathbb{R}$. In the context of EFG solving, $\phi(x, y)$ is simply the inner product given in (3.1).

The BSPP $(\mathcal{S})$ gives rise to two convex optimization problems that are dual to each other:

$$\begin{array}{rcll} \mathrm{Opt}(P) & = & \min_{x\in\mathcal{X}}[\overline{\phi}(x) := \max_{y\in\mathcal{Y}} \phi(x, y)] & (P) \\ \mathrm{Opt}(D) & = & \max_{y\in\mathcal{Y}}[\underline{\phi}(y) := \min_{x\in\mathcal{X}} \phi(x, y)] & (D) \end{array}$$

with $\mathrm{Opt}(P) = \mathrm{Opt}(D) = \mathrm{Opt}$. It is well known that the solutions to $(\mathcal{S})$ — the saddle points of ϕ on $\mathcal{X} \times \mathcal{Y}$ — are exactly the pairs $z = [x; y]$ comprised of optimal solutions to the problems (P) and (D). We quantify the accuracy of a candidate solution $z = [x; y]$ with the *saddle point residual*

$$\epsilon_{\mathrm{sad}}(z) \;:=\; \overline{\phi}(x) - \underline{\phi}(y) = \underbrace{\left[\overline{\phi}(x) - \mathrm{Opt}(P)\right]}_{\geq 0} + \underbrace{\left[\mathrm{Opt}(D) - \underline{\phi}(y)\right]}_{\geq 0}.$$

In the context of EFG, $\epsilon_{\mathrm{sad}}(z)$ measures the proximity to being an ϵ-Nash equilibrium.

The problems (P) and (D) also give rise to the following *variational inequality*: find $z_* \in \mathcal{Z}$ s.t.

$$\langle F(z), z - z_*\rangle \geq 0 \;\text{ for all } z \in \mathcal{Z}, \tag{3.2}$$

where $F : \mathcal{Z} \mapsto \mathbf{E}_x \times \mathbf{E}_y$ is the affine monotone operator defined by

$$F(x, y) := \left[F_x(y) = \frac{\partial\phi(x, y)}{\partial x}; F_y(x) = -\frac{\partial\phi(x, y)}{\partial y}\right].$$

For EFG-solving purposes, $F(x, y)$ corresponds to the map returning the gradients $[Ay, -A^\top x]$ from (3.1). Then (3.2) states that for any other strategy pair $z = [x, y]$, each player weakly prefers deviating to their part of $z_* = [x_*, y_*]$.

General framework for FOMs

Most FOMs capable of solving BSPP $(\mathcal{S})$ are quite flexible in terms of adjusting to the geometry of the problem characterized by the domains $\mathcal{X}, \mathcal{Y}$ of the BSPP $(\mathcal{S})$. The following components are standard in forming the setup for such FOMs:

- *Norm*: $\|\cdot\|$ on the Euclidean space $\mathbf{E}$ where the domain $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ of $(\mathcal{S})$ lives, along with its dual norm $\|\zeta\|_* = \max_{\|z\| \le 1} \langle \zeta, z \rangle$.
- *Distance-Generating Function* (DGF): A function $\omega(z) : \mathcal{Z} \to \mathbb{R}$, which is convex and continuous on $\mathcal{Z}$, and admits a continuous selection of subgradients $\omega'(z)$ on the set $\mathcal{Z}^\circ := \{z \in \mathcal{Z} : \partial\omega(z) \neq \emptyset\}$ (here $\partial\omega(z)$ is a subdifferential of ω taken at z), and is strongly convex with modulus φ w.r.t. the norm $\|\cdot\|$:
$$\forall z', z'' \in \mathcal{Z}^\circ : \ \langle \omega'(z') - \omega'(z''), z' - z'' \rangle \ge \varphi \|z' - z''\|^2. \tag{3.3}$$
- *Bregman distance*: $V_z(u) := \omega(u) - \omega(z) - \langle \omega'(z), u - z \rangle$ for all $z \in \mathcal{Z}^\circ$ and $u \in \mathcal{Z}$.
- *Prox-mapping*: Given a *prox center* $z \in \mathcal{Z}^\circ$,
$$\mathrm{Prox}_z(\xi) := \operatorname*{argmin}_{w \in \mathcal{Z}} \{\langle \xi, w \rangle + V_z(w)\} : \mathbf{E} \to \mathcal{Z}^\circ.$$
For properly chosen stepsizes, the prox-mapping becomes a contraction. This is critical in the convergence analysis of FOMs. Furthermore, when the DGF is taken as the squared ℓ_2 norm, the prox mapping becomes the usual projection operation of the vector $z - \xi$ onto $\mathcal{Z}$.
- *ω-center*: $z_\omega := \operatorname*{argmin}_{z \in \mathcal{Z}} \omega(z) \in \mathcal{Z}^\circ$ of $\mathcal{Z}$.
- *Set width*: $\Omega = \Omega_z := \max_{z \in \mathcal{Z}} V_{z_\omega}(z) \le \max_{z \in \mathcal{Z}} \omega(z) - \min_{z \in \mathcal{Z}} \omega(z)$.
- *Lipschitz constant*: $\mathcal{L}$ of F from $\|\cdot\|$ to $\|\cdot\|_*$, satisfying $\|F(z) - F(z')\|_* \le \mathcal{L}\|z - z'\|, \forall z, z'$.

In the standard customization of a FOM to solving a BSPP $(\mathcal{S})$, we associate a norm $\|\cdot\|_x$ and a DGF $\omega_x(\cdot)$ with the domain $\mathcal{X}$, and similarly $\|\cdot\|_y$, $\omega_y(\cdot)$ with the domain $\mathcal{Y}$. Given two scalars $\alpha_x, \alpha_y > 0$, we build the DGF $\omega(z)$ and ω-center z_ω for $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ as

$$\omega(z) = \alpha_x \omega_x(x) + \alpha_y \omega_y(y) \quad \text{and} \quad z_\omega = [x_{\omega_x}; y_{\omega_y}],$$

where $\omega_x(\cdot)$ and $\omega_y(\cdot)$ as well as x_{ω_x} and y_{ω_y} are customized based on the geometry of the domains $\mathcal{X}, \mathcal{Y}$. In this construction, the flexibility in determining the scalars $\alpha_x, \alpha_y > 0$ is useful in optimizing the overall convergence rate. Moreover, by letting $\xi = [\xi_x; \xi_y]$ and $z = [x; y]$, the prox mapping becomes decomposable as

$$\mathrm{Prox}_z(\xi) = \left[\mathrm{Prox}_x^{\omega_x}\left(\frac{\xi_x}{\alpha_x}\right) ; \mathrm{Prox}_y^{\omega_y}\left(\frac{\xi_y}{\alpha_y}\right) \right],$$

where $\mathrm{Prox}_x^{\omega_x}(\cdot)$ and $\mathrm{Prox}_y^{\omega_y}(\cdot)$ are respectively prox mappings w.r.t. $\omega_x(\cdot)$ in domain $\mathcal{X}$ and $\omega_y(\cdot)$ in domain $\mathcal{Y}$.

Based on this setup, we formally state the *Mirror Prox* (MP) algorithm in Algorithm 1.

Suppose the step sizes in the MP algorithm satisfy $\gamma_t = \mathcal{L}^{-1}$. Then, at every iteration $t \ge 1$ of the MP algorithm, the corresponding solution $z^t = [x^t; y^t]$ satisfies $x^t \in \mathcal{X}$, $y^t \in \mathcal{Y}$, and

$$\overline{\phi}(x^t) - \underline{\phi}(y^t) = \epsilon_{\text{sad}}(z^t) \le \frac{\Omega \mathcal{L}}{\varphi t}.$$

ALGORITHM 1: Mirror Prox

input : ω-center z_ω, positive step sizes $\{\gamma_t\}$, and $\epsilon > 0$
output: $z^t (= [x^t; y^t])$

1 $t = 1; z_1 := z_\omega;$
2 **while** $\epsilon_{\text{sad}}(z^t) > \epsilon$ **do**
3 $\quad v_t = \text{Prox}_{z_t}(\gamma_t F(z_t));$
4 $\quad z_{t+1} = \text{Prox}_{z_t}(\gamma_t F(v_t));$
5 $\quad z^t = \left[\sum_{s=1}^t \gamma_s\right]^{-1} \sum_{s=1}^t \gamma_s v_s;$
6 $\quad t = t + 1;$

Consequently, the MP algorithm [Nemirovski, 2004] has a convergence rate of $O(1)\frac{\Omega\mathcal{L}}{\varphi\epsilon}$. Similarly, the EGT algorithm [Nesterov, 2005a] can be set up using the same framework to achieve a convergence rate of $O(1)\frac{\Omega\mathcal{L}}{\varphi\epsilon}$.

Alternatively, it is also possible to set up a $1/\epsilon^2$ algorithm that has a better dependence on game constants. The *mirror descent* MD algorithm is similar to Algorithm 1, but only does one prox mapping and uses that point both for averaging and as the new iterate. With the same setup as given above, the MD algorithm achieves a convergence rate of $O(1)\frac{\sqrt{\Omega}\mathcal{L}}{\sqrt{\varphi\epsilon^2}}$

Stochastic FOMs

In practice, each iteration of an FOM might be prohibitively expensive, usually because the gradient is expensive to compute. In such a case, stochastic FOMs that work with unbiased estimates of the gradients constructed through sampling are often advantageous. Juditsky et al. [2011] discuss a stochastic variant of MP for BSSP, namely *Stochastic MP* (SMP). Algorithm 1 is modified as follows. Steps 3 and 4 do not use the exact gradients, $F(z_t)$ and $F(v_t)$, but rather unbiased estimators thereof. Specifically, for every $x \in \mathcal{X}$, we assume access to a probability distribution Π_x such that $\mathbb{E}_{\xi\sim\Pi_x}[\xi] = x$. Similarly, we assume access to P_y for $y \in \mathcal{Y}$ such that $\mathbb{E}_{\eta\sim P_y}[\eta] = y$. We define variance as follows:

$$\sigma_x^2 = \sup_{x\in\mathcal{X}} \mathbb{E}\left\{\|A^\top[\xi_x - x]\|_{y,*}^2\right\}, \quad \sigma_y^2 = \sup_{y\in\mathcal{Y}} \mathbb{E}\left\{\|A[\eta_y - y]\|_{x,*}^2\right\}$$

Juditsky et al. [2011] prove that after T iterations of SMP with stepsizes $\gamma_t = \min\left[\frac{1}{\sqrt{3}\mathcal{L}}, \sqrt{\frac{4\Omega}{7T\varphi(\sigma_x^2+\sigma_y^2)}}\right]$, the solution satisfies:

$$\mathbb{E}\left[\epsilon_{sad}(z^T)\right] \le \max\left[\frac{7\Omega\mathcal{L}}{2T\varphi}, 7\sqrt{\frac{2\Omega(\sigma_x^2+\sigma_y^2)}{3T\varphi}}\right]. \tag{3.4}$$

3.2.4 Treeplexes

Hoda et al. [2010] introduce the *treeplex*, a class of convex polytopes that encompass the sequence-form description of strategy spaces in perfect-recall EFGs.

Definition 5. *Treeplexes are defined recursively:*

1. Basic sets*: The standard simplex Δ_m is a treeplex.*
2. Cartesian product*: If $Q_1, \ldots, Q_k$ are treeplexes, then $Q_1 \times \cdots \times Q_k$ is a treeplex.*

3. Branching*: Given a treeplex $P \subseteq [0,1]^p$, a collection of treeplexes $Q = \{Q_1, \ldots, Q_k\}$ where $Q_j \subseteq [0,1]^{n_j}$, and $l = \{l_1, \ldots, l_k\} \subseteq \{1, \ldots, p\}$, the set defined by*

$$P\boxed{l}Q \coloneqq \left\{(x, y_1, \ldots, y_k) \in \mathbb{R}^{p+\sum_j n_j} : x \in P,\, y_1 \in x_{l_1} \cdot Q_1,\, \ldots,\, y_k \in x_{l_k} \cdot Q_k\right\}$$

is a treeplex. In this setup, we say x_{l_j} is the branching variable for the treeplex Q_j.

A treeplex is a tree of simplices where children are connected to their parents through the branching operation. In the branching operation, the child simplex domain is scaled by the value of the parent branching variable. Understanding the treeplex structure is crucial because the proofs of our main results rely on induction over these structures. For EFGs, the simplices correspond to the information sets of a single player and the whole treeplex represents that player's strategy space. The branching operation has a sequential interpretation: The vector x represents the decision variables at certain stages, while the vectors y_j represent the decision variables at the k potential following stages, depending on external outcomes. Here $k \leq p$ since some variables in x may not have subsequent decisions. For treeplexes, von Stengel [1996] has suggested a polyhedral representation of the form $Ex = e$ where the matrix E has its entries from $\{-1, 0, 1\}$ and the vector e has its entries in $\{0, 1\}$.

For a treeplex Q, we denote by S_Q the index set of the set of simplices contained in Q (in an EFG S_Q is the set of information sets belonging to the player). For each $j \in S_Q$, the treeplex rooted at the j-th simplex Δ^j is referred to as Q_j. Given vector $q \in Q$ and simplex Δ^j, we let $\mathbb{I}_j$ denote the set of indices of q that correspond to the variables in Δ^j and define q^j to be the sub vector of q corresponding to the variables in $\mathbb{I}_j$. For each simplex Δ^j and branch $i \in \mathbb{I}_j$, the set $\mathcal{D}_j^i$ represents the set of indices of simplices reached immediately after Δ^j by taking branch i (in an EFG $\mathcal{D}_j^i$ is the set of potential next-step information sets for the player). Given a vector $q \in Q$, simplex Δ^j, and index $i \in \mathbb{I}_j$, each child simplex Δ^k for every $k \in \mathcal{D}_j^i$ is scaled by q_i. Conversely, for a given simplex Δ^j, we let p_j denote the index in q of the parent branching variable q_{p_j} that Δ^j is scaled by. We use the convention that $q_{p_j} = 1$ if Q is such that no branching operation precedes Δ^j. For each $j \in S_Q$, d_j is the maximum depth of the treeplex rooted at Δ^j, that is, the maximum number of simplices reachable through a series of branching operations at Δ^j. Then d_Q gives the depth of Q. We use b_Q^j to identify the number of branching operations preceding j-th simplex in Q.

Figure 3.1 illustrates an example treeplex Q. Q is constructed from nine two-to-three-dimensional simplices $\Delta^1, \ldots, \Delta^9$. At level 1, we have taken the Cartesian product, denoted by $\times$, of the simplices Δ^1 and Δ^2. We have maximum depths $d_1 = 2$, $d_2 = 1$ beneath them. Since there are no preceding branching operations, the parent variables for these simplices Δ^1 and Δ^2 are $q_{p_1} = q_{p_2} = 1$. For Δ^1, the corresponding set of indices in the vector q is $\mathbb{I}_1 = \{1, 2\}$, while for Δ^2 we have $\mathbb{I}_2 = \{3, 4, 5\}$. At level 2, we have the simplices $\Delta^3, \ldots, \Delta^7$. The parent variable of Δ^3 is $q_{p_3} = q_1$; therefore, Δ^3 is scaled by the parent variable q_{p_3}. Similarly, each of the simplices $\Delta^3, \ldots, \Delta^7$ is scaled by their parent variables q_{p_j} that the branching operation was performed on. So on for Δ^8 and Δ^9 as well. The number of branching operations required to reach simplices Δ^1, Δ^3 and Δ^8 is $b_Q^1 = 0, b_Q^3 = 1$ and $b_Q^8 = 2$, respectively.

Note that we allow more than two-way branches; hence our formulation follows that of Kroer et al. [2015] and differs from that of Hoda et al. [2010]. As discussed in Hoda et al. [2010], it is possible to model sequence-form games by treeplexes that use only two-way branches. Yet, this can cause a large increase in the depth of the treeplex, thus leading to significant degradation in the strong convexity parameter. Because we handle multi-way branches directly in our framework, our approach is more effective in taking into account the structure of the sequence-form game and thereby resulting in better bounds on the associated strong convexity parameters and thus overall convergence rates.

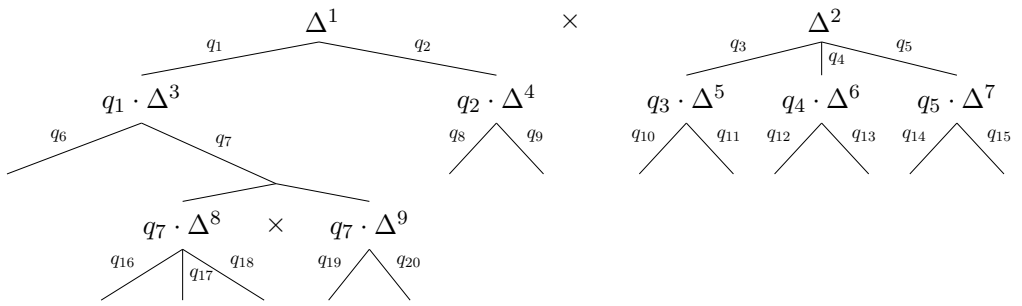


Figure 3.1: An example treeplex constructed from 9 simplices. Cartesian product operation is denoted by $\times$.

Our analysis requires a measure of the size of a treeplex Q. Thus, we define

$$M_Q \coloneqq \max_{q \in Q} \|q\|_1. \tag{3.5}$$

In the context of EFGs, suppose Q encodes player 1's strategy space; then M_Q is the maximum number of information sets with nonzero probability of being reached when player 1 has to follow a pure strategy while the other player may follow a mixed strategy. We also let

$$M_{Q,r} \coloneqq \max_{q \in Q} \sum_{j \in S_Q : b_Q^j \leq r} \|q^j\|_1. \tag{3.6}$$

Intuitively, $M_{Q,r}$ gives the maximum value of the ℓ_1 norm of any vector $q \in Q$ after removing the variables corresponding to simplices that are not within r branching operations of the root of Q.

3.2.5 Distance-generating functions with bounded strong convexity

In this section we introduce DGFs for domains with treeplex structures and establish their strong convexity parameters with respect to a given norm (see (3.3)).

The basic building block in our construction is the *entropy* DGF given by $\omega_e(z) = -\sum_{i=1}^n z_i \log(z_i)$, for the simplex Δ_n. It is well-known that $\omega_e(\cdot)$ is strongly convex with modulus 1 with respect to the ℓ_1 norm on Δ_n (see Juditsky and Nemirovski [2011a]). We will show that a suitable modification of this function achieves a desirable strong convexity parameter for the treeplex domain.

The treeplex structure is naturally related to the *dilation operation* [Hiriart-Urruty and Lemaréchal, 2001] defined as follows: Given a compact set $K \subseteq \mathbb{R}^d$ and a function $f : K \to \mathbb{R}$, we first define

$$\bar{K} \coloneqq \left\{ (t, z) \in \mathbb{R}^{d+1} : \ t \in [0, 1], \ z \in t \cdot K \right\}.$$

Definition 6. *Given a function $f(z)$, the* dilation operation *is the function $\bar{f} : \bar{K} \to \mathbb{R}$ given by*

$$\bar{f}(z, t) = \begin{cases} t \cdot f(z/t) & \text{if } t > 0 \\ 0 & \text{if } t = 0 \end{cases}.$$

Due to the recursive definition of a treeplex as a tree of simplexes, any function on a simplex can be converted to a function for an individual simplex in a treeplex by dilating with the parent variable q_{p_j}.

Based on the dilation operation, we define the dilated entropy function over the simplices of a treeplex. At each simplex j, we dilate the entropy function by the parent variable q_{p_j}:

Definition 7. *Given a treeplex Q and weights $\beta_j > 0$ for each $j \in S_Q$, we define the* dilated entropy function *as*

$$\omega(q) = \sum_{j \in S_Q} \beta_j q_{p_j} \sum_{i \in \mathbb{I}_j} \frac{q_i}{q_{p_j}} \log \frac{q_i}{q_{p_j}} = \sum_{j \in S_Q} \beta_j \sum_{i \in \mathbb{I}_j} q_i \log \frac{q_i}{q_{p_j}} \quad \text{for any } q \in Q,$$

where we follow the treeplex notation and p_j is the index of the branching variable preceding Δ^j, with the convention that $q_{p_j} = 1$ if Δ^j has no branching operation preceding it.

Remark 1. *Note that the dilated entropy function $\omega(\cdot)$ defined above is twice differentiable in the relative interior of treeplex Q and admits a continuous gradient selection. Moreover, for large enough weights β_j, we will demonstrate that it is strongly convex w.r.t. the ℓ_1 norm. Thus, the dilated entropy function is compatible with the ℓ_1 norm, as required by the BSPP setup.* ∎

Definition 7 above leads to a subset of the DGFs considered by Hoda et al. [2010]. Our main theoretical result shows that by a careful selection of the weights β_j, we can significantly improve the strong convexity bounds associated with the dilated entropy function.

Theorem 1. *For any treeplex Q, the dilated entropy function with weights $\beta_j = 2 + \sum_{r=1}^{d_j} 2^r (M_{Q_j,r} - 1)$ for all $j \in S_Q$ is strongly convex with modulus $\frac{1}{M_Q}$ w.r.t. the ℓ_1 norm.*

Previously, Hoda et al. [2010] have proven strong convexity of this scheme under weaker assumptions on the norm. But their analysis is only for the degradation associated with a single two-way branching step which fails to provide explicit strong convexity bounds. For the special case of *uniform treeplexes* (a significant restriction on the treeplex construction), Hoda et al. [2010] do give explicit bounds. Nevertheless, in Section 3.2.5 we show that their bounds are significantly looser than ours even for this special case.

To our knowledge, the first and also best known strong convexity bounds for general treeplexes were proved in Kroer et al. [2015], the preliminary conference version of this work. Theorem 1 improves upon our prior bounds by exchanging a factor of $|S_Q|$ with a factor of M_Q. Note that $|S_Q|$ is tied to the branching factor associated with branching operations in the treeplex Q whereas M_Q is not. Therefore, our result removes all of the dependence of the strong convexity parameter on the branching factor and hence significantly improves upon our preliminary results.

In Theorem 2 we use our strong convexity result to show a polytope diameter that has only a logarithmic dependence on the branching factor. As a consequence, the associated dilated entropy DGF when used in FOMs such as MP and EGT for solving EFGs leads to the same improvement in their convergence rate.

Treeplex width

The convergence rates of FOMs such as MP and EGT algorithms depend on the diameter-to-strong convexity parameter ratio $\frac{\Omega}{\varphi}$, as described in Section 3.2.3. In order to establish full results on the convergence rates of these FOMs, we now bound this ratio using Theorem 1.

Theorem 2. *For a treeplex Q, the dilated entropy function with weights $\widehat{\beta}_j = M_Q(2 + \sum_{r=1}^{d_j} 2^r (M_{Q_j,r} - 1))$ for each $j \in S_Q$ results in $\frac{\Omega}{\varphi} \leq M_Q^2 2^{d_Q+2} \log m$ where m is the dimension of the largest simplex Δ^j for $j \in S_Q$ in the treeplex structure.*

3.2.6 Instantiating FOMs for extensive-form game solving

We now describe how to instantiate MP and MD for solving two-player zero-sum EFGs of the form given in (3.1) for treeplex domains. The instantiation of EGT is similar. Below we state the customization of all the definitions from Section 3.2.3 for our problem.

Let $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ be the treeplex corresponding to the cross product of the strategy spaces $\mathcal{X}, \mathcal{Y}$ for the two players in (3.1) and m be the size of the largest simplex in this treeplex. Because $\mathcal{X}$ and $\mathcal{Y}$ are treeplexes, it is immediately apparent that they are closed, convex, and bounded. We use the ℓ_1 norm on both of the embedding spaces $\mathbf{E}_x, \mathbf{E}_y$; and as our DGF for $\mathcal{Z}$ compatible with ℓ_1 norm, we use the dilated entropy DGF scaled with weights given in Theorem 2. Then Theorem 2 gives our bound on $\frac{\Omega}{\varphi}$. Because the dual norm of ℓ_1 norm is the ℓ_∞ norm, an upper bound on the Lipschitz constant of our monotone operator, namely $\mathcal{L}$, is given by:

$$\mathcal{L} = \max\left\{\mathcal{L}_1, \mathcal{L}_2\right\}, \text{where}$$

$$\mathcal{L}_1 = \max_{y \in \mathcal{Y}} \left| \max_j (Ay)_j - \min_j (Ay)_j \right|, \quad \mathcal{L}_2 = \max_{x \in \mathcal{X}} \left| \max_i (A^\top x)_i - \min_i (A^\top x)_i \right|.$$

Remark 2. *Note that $\mathcal{L}$ is not at the scale of the maximum payoff difference in the original game. The differences involved in $\mathcal{L}_1, \mathcal{L}_2$ are scaled by the probability of the observed nature outcomes on the path of each sequence. Thus, our Lipschitz constant $\mathcal{L}$ is exponentially smaller (in the number of observed nature steps on the path to the maximizing sequence) than the maximum payoff difference in the original EFG.* ∎

Theorem 2 immediately leads to the following convergence rate result for MP or EGT equipped with dilated entropy DGFs to solve EFGs (and more generally BSPPs over treeplex domains).

Theorem 3. *Consider a BSPP over a treeplex domain $\mathcal{Z}$. Suppose MP (or EGT) equipped with the dilated entropy DGF with weights $\beta_j = 2 + \sum_{r=1}^{d_j} 2^r (M_{\mathcal{Z}_j, r} - 1)$ for all $j \in S_{\mathcal{Z}}$ is used to solve the BSPP. Then the convergence rate of this algorithm is*

$$\frac{\mathcal{L}\, M_{\mathcal{Z}}^2\, 2^{d_{\mathcal{Z}}+2} \log m}{\epsilon}.$$

This rate is, to our knowledge, the state-of-the-art for FOMs with $O(1/\epsilon)$ convergence rate for EFGs.

We can also instantiate MD, which has a better dependence on game constants, but sacrifices the linear dependence on ϵ.

Theorem 4. *Consider a BSPP over a treeplex domain $\mathcal{Z}$. Suppose MD equipped with the dilated entropy DGF with weights $\beta_j = 2 + \sum_{r=1}^{d_j} 2^r (M_{\mathcal{Z}_j, r} - 1)$ for all $j \in S_{\mathcal{Z}}$ is used to solve the BSPP. Then the convergence rate of this algorithm is*

$$\frac{\mathcal{L}\, M_{\mathcal{Z}}\, 2^{d_{\mathcal{Z}}/2+1} \log m}{\epsilon^2}.$$

For SMP we further need unbiased gradient estimators.

3.2.7 Sampling

We now introduce a broad class of unbiased gradient estimators for EFGs. We describe how to generate a gradient estimate η of $x^\top A$ given $x \in \mathcal{X}$, that is, the gradient for player 2. Given $y \in \mathcal{Y}$, the estimate ξ of $A^\top y$ is generated analogously. This class explicitly uses the tree structure representation of EFGs, so we

introduce some notation for it. We define H to be the set of nodes (or equivalently, histories) in the game tree. For any node h, we define $A(h)$ to be the set of actions available at h. The node reached by taking action $a \in A(h)$ at h is denoted by $h[a]$. We let $p_{h,x}$ be the probability distribution over actions $A(h)$ given by the current iterate x. If h is a chance node, x has no effect on the distribution. We let $u_2(h)$ be the utility of player 2 for reaching a terminal node h, and η_h be the index in η corresponding to a given terminal node h.

An *EFG gradient estimator* is defined by a sampling-description function $\mathcal{C} : H \to \mathbb{N} \cup \{all\}$, where $\mathcal{C}(h)$ gives the number of samples drawn at the node $h \in H$. The estimate η of $x^\top A$ is then generated using the following recursive sampling scheme:

$$\mathbf{Sample}(h,\tau): \begin{cases} \text{if } h \text{ is a terminal node}: \eta_h = \tau \cdot u_2(h) \\ \text{else if } h \text{ belongs to player 2: } \forall a \in A(h): \mathbf{Sample}(h[a], \tau) \\ \text{else if } \mathcal{C}(h) = all\text{: } \forall a \in A(h): \mathbf{Sample}(h[a], p_{h,x}(a) \cdot \tau) \\ \text{else: } \begin{cases} \text{Draw } \left\{a_1, \ldots, a_{\mathcal{C}(h)}\right\} \sim p_{h,x}, \\ \forall j = 1, \ldots, \mathcal{C}(h): \mathbf{Sample}(h[a_j], \tau \cdot \frac{1}{\mathcal{C}(h)}) \end{cases} \end{cases}.$$

Sampling is initiated with $h = r, \tau = 1$, where r is the root node of the game tree. From the definition of this sampling scheme, it is clear that the expected value of any EFG estimator is exactly $x^\top A$ as desired. It is possible to generalize this class to sample $A(h)$ at nodes h belonging to player 2 as well.

Lanctot et al. [2009] introduced several unbiased sampling schemes for EFGs. These correspond to certain specializations of our general sampling scheme. *Chance sampling*, $\mathcal{C}_c(h)$, is where a single sample is drawn if the node is a nature node, and otherwise all actions are chosen. This corresponds to the following EFG estimator:

$$\mathcal{C}_c(h) = \begin{cases} 1 & \text{if } h \text{ is a chance node} \\ all & \text{else} \end{cases}.$$

In *external sampling*, $\mathcal{C}_e(h)$, a single sample is drawn for the nodes that do not belong to the current player. Hence, when we estimate $x^\top A$, we get $\mathcal{C}_e(h) = 1$.

In our experiments, for a given positive integer k, we focus on the following estimator

$$\mathcal{C}_{c,k}(h) = \begin{cases} k & \text{if } h \text{ is a chance node} \\ all & \text{else} \end{cases}. \tag{3.7}$$

We now provide a simple upper bound for the variance of this class of gradient estimators. We will need the maximum payoff difference for Player 2 in the EFG, which we denote by $\Gamma_2 = \max_{h,h'} u_2(h) - u_2(h')$. An analogous theorem holds for σ_y^2.

Theorem 5. *Any gradient estimator $\mathcal{C} : H \to \mathbb{N} \cup \{all\}$, has variance at most $\sigma_x^2 \leq \Gamma_2^2$.*

This result immediately leads to the following first-of-its-kind bound on the convergence rate of a stochastic FOM for EFG solving:

Theorem 6. *Consider a BSPP over a treeplex domain $\mathcal{Z}$. Suppose SMP equipped with the dilated entropy DGF with weights $\beta_j = 2 + \sum_{r=1}^{d_j} 2^r (M_{\mathcal{Z}_j,r} - 1)$ for all $j \in S_{\mathcal{Z}}$ and any gradient estimator $\mathcal{C}$ is used to solve the BSPP. Using stepsizes $\gamma_t = \min\left[\frac{1}{\sqrt{3}\mathcal{L}}, \sqrt{\frac{4M_{\mathcal{Z}}^2 2^{d_{\mathcal{Z}}+2} \log m}{7T(\Gamma_1^2+\Gamma_2^2)}}\right]$, the expected convergence rate of this algorithm is*

$$\mathbb{E}\left[\epsilon_{sad}(z^T)\right] \leq \max\left[\frac{7\mathcal{L}M_{\mathcal{Z}}^2 2^{d_{\mathcal{Z}}+2}\log m}{2T},\ 7\sqrt{\frac{2M_{\mathcal{Z}}^2 2^{d_{\mathcal{Z}}+2}\log m(\Gamma_1^2+\Gamma_2^2)}{3T}}\right]. \tag{3.8}$$

3.2.8 Discussion of theoretical improvements

We first discuss the theoretical improvement in convergence rate originating from our new strong convexity and set width parameter bounds for FOMs achieving $O(\frac{1}{\epsilon})$ rate and then elaborate on how our convergence rate result is positioned with respect to other equilibrium-finding algorithms from the literature.

Improvement on strong convexity and set width parameter bounds

The ratio $\frac{\Omega}{\varphi}$ of set diameter over the strong convexity parameter is important for FOMs that rely on a prox function, such as Nesterov's EGT, MP, and SMP.

Compared to the rate obtained in our preliminary conference version [Kroer et al., 2015], we get the following improvement: Assuming that the number of actions available at each information set is on average a, our bound improves the convergence rate of [Kroer et al., 2015] by a factor of $O(d_{\mathcal{Z}} \cdot a^{d_{\mathcal{Z}}})$.

As mentioned previously, Hoda et al. [2010] proved only explicit bounds for the special case of uniform treeplexes that are constructed as follows: 1) A base treeplex Q along with a subset of b indices from it for branching operations is chosen. 2) At each depth d, a Cartesian product operation of size k is applied. 3) Each element in a Cartesian product is an instance of the base treeplex with a size b branching operation leading to depth $d-1$ uniform treeplexes constructed in the same way. Given bounds Ω_b, φ_b for the base treeplex, the bound of Hoda et al. [2010] for a uniform treeplex with d uniform treeplex levels (note that the total depth of the constructed treeplex is $d \cdot d_Q$, where d_Q is the depth of the base treeplex Q) is

$$\frac{\Omega}{\varphi} \leq O\left(b^{2d-2} k^{2d+2} d^2 M_{Q_b}^2 \frac{\Omega_b}{\varphi_b}\right).$$

Then when the base treeplex is a simplex of dimension m, their bound for the dilated entropy on a uniform treeplex $\mathcal{Z}$ becomes

$$\frac{\Omega}{\varphi} \leq O\left(|S_{\mathcal{Z}}|^2 d_{\mathcal{Z}}^2 \log m\right).$$

Even for the special case of a uniform treeplex with a base simplex, comparing Theorem 2 to their bound, we see that our general bound improves the associated constants by exchanging $O(|S_Q|^2 d_{\mathcal{Z}}^2)$ with $O(M_Q^2 2^{d_{\mathcal{Z}}})$. Since M_Q does not depend on the branching operation in the treeplex, whereas $|S_Q|$ does, these are also the first bounds to remove any dependence on the branching operation.

A more intuitive bound on CFR convergence rate

The CFR literature has had several papers improving previous bounds on the convergence rate of the algorithm [Zinkevich et al., 2007, Lanctot et al., 2009, Burch et al., 2012]. The most recent such result shows a bound of [Burch et al., 2012]:

$$\frac{\Delta_i M_i(\sigma_i^*)\sqrt{|A_i|}}{\sqrt{T}}$$

where Δ_i is the maximum payoff difference in the game, and $M_i(\sigma_i^*)$ is equal to $\sum_{B \in \mathcal{B}_i} \pi_i^{\sigma^*}(B)\sqrt{|B|}$. This bound is somewhat difficult to compare to the original CFR bound as well as our results. In this section we express this bound in terms of a uniform game where players 1, 2, and nature alternate d times and have the same number of actions $|A_1| = |A_2| = |A_0| = b$. Note that this corresponds to each player's strategy space being a treeplex with depth d, branching operations of size b, and Cartesian product operations of size k.

Let player i be the player that goes second. For such a game, we get

$$\begin{aligned}
&\sum_{B\in\mathcal{B}_i} \pi_i^{\sigma^*}(B)\sqrt{|B|} = \sum_{B\in\mathcal{B}_i} \pi_i^{\sigma^*}(B)\sqrt{k} \\
=&\sum_{B\in\mathcal{B}_i} \pi_i^{\sigma^*}(B)\sqrt{k} = \sum_{B\in\mathcal{B}_i} \frac{1}{b^{d-d_B}}\sqrt{k} \\
=&\sum_{h=1}^{d} \sum_{B\in\mathcal{B}_i, d_B=h} \frac{1}{b^{d-h}}\sqrt{k} = \sum_{h=1}^{d} (bk)^{d-h}\frac{1}{b^{d-h}}\sqrt{k} \\
=&\sum_{h=1}^{d} k^{d-h}\sqrt{k} = \sqrt{k}\frac{k^h-1}{k-1},
\end{aligned}$$

As can be seen this value is essentially the same as M_Q, i.e. the maximum value of the l_1 norm over Q.

Improvements in extensive-form game convergence rate

CFR, EGT, MP, and SMP all need to keep track of a constant number of current and/or average iterates, so the memory usage of all four algorithms is of the same order: when gradients are computed using an iterative approach as opposed to storing matrices or matrix decompositions, each algorithm requires a constant times the number of sequences in the sequence-form representation. Therefore, we compare mainly the number of iterations required by each algorithm.

Theorem 2 establishes the best known results on strong convexity parameter and set width of prox functions based on dilated entropy DGF over treeplex domains; and consequently, Theorems 3 and 6 establish the best known convergence results for all FOMs based on dilated entropy DGF such as MP, MD, SMP, and EGT algorithms.

In comparison to such FOMs, CFR and its stochastic counterpart MCCFR are the most popular algorithms for practical large-scale equilibrium finding. CFR has a $O(\frac{1}{\epsilon^2})$ convergence rate; but its dependence on the number of information sets is only linear (and sometimes sublinear [Lanctot et al., 2009]). Since our results have a quadratic dependence on M_Q^2, CFR has a better dependence on game constants and can be more attractive for obtaining low-quality solutions quickly for games with many information sets. MCCFR has a similar convergence rate [Lanctot et al., 2009]. When we apply SMP with our results, we achieve an expected convergence rate of $O(\frac{1}{\epsilon^2})$, but with only a square root dependence on the ratio $\frac{\Omega}{\varphi}$ in (3.4). Despite their slower convergence rates, it is important to note that both SMP and MCCFR algorithms have much cheaper iterations compared to the deterministic algorithms.

On a separate note, Gilpin et al. [2012] give an equilibrium-finding algorithm presented as $O(\ln(\frac{1}{\epsilon}))$; but this form of their bound has a dependence on a certain condition number of the A matrix. Specifically, their iteration bound for sequential games is $O(\frac{\|A\|_{2,2}\cdot\ln(\|A\|_{2,2}/\epsilon)\cdot\sqrt{D}}{\delta(A)})$, where $\delta(A)$ is the condition number of A, $\|A\|_{2,2} = \sup_{x\neq 0}\frac{\|Ax\|_2}{\|x\|_2}$ is the Euclidean matrix norm, and $D = \max_{x,\bar{x}\in\mathcal{X},y,\bar{y}\in\mathcal{Y}} \|(x,y)-(\bar{x},\bar{y})\|_2^2$. Unfortunately, the condition number is only shown to be finite for these games. Without any such unknown quantities based on condition numbers, Gilpin et al. [2012] establish an upper bound of $O(\frac{\|A\|_{2,2}\cdot D}{\epsilon})$ convergence rate. This algorithm, despite having the same dependence on ϵ as ours in its convergence rate, i.e., $O(\frac{1}{\epsilon})$, suffers from worse constants. In particular, there exist matrices such that $\|A\|_{2,2} = \sqrt{\|A\|_{1,\infty}\|A\|_{\infty,1}}$, where $\|A\|_{1,\infty}$ and $\|A\|_{\infty,1}$ correspond to matrix norms given by the maximum absolute column and row sums, respectively. Then together with the value of D, this leads to a cubic dependence on the dimension of

Q. For games where the players have roughly equal-size strategy spaces, this is equivalent to a constant of $O(M_Q^4)$ as opposed to our constant of $O(M_Q^2)$.

3.2.9 Numerical experiments

We carry out numerical experiments to investigate the practical performance of several iterative methods used in conjunction with EFGs and demonstrate the practical impact of our theoretical results. Specifically, we consider first-order methods EGT, MP, SMP together with regret-based algorithms CFR and MCCFR. For EGT, MP, and SMP we employ the dilated entropy DGF with weights chosen according to Theorem 2. To enhance the practical performance of EGT, we employ the balancing heuristic described by Hoda et al. [2010]. We run CFR with pruning, which was shown to improve its performance Lanctot et al. [2009]. For MCCFR and SMP we tested 4 different EFG gradient estimators: $\mathcal{C}_{c,1}, \mathcal{C}_{c,5}, \mathcal{C}_{c,10}, \mathcal{C}_{c,20}$ (c.f. (3.7)). In our experiments we report only the results of the best estimater for each: MCCFR was best with $\mathcal{C}_{c,1}$ while SMP was best with $\mathcal{C}_{c,20}$. For SMP, rather than employing the very small fixed stepsize given in (3.4), we use a decreasing stepsize that approaches this value from above.

We test these algorithms on a scaled up variant of the poker game Leduc holdem [Southey et al., 2005], a benchmark problem in the imperfect-information game-solving community. In our version, the deck consists of k pairs of cards $1 \dots k$, for a total deck size of $2k$. Each player initially pays one chip to the pot, and is dealt a single private card. After a round of betting, a community card is dealt face up. After a subsequent round of betting, if neither player has folded, both players reveal their private cards. If either player pairs their card with the community card they win the pot. Otherwise, the player with the highest private card wins. In the event both players have the same private card, they draw and split the pot.

The results are shown in Figure 3.2. Each graph is a loglog plot that shows the results for a particular instance of Leduc with $6, 10, 16$ and 30 card decks, respectively. For each graph, we show the performance of all five algorithms, with the x-axis showing the number of nodes in the game tree touched and the y-axis showing the maximum regret over the two players. In these plots, we use the number of nodes touched in order to account for the pruning in the CFR version that we run together with the effect of other constants such as the fact that the algorithms perform different numbers of tree traversals per iteration (for example, MP and SMP require two prox function applications). In line with what the theory predicts, FOMs with convergence rate $O(\frac{1}{\epsilon})$ such as EGT and MP convergence faster despite the fact that they initially start out slower. These results suggest that they might be better suited for medium-to-high-precision equilibrium computation and smaller size instances. SMP does not perform as well on smaller instances; but its performance improves as the instances get larger. For example, while SMP is initially preferable to EGT and MP for the 30-card game, its performance deteriorates as the accuracy needed increases. We attribute this behavior of SMP to the small stepsizes required by SMP.

In our preliminary experiments, we observed that both MP and SMP are very sensitive to parameter selection. Nevertheless, we did not perform much engineering of the parameters of these algorithms in our experiments. This suggests the possibility of improving their practical performance with a more thorough experimental investigation which we leave as future work.

We also investigate the practical impact of applying the weights used in Theorem 3 (henceforth referred to as new weights), rather than the weights proposed in the preliminary conference version of this paper Kroer et al. [2015] (henceforth referred to as old weights). Figure 3.3 shows the result of running EGT and MP with the old and the new weights. As the stepsize parameter for MP (γ) and smoothing parameter for EGT (μ) required by the theory were found to be too conservative, in the left figure, we show the results after tuning these parameters for the corresponding algorithms to yield the best results for each weight vectors. The theory suggests the same stepsize and smoothing values for both old and new weights.

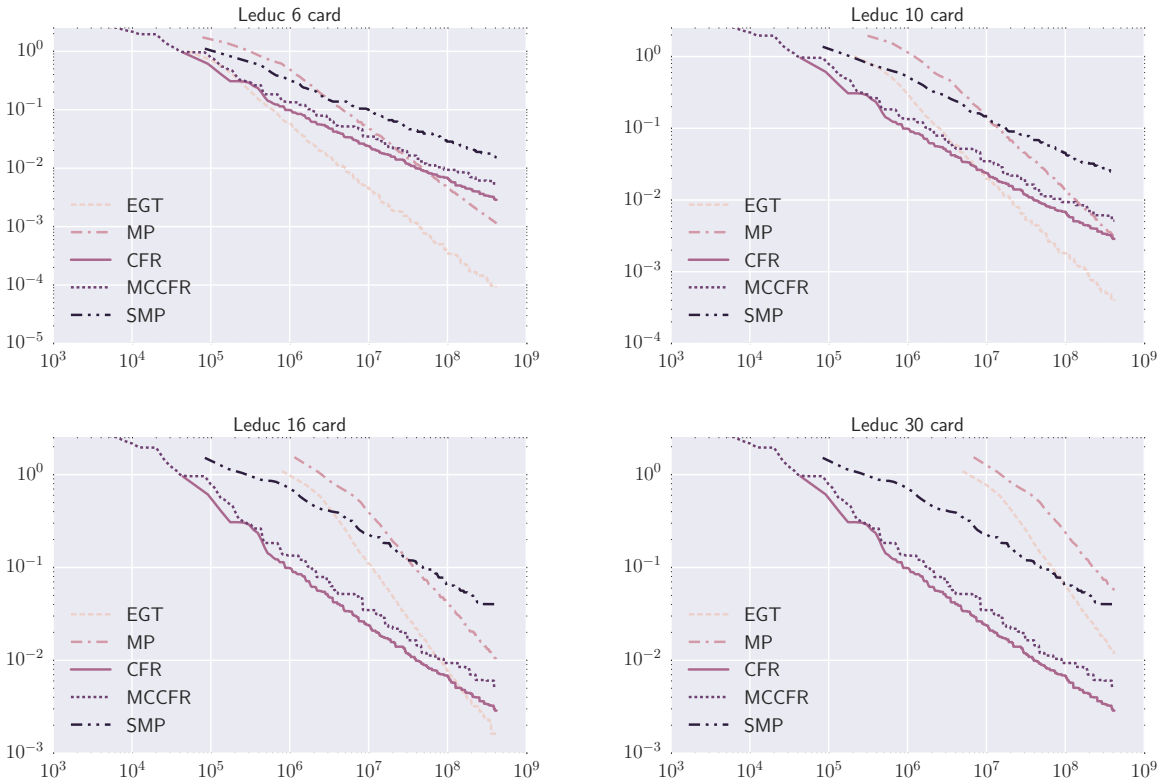


Figure 3.2: Regret as a function of the number of nodes touched in four different variants of Leduc holdem for the CFR, MCCFR, EGT, MP, and SMP algorithms.

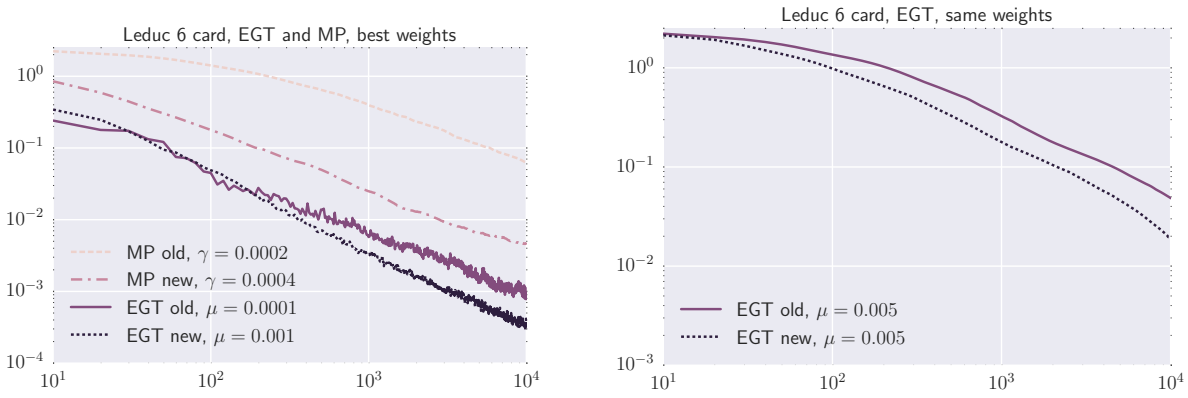


Figure 3.3: Regret as a function of the number of iterations for MP and EGT. Both axes are on a log scale. Throughout *old* corresponds to the weights from Kroer et al. [2015] and *new* corresponds to the weights from Theorem 3. The left plot shows MP with the old and new weights (γ denotes the stepsize) and EGT with the old and new weights (μ denotes the initial smoothing parameter). The right plot shows EGT with old and new weights using a fixed smoothing parameter μ.

Regardless of this, we noticed that the old weights are more sensitive to tuning and have smaller stepsize and smoothing values. The right figure shows results for EGT when applying a single smoothing parameter to both the old and new weights. We observe that the positive impact of our new weighting scheme is more pronounced for MP. MP performs significantly better with the new stepsizes, with about an order of magnitude improvement. EGT also performs better with the new weights, both when using the same μ (on the right), and when using individually tuned μ (on the left).

3.2.10 Conclusions

We have investigated FOMs for computing Nash equilibria in two-player zero-sum perfect-recall EFGs. On the theoretical side, we analyzed the strong convexity properties of the dilated entropy DGF over treeplexes. By introducing specific weights that are tied to the structure of the treeplex, we improved prior results on treeplex diameter from $O(|S_Q|^2 d_{\mathcal{Z}}^2)$ to $O(M_Q^2 2^{d_{\mathcal{Z}}})$, thereby removing all dependence on branching from the Cartesian product operator in the treeplex definition. Our results generalize to any treeplex, whereas the prior results were for only uniform treeplexes, a significant restriction. These results lead to significant improvements in the convergence rates of many FOMs that can be equipped with dilated entropy DGFs and used for EFG solving including but not limited to EGT, MP, and SMP. Finally, we designed a class of sampling schemes for gradient estimation, generalizing prior schemes considered for MCCFR.

We numerically investigated the performance of EGT, MP, SMP, CFR, and MCCFR. For moderately-sized games, our experiments indicate that FOMs such as MP or EGT with the dilated entropy DGF have better practical, as well as theoretical, convergence rate than the current common practice of using CFR algorithms. However, there is a cross-over point at which FOMs beat CFR. Our experiments demonstrated that the cross-over point required more iterations to reach as the games got larger. This begs the question of to what extent the improvements resulting from our approach can make FOMs competitive with CFR for very large games where exact solutions are not computed. Because both MP and SMP are much more customizable than the other FOMs and here we only considered their basic variants, a compelling research direction is to investigate various strategies for step sizes and parameter choices for MP and SMP and see their impact on the practical performance of these algorithms and how they compare against EGT.

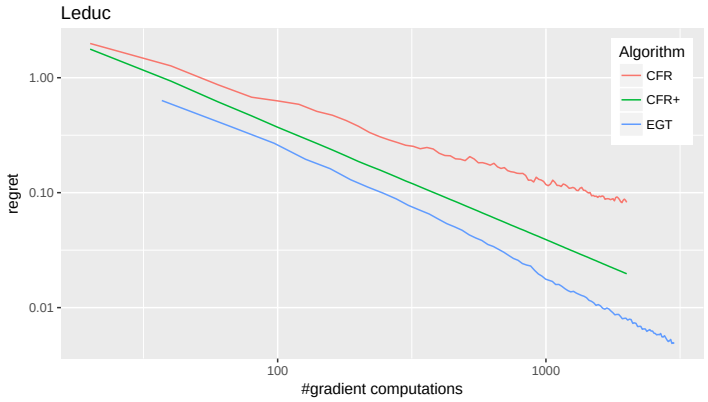


Figure 3.4: Regret as a function of the number of gradient computations in Leduc holdem for the CFR, CFR+, and EGT algorithms.

On a separate note, in practice CFR is often paired with an abstraction technique [Sandholm, 2010] such as those mentioned in Section 3.1. This is despite the lack of any theoretical justification. Effective ways to pair FOMs such as MP and EGT with practical abstraction techniques [Brown et al., 2015] or abstraction techniques that achieve solution-quality guarantees [Lanctot et al., 2012, Kroer and Sandholm, 2014, 2016a] are also worth further consideration.

3.3 Proposed work

While our current results are state-of-the-art for theoretical convergence rates for sparse iterative methods, they are not the practical state-of-the-art. In practice, CFR algorithms are still preferred. We propose extending our theoretical work by experimentally findings ways to scale up FOMs to be competitive with CFR algorithms. We propose leveraging more aggressive stepsizing and/or better initialization in the EGT algorithm. Some preliminary experiments suggest that these two ideas can have an orders-of-magnitude impact on the convergence rate. While these ideas seem heuristic, they can be made theoretically sound through intermittent checking of the excessive gap condition, an inequality associated with EGT, such that we are guaranteed convergence. Figure 3.4 shows some encouraging but very preliminary experiments suggesting that EGT can indeed be made to dominate CFR, rather than only being better after many iterations. This performance was achieved by trying several potential scaling parameters applied to the simplex weights used in Theorem 1. This plot is presented in this proposed work section because we do not yet have a reliable method for generating this kind of performance beyond ad-hoc parameter tuning.

The success of this approach will be measured by showing that FOMs can outperform the leading CFR-like algorithm on real-world games. Specifically, we will consider one or both of the following games:

- Large texas holdem poker endgames. The best current pokerbots run an endgame solver as part of their overall approach. An endgame is, at a high level, an extensive-form game constructed ad-hoc to represent the remainder of the game. Even endgames are very large games, on the order of 100 million nodes in the game tree. These endgames are solved on the spot while the opponent waits for a response, and as such must be solved within 10-20 seconds. Endgames are currently solved with CFR+, and represent the smallest class of games where FOMs can have practical impact.
- No limit Texas hold'em (NLTHE) game abstractions. NLTHE is such a large game that current approaches still use abstraction on the original game. One advantage of this is that we have many otions

for the exact kind of abstraction we want to solve, By increasing the scalability of equilibrium algorithms, we can enable the solution of larger game abstractions. NLTHE is the current frontier of large-scale EFG solving, where many of the state-of-the-art contemporary bots are more exploitable than always folding [Lisy and Bowling, 2017].

Theoretically, we propose extending our results by developing corresponding lower bounds on the convergence rates achievable with the class of entropy-based smoothing functions we consider. Furthermore, we propose showing corresponding upper and/or lower bounds on the convergence rate achievable with dilation-based l_2 smoothing. Ideally, this will allow us to show that entropy-based smoothing is theoretically superior to l_2 smoothing for EFGs. This would complement experimental results from Hoda et al. [2010], which found entropy smoothing to be superior, and parallel the simplex case, where the entropy DGF is known to be superior.

Finally, we propose the development of sparse iterative methods for computing Nash equilibrium refinements. In the case of zero-sum games, polynomial time algorithms are known for refinements such as normal-form proper and quasi-perfect equilibria [Miltersen and Sørensen, 2008, 2010, Farina and Gatti, 2017]. However, these algorithms rely on solving one or more linear programs which are modifications to the sequence-form LP of von Stengel [1996]. This leads to LPs whose size is linear in the size of the game tree. As with regular Nash equilibrium computation, such LP formulations are not practical for large-scale games. Instead, we propose leveraging a perturbed sequence-form polytope (such as that of Miltersen and Sørensen [2010]) in the standard bilinear saddle-point formulation, and then focus on solving this perturbed saddle-point problem directly. For a small enough perturbation, solving this perturbed problem should lead to an approximate equilibrium refinement. We propose combining our results from this chapter with a linearly perturbed dilated entropy DGF, and then applying solvers such as EGT or MP to the computation of approximate quasi-perfect equilibria or extensive-form perfect equilibria.

Chapter 4

Abstraction for large games

Initially, game abstractions were created by hand, using domain dependent knowledge [Shi and Littman, 2002, Billings et al., 2003]. More recently, automated abstraction has taken over [Gilpin and Sandholm, 2006, 2007c, Zinkevich et al., 2007]. This has typically been used for information abstraction, whereas action abstraction is still largely done by hand [Gilpin et al., 2008, Schnizlein et al., 2009]. Recently, automated action abstraction approaches have also started to emerge [Hawkin et al., 2011, 2012, Sandholm and Singh, 2012, Brown and Sandholm, 2014, 2015].

Ideally, abstraction would be performed in a lossless way, such that implementing an equilibrium from the abstract game results in an equilibrium in the full game. Abstraction techniques for this were introduced by Gilpin and Sandholm [2007c] for a subclass of games called *game of ordered signals*. Unfortunately, lossless abstraction often leads to games that are still too large to solve. Thus, we must turn to lossy abstraction. However, significant abstraction *pathologies* (*nonmonotonicities*) have been shown in games that cannot exist in single-agent settings: if an abstraction is refined, the equilibrium strategy from that new abstraction can actually be worse in the original game than the equilibrium strategy from a coarser abstraction Waugh et al. [2009a]! Until recently, all lossy abstraction techniques for general games of imperfect information were without any solution quality bounds. Basilico and Gatti [2011] give bounds for the special game class *Patrolling Security Games*. Johanson et al. [2013] provide computational methods for evaluating the quality of a given abstraction via computing a best response in the full game after the fact. Lanctot et al. [2012] present regret bounds for strategies with low immediate regret computed in imperfect recall abstractions, with their result also extending to perfect-recall abstractions. Their result depends on the counterfactual regret minimization (CFR) algorithm being used for equilibrium computation in the abstraction, and focuses on abstraction via information coarsening, thus allowing neither action abstraction nor reduction of the size of the game tree in terms of nodes. Waugh et al. [2015] and Brown and Sandholm [2015] develop iterative abstraction-refinement schemes that converge to a Nash equilibrium in the limit, but do not give bounds when an abstraction of the game is solved. Sandholm and Singh [2012] provide lossy abstraction algorithms with bounds for stochastic games. They leave as an open problem whether similar bounds can be achieved for extensive-form games. A key complication keeping the analysis in their paper from directly extending to extensive-form games is information sets. In stochastic games, once the opponent strategies are fixed, the best response analysis can be approached on a node-by-node basis. With information sets this becomes much more complex, as strategies have to be evaluated not only according to how they perform at a given node, but also how they perform according to the distribution of nodes in the information set.

This chapter develops the first algorithm-agnostic theoretical bounds on solution quality for (possibly approximate) Nash equilibria computed in abstractions of EFGs rather than the full game. In addition, the

bounds we develop (both for perfect and imperfect-recall abstractions) are exponentially stronger than the only prior (algorithm specific) bounds for EFGs [Lanctot et al., 2012].

4.1 Perfect-recall abstraction

4.1.1 Introduction

This section develops the first bounds on lossy abstraction in general perfect-recall extensive-form games. These results are independent of the equilibrium computation approach, and allow both information coarsening and removal of nodes and actions to make the game tree smaller.

To achieve these more general results, we introduce several new analysis tools:

1. Most importantly, much of our analysis operates on a per information set basis, which fundamentally changes how the analysis can be performed.
2. To enable high expressiveness in the types of abstractions that can be computed, our mappings between real and abstract games are not all surjective functions.
3. We introduce a stricter notion of strategy mapping, called *undivided lifted strategies*.
4. We introduce a new solution concept, *self-trembling equilibrium*, the concept of which is used in our analysis to achieve bounds for general Nash equilibria (defined in Section 2.3).
5. We introduce several new notions of error in abstractions that are used to characterize our bounds.

As a special case, our results generalize the results of Sandholm and Singh [2012], since any DAG-structured stochastic game can be converted to an extensive form game. We also tighten the bound given in their paper. Since our results apply to general Nash equilibria, as well as subgame-perfect equilibria, we also generalize the set of equilibrium concepts that the analysis applies to; theirs was for subgame-perfect equilibrium.

Our framework is the first general theoretical framework for reasoning about quality of equilibria computed in abstract games. The only prior work [Gilpin and Sandholm, 2007c] is for a very restricted setting that closely resembles poker games, and only concerns lossless abstraction. In contrast, our results apply to all extensive-form games of perfect recall, and can be used to reason about both lossless and lossy abstraction. Further, we show that, even in the restricted setting considered in their paper, our framework can find lossless abstractions that theirs cannot.

We also present several new algorithmic characterizations and algorithms for building abstractions that minimize our bounds. We show several hardness results related to computing abstractions on a level-by-level basis, and show how only considering level-by-level abstraction can lead to not finding the smallest, or any, lossless abstractions. Motivated by these results, we develop the first abstraction algorithm with bounded loss, which operates on all levels of the tree at once. We then present experimental results, where we apply our algorithm to a relatively simple poker game. Finally, we discuss additional results, connections to other abstraction approaches, and directions for future research.

4.1.2 Framework

We will consider two extensive form games, the original game Γ and the abstract game Γ'. Both games have the same set of players N. The same elements are defined for Γ', except each one is denoted by a prime superscript. For example, the node space for Γ' is denoted by S'. The value for player i at leaf node $z' \in Z'$ in Γ' is denoted $u_i(z')$. We define the largest utility at any leaf node in the abstract game to be $\overline{W} = \max_i \max_{z' \in Z'} u_i(z')$, which will be useful later.

We will use $W_i^{\sigma'} : \mathcal{S}'_i \to \mathbb{R}$ and $W_i^{\sigma'} : \mathcal{I}'_i \to \mathbb{R}$ to denote the value functions for the abstract game Γ' (completely analogous to the value functions for the original game defined in Section 2.2). Thus, by Proposition 1 we can write the value function for the abstract game as

$$W_i^{\sigma'}(I') = \sum_{a' \in A_{I'}} \sigma'(I', a') \sum_{\hat{I}' \in \mathcal{D}_{I'}^{a'}} \frac{\pi_{-i}^{\sigma'}(\hat{I}')}{\pi_{-i}^{\sigma'}(I')} W_i^{\sigma'}(\hat{I}').$$

Abstraction functions

We define an information set abstraction function $f : \mathcal{I} \to \mathcal{I}'$ which maps from original information sets to abstract information sets. Similarly, $h : \mathcal{S} \to \mathcal{S}'$ is a node abstraction function that maps from original nodes to abstract nodes. Finally, $g : A \to A'$ is an action abstraction function that maps from original actions to abstract actions. All three functions are surjections, so that the game Γ' is no bigger than Γ, and ideally significantly smaller. We define $f^{-1} : \mathcal{I}' \to \mathcal{I}$ to be the set-valued inverse of the information set abstraction function. For a given I', $f^{-1}(I')$ returns all information sets that map to I'. We define h^{-1} and g^{-1} similarly. We also define the function $h_I^{-1} : \mathcal{S}' \to \mathcal{S}$ which for given $I, s' \in f(I)$ returns $h^{-1}(s') \cap I$, the subset of nodes in I that map to s'. Similarly, we define, $h_{\mathcal{P}_s}^{-1} : \mathcal{S}' \to \mathcal{S}$ which for a given s, s' returns $h^{-1}(s') \cap \mathcal{P}_s$, where $\mathcal{P}_s$ is the set of all ancestors and descendants of s.

We require that for any two actions that map to the same abstract action, the two nodes respectively reached by performing the actions at a given node map to the same abstract node. Formally, for all $s \in S$, if $g(a_1) = g(a_2)$ for $a_1, a_2 \in A_s$, then $h(t_{a_1}^s) = h(t_{a_2}^s)$, where t_a^s is the node reached by performing action a at state s, as defined above.

As with prior abstraction approaches, these abstraction functions allow us to map a larger game onto a smaller game, through many-to-one mapping of nodes. We now present a novel approach for mapping nodes one-to-many. This leads to a larger set of valid abstractions, thereby sometimes allowing either smaller abstractions for a given bound, or the same size abstraction with a smaller bound. We call this "multi-mapping". An example where it is useful is given in Figure 4.1. In this game, nature starts out choosing among four actions. Player one then has two information sets, one for when either of the two leftmost nature actions were chosen, and one for when either of the two rightmost nature actions were chosen. If player 1 goes left at any node, player 2 then has two information sets. Player 2 can also only distinguish between nature's actions in sets of two, but different sets than player 1. In this game, we can losslessly treat player 2's information sets $\{2A, 2C\}$ and $\{2B, 2D\}$ as one big information set, since there is a mapping of actions at $2A$ to actions at $2B$ such that the utilities for Player 2 are the same, and similarly for $2C$ and $2D$. However, we need to formalize the notion that $2A$ is similar to $2B$ without merging the nodes, as they are not similar for Player 1. In mapping the smaller information set $\{2A, 2C\}$ onto the abstract information set $\{2A, 2C, 2B, 2D\}$, we wish to specify that $2A$ is represented by the two nodes $2A$ and $2B$. [1]

We will now incorporate "multi-mapping" into the abstraction framework. To our knowledge, we are the first to allow for such abstraction. We define a mapping function $\phi : \mathcal{S}_\phi \to 2^{S'}$ which maps from a subset $\mathcal{S}_\phi \subset S$ of the original nodes to the powerset $2^{S'}$ of the abstract nodes. For all information sets I, we define the sets $I_\phi, I_{\neg\phi}$, the partition of I into two sets such that $s \in \mathcal{S}_\phi$ for $s \in I_\phi$ and $s \notin \mathcal{S}_\phi$ for $s \in I_{\neg\phi}$.

Since in our abstraction framework we are allowing "multi-mapping" (one original node mapped to multiple abstract nodes), we have to ensure that all those abstract nodes are similar (in terms of their dependence

[1] This does not create a smaller abstraction in terms of the tree size, but it decreases the number of information sets. This is relevant to equilibrium-finding algorithms, such as CFR, whose complexity depends on the number of information sets rather than the number of nodes in the game tree. A similar notion of abstraction was introduced by Lanctot et al. [2012].

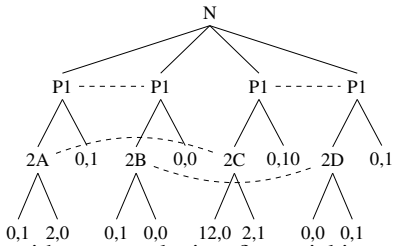


Figure 4.1: Extensive form game, with nature playing first, picking among four outcomes uniformly at random. The next level, with nodes denoted P1, belongs to player 1, and the last level of internal nodes named 2A-2D belong to player 2.

on the other players). Specifically, we require the following constraints for each information set I such that $I_\phi > 0$.

1. If an abstract node $s' \in f(I)$ is part of $\phi(s)$ for some $s \in I_\phi$, then no other nodes in I can map to s'.
2. For all nodes s and pairs of nodes $s'_1, s'_2 \in \phi(s)$ at level k, each level $l \in \mathcal{H}_{-0}, l > k$, the edges on the respective paths to s'_1, s'_2 must be in an information set together, and belong to the same action, or both be in information sets with a single action.
3. For all abstract nodes $s' \in \phi(s)$, we require that the subtree rooted at s also maps onto the subtree rooted at s'. For all such pairs of nodes, we define a surjective function $\phi_{s,s'}$, completely analogous to the function h, defining this mapping.
4. For all descendants $\hat{s}$ of s and pairs of nodes $s'_1, s'_2 \in \phi(s)$, $\phi_{s,s'_1}(\hat{s})$ must be in the same information set as $\phi_{s,s'_2}(\hat{s})$.

The example presented above can be described in terms of our ϕ function: $\phi(2A) = \{2A, 2B\}, \phi(2B) = \{2A, 2B\}, \phi(2C) = \{2C, 2D\}, \phi(2D) = \{2C, 2D\}$. Since the children of these nodes are all leaf nodes, we need not define the subtree mapping functions $\phi_{2A,2B}, \phi_{2C,2B}$.

Mapping behavioral strategies from the abstract game back to the original game

When computing a strategy profile σ' in Γ', we need some notion of how to interpret this profile in Γ. Here we introduce two classes of mappings from abstract strategies to real strategies. First, we present the natural extension of lifted strategies introduced by Sandholm and Singh [2012] to extensive-form games. In this definition, a valid mapping $\sigma^{\uparrow\sigma'}$ is any strategy profile such that for each information set I and abstract action $a' \in A_{f(I)}$, the probability $\sigma'(f(I), a')$ of selecting that abstract action is divided any way across the actions $g^{-1}(a') \cap I$.

Definition 8. *(Strategy lifting) Given an abstract game Γ' and a strategy profile σ', a lifted strategy for Γ is a strategy profile $\sigma^{\uparrow\sigma'}$ such that for all I, all $a' \in A_{f(I)}$: $\sum_{a \in g^{-1}(a')} \sigma^{\uparrow\sigma'}(I, a) = \sigma'(f(I), a')$.*

For our game-theoretic results, we require a stronger notion of similarity in the mapping. We introduce *undivided lifted strategies*, a refinement of lifted strategies. In addition to the lifted strategy constraint, we require that the probability distribution over nodes in an information set $I \in \Gamma$ such that $\sigma_{-0}(I) > 0$, when disregarding nature probabilities, conditioned on reaching the information set, is the same as for the abstract information set, when summing over node mappings.

Definition 9. *(Undivided strategy lifting)) Given an abstract game Γ' and a strategy profile σ', an undivided lifted strategy for Γ is a lifted strategy profile σ such that for all $I \in \Gamma$ we have*

1. *For $s \in I_\phi$: $\frac{\pi^\sigma_{-0}(s)}{\pi^\sigma_{-0}(I)} = \sum_{s' \in \phi(s)} \frac{\pi^\sigma_{-0}(s')}{\pi^{\sigma'}_{-0}(f(I))}$, and*

2. *For* $s' \in f(I_{\neg\phi})$: $\frac{\pi^{\sigma'}_{-0}(s')}{\pi^{\sigma'}_{-0}(f(I))} = \sum_{s \in h_I^{-1}(s')} \frac{\pi^{\sigma}_{-0}(s)}{\pi^{\sigma}_{-0}(I)}$

One natural family of action abstraction techniques is to select actions for the abstract game from the actions in the original game (as opposed to creating new actions), or, in other words, to simply remove actions from the original game to generate the abstract game. That is, the abstraction is created by restricting play only to a subset of the actions, and potentially also restricting nature to only choosing a subset of its outcomes. Then, the strategy mapping just consists of playing the strategy computed in the restricted game and ignoring the remaining actions. All prior action abstraction algorithms have operated within that family (e.g., Gilpin et al. [2008], Schnizlein et al. [2009], Hawkin et al. [2011, 2012], Sandholm and Singh [2012], Ganzfried and Sandholm [2013], Brown and Sandholm [2014]). Such lifting (as long as the information abstraction is done within the framework of our paper, as described above) is a form of undivided strategy lifting.

4.1.3 Reward-approximation and information-approximation error terms

We define approximation error terms, that give us a notion of the distance between the real game and the abstract game. These will later be used to prove bounds on the solution quality of equilibria computed using the abstract game.

First we define *reward approximation error*. Intuitively, we define the reward error at leaf nodes to be the maximum difference between values of the real leaf and the abstract leaves that it maps to, at player nodes to be the maximum error of the child nodes, and at nature nodes to be a weighted sum of the error of the child nodes. Formally, the reward approximation error for the mapping from Γ to Γ' for a given agent i, and node s is

$$\epsilon^R_{s,i} = \begin{cases} \max_{s' \in \Phi(s)} |u_i(s) - u_i(s')| & \text{if } s \text{ is a leaf node} \\ \max_{a \in A_s} \epsilon^R_{t^s_a,i} & \text{if } s \text{ is a player node} \\ \sum_{a \in A_s} \sigma_0(s,a) \epsilon^R_{t^s_a,i} & \text{if } s \text{ is a nature node} \end{cases}$$

Here $\Phi^{-1}(s) = \bigcup_{\hat{s} \in \mathcal{S}_\phi, s' \in \phi(\hat{s})} \phi_{\hat{s},s'}(s) \cup h(s)$ is the set of all abstract nodes that s maps to through either some $\phi_{\hat{s},s'}$ function or h. For a given strategy profile, the error for node s for a non-leaf, non-nature node could be defined as the weighted sum of the errors of the actions at the state. But since we do not know which strategy will be played, we have to take the maximum. The reward error for Player i of an information set I is the maximum error over the nodes in I, and the total reward error is the maximum of this quantity over all players:

$$\epsilon^R_{I,i} = \max_{s \in I} \epsilon^R_{s,i}, \quad \epsilon^R = \max_i \epsilon^R_{r,i}$$

Similarly, we define the *distribution error* for an information set I and nodes $s' \in f(I_{\neg\phi})$ and $s \in I_\phi$ to be:

$$\epsilon^D_{I,s'} = \left| \frac{\sum_{s \in h_I^{-1}(s')} \pi^{\sigma}_{-i}(s)}{\pi^{\sigma}_{-i}(I)} - \frac{\pi^{\sigma'}_{-i}(s')}{\pi^{\sigma'}_{-i}(f(I))} \right|, \quad \epsilon^D_{I,s} = \left| \frac{\sum_{s' \in \phi(s)} \pi^{\sigma'}_{-i}(s')}{\pi^{\sigma'}_{-i}(f(I))} - \frac{\pi^{\sigma}_{-i}(s)}{\pi^{\sigma}_{-i}(I)} \right|$$

The distribution error for information set I and error over all information sets at height k is:

$$\epsilon^D_I = \sum_{s' \in f(I_{\neg\phi})} \epsilon_{I,s'} + \sum_{s \in I_\phi} \epsilon^D_{I,s}, \quad \epsilon^D_k = \max_{I \in \mathcal{I}_k} \epsilon^D_I$$

For all distribution error notions, they depend on the strategy profile played, but we omit this in the subscripts for ease of readability. For all proofs, it will be clear what the strategy profile is. We define the nature distribution error of an information set I and nodes $s' \in f(I_{\neg\phi}), s \in I_\phi$ to respectively be

$$\epsilon^0_{I,s'} = \left| \frac{\sum_{s \in h_I^{-1}(s')} \pi_0^\sigma(s)}{\pi_0^\sigma(I)} - \frac{\pi_0^{\sigma'}(s')}{\pi_0^{\sigma'}(f(I))} \right|, \quad \epsilon^0_{I,s} = \left| \frac{\sum_{s' \in \phi(s)} \pi_0^{\sigma'}(s')}{\pi_0^{\sigma'}(f(I))} - \frac{\pi_0^\sigma(s)}{\pi_0^\sigma(I)} \right|$$

This is the difference between nature's probability of reaching s' or s and nature's probability of reaching any node in $h_I^{-1}(s')$ and in $\phi(s)$ respectively, all normalized by the probability of reaching the given information sets. The nature error for information set I is

$$\epsilon^0_I = \sum_{s' \in f(I_{\neg\phi})} \epsilon^0_{I,s'} + \sum_{s \in I_\phi} \epsilon^0_s$$

For a nature node s at height $k \in \mathcal{H}_0$ and $s' \in \Phi(s)$, we define the nature action $a' \in A_{s'}$ error and node error to respectively be

$$\epsilon^0_{s,s',a'} = \left| \sigma_0'(s', a') - \sum_{a \in g^{-1}(a') \cap A_s} \sigma_0(s, a) \right|, \qquad \epsilon^0_s = \max_{s' \in \Phi(s)} \sum_{a' \in A_{s'}} \epsilon^0_{s,s',a'}$$

The nature error at height k is

$$\epsilon^0_k = \begin{cases} \max_{I \in \mathcal{I}_k} \epsilon^0_I & \text{if } k \notin \mathcal{H}_0 \\ \max_{s \in S_k} \epsilon^0_s & \text{if } k \in \mathcal{H}_0 \end{cases}$$

In contrast to distribution error, nature distribution error does not depend on the strategy profile. For any undivided lifted strategy, the distribution error is bounded by the nature distribution error.

Proposition 2. *For any undivided lifted strategy σ and information set I, $\epsilon^D_I \leq \epsilon^0_I$.*

We show that for a given node s and abstract node s' that is a descendant of $h(s)$, the probability of reaching s' from $h(s)$ is at least as high as the probability of reaching any node in $h^{-1}(s')$ from s, disregarding error in nature:

Proposition 3. *For any node s, abstract node s' that's a descendant of $h(s)$, and strategy σ lifted from σ',*

$$\sum_{\hat{s} \in h^{-1}(s')} \pi^\sigma(s, \hat{s}) \leq \pi^{\sigma'}(h(s), s') + \sum_{l \in \mathcal{H}_0, k \geq l > k'} \epsilon^0_l$$

The same result holds for the set of nodes $\phi^{-1}_{\hat{s},\hat{s}'}(s')$ for all $\hat{s}, \hat{s}'$ such that $\phi_{\hat{s},\hat{s}'}$ is defined.

4.1.4 Lifted strategies from abstract equilibria have bounded regret

We are now ready to move to the main part of the paper, which is a mathematical framework for bounding the loss from abstraction in extensive-form games.

We first show that the utility obtained from implementing any lifted strategy in Γ is close to the utility of the original strategy in Γ'. We will use this to prove our main result.

Proposition 4. *For any player i, abstract strategy σ', and lifted strategy $\sigma^{\uparrow\sigma'}$:*

$$|V_i^{\sigma^{\uparrow\sigma'}}(r) - W_i^{\sigma'}(f(r))| \leq \epsilon^R_i + \sum_{j \in \mathcal{H}_0} \epsilon^0_j \overline{W}$$

We are now ready to show our main result, which is game theoretic. We show that any Nash equilibrium computed in the abstract game, when implemented as an undivided lifted strategy in the original game, has bounded regret.

Theorem 7. *For any Nash equilibrium σ' in Γ', any undivided lifted strategy σ is a $2\epsilon^R + \sum_{j \in \mathcal{H}_i} \epsilon_j^0 \overline{W} + \sum_{j \in \mathcal{H}_0} 2\epsilon_j^0 \overline{W}$-Nash equilibrium in Γ.*[2]

It is possible to make the nature error bound tighter in the above, by not using the error over the entire level at a given height k, but instead keeping the errors defined in terms of the current subtree. For expositional ease we avoided this, since we later show that this error can often be kept at 0, but point out that it could be relevant in certain applications.

Using exactly the same technique as above, except without using the self-trembling property for subgame-inducing nodes, it is possible to show that any subgame-perfect equilibrium in the abstract game maps to an approximate subgame-perfect equilibrium in the full game.

Example where the bound is tight

We now investigate the tightness of our bounds. First, we show that the bound over payoff error is tight.

Proposition 5. *There exist games such that requiring any amount of abstraction leads to every Nash equilibrium of the abstraction having regret $2\epsilon^R$ for some player when implemented in the original game with any undivided strategy.*

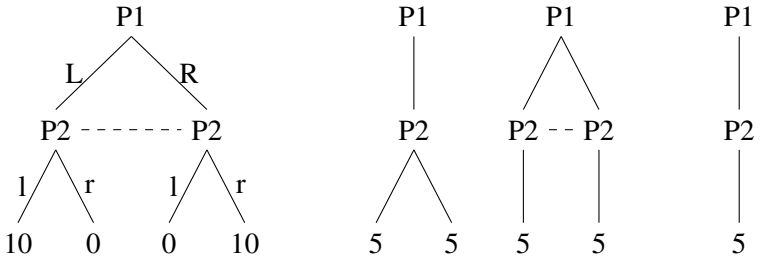


Figure 4.2: On the left is a zero-sum extensive form game with one decision node for Player 1 and two decision nodes (in the same information set) for Player 2. In the middle are the two possible results from performing a single action abstraction, either merging actions L and R, or l and r. On the right is the final abstraction which both intermediate abstractions reduce to.

For the nature error terms, we show that the bound is tight up to a constant factor of 6.

Proposition 6. *There exist games such that requiring any amount of abstraction leads to some Nash equilibrium of the abstraction having regret $\frac{1}{6}(\sum_{j \in \mathcal{H}_i} \epsilon_j^0 \overline{W} + \sum_{j \in \mathcal{H}_0} 2\epsilon_j^0 \overline{W})$ for some player when implemented in the original game with any lifted strategy.*

In both the above tightness examples, we relied on either mapping only undivided lifted strategies (in the first case), or showing the bound is tight for only some equilibria (in the latter case). However, computing the best lifted strategy can, in general, amount to computing an equilibrium in the full game, as it does in the first case. Likewise, avoiding certain equilibria in the abstract game would require somehow adding more information to the abstraction computation, thus linking it more intimately to the original game. This was

[2]With the same proof, except without using the self-trembling property for subgame-inducing nodes, we can show that any subgame-perfect equilibrium in the abstract game maps to an approximate subgame-perfect equilibrium in the original game. This requires that subgame-inducding nodes in the real game map to subgame-inducing nodes in the abstract game. This is easily achieved by disallowing multi-mapping on subgame-inducing nodes.

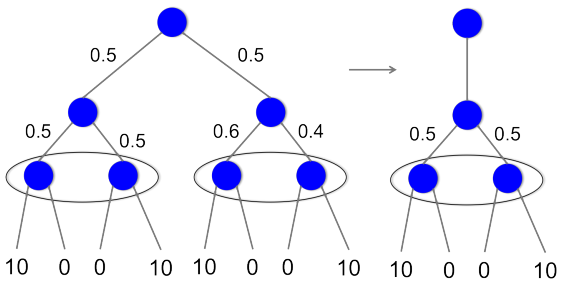


Figure 4.3: On the left is a simplified extensive form game with two layers of nature nodes, followed by decision nodes for Player 1 in two different information sets. On the right is given the only possible abstraction that does not lead to a loss of 10.

considered by Johanson et al. [2013], but their evaluation requires computing best responses in the original game.

4.1.5 Abstraction algorithms and complexity

In this section we develop several algorithms for computing abstractions in extensive form games, such that Nash equilibria computed in these abstractions satisfy the bound derived in Theorem 7 when mapped to an undivided lifted strategy in the original game.

Level-by-level abstraction

Prior game abstraction algorithms typically proceed level by level in the game tree—possibly moving both up and down—e.g. Gilpin and Sandholm [2006, 2007a,c], Gilpin et al. [2007, 2008], Zinkevich et al. [2007], Sandholm and Singh [2012], Johanson et al. [2013], Ganzfried and Sandholm [2014]. We give a general framework for a large class of level-by-level abstraction algorithms, and investigate the required subroutines. We then present an impossibility result for level-by-level abstraction.

First we give a general framework for computing abstractions level by level. The algorithm operates on a game Γ, and a set of nodes S at some level k, where S would usually be an information set. It proceeds to build the set A_{Iso}, a set of action pairs that are eligible for merging, along with the cost c of merging them. This is done by iterating over all action pairs a_1, a_2 available at the nodes, and calling the subroutine APPROXISOMORPHIC, which computes whether the actions can be merged and at what cost, for each node $s \in S$. Once the set A_{Iso} has been constructed, the algorithm calls a subroutine COMPUTEPROTOTYPES, which selects the subset of actions that are kept (we call these *prototypes*), with the remaining branches mapped onto the prototypes.[3]

The two subroutines APPROXISOMORPHIC and COMPUTEPROTOTYPES perform the brunt of the work in this algorithm. We will now, in turn, examine what the subroutines entail.

[3]The theory from earlier in this paper also applies to the setting where the abstract game has new actions, if the new action is constructed from an action in the original game (and that subtree under that action) by simply changing payoffs in that subtree. However, as in all prior abstraction algorithms, in this algorithms section we focus on abstraction that selects action prototypes from the actions in the original game.

ALGORITHM 2: A framework for level-by-level abstraction with bounded loss in extensive-form games.

Input: A game Γ, a set of nodes S at some level k such that they have the same action set available
Output: An abstract game Γ'

```
A_Iso ← ∅
for each action pair a_1, a_2 ∈ A_S do
    for each node s ∈ S do Test ApproxIsomorphic(t^s_{a_1}, t^s_{a_2}) ;
    Let c be the cost of merging a_1, a_2 over all s ∈ S
    if isomorphic for all s ∈ I then add ((a_1, a_2), c) to A_Iso ;
    Call ComputePrototypes(A_Iso) to compute a set of prototypes that the remaining actions map onto.
```

Extensive-form game tree isomorphism

In order to determine whether two given actions can be merged at some information set (APPROXISOMORPHIC function above), it must be determined what the cost of merging the subtrees are at each node in the information set. A special, and simple, case is when the subtrees are subgames, and there is only a single node in the information set. Consider such two subgames rooted at nodes s_1, s_2. Since we are assuming that abstraction is conducted level by level, the algorithmic problem amounts to finding the cheapest onto mapping of s_1 onto s_2. To reason about this problem, we introduce the *extensive-form game tree isomorphism* problem. For the purposes of this section, we will consider the node mapping function h an onto function, rather than a surjection, since abstraction is not performed as part of this subroutine.

First we introduce a simple notion of isomorphism. It does not capture everything we need for the APPROXISOMORPHIC subroutine, but we will later show that even this simple notion is hard to compute. For two game trees to be isomorphic, this definition requires two parts. First, the trees must be isomorphic in the traditional sense. Second, for any node pairs from the same information set, their mapped nodes have to be in the same information set, and for any node pairs in different information sets, their mapped nodes must be in different information sets.

Definition 10. *(Extensive form game tree topology isomorphism) A topology isomorphism (TI) between extensive form games Γ, Γ' is an onto mapping h such that*

1. *For nodes $s_1, s_2 \in \Gamma$ belonging to the same information set I, $h(s_1)$ and $h(s_2)$ belong to the same information set in Γ'.*
2. *For nodes $s_1 \in I_1$ and $s_2 \in I_2$, where $I_1 \neq I_2$, $h(s_1)$ and $h(s_2)$ belong to different information sets in Γ'.*
3. *For nodes s and s_c, where s_c is a child of s, $h(s_c)$ is a child of $h(s)$.*

For any tree isomorphism on two extensive-form game trees (that is, ignoring information sets), any node at level k in Γ will map to a node $h(k)$ at level k in Γ'. For leaf nodes, they are the only nodes with out-degree one, so leaf nodes must map to leaf nodes. Likewise, the root node r has some distance d to each leaf node, since we assumed uniform depth, so $h(r)$ must map to the root node in Γ'. Now consider some node s at height k, at depth d. $h(s)$ must be at height k in order to be isomorphic, and the shortest path to the root node must be of length d. Thus, our requirement that a TI must respect player's turn taking is achieved already by it being a tree isomorphism.

We now present a refinement of TI, which we call *extensive form game tree strategic isomorphism (SI)*, that captures the chance and utility structure of the game. This refinement is what we need to compute for the APPROXISOMORPHIC subroutine. For subtrees where this isomorphism does not exist, we want to compute the mapping that minimizes the distance to strategic isomorphism.

Definition 11. *(Extensive form game tree strategic isomorphism) An extensive form game tree strategic*

isomorphism (SI) is a TI that satisfies the following further constraints.

1. *For all nodes s at some height $k \in \mathcal{H}_0$, for each $a \in A_s$, $\sigma(s, a) = \sigma(h(s), g(a))$.*
2. *For all leaf nodes $z \in Z$ and players $i \in N$, $V_i(z) = V_i(h(z))$.*

We now show that both TI and SI are graph isomorphism-complete. A graph isomorphism complete problem is a problem such that there exists a polynomial time reduction both to and from the graph isomorphism problem [Booth and Colbourn, 1979].

Theorem 8. *Deciding whether two game trees are extensive-form game tree topologically isomorphic or extensive-form game tree strategically isomorphic is graph isomorphism-complete.*

Game isomorphism has been studied to a limited extent in the literature. The more general question of whether two games are isomorphic across game types has been studied by Gabarró et al. [2007]. Peleg et al. [1999] and Sudhölter et al. [2000] study extensive form game tree isomorphism in the context of strategic equivalence, but do not consider the computational problem. To our knowledge, we are the first paper to introduce the question of computing extensive-form game tree isomorphisms.

Choosing the set of prototypes

After computing the loss incurred by merging the different action pairs available at the set of nodes under consideration, the subroutine COMPUTEPROTOTYPES computes a subset of the action pairs whose subtrees are kept as the abstract game, with the remaining subtrees mapped onto these. We now show that, even when the values of merging all action pairs have been computed, the problem of choosing the optimal subset of actions is NP-complete.

Definition 12. *(Action subset selection problem) The input is a set of actions A, a cost c_{a_1,a_2} for merging each action pair $a_1, a_2 \in A$, some value k that specifies the number of actions to be selected, and a cost c. The action subset selection problem asks if there exists a set of k actions and a mapping of the remaining actions onto this set, such that the cost is less than or equal to c.*

Theorem 9. *The action subset selection problem is NP-complete.*[4]

The problem of computing an extensive form game abstraction that minimizes the bound given in Theorem 7 is easily seen to be NP-complete as well—even with just one player, and thus it holds for zero-sum games also. The action subset selection problem itself reduces to a two-level game tree by making a chance node as the root, with all children being in the same information set of size $|A|$, with $|A|$ actions available, where each node is used to ensure the correct cost of merging for a single action pair, with all other merges being free at the node.

A sequence of single abstraction transformations to the optimal abstraction might not exist

We now show a fundamental issue with level-by-level abstraction. It can leave valid (even lossless) abstractions undiscovered: reasoning across merges at different levels and in different subtrees simultaneously is required in order to stay within the set of valid abstractions throughout the traversal of the game tree. To show this, we show a slightly broader result: determining whether a merge is within the desired error bound can require reasoning about merges in subtrees different from the ones considered for merging.

Theorem 10. *Any abstraction technique that computes the abstraction by merging subtrees must satisfy at least one of the following properties.*

[4] Sandholm and Singh [2012] show that in stochastic games it is NP-complete to compute an abstraction where one of the players does not automatically receive the worst possible payoff. However, some bound-minimizing solutions do not solve the problem that is reduced from. Thus that result does not imply that minimizing the bound is NP-complete.

1. *It does not always find the smallest abstraction in the space of valid abstractions for a given bound.*
2. *When merging two nodes at a level, it must ensure that future merges at other levels in the tree, outside of the two subtrees below those two nodes, will enable returning to an abstraction within the bound.*

This applies even if the game is a game of ordered signals Gilpin and Sandholm [2007c], zero-sum, and only information abstraction is performed.

Proof. Consider a two-player poker game where the card deck consists of two kings and two jacks. Each player is dealt a private card and then a single public card is dealt. We consider two variants based on what the payoffs at the leaves are. In the first variant, Player 1 always wins. The second variant is a slight modification, where if Player 2 pairs a private $J2$ with a public $K2$ he wins, and otherwise Player 1 wins. Both variants are zero-sum games of ordered signals. Figure 4.4 shows the possible sequences of cards dealt. The two variants are obtained by setting δ equal to 1 or -1 in Figure 4.4, respectively. If Player 1 always

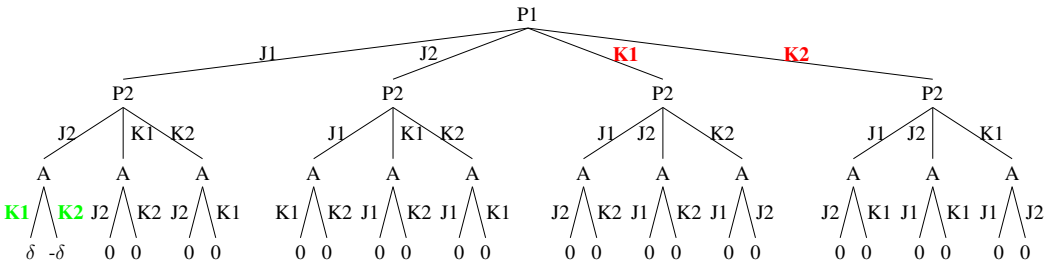


Figure 4.4: A signal tree for a simple poker game. Nodes labeled P1 or P2 denote the card being dealt privately to player 1 or 2. Nodes labeled A denote a public card. A leaf node labeled 1 indicates Player 1 winning.

wins ($\delta = 1$), the whole game can be abstracted into a single string of cards dealt, as none of them matter. If Player 2 wins ($\delta = -1$), the two public kings cannot be abstracted into a single king when Player 1 holds $J1$ and Player 2 holds $J2$ based on the following reasoning. Consider $K1$ and $K2$ dealt to Player 1 at the root. Player 2 has two information sets that consist of him holding a $J2$ and the public card being a king. Each of those two information sets has two nodes corresponding to Player 1 holding the other jack or the other king. The two information sets differ based on whether the public king is $K1$ or $K2$. If $K1$ and $K2$ at the root are abstracted into a single tree, then one of these two information sets loses a node, since the sequence $K1, J2, K2$ would map onto $K2, J2, K1$, or vice versa. This leads to Player 2 being able to deduce exactly which card Player 1 has for the information set that is now a singleton. For the other information set, the probability on the node where Player 1 has a private king would be higher than the probability on the node where Player 1 has a private jack. Thus, if a lossless abstraction is desired, the choice of whether to abstract $K1$ and $K2$ at the root hinges on whether δ is set to 1 or not. □

One consequence is that the GameShrink lossless abstraction algorithm Gilpin and Sandholm [2007c] does not work correctly for all games of ordered signals.

Generating an abstraction by considering all levels at once

Motivated by the problems described in Section 4.1.5, we develop an algorithm for computing abstractions with bounded loss that operates on all levels at once. The only assumption we make about structure of the game, is that we only allow nature nodes to merge if they have the same number of actions, and only allow different branches to merge if they have the same probability of occurring.

Using integer-programming (IP), we develop two similar models for computing abstractions. One takes as input the maximum number of game tree nodes allowed and computes an abstraction that minimizes the bound given in Theorem 7. The other takes as input a desired error bound and computes the smallest abstraction subject to satisfying the bound.

Variables

For each node s_i, we introduce a variable $p_i \in \{0, 1\}$ denoting whether s_i is a prototype. For each level $k \in \mathcal{H}$ and ordered pair of nodes at height k, $s_i, s_j \in S_k$, we introduce a variable $\delta_{i,j} \in \{0, 1\}$ denoting whether s_i is mapped onto s_j. For each unordered pair of information sets I_i, I_j at height k, we introduce a variable $I_{i,j} \in \{0, 1\}$ denoting whether the two information sets map to the same abstract information set.

Objective functions

In the case where we are given a bound to satisfy and wish to compute the smallest abstraction, we maximize the sum over all abstraction variables $\delta_{i,j}$, thereby minimizing the number of nodes in the game tree:

$$\max_{\delta_{i,j}, p_i, I_{i,j}} \sum_{i,j} \delta_{i,j} \tag{4.1}$$

In the case where we are given a maximum tree size and want to minimize the bound, we minimize the sum over leaf nodes mapped onto each other, weighted by the absolute difference in utility at the leaves:

$$\min_{\delta_{i,j}, p_i, I_{i,j}} \sum_{z \in Z} \sum_{\hat{z} \in Z} \max_{i \in N} |u_i(z) - u_i(\hat{z})| \delta_{i,j} \tag{4.2}$$

Constraints

To ensure that nodes are either prototypes or mapped onto a single prototype, we introduce the following three constraints. The first and second constraints below are introduced for all nodes s_i, and the third for all pairs of nodes s_i, s_j at the same height.

$$\sum_{j \in S_k} \delta_{i,j} \leq 1, \forall s_i \in S, \qquad p_i + \sum_{j \in S_k} \delta_{i,j} \geq 1, \forall s_i \in S, \qquad \delta_{j,i} - p_i \leq 0, \forall k \in \mathcal{H}, s_i, s_j \in S_k \tag{4.3}$$

The first constraint ensures that each node is mapped onto at most one other node. The second and third constraints ensure that the variables p_i accurately reflect whether s_i is a prototype. The second constraint requires that $p_i = 1$ unless s_i is mapped to some other node. The third constraint sets $p_i = 0$ if s_i is mapped onto any other node s_j. Together, the last two constraints ensure that nodes are only mapped onto prototypes, and that prototype nodes are not mapped onto anything.

If a node s_i is mapped onto another node s_j, one of the constraints specified in Section 4.1.2 requires that the children at s_i map onto the children at s_j. This is ensured by the following constraint, where C_i is the set of all child node indices of s_i.

$$\delta_{i,j} - \sum_{c_j \in C_j} s_{c_i, c_j} \leq 0, \forall s_i \in S \setminus Z, c_i \in C_i, \tag{4.4}$$

If $\delta_{i,j} = 1$, this constraint requires that s_{c_i} is mapped onto some child node of s_j by requiring that at least one of them is set to one. Similarly, if s_i is mapped onto s_j, we require that the parent node of s_i, denoted

by s_{p_i}, is mapped onto the parent node of s_j, denoted by s_{p_j}. This gives the following constraint, where the parent mapping variable δ_{p_i,p_j} is required to be set to one if $\delta_{i,j} = 1$.

$$\delta_{i,j} - \delta_{p_i,p_j} \leq 0, \forall k \in \mathcal{H}, s_i, s_j \in S_k \tag{4.5}$$

For each node pair s_i, s_j at some height k, let I_i, I_j be their respective information sets and $I_{i,j}$ the variable denoting whether the information sets are merged. If the nodes are merged, we require that their information sets are also merged, which is achieved by the following constraint.

$$\delta_{i,j} - I_{i,j} \leq 0, \forall k \in \mathcal{H}, s_i, s_j \in S_k \tag{4.6}$$

Information set merges are transitive, so if two information sets are both merged with the same third information set, we require that they are themselves merged:

$$I_{i,j} + I_{i,l} - I_{j,l} \leq 1, \forall k \in \mathcal{H}, I_i, I_j, I_l \in \mathcal{I}_k \tag{4.7}$$

Using the variable $I_{i,j}$ for two information sets I_i, I_j, we can ensure that each prototype node in the abstract merged information set has a node from each information set mapping onto it. Without loss of generality, assume $s_l \in I_i$, we add the following constraint.

$$p_l + I_{i,j} - \sum_{s_m \in I_j} \delta_{m,l} \leq 1, \forall I_i, I_j, s_l \in I_i \tag{4.8}$$

This requires that if s_l is a prototype, $p_l = 1$, and if $I_{i,j} = 1$, at least one node from I_j maps onto s_l.

As mentioned above, we only allow nature outcomes to map onto each other if their probability is the same. Further, if for some nature node s_j mapped onto a nature node s_i we have that $m > 1$ child nodes of s_j map onto the same child node of s_i, then we must ensure that $m - 1$ child nodes at s_i map onto c_i, in order to keep the nature distribution error at zero. This is achieved by the following two constraints.

$$\delta_{c_i,c_j} = 0, \forall s_i, s_j, c_i \in C_i, c_j \in C_j, \sigma(s_i, c_i) \neq \sigma(s_j, c_j) \tag{4.9}$$

$$\sum_{c_j \in C_j} \delta_{c_j,c_i} = \sum_{c_k \in C_i} \delta_{c_k,c_i} + 1, \forall s_i, s_j \in S \setminus Z, c_i \in C_i \tag{4.10}$$

The first constraint just disallows mapping children of nature nodes that do not have equal probability of occurring. The second constraint ensures that the probability of a prototype child being chosen at the nature node, which is equal to the sum of the probabilities of outcomes at the node that are mapped to it, is equal to the sum of probabilities over outcomes mapped to it from s_j.

For the case where a specific bound is given as input and we wish to compute the minimum size abstraction, we add the following constraint.

$$\sum_{z_i,z_j \in Z} \max_{i \in N} |u_i(z_i) - u_i(\hat{z_j})| \delta_{i,j} \leq \epsilon^R \tag{4.11}$$

This constraint sums over all leaf node pair mappings, and weights them by the difference in utility at the pair. We require that this sum be less than ϵ^R, the given bound. For the case where a maximum tree size K is given as input and we wish to minimize the bound, we add the following constraint.

$$\max_{\delta_{i,j},p_i,I_{i,j}} \sum_{i,j} \delta_{i,j} \geq |S| - K \tag{4.12}$$

This constraint requires that at least $|S| - K$ merges are performed, so the abstraction has at most K nodes.

The number of variables in the model is $O(|Z|^2)$. The largest constraint sets are those in Equation 4.7 and those over all variable pairs at each level. The former is $O(\max_{k \in \mathcal{H}} \mathcal{I}_k^3)$ and the latter is $O(|Z|^2)$.

4.1.6 Experiment

We applied our IP model to a simple poker game. Our game has a deck of five cards: two jacks, a queen, and two kings. Two players play the game, and after each round of cards is dealt, up to two rounds of betting occur. A full description of the game is given in the appendices.

One advantage of poker games for testing our approach is that the chance outcomes can be decoupled from the player actions using the *signal tree*. The signal tree is defined conceptually by removing all player actions from the game tree, and only considering the tree of possible nature moves (aka signals). Clearly, for this decoupling to be possible, the nature moves can be conditioned only on which prior nature events have occurred, not on player actions. Any abstraction computed on the signal tree can easily be converted to an abstraction of the full game. Gilpin and Sandholm [2007c] introduced the signal tree in the specific setting of games of ordered signals, a game class closely resembling poker. More generally, in any game where the player's action options are independent of nature's moves, abstraction can be performed on the signal tree.

In our poker game, the unabstracted signal tree has 86 nodes; the game tree has 4806 nodes. The IP has 4,427 binary variables (one for each pair of nodes at each level of the signal tree, plus the additional bookkeeping ones) and 38,813 constraints. To solve the IP, we used CPLEX version 12.5.

For the model where a bound is given as input and the objective is to minimize tree size, we ran experiments with a bound ranging from 0 to 20 chips. Figure 4.5 Left shows a plot of the game sizes as a function of the bound. As can be seen, tree size is a step function of the given bound. Four different abstraction sizes were found. The coarsest abstraction has four signal tree nodes, and thus represents a single sequence of outcomes where the players act essentially without seeing any cards. The coarsest lossless abstraction has size 30. It is not until we allow a loss bound of 5 that the algorithm finds a lossy abstraction (of size 14). For

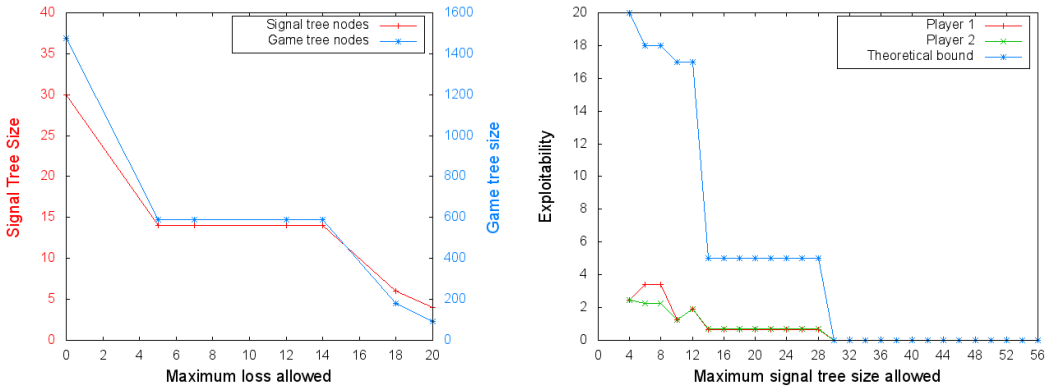


Figure 4.5: Left: Number of nodes in the signal tree (left y-axis) and in the game tree (right y-axis) as a function of the allowed loss in the IP model when minimizing tree size. Right: Exploitability as a function of the allowed signal tree size. Exploitability for both players is shown, along with our theoretical bound.

the model where a maximum tree size is given as input and the objective is to minimize the regret bound, we ran experiments for signal tree size bounds from 4 to 54. Figure 4.5 Right shows exploitability as a function of allowed signal tree size. Three plots are shown, the exploitability of each of the two players, and the exploitability bound given by Theorem 7. By and large, the three plots decrease with signal tree size, with a non-monotonicity at size 6, where Player 1's exploitability goes up compared to that at size 4. This is an instance of abstraction pathology, which exists in games: refining an abstraction can cause the equilibrium strategies to have higher exploitability in the original game Waugh et al. [2009a].

The different abstractions (signal trees) that the IP generated are presented in the appendix. Interestingly, sometimes the IP generated an abstraction that is asymmetric between the players.

4.1.7 Discussion

We presented the first framework for giving theoretical guarantees on solution quality in lossy abstraction in extensive-form games. We defined notions of payoff and nature error between the abstraction and the original game, and developed analysis tools that enable one to give an upper bound on regret and exploitability of equilibria computed in the abstract game, when mapped to the original game. To do this, we introduced a notion of strategy mappings, undivided lifted strategies. We also introduced an equilibrium refinement, self-trembling equilibrium, that we used to analyze the quality of general Nash equilibria from abstract games.

Using this framework, we developed the first lossy extensive-form game abstraction algorithm with bounds on solution quality. We did this by designing an IP that computes either the minimal size abstraction given a bound on solution quality, or a solution with minimum bound given a cap on tree size. Experiments on a relatively simple poker game showed that our method finds a lossless abstraction when one is available and lossy abstractions when smaller abstractions are desired.

While our framework can be used for lossy abstraction, it is also a powerful tool for lossless abstraction if we set the bound to zero. This is, to our knowledge, the first framework to provide tools for characterizing lossless abstractions in general extensive-form games of perfect recall. Likewise, our IP model is the first algorithm to compute abstractions with such guarantees. It is also the first algorithm to automatically detect the canonical representation of poker games while guaranteeing losslessness, without resorting to any domain-specific tricks. Until now, there has been no theory for algorithmically detecting such symmetries in a general sense that goes beyond poker. Further, as we described in Section 6, the only similar work with theoretical guarantees Gilpin and Sandholm [2007c] has the problem that it might not be able to reach the lossless abstractions due to discontinuity between the regions of lossless abstractions in the space of all abstractions. This is in spite of their work analyzing a much more restricted game setting, that very closely resembles poker, whereas our theoretical work makes no assumptions about the game beyond perfect recall.

As mentioned in Section 4.1.1, our quality results for lossy abstraction generalize similar results for the special case of stochastic games (expanded to a DAG) Sandholm and Singh [2012]. They get a cubic dependence on the depth of the game tree in their bounds. In our work, if we consider the sum over error terms in the bound in Theorem 7, we have $2\epsilon^R + \sum_{j \in \mathcal{H}_i} \epsilon_j^0 \overline{W} + \sum_{j \in \mathcal{H}_0} 2\epsilon_j^0 \overline{W}$. The two expressions $\sum_{j \in \mathcal{H}_i} \epsilon_j^0 \overline{W}$ and $\sum_{j \in \mathcal{H}_0} 2\epsilon_j^0 \overline{W}$ have the biggest dependence on depth in our bound. In converting the stochastic game, $\overline{W}$ gets a linear dependence on the depth, and the summations are linear in the depth. The error term ϵ_j^0 has no dependence on depth for either term, since each state of the stochastic game induces a subgame in the extensive-form representation. Our bound, when applied to a stochastic game DAG thus has only a quadratic dependence on depth, while theirs was cubic.

We introduced the extensive form game tree isomorphism and action subset selection problems, both important for computing abstractions on a level-by-level basis. We proved that the former is graph isomorphism complete, and the latter NP-complete. We showed that level-by-level abstraction can be too myopic and thus fail to find even obvious lossless abstractions. At the same time, it can be challenging to scale up the IP that considers all levels at once, depending on the game.

The hardness results do not mean that abstraction is not helpful. For one, the worst-case hardness results do not imply hardness in practice. Furthermore, many games can be decomposed in terms of information structure. Our algorithm can be used to compute abstractions on the decomposed parts, leading to a smaller game size in the full game. One example of such decomposition is conducting abstraction on the signal tree

instead of the full game tree. In fact, essentially all information abstraction algorithms for poker proceed level by level in tree data structures that are like the signal tree, except that each tree consists of signals to only one player. Such algorithms have proven to be key for being able to address large games even if the abstraction algorithm solves NP-complete subproblems when dealing with each level in turn [Gilpin and Sandholm, 2007a], even in two-player zero-sum games which are solvable in worst-case polynomial time.

There are many possible future directions for developing abstraction algorithms within our framework that provides bounds, for example, algorithms that proceed level by level but do not necessarily find an optimal abstraction (and may not even optimize within a level), or algorithms that consider multiple levels at once, but potentially not all levels and potentially not optimally.

While our algorithms work for any game of perfect recall, the set of abstractions they can compute is only a subset of all possible abstractions. Recent progress on abstraction for the specific application of solving poker games has been focused on the use of imperfect-recall abstractions [Waugh et al., 2009b, Johanson et al., 2013]. They happen to work well for poker, but are poorly understood. Other practical abstraction algorithms that fall outside our framework are ones that merge different information sets at different branches in the tree without requiring that nodes in the subtrees map to each other, nor that they are in the same information set for the other players. It is easy to construct examples where such abstraction has arbitrarily high loss, but it seems to work well in practice—at least for poker. It would be interesting to explore whether our bounding framework could be extended to these kinds of abstractions.

4.2 Imperfect-recall abstraction

4.2.1 Introduction

In recent years, imperfect-recall abstractions [Waugh et al., 2009b] have been the leading approach in the Annual Computer Poker Competition (ACPC). For the last several years, the winning no-limit Texas Hold'em bots have employed it (for example, see the winners "Baby Tartanian8" by Brown and Sandholm and "Slumbot" by Jackson, from the ACPC [2016]). Recently, this approach was also used in security games Lisy et al. [2016]. We expect that it could also be relevant to settings such as sequential auctions and trading, like the trading-agent competition, where abstraction approaches have previously been considered Wellman et al. [2005]. The motivation for imperfect-recall is computational. It allows the agent to forget less important details, in order to afford–from a computational perspective–to have a finer-grained abstraction of the present. Halpern and Pass [2013] showed this same phenomenon in "computational games," where agents are charged for computation. In this setting, choosing to forget information can be rational, because it decreases computational cost.

To the best of our knowledge, the only prior results for imperfect-recall abstraction were developed by Lanctot et al. [2012]. They present regret bounds in a class of imperfect-recall abstractions for equilibria computed using CFR [Zinkevich et al., 2007].

In contrast to prior work, our results are for a fairly general class of imperfect-recall abstractions, and are algorithm agnostic; they apply to both Nash equilibria, and strategies with bounded counterfactual regret. We focus on these two classes of strategies because they are, to the best of our knowledge, the only types of strategies used in practical large-scale game solving. While refinements of Nash equilibria are desirable in some settings, they are usually too expensive to compute in the large-scale games where abstractions are applied.

To develop our results, we generalize the notion of *skew well-formed games* (SWFGs) introduced by Lanctot et al. [2012], by introducing a new game class *chance-relaxed skew well-formed (CRSWF) games*. CRSWF games generalize SWFGs by also allowing chance error. Enabling chance error allows for a richer

set of abstractions where nodes can go in the same abstract information set even if the nature probabilities of reaching those nodes and going from those nodes are not the same (this was left as an open problem by Lanctot et al. [2012]). This enables dramatically smaller abstractions for some games. We extend the techniques from Section 4.1 to give theoretical solution-quality bounds for this class. The solution quality bounds we derive are exponentially stronger than those of Lanctot et al. [2012] which had a linear dependence on the number of information sets in the game tree, and did not weight the leaf reward error by the probability of a given leaf being reached. The reward error term in our result has only a linear dependence on tree height (actually, just the number of information sets any single player can experience on a path of play). Our leaf reward error term is weighted by the probability of the leaf being reached. Furthermore, our bounds are independent of the equilibrium computation method, while that prior work was only for CFR.

For games where abstraction of a subset of information sets at a single level is guaranteed to result in a CRSWF game, we show an equivalence between the problem of computing a single-level abstraction that minimizes our theoretical solution-quality guarantees and a class of clustering problems. Using the decrease in solution-quality bound from abstracting a pair of information sets as a distance function, we show that such abstraction problems form a metric space. This yields a 2-approximation algorithm for performing abstraction at a single level in the game tree when information sets differ only by the actions taken by players. When information sets differ based on nature's actions, our equivalence yields a new clustering objective that has not, to our knowledge, been previously studied. Our clustering results yield the first abstraction algorithm for computing imperfect-recall abstractions with solution-quality bounds.

Finally, we use our theoretical results to conduct experiments on a simple die-based poker game that has been used as a benchmark for game abstraction in prior work. The experiments show that the CFR algorithm works well even on abstractions where different nature probabilities are abstracted together, and that the theoretical bound is within 0 to 2 orders of magnitude of the regrets at CFR convergence.

4.2.2 Perfect-recall refinements and abstraction concepts

For games $\Gamma' = \langle N, A, S, Z, \mathcal{H}, \sigma_0, u, \mathcal{I}' \rangle$ and $\Gamma = \langle N, A, S, Z, \mathcal{H}, \sigma_0, u, \mathcal{I} \rangle$, we say that Γ is a perfect-recall refinement of Γ' if Γ has perfect-recall, and for any information set $I \in \mathcal{I} : \exists I' \in \mathcal{I}', I \subseteq I'$. That is, the game Γ can be obtained by partitioning the nodes of each information set in $\mathcal{I}'$ appropriately. For any perfect-recall refinement Γ, we let $\mathcal{P}(I')$ denote the information sets $I \in \mathcal{I}$ such that $I \subseteq I'$ and $\bigcup_{I \in \mathcal{P}(I')} = I'$. For an information set I in Γ, we let f_I denote the corresponding information set in Γ'.

We will focus on the setting where we start out with some perfect-recall game Γ, and wish to compute an imperfect-recall abstraction such that the original game is a perfect-recall refinement of the abstraction. Imperfect-recall abstractions will be denoted by $\Gamma' = \langle N, A, S, Z, \mathcal{H}, \sigma_0, u, \mathcal{I}' \rangle$. That is, they are the same game, except that some information sets have been merged.

For imperfect-recall information sets, we let $W(I') = \sum_{s \in I'} \frac{\pi^\sigma(s)}{\pi^\sigma(I')} V(s)$ be the value of an information set. Note that this definition is slightly different from that used in the section on perfect-recall abstraction. In particular we use the reach probabilities of all players here, whereas for perfect-recall abstraction we used the reach probabilities for all players except i.

We will later be concerned with a notion of how much better a player i could have done at an information set: the regret for information set I and action a is $r(I, a) = V_i^{\sigma_{I \to a}}(I) - V_i^\sigma(I)$. That is, the increase in expected utility for Player i obtained by deviating to taking action a at I. The immediate regret at an information set I given a strategy profile σ is $r(I) = \max_{a \in A_I} r(I, a)$. Regret is defined analogously for imperfect-recall games using $W(I)$.

Chance-relaxed skew well-formed (CRSWF) games

Now we introduce the class of imperfect-recall games that we consider as potential abstractions of a perfect-recall game. We call this class CRSWF games. A CRSWF game is an imperfect-recall game where there exists a perfect-recall refinement of the game that satisfies a certain set of properties that we introduce below. We will focus on the general problem of computing solution concepts in CRSWF games and mapping the solution concept to a perfect-recall refinement. Typically, the perfect-recall refinement is the original game, and the CRSWF game is an abstraction that is easier to solve.

Intuitively, a CRSWF game is an imperfect-recall game where there exists a refinement that satisfies two intuitive properties for any pair of information sets that are separated in the perfect-recall refinement. The first is that a bijection exists between the leaf nodes of the information set pair such that leaves mapped to each other pass through the same non-nature actions on the path from the information set to the leaf. This ensures that the probability of reaching pairs of leaves that map to each other is similar. The second is that a bijection exists between the nodes in the pair of information sets, such that the path leading to two nodes mapped to each other passes through the same information set-action pairs over all players except the acting player and nature. This ensures that the conditional distribution over nodes in the information sets is similar.

Definition 13. *For an EFG Γ', and a perfect-recall refinement Γ, we say that Γ' is a* CRSWF game with respect to Γ *if for all $i \in N, I' \in \mathcal{I}'_i, I, \breve{I} \in \mathcal{P}(I')$, there exists a bijection $\phi : Z_I \to Z_{\breve{I}}$ such that for all $z \in Z_I$:*

1. *In Γ', $X_{-\{i,0\}}(z) = X_{-\{i,0\}}(\phi(z))$, that is, for two leaf nodes mapped to each other (for these two information sets in the original game), the action sequences of the other players on those two paths must be the same in the abstraction.*[5]
2. *In Γ', $X_i(z[I], z) = X_i(\phi(z)[\breve{I}], \phi(z))$, that is, for two leaf nodes mapped to each other (for information sets I and $\breve{I}$ in the original game), the action sequence of Player i from I to z and from $\breve{I}$ to $\phi(z)$ must be the same in the abstraction.*

Our definition implicitly assumes that leaf nodes are all at the same level. This is without loss of generality, as any perfect-recall game can be extended to satisfy this.

With this definition, we can define the following error terms for a CRSWF refinement Γ of an imperfect-recall game Γ' for all $i \in N, I' \in \mathcal{I}'_i, I, \breve{I} \in \mathcal{P}(I'), z \in I$

- $\left|u_i(z) - \delta_{I,\breve{I}} u_i(\phi(z))\right| \leq \epsilon^R_{I,\breve{I}}(z)$, the reward error at z, after scaling by $\delta_{I,\breve{I}}$ at $\breve{I}$.
- $\left|\pi_0(z[I], z) - \pi_0(\phi(z)[\breve{I}], \phi(z))\right| = \epsilon^0_{I,\breve{I}}(z)$, the leaf probability error at z.
- $\left|\frac{\pi_0(z[I])}{\pi_0(I)} - \frac{\pi_0(\phi(z)[\breve{I}])}{\pi_0(\breve{I})}\right| = \epsilon^D_{I,\breve{I}}(z[I])$, the distribution error of $z[I]$.

The scaling term $\delta_{I,\breve{I}}$ does not affect the abstract game; it can be chosen by the user to minimize the bounds proved later. The reward error uses an inequality rather than equality because the error at a leaf node can vary by player.

Instead of our probability and distribution error terms, Lanctot et al. [2012] require $\pi_0(z) = l_{I,\breve{I}} \pi_0(\phi_{I,\breve{I}}(z))$, where $l_{I,\breve{I}}$ is a scalar defined on a per information set pair basis. We omit any such constraint, and instead introduce probability and distribution error terms as above. Our definition allows for a richer class of abstractions. Consider some game where every nature probability in the game differs by a small amount. For such a game, no two information sets can be merged according to the SWFG definition, whereas our definition allows such abstraction.

[5] It is possible to relax this notion slightly: if two actions of another player are not the same, as long as they are on the path (at the same level) to all nodes in their respective full-game information sets (I and $\breve{I}$), they do not affect the distribution over nodes in the information sets, and are thus allowed to differ in the abstraction.

We define $\overline{u}_{I,\breve{I}}(s) = \max_{i \in N, z \in Z_s} u_i(z) + \epsilon^R_{I,\breve{I}}(z)$, the maximum utility plus its scaled error achieved at any leaf node. This will simplify notation when we take the maximum over error terms related to probability transitions.

We now define additional aggregate approximation error terms. These will be useful when reasoning inductively about more than one height of the game at a time. We define the *reward approximation error* $\epsilon^R_{I,\breve{I}}(s)$ for information sets $I, \breve{I} \in \mathcal{P}(I')$ and any node s in $\breve{I}$ to be

$$\epsilon^R_{I,\breve{I}}(s) = \begin{cases} \epsilon^R_{I,\breve{I}}(z) & \text{if } \exists z \in Z : z = s \\ \sum_{a \in A_s} \sigma_0(s,a) \epsilon^R_{I,\breve{I}}(t^s_a) & \text{if } s \in S_0 \\ \max_{a \in A_s} \epsilon^R_{I,\breve{I}}(t^s_a) & \text{if } s \notin S_0 \wedge s \notin Z \end{cases},$$

We define the *transition approximation error* $\epsilon^0_{I,\breve{I}}(s)$ for information sets $I, \breve{I} \in \mathcal{P}(I')$ and any node s in $\breve{I}$ to be

$$\epsilon^0_{I,\breve{I}}(s) = \max_{\vec{a} \in X^b_{-0}(s)} \sum_{z \in Z^{\vec{a}}_s} \epsilon^0_{I,\breve{I}}(z) \overline{u}_{I,\breve{I}}(s)$$

We define the *distribution approximation error* for an information set pair $I, \breve{I} \in \mathcal{P}(I')$ to be

$$\epsilon^D_{I,\breve{I}} = \sum_{s \in I} \epsilon^D_{I,\breve{I}}(s) \overline{u}_{I,\breve{I}}(s).$$

Figure 4.6 shows two subtrees of an example game tree. Dotted lines with arrow heads show a CRSWF abstraction of the game. First consider the left node for P2, which maps to the right P2 node. It has distribution approximation error of zero (as is always the case for singleton information sets). It has transition approximation error of $0.2 \cdot 10 = 2$ (the maximizing sequences of player actions $\vec{a}$ are $[a, l]$ or $[a, r]$). Finally, the node has reward approximation error $1 \cdot 0.5$, since the biggest utility difference between nodes mapped to each other is 1, and the definition of reward approximation error allows taking a weighted sum at nature nodes. Now consider the leftmost information set for P1. The distribution approximation error at this information set is $0.2 \cdot 10$, since the conditional probability of being at each of the two nodes in the information set differs by 0.1 from the node in the information set that it is mapped to. The transition approximation error is zero for both nodes in the information set. The reward approximation error is zero, since all leaf nodes under the information set are mapped to leaf nodes with the same utility.

4.2.3 Strategies from abstract near-equilibria have bounded regret

To prove our main result, we first show that strategies with bounded regret at information sets in CRSWF games have bounded regret at their perfect-recall refinements.

Proposition 7. *For any CRSWF game Γ', refinement Γ, strategy profile σ, and information set $I' \in \mathcal{I}'$ such that Player i has bounded regret $r(I', a)$ for all $a \in A_{I'}$, the regret $r(I, a^*)$ at any information set $I \in \mathcal{P}(I')$ and action $a^* \in A_I$ is bounded by*

$$r(I, a^*) \leq \max_{\breve{I} \in \mathcal{P}(I')} \delta_{I,\breve{I}} r(I', a^*) + 2 \sum_{s \in I} \frac{\pi^\sigma(s)}{\pi^\sigma(I)} \left(\epsilon^0_{I,\breve{I}}(s) + \epsilon^R_{I,\breve{I}}(s) \right) + \epsilon^D_{I,\breve{I}}.$$

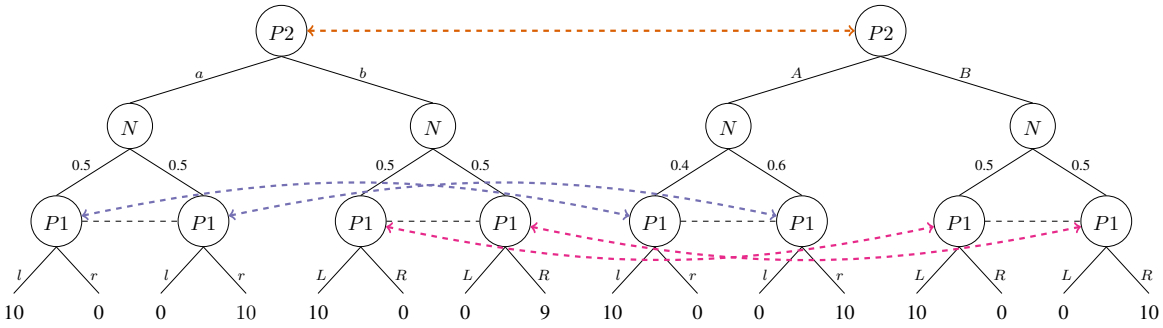


Figure 4.6: Two subtrees of a game tree. Information sets are denoted by dotted lines. A CRSWF abstraction is shown, with merged information sets and their node mapping denoted by dotted lines with arrowheads. All actions are mapped to their corresponding upper/lower-case actions in the merged information sets.

Intuitively, the scaling variable $\delta_{I,\check{I}}$ ensures that if the regret at I' is largely based on some other information set, then the regret is scaled to fit with the payoffs at I.

With this result, we are ready to prove our main results. First, we show that strategies with bounded regret at each information set in CRSWF games are ϵ-self-trembling equilibria when implemented in any perfect-recall refinement.

Theorem 11. *For any CRSWF game Γ' and strategy σ with bounded immediate regret $r_{I'}$ at each information set $I' \in \Gamma'$ where $\sigma_{-i}(I') > 0$, σ is an ϵ-self-trembling equilibrium when implemented in any perfect-recall refinement Γ, where $\epsilon = \max_{i \in N} \epsilon_i$ and*

$$\epsilon_i = \max_{\vec{a} \in X_i^b(r)} \sum_{j \in \mathcal{H}_i} \sum_{I \in \mathcal{D}_r^{\vec{a},j}} \pi_{-i}^{\sigma}(I) \left(\max_{\check{I} \in \mathcal{P}(f_I)} \delta_{I,\check{I}} r(f_I) + 2 \sum_{s \in I} \frac{\pi^{\sigma}(s)}{\pi^{\sigma}(I)} \left(\epsilon^0_{I,\check{I}}(s) + \epsilon^R_{I,\check{I}}(s) \right) + \epsilon^D_{I,\check{I}} \right).$$

This version of our bound weights the error at each information set by the probability of reaching the information set, and similarly, the error at each of the nodes in the information set is weighted by the probability of reaching it. This is important for CFR-style algorithms, where the regret at each information set I only goes to zero when weighted by $\pi_{-i}^{\sigma}(I)$. If one wishes to compute an abstraction that minimizes the bound independently of a specific strategy profile, it is possible to take the maximum over all player actions. Importantly, this preserves the probability distribution over errors at nature nodes. In the previous CFR-specific results of Lanctot et al. [2012], the reward error bound for each information set was the maximum reward error at any leaf node. Having the reward error be a weighted sum over the nature nodes and only maximized over player action sequences allows significantly finer-grained measurement of similarity between information sets. Consider any poker game where an information set represents the hand that the player holds, and consider three hands: a pair of aces I_A, pair of kings I_K, or pair of twos I_2. When the reward error is measured as the maximum over nodes in the information set, I_A and I_K are as dissimilar as I_A, I_2, since the winner changes for at least one hand held by the opponent for both information sets. In contrast to this, when reward errors are weighted by the probability of them being reached, we get that I_A and I_K are much more similar than I_A and I_2.

Our proof techniques have their root in those of Section 4.1 as well as the notion of refinements used by Lanctot et al. [2012]. We devise additional machinery, mainly Proposition 7 and the notion of CRSWF abstractions, to deal with imperfect recall. In doing so, our bounds get a linear dependence on height for the reward approximation error. The prior bounds [Kroer and Sandholm, 2014] have no dependence on height for the reward approximation error, and are thus tighter for perfect-recall abstractions.

In general, it is known that imperfect-recall games are harder to solve than perfect-recall games: In the two-player zero-sum case, the problem is NP-hard Koller and Megiddo [1992]. However, our game class, CRSWF games, is not so broad that it encompasses the type of game used in the proof by Koller and Megiddo [1992]. More importantly, we are not necessarily interested in solving the imperfect-recall game. Ultimately, we wish to find a strategy that we can map back to the original game, and get a good strategy. Theorem 11 shows that we do not need to solve the imperfect-recall game; we can just compute a strategy with low regret at each information set. To do this, we can employ the CFR algorithm, similar to Lanctot et al. [2012].

We now show a second result, which concerns the mapping of Nash equilibria in CRSWF games to approximate Nash equilibria in perfect-recall refinements.

Theorem 12. *For any CRSWF game Γ' and Nash equilibrium σ, σ is an ϵ-Nash equilibrium when implemented in any perfect-recall refinement Γ, where $\epsilon = \max_{i \in N} \epsilon_i$ and*

$$\epsilon_i = \max_{\vec{a} \in X_i^b(r)} \sum_{j \in \mathcal{H}_i} \sum_{I \in \mathcal{D}_r^{\vec{a},j}} \pi^\sigma_{-i}(I) \left(\max_{\check{I} \in \mathcal{P}(f_I)} 2 \sum_{s \in I} \frac{\pi^\sigma(s)}{\pi^\sigma(I)} \left(\epsilon^0_{I,\check{I}}(s) + \epsilon^R_{I,\check{I}}(s) \right) + \epsilon^D_{I,\check{I}} \right).$$

For practical game solving, Theorem 11 has an advantage over Theorem 12: any algorithm that provides guarantees on immediate counterfactual regret in imperfect-recall games can be applied. For example, the CFR algorithm can be run on a CRSWF abstraction, and achieve the bound in Theorem 11, with the information set regrets $\pi^\sigma_{-i}(I) r(f_I)$ decreasing at a rate of $O(\sqrt{(T)})$. Conversely, no good algorithms are known for computing Nash equilibria in imperfect-recall games.

4.2.4 Complexity and algorithms

We now investigate the problem of computing CRSWF abstractions with minimal error bounds. First, we show that this is hard, even for games with a single player and a game tree of height two.[6]

Theorem 13. *Given a perfect-recall game and a limit on the number of information sets, determining whether a CRSWF abstraction with a given bound as in Theorem 11 or 12 exists is NP-complete. This holds even if there is only a single player, and the game tree has height two.*

The hardness proof is by reduction from clustering, which also hints that clustering techniques could be used in an abstraction algorithm within our framework. Performing abstraction at a single level of the game tree that minimizes our bound reduces to clustering if the information sets considered for clustering satisfy Conditions 1 and 2. The distance function for clustering depends on how the trees match on utility and nature error, and the objective function depends on the topology higher up the tree. In such a setting, an imperfect-recall abstraction with solution quality bounds can be computed by clustering valid information sets level-by-level in a bottom-up fashion. In general, a level-by-level approach has no optimality guarantees, as some games allow *no* abstraction unless coupled with other abstraction at different levels (a perfect-recall abstraction example of this is shown in Section 4.1.5). However, considering all levels simultaneously is often impossible in practice. A medical example of a setting where a level-by-level scheme would work well is given by Chen and Bowling [2012], where an opponent initially chooses a robustness measure, which impacts nature outcomes and utility, but not the topology of the different subtrees. Similarly, the *die-roll poker* game introduced by Lanctot et al. [2012] as a game abstraction benchmark is amenable to this approach.

[6] Sandholm and Singh [2012] already showed hardness of computing an optimal abstraction when minimizing the actual loss of a unique equilibrium.

We now show that single-level abstraction problems (SLAPs) where Conditions 1 and 2 of Definition 13 are satisfied for all merges form a metric space together with the distance function that measures the error bound for merging information set pairs. Clustering problems over metric spaces are often computationally easier, yielding constant-factor approximation algorithms Gonzalez [1985], Feder and Greene [1988].

Definition 14. *A* metric space *is a set M and a distance function $d : M \times M \to \mathbb{R}$ such that the following holds for all $x, y, z \in M$: **(a)** $d(x, y) \geq 0$ **(b)** $d(x, y) = 0 \Leftrightarrow x = y$ (identity of indiscernibles) **(c)** $d(x, y) = d(y, x)$ (symmetry) **(d)** $d(x, y) \leq d(x, z) + d(z, y)$ (triangle inequality) .*

Proposition 8. *For a set of information sets $\mathcal{I}^m$ such that any partitioning of $\mathcal{I}^m$ yields a CRSWF abstraction (with no scaling, i.e. $\delta_{I,\breve{I}} = 1, \forall I, \breve{I} \in \mathcal{I}^m$), and a function $d : \mathcal{I}^m \times \mathcal{I}^m \to \mathbb{R}$ describing the loss incurred in the error bound when merging $I, \breve{I} \in \mathcal{I}^m$, the pair $(\mathcal{I}^m, d)$ forms a metric space.*

Conversely to our result above, if the scaling variables can take on any value, the triangle inequality does not hold, so $(\mathcal{I}^m, d)$ is not a metric space.

Consider three information sets I_1, I_2, I_3, each with two nodes reached with probability 0.9 and 0.1, respectively. Let there be one action at each information set, leading directly to a leaf node in all cases. Let $I_1 = \{1, 2\}, I_2 = \{5, 11\}, I_3 = \{10, 23\}$, where the name of the node is also the payoff of Player 1 at the node's leaf. We have that I_1 and I_2 map onto each other with scaling variable $\delta_{I_1,I_2} = 5$ to get $\epsilon^R_{I_1,I_2} = 1$ and $I_{2,3}$ with $\delta_{I_2,I_3} = 2, \epsilon^R_{I_2,I_3} = 1$. However, I_1 and I_3 map onto each other with $\delta_{I_1,I_3} = 10$ to get $\epsilon^R_{I_1,I_3} = 3$ which is worse than the sum of the costs of the other two mappings, since all reward errors on the right branches are multiplied by the same probability 0.1, i.e., $0.1 \cdot \epsilon^R_{I_1,I_2} + 0.1 \cdot \epsilon^R_{I_2,I_3} < 0.1 \cdot \epsilon^R_{I_1,I_3}$.

The objective function for our abstraction problem has two extreme versions. The first is when the information set that is reached depends entirely on players not including nature. In this case, the error bound over the abstraction at each level is the maximum error of any single information set. This is equivalent to the minimum diameter clustering problem, where the goal is to minimize the maximum distance between any pair of nodes that share a cluster; Gonzalez [1985] gave a 2-approximation algorithm when the distance function satisfies the triangle inequality. Coupled with Proposition 8 this gives a 2-approximation algorithm for minimizing our bound on SLAPs. The other extreme is when each of the information sets being reached differ only in nature's actions. In this setting, the error bound over the abstraction is a weighted sum of the error at each information set. This is equivalent to clustering where the objective function being minimized is the weighted sum over all elements, with the cost of each element being the maximum distance to any other element within its cluster. To our knowledge, clustering with this objective function had not been studied in the literature, even when the weights are uniform, prior to our introduction of this objective function. However, inspired by the problem-setting presented in this chapter, Gupta et al. [2016] developed approximation algorithms for this problem based on a pre-publication manuscript of our work.

In its most general setting, the objective function can be thought of as a tree, where a given leaf node represents some information set, and takes on a value equal to the maximum distance to any information set with which it is clustered. Each internal node either takes the maximum or weighted sum of its child-node errors. The goal is to minimize the error at the root node. In practice, integer programs (IPs) have sometimes been applied to clustering information sets for EFG abstraction [Gilpin and Sandholm, 2007a, Gilpin et al., 2007] (without bounds on solution quality, and just for perfect-recall abstractions), and are likely to perform well in our setting. An IP can easily be devised for any objective function in the above form.

For abstraction problems across more than a single level, Proposition 8 does not give any guarantees. While the result can be applied level-by-level, the abstraction performed at one level affects which information sets are valid for merging at other levels, and thus the approximation factor is not preserved across the levels.

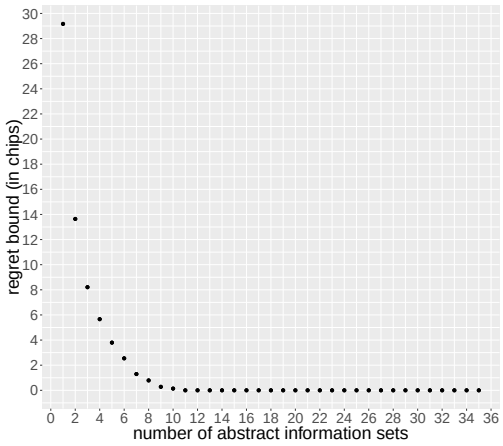


Figure 4.7: Regret bounds for varying SLAP sizes in DRP. The x-axis shows the number of information sets in the abstraction, and the y-axis shows the theoretical bound on soluation quality. The total number of information sets in the original game is 36

4.2.5 Experiments

We now investigate what the optimal SLAP bounds (in terms of Theorem 12) look like for the *die roll poker (DRP)* game, a benchmark game for testing abstraction [Lanctot et al., 2012]. Die-roll poker is a simple two-player zero-sum poker game where dice, rather than cards, are used to determine winners. At the beginning of the game, each player antes one chip to the pot. The game then consists of two rounds. In each round, each player rolls a private die (making the game imperfect information). Afterwards a betting round occurs. During betting rounds, a player may fold (causing the other player to win), call (match the current bet), or raise by a fixed amount, with a maximum of two raises per round. In the first round, each raise is worth two chips. In the second round, each raise is worth four chips. The maximum that a player can bet is 13 chips, if each player uses all their raises. At the end of the second round, if neither player has folded, a showdown occurs. In the showdown, the player with the largest sum of the two dice wins all the chips in the pot. If the players are tied, the pot is split.

DRP has the nice property that abstractions computed at the bottom level of the tree satisfy the conditions of Definition 13. At heights above that one we can similarly use our clustering approach, but where two information sets are eligible for merging only if there is a bijection between their future die rolls such that the information sets for the future rolls in the bijection have been merged. A clustering would be computed for each set in the partition that represents a group of information sets eligible for merging. In the experiments in this paper we will focus on abstraction at the bottom level of the tree. We use CPLEX to solve an IP encoding the SLAP of minimizing our bound given a limit on the number of abstract information sets. The results are shown in Figure 4.7. For one or two clusters, the bound is bigger than the largest payoff in the game, but already at three clusters it is significantly lower. At eight clusters, the bound is smaller than that of always folding, and decreases steadily to zero at eleven clusters (the original game has 36 information sets). While these experiments show that our bound is relatively small for the DRP game, they are limited in that we only performed abstraction at a single level. If abstraction at multiple levels is performed, the bound is additive in the error over the levels.

Another important question is how well strategies computed in abstractions that are good—as measured by our bound—perform in practice. Lanctot et al. [2012] conducted experiments to investigate the performance of CFR strategies computed in imperfect-recall abstractions of several games: DRP, Phantom

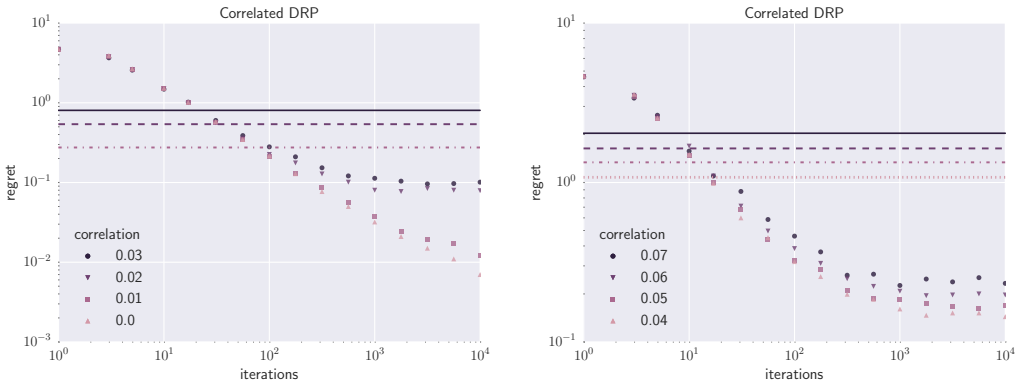


Figure 4.8: Log-log plots of the sum of the two players' regrets as a function of CFR iterations on the bound-minimizing abstraction of CDRP. The legends give the amount of correlation in the die rolls of the different CDRP games on which we ran experiments. The horizontal lines show the respective ex-ante regret bound of Theorem 12 for each of the CDRP games. (In the first game on the left where the correlation is zero, the abstraction is lossless, so the horizontal line (not shown) would be at zero.)

tic-tac-toe (where moves are unobserved), and Bluff. They found that CFR computes strong strategies in imperfect-recall abstractions of all these games, even when the abstraction did not necessarily fall under their framework. Their experiments validate a subset of the class of CRSWF abstractions: ones where there is no nature error. Due to this existing experimental work, we focus our experiments on problems where abstraction does introduce nature error. One class of problems where such error can occur are settings where players observe imperfect signals of some phenomenon. For such settings, one would expect that there is correlation between the observations made by the players. Examples include negotiation, sequential auctions, and strategic acquisition.

DRP can be thought of as a game where the die rolls are the signals. Regular DRP has a uniform distribution over the signals. We now consider a generalization of DRP where die rolls are correlated: *correlated die-roll poker* (CDRP). There are many variations on how one could make the rolls correlated; we use the following. We have a single correlation parameter c, and the probability of any pair of values (v_1, v_2), for Player 1 and 2 respectively, is $\frac{1}{\#sides^2} - c\,|v_1 - v_2|$. The probabilities for the second round of rolls is independent of the first round. As an example, the probability of Player 1 rolling a 3 and Player2 rolling a 5 with a regular 6-sided die in either round would be $\frac{1}{36} - 2c$. We generate DRP games with a 4-sided die and $c \in \{0, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07\}$.

For each value of c, we compute the optimal bound-minimizing abstraction for the second round of rolls, with a static mapping between information sets such that for any sequence of opponent rolls, the nodes representing that sequence in either information set are mapped to each other. The bound cost of the mappings is precomputed, and the optimal abstraction is found with a standard MIP formulation of clustering. After computing the optimal abstraction for a given game, we run CFR on the abstraction, and measure the regret for either player in terms of their regret in the full game. Figure 4.8 shows the results of these experiments. On the x-axis is the number of CFR iterations. On the y-axis is $r_1 + r_2$, where r_i is the regret for Player i for the strategy at a given iteration. Furthermore, the horizontal lines denote the regret bound of Theorem 12 for an exact Nash equilibrium. On the left in Figure 4.8 is shown the results for the four smallest values of c, on the right the four largest values. As can be seen, CFR performs well on the abstractions, even for large values of c: when $c = 0.7$, a very aggressive abstraction, the sum of regrets still

goes down to ~ 0.25 (for reference, always folding has a regret of 1). We also see that for $c \geq 0.2$, the regret stops decreasing after around 1000 iterations. This is likely where CFR converges in the abstraction, with the remaining regret representing the information lost through the abstraction. We also see that our theoretical bound is at the same order of magnitude as the actual bound even when CFR converges.

4.2.6 Discussion

In this section, we proved bounds for abstractions obtained through merging information sets. The perfect-recall results of Section 4.1 also allow abstraction by removing actions available to players. The following approach can be adopted for imperfect-recall abstraction with such branch removal, while still obtaining solution-quality guarantees. First, a valid perfect-recall abstraction is computed, where the desired branches are removed. The results by Section 4.1 give bounds on the solution quality of equilibria computed in this abstraction. An imperfect-recall abstraction can then be computed from this perfect-recall abstraction, with our results providing bounds on solution quality for this step. Solution quality bounds can then be achieved for the final abstraction by taking the sum of the bounds for the two steps. It is likely that tighter bounds could be derived by analyzing the distance between the original game and the final abstraction directly. We leave this as future research.

An important avenue for future research is how to merge the solution-quality bounds obtained in this work with practical algorithms for generating abstractions. We showed how single-level abstraction problems can be addressed. For multi-level abstraction, a similar approach can be adopted, but where the abstraction is either computed greedily level-by-level, or using IP or search algorithms that ensure that the abstraction satisfies Conditions 1 and 2 of Definition 13 across levels.

4.3 Discretizing continuous action spaces

4.3.1 Introduction

The computational EFG literature has largely focused on games where the action spaces are discrete at every node of the game. Indeed, most algorithms for solving extensive-form games require this (e.g. von Stengel [1996], Zinkevich et al. [2007], Lanctot et al. [2009], Hoda et al. [2010], Pays [2014]). (One notable exception to this was introduced by Johanson et al. [2012], where a technique was demonstrated for handling continuous action spaces for nature in a fairly restricted sense.) However, not all games encountered in practice are discrete. Sources of continuity include noise (e.g. Gaussian) being modeled by a nature node, bid sizes in auctions, betting sizes in no-limit poker games, type profiles in mechanism design, and power levels in jamming games.

In the past, such continuity has largely been handled by heuristic discretization, with no theoretical guarantees on solution quality. In contrast, we develop the first general bounds on solution quality for discretizations of continuous action spaces in a very broad class of games. Building on our results on perfect-recall abstraction in Section 4.1, we show how to discretize continuous action spaces in a way that gives theoretical bounds on solution quality when using any Nash equilibrium computed in the discretized game to form a strategy in the full (continuous) game.

We then proceed to investigate the computation of discretizations that minimize an error bound. We first formulate the general problem, making only the assumption that the error bound function is *monotonic*, which intuitively requires that the error gets worse the farther a point gets from the discrete point to which it maps. Since our problem formulation consists of linear constraints, this immediately leads to the conclusion that convex error functions can typically be minimized in polynomial time. We then go on to consider error

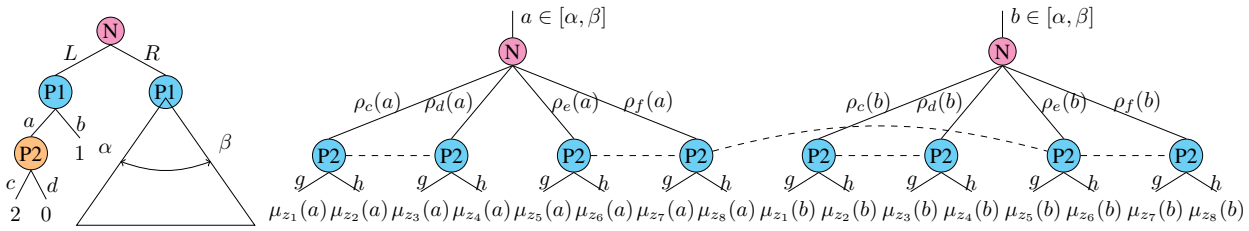


Figure 4.9: An example of a game tree with the triangle representing a subtree with a continuous action space at the node.

functions of the form derived in our theoretical solution-quality bounds. We show that when individual leaf and nature node error functions are linear, and nature always picks uniformly over continuous action intervals, the bound-minimizing solution is to uniformly discretize each continuous interval. We further show how to decompose the problem and we develop a general framework for optimizing the decomposed formulation for convex functions. We then develop a mixed-integer program (MIP) for computing optimal discretizations when only player action spaces are being discretized and the error functions are piecewise linear.

4.3.2 Continuous action spaces

We will assume that we are dealing with a game Γ, where one or more nodes $s \in S$ have one or more continuous action intervals $A_{s,c} = [\alpha_{s,c}, \beta_{s,c}] \subseteq A_s$ for $c \in C_s$, where C_s is an index set of the continuous action spaces at s.

Let S_a be the set of nodes in the subtree reached by taking action $a \in A_{s,c}$ at node s. We assume there is a one-to-one mapping $\phi_{a,\hat{a}} = \phi_{\hat{a},a}$ between S_a and $S_{\hat{a}}$ for any two actions $a, \hat{a} \in A_{s,c}$, where nodes at a given height map onto nodes at the same height, and the condition of Definition 15 is satisfied. Intuitively, the condition requires that nodes that are mapped to each other are either (1) in the same information set, or (2) their information sets contain no nodes from outside the (infinitely large) set of subtrees reachable by taking actions from $A_{s,c}$ at s, and for any other node in the same information set and same subtree, the nodes mapped to must similarly be in the same information set.

Definition 15. *For any node $\hat{s} \in S_a$ in the subtree at $a \in A_{s,c}$, we require one of the two following conditions to hold for all a':*

1. $\phi_{a,a'}(\hat{s}) \in I_{\hat{s}}$
2. $I_{\hat{s}} \subset \bigcup_{\bar{a} \in [\alpha_{s,c}, \beta_{s,c}]} S_{\bar{a}}$, and for any other node $\bar{s} \in I_{\hat{s}}, \bar{s} \in S_a$: $\phi_{a,a'}(\bar{s}) \in I_{\phi_{a,a'}(\hat{s})}$

For each leaf node $z \in Z_{t^s_a}$ for some $a \in A_{s,c}$, we assume that the payoff to Player i can be described as $u_i(z) = \mu^i_z(a)$, where $\mu^i_z = \mu^i_{\phi_{a,\hat{a}}(z)}$ for all $\hat{a} \in A_{s,c}$. Intuitively, all leaf nodes that map to each other have their payoffs described by the same function, with the actual value depending on the choice of a. Similarly, the probability of each outcome at each nature node s' in the subtree rooted at t^s_a is described by a function $\rho_{s'}(a)$ where $\rho_{s'} = \rho_{\phi_{a,\hat{a}}}(s')$ for all $\hat{a} \in A_{s,c}$.

Since we require a bijection mapping $\phi_{a,\hat{a}}$ between the subtrees for any two actions $a, \hat{a} \in [\alpha_{s,c}, \beta_{s,c}]$ for each node s with continuous action interval c, the infinitely many subtrees can equivalently be thought of as infinitely many instantiations of a single game-tree prototype, where payoffs and nature outcome probabilities are parameterized by the choice of $a \in [\alpha_{s,c}, \beta_{s,c}]$, and the information set topology of the instantiations satisfy Definition 15.

An example game is shown in Figure 4.9. On the left is a game with three players: nature(N), Player 1 ($P1$), and Player 2 ($P2$). Nature first chooses between L and R (with some unspecified fixed probability distribution). If L is chosen, a discrete subgame is reached. If R is chosen, $P1$ has a continuous action space $[\alpha, \beta]$. In the middle and right are shown two specific subtrees under the continuous action space, for $P1$ choosing actions $a, b \in [\alpha, \beta]$, respectively. The information set that spans across the trees is an example of one that satisfies Condition 1 of Definition 15, while the one that does not span across the trees satisfies Condition 2.

So far we have described our notation in terms of discretizing a single continuous action space at some node. When there is more than one node with a continuous action space that is being discretized, we have to consider two groups of nodes with continuous action spaces.

The first group is the set of all nodes s that have one or more continuous action intervals, and s is the first node on the path from the root to s with a continuous action space. Let the set of all such nodes be $\mathcal{S}^1 \subseteq S$. These nodes are handled exactly as described above. If there are additional continuous action spaces in the subtree after taking some action $a \in A_{s,c}$ for some s, c, then the bijections between subtrees simply map uncountably many nodes.

The second group is the set of all nodes s' such that there is at least one node-action pair s, a on the path from the root to s', where s has a continuous action space $A_{s,c}$ such that $a \in A_{s,c}$. Let this set of nodes be called $\mathcal{S}^2 \subset S$. Let $\vec{a} \in \mathbb{R}^n$, where n is the number of continuous action spaces leading to s' and including the one at s', such that $a_i \in [\alpha_i, \beta_i]$ where $[\alpha_i, \beta_i]$ is the continuous action space for the i'th node with a continuous action space on the path to s'. We then require that the payoff and nature functions are functions of $\vec{a}$ rather than a single action choice. For fixed choices in the past, the functions then behave exactly as the functions for nodes in $\mathcal{S}^1$.

We fix some ordering of all continuous intervals, and define $\mathbb{A} = \times_{s \in \mathcal{S}^1 \cup \mathcal{S}^2, c \in C_s} A_{s,c}$ to be the Cartesian product over all continuous intervals. We will use j to denote indices into this ordering $\mathbb{I}$. We let $[\alpha_j, \beta_j]$ denote the endpoints of interval $j \in \mathbb{I}$. From now on we will use $\vec{a} \in \mathbb{A}$ to denote elements of this set. A *discretization* is a finite set of points $\mathbb{A}' \subset \mathbb{A}$. The size $|\mathbb{A}'| = m$ is the number of discrete points chosen for each interval. If fewer than m points are desired for some interval $A_{s,c}$ with index $j \in \mathbb{I}$, we can simply let $\vec{a}_j = \vec{a}'_j$ for distinct points $\vec{a}, \vec{a}' \in \mathbb{A}'$. For a node s, we will use $\vec{a}^s$ to refer to the subset of actions taken on the path to s such that the action is part of a continuous action interval. We will overload the notation of the payoff and nature error functions so that for a given f or h for a node s, they take elements $\vec{a} \in \mathbb{A}$ as input, with the implicit understanding that the value of the function depends only on $\vec{a}^s$. We will let $\pi_0(\vec{a})$ denote the product of probabilities over each index j into $\vec{a}$ such that nature acts at $\vec{a}_j$.

The game $\Gamma' = \langle N, A', S', Z', \mathcal{H}, \sigma_0, u, \mathcal{I}' \rangle$ is the discrete extensive-form game obtained by restricting players to selecting actions that are part of the discretization $\mathcal{A}'$.

4.3.3 Discretization model

Discretization mapping and error terms

We will need to reason about the similarity of the full game Γ and the induced game Γ' for a given discretization $\mathbb{A}'$. To do this we will require a mapping of the continuous action space onto the discretization:

Definition 16. *A* discretization mapping *is a surjective function $g : \mathbb{A} \to \mathbb{A}'$ that maps the continuous action space $\mathbb{A}$ onto the discretization $\mathbb{A}'$. We require that g is decomposable, so for all $\vec{a} \in \mathbb{A}$, $g(\vec{a})_j$ depends only on $\vec{a}_j$. $\mathcal{G}_{\mathbb{A}'}$ denotes the set of legal discretization maps for a given discretization $\mathbb{A}'$.*

The discretization mapping along with the bijections $\phi_{a,a'}$ for all $s \in S, c \in C_s, a, a' \in A_{s,c}$ immediately defines a mapping of the nodes S onto the nodes S'. Denote this node mapping function by $h : S \to S'$.

For any node $s \in S \cap S'$, $h(s) = s$. For $s \in S, s \notin S'$, $h(s)$ is the node in S' reached by inductively applying the maps g and $\phi_{\vec{a}^s_j, g(\vec{a}^s)_j}$ at each continuous action space on the path to s.

Due to the constraints in Definition 15, g also leads to an information set mapping, as any two nodes $s_1, s_2 \in I$ for some I must map to the same information set: $h(s_1), h(s_2) \in I'$ for some I'. We let $f : \mathcal{I} \to \mathcal{I}'$ be this information set mapping.

For all three functions g, h, f we also define their inverses g^{-1}, h^{-1}, f^{-1}, that return all intervals $\bar{\mathbb{A}} \subseteq \mathbb{A}$, nodes $\bar{S} \subset S$, and information sets $\bar{\mathcal{I}} \subseteq \mathcal{I}$, respectively, that map onto given $\vec{a} \in \mathbb{A}', s' \in S'$, and $I' \in \mathcal{I}'$, respectively. We denote by $h_I^{-1}(s')$ the intersection $h^{-1}(s') \cap I$.

Given a discretization mapping g, it will be convenient to define aggregate utility error terms for nodes of the real game:

$$\epsilon^R_{s,i} = \begin{cases} \max_{a \in A_s} \epsilon^R_{t^s_a, i} & \text{if } s \text{ is a player node} \\ \int_{a \in A_s} \sigma_0(s,a) \epsilon^R_{t^s_a, i} & \text{if } s \text{ is a nature node} \\ \left| \mu^i_s(\vec{a}) - \mu^i_s(g(\vec{a})) \right| & \text{if } s \text{ is a leaf node} \end{cases}$$

Similarly, we define aggregate error terms for nature error. We define the nature distribution error of an information set I and node $s' \in f(I)$ to be

$$\epsilon^0_{I,s'} = \left| \frac{\int_{s \in h_I^{-1}(s')} \sigma_0(s)}{\sigma_0(I)} - \frac{\sigma'_0(s')}{\sigma'_0(f(I))} \right|$$

This is the difference between nature's probability of reaching s' and its probability of reaching any node in $h_I^{-1}(s')$, normalized by the probability of reaching the given information sets. The nature error for information set I is

$$\epsilon^0_I = \sum_{s' \in f(I)} \epsilon^0_{I,s'}$$

For a nature node s at height $k \in \mathcal{H}_0$ and $s' = h(s)$, we define the nature action $a' \in A_{s'}$ error and node error to respectively be

$$\epsilon^0_{s,s',a'} = \left| \sigma'_0(s', a') - \int_{a \in g^{-1}(a') \cap A_s} \sigma_0(s,a) \right|$$
$$\epsilon^0_s = \sum_{a' \in A_{s'}} \epsilon^0_{s,s',a'}$$

The nature error at height k is

$$\epsilon^0_k = \begin{cases} \int_{I \in \mathcal{I}_k} \pi(I) \epsilon^0_I & \text{if } k \notin \mathcal{H}_0 \\ \int_{s \in S_k} \pi(s) \epsilon^0_s & \text{if } k \in \mathcal{H}_0 \end{cases}$$

Finally, we let $\bar{W} = \max_{i \in N, z \in Z} u_i(z)$ be the maximum payoff at any leaf node.

Strategy mapping

Once a Nash equilibrium has been computed in the discretized game, we need to define a way of converting that strategy profile to a strategy profile for the full game. We perform this mapping in the following simple way. Since all subtrees under discretized intervals have the same shape (based on how we defined the

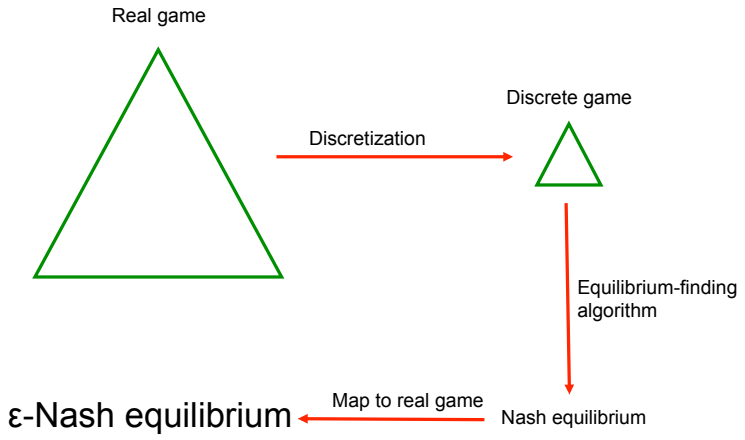


Figure 4.10: An overview of our discretization approach.

discretization problem in Section 4.3.2) and thus same number of actions, we can simply implement the same strategy that we computed for a given discretized subtree at all subtrees that map to it. Specifically, let σ' be the strategy profile computed for the discretized game. Then for each real node $s \in S$, we set $\sigma(s, a) = \sigma'(h(s), a')$, where a' is the action at s' that leads to $h(t^s_a)$. For continuous intervals, we simply pick each discrete point a with the probability that it was chosen in the discrete game, and every other action with probability zero. This mapping yields a strategy profile that satisfies the following property, which we will use to derive bounds later:

Proposition 9. *For a strategy profile σ' computed in a discretized game Γ', our method of strategy conversion leads to a strategy profile σ for the full game Γ so that for any information set pair I, I' such that I maps onto I', $\sigma_{-0}(I) > 0$, and $\sigma_i(I) > 0$,*

$$\left| \frac{\sigma(s')}{\sigma(I')} - \sum_{s \in g_I^{-1}(s')} \frac{\sigma(s)}{\sigma(I)} \right| \leq \epsilon^0_{I,s'}$$

4.3.4 Overview of our approach

Given some game with continuous action spaces, the goal in this work is to pick a finite set of points for each continuous action interval. This will induce a finite extensive-form game. A (potentially approximate) Nash equilibrium is then computed in the discrete game. The computed Nash equilibrium is then mapped to a strategy profile in the full (continuous) game. Figure 4.10 illustrates the approach.

Under reasonable assumptions, we will derive solution quality bounds for any Nash equilibrium computed in the abstraction when implemented in the full game. More specifically, we will show that such strategy profiles constitute ϵ-Nash equilibria in the full game, where the ϵ depends on the error terms we defined in the previous section. These results are anlogous to the solution-quality results for perfect-recall abstraction in Section 4.1.

4.3.5 Discretization quality bounds

We start by showing an error bound on solution quality for any given discretization and discretization mapping, leveraging Theorem 7.

Theorem 14. *For any game Γ with continuous action spaces $\mathbb{A}$ that satisfy the constraint given in Definition 15, discretization $\mathbb{A}' \subset \mathbb{A}$, and discretization mapping $g : \mathbb{A} \to \mathbb{A}'$, any Nash equilibrium σ computed in Γ' constitutes an ϵ-Nash equilibrium when implemented in Γ, where*

$$\epsilon = \max_i \left\{ 2\epsilon^R_{s,i} + \sum_{k \in \mathcal{H}_i} \epsilon^0_k \overline{W} \right\} + 2 \sum_{k \in \mathcal{H}_0} \epsilon^0_k \overline{W}$$

The bounds as given here are in their most general form. In particular, the two nature error terms $\sum_{k \in \mathcal{H}_i} \epsilon^0_k \overline{W}$ and $2 \sum_{k \in \mathcal{H}_0} \epsilon^0_k \overline{W}$ are not given in terms of the functions ρ_s that describe the change in nature outcome probabilities. ϵ^0_k can easily be bounded for all k in $\mathcal{H}_i$ and $\mathcal{H}_0$ respectively:

$$\epsilon^0_k \leq \int_{I \in \mathcal{I}_k} \pi(I) \sum_{s' \in f(I)} \left| \int_{s \in h_I^{-1}(s')} \pi_0(\vec{a}^s) - \pi_0(\vec{a}^{s'}) \right| = \xi^0_k$$

$$\epsilon^0_k \leq \int_{s \in S_k} \pi(s) \sum_{a' \in A_{s'}} \left| \int_{a \in g_I^{-1}(a')} \sigma_0(s, a) - \sigma_0(s', a') \right| = \xi^0_k$$

This gives the following corollary:

Corollary 1. *For any game Γ with continuous action spaces $\mathbb{A}$ that satisfy the constraint given in Definition 15, discretization $\mathbb{A}' \subset \mathbb{A}$, and discretization mapping $g : \mathbb{A} \to \mathbb{A}'$, any Nash equilibrium σ computed in Γ' constitutes an ϵ-Nash equilibrium when implemented in Γ, where*

$$\epsilon = \max_i \left\{ 2\epsilon^R_{s,i} + \sum_{k \in \mathcal{H}_i} \xi^0_k \overline{W} \right\} + 2 \sum_{k \in \mathcal{H}_0} \xi^0_k \overline{W}$$

The ξ^0_k terms do not diverge. While an infinite sum is taken in both cases, the terms are probability weighted, and $\int_{I \in \mathcal{I}_k} \pi(I) = 1, \int_{s \in S_k} \pi(s) = 1$. Since we are dealing with probability distributions, we can take the maximum over $\mathcal{I}_k$ or S_k. In practical settings, it may be desireable to take several maxima. First, in the current form of both Theorem 14 and Corollary 1, the bound depends on the strategy profile of the players, not just nature. To make the bound independent of player actions, one can take the maximum over player actions. Second, instead of computing the infinite sum of errors over $\mathcal{I}_k$ or S_k, it may be useful to take the probability-weighted sum of errors over discrete points, and then compute the maximum error over each discrete point.

4.3.6 Discretization algorithms

In this section we consider general bounded discretization problems that are largely independent of the specific error bound to be minimized. This means that our algorithms will apply to the results from Section 4.3.5, and also to any (potentially stronger or more general) bounds obtained in the future, as long as they fall under the setting described in Section 4.3.6.

Optimal discretization as an optimization problem

We consider a more general class of problems than those described in Section 4.3.2. We consider a game Γ where one or more $s \in S$ has a continuous action space. We again let $\mathbb{A}$ be the set of all intervals to be discretized with index set $\mathbb{I}$ and let $\vec{a}_j^l$ refer to the k_j discrete points chosen for interval j. We assume the points in the discretization are ordered, so $\vec{a}_j^l < \vec{a}_j^{l+1}$ for all $j \in \mathbb{I}, l \in [k_j]$.

We start by formulating the optimization problem very generally. We assume that we have an error bounding function $\Psi : \mathbb{A}^k \times \mathcal{G}_{\mathbb{A}} \to \mathbb{R}$ that takes as input a discretization $\mathbb{A}'$ and discretization map g and returns a real-valued error bound $\Psi(\mathbb{A}', g)$. We make two innocuous assumptions about Ψ to represent the natural condition that the error increases the further an actual point is from the discrete point to which it maps.

First, each point maps to its nearest lower or upper discrete point. Formally, for all $j \in \mathbb{I}$, and any point a_l, $\vec{a}_j^l < a_l < \vec{a}_j^{l+1}$ not in the discretization, Ψ is minimized at $g(a_l) = \vec{a}_j^l$ or $g(a_l) = \vec{a}_j^{l+1}$.

Second, for all $j \in \mathbb{I}$, and any two points $a_l, \hat{a}_l$, $\vec{a}_j^l < a_l < \hat{a}_l < \vec{a}_j^{l+1}$ not in the discretization, if Ψ is minimized when $g(\hat{a}_l) = \vec{a}_j^l$ then Ψ is minimized when $g(a_l) = \vec{a}_j^l$, and if Ψ is minimized when $g(a_l) = \vec{a}_j^{l+1}$ then Ψ is minimized when $g(\hat{a}_l) = \vec{a}_j^{l+1}$.

We will say that an error function Ψ that satisfies our two assumptions is *monotonic*. Given a monotonic Ψ and a discretization $\mathbb{A}'$, the optimal mapping for each interval $j \in \mathbb{I}$ can be determined by finding the splitting point between each interval $\left[\vec{a}_j^l, \vec{a}_j^{l+1}\right]$ such that the left side of the splitting point is mapped onto $\vec{a}_j^l$ and right side is mapped onto $\vec{a}_j^{l+1}$.

For each interval $j \in \mathbb{I}$, we introduce real-valued variables $\vec{a}_j^l \in \mathbb{A}_j, l \in [k_j]$ and $g_{s,c}^\tau, \tau \in [k_j - 1]$, where k_j is the desired number of discrete points for interval $\mathbb{A}_j$. The variables $\vec{a}_j^l$ represent the discrete points, while g_j^τ represents the point between $\vec{a}_j^\tau$ and $\vec{a}_j^{\tau+1}$ that separates the interval, such that the points in the interval $\left[\vec{a}_j^\tau, g_j^\tau\right]$ map onto $\vec{a}_j^\tau$ and the points in the interval $\left[g_j^\tau, \vec{a}_j^{\tau+1}\right]$ map onto $\vec{a}_j^{\tau+1}$. Since any set of values for $\vec{a}_j^l, g_j^\tau$ over all $j \in \mathbb{I}, l \in [k_j], \tau \in [k_j - 1]$ completely specifies a discretization and discretization mapping, we let $\mathbb{A}'_v, g_v$ denote the discretization and mapping obtained by a solution. The feasible set of this problem is

$$\mathcal{F} = \left\{ \mathbb{A}'_v, g_v : \begin{array}{ll} \vec{a}_j^l \le \vec{a}_j^{l+1} & \forall j \in \mathbb{I}, l \in [k_j - 1] \\ \vec{a}_j^\tau \le g_j^\tau \le \vec{a}_j^{\tau+1} & \forall j \in \mathbb{I}, \tau \in [k_j - 1] \\ \vec{a}_j^l \in \mathbb{A}_j & \forall j \in \mathbb{I}, l \in [k_j] \\ g_j^\tau \in \mathbb{A}_j & \forall j \in \mathbb{I}, \tau \in [k_j - 1] \end{array} \right\} \tag{4.13}$$

With this notation, we arrive at the most generic form of the optimal discretization problem for monotonic error functions:

$$\min \left\{ \Psi(\mathbb{A}'_v, g_v) : \left(\mathbb{A}'_v, g_v\right) \in \mathcal{F} \right\} \tag{4.14}$$

Setting $k = 1$, one can see that this is equivalent to minimizing *any* function, so this general form will not get us far. Fortunately, there is often further structure in practical games. In the rest of the algorithms section, we will consider various forms of structure that enable us to design efficient algorithms for finding good discretizations.

Convex error function The constraints specified in (4.13) are all linear. Thus, if Ψ is convex, solving (4.14) becomes a convex minimization problem over linear constraints. These are solvable in polynomial

time under mild assumptions [Ben-Tal and Nemirovski, 2001][7]. Perhaps more importantly, large subsets of this class of error functions have practically efficient solution methods—e.g., an error function that is conic-quadratic or semi-definite representable (Ben-Tal and Nemirovski [2001] give a thorough discussion of such representability), smooth, or non-smooth with certain structure [Nesterov, 2005b, Beck and Teboulle, 2012].

Decomposable error function

In Section 4.3.5, we considered error functions that are a mixture of maximums (player nodes) and probability-weighted sums (nature nodes) of the error functions at individual nodes. We now consider how to (recursively) represent such error functions using linear inequalities, for the purpose of computation.

Let $\xi_j : \mathbb{A}_j \times g_j \to \mathbb{R}$ be an error function that gives the error incurred at interval $j \in \mathbb{I}$ when choosing discretization $\mathbb{A}'_j$ and mapping g_j at $\mathbb{A}_j$. Recursively taking the maximum or weighted sum can be implemented by recursively applying the following linear inequalities, where variable e_δ represents the error at a given node or information set δ:

$$\mathcal{E} = \left\{ \begin{array}{l} e_s \geq \sum_{a \in A_s} \sigma_0(s,a) e_{t^s_a} \\ e_s \geq \max_{a \in A_s} e_{t^s_a} \\ e_s \geq \xi_j \end{array} \right\} \tag{4.15}$$

The same linear formulation can also be used to formulate the error bound in the case where $\xi_s : \mathbb{A} \times g$ are functions that give the error for each leaf and nature node. In a slight abuse of notation, we denote the set of error function solutions described in (4.15) by $\mathcal{E}$.

If each error function ξ_j depends only on the subset of $\mathbb{A}$ that consists of interval j and any descendant intervals, the discretization problem can be decomposed: find each interval $j \in \mathbb{I}$ that is the first continuous interval from the root to the interval. Each such interval, along with its subtrees and any intervals therein, can be minimized separately.

Minimizing our bound

We will now study the game class that satisfies Definition 15 and minimization of the bound given in Corollary 1. We will consider various types of error functions.

Linear error functions In this section we consider games Γ where the utility and nature outcome distribution functions μ_z, ρ_s, at all $z \in Z, s \in S$ that are descendants of a continuous interval, are Lipschitz continuous with Lipschitz constant $L_{z/s}$. This encompasses two important practical classes of game: (1) all the functions μ and ρ are linear, and (2) the functions are non-linear but the bound being derived is based on knowing (only) that the functions are Lipschitz continuous.

If all continuous intervals at nature nodes have a uniform distribution, we get the following simple results: all intervals should be discretized into uniform interval sizes.

Theorem 15. *For a game Γ with continuous action spaces $\mathbb{A}$, where each utility and nature outcome distribution function is Lipschitz continuous with constant $L_{z/s}$ for each leaf z or nature node s, and all continuous nature intervals are uniformly distributed, the bound-minimizing k_j-point discretization at each*

[7] A heavily updated version of this book is at http://www2.isye.gatech.edu/~nemirovs/.

interval $\mathbb{A}'_j$ is:

$$\vec{a}_j^l = \alpha_j + \left(l - \frac{1}{2}\right) \cdot \left(\frac{\beta_j - \alpha_j}{k_j}\right) \qquad \forall l \in [k_j]$$

$$g_j^\tau = \alpha_j + \tau \cdot \left(\frac{\beta_j - \alpha_j}{k_j}\right) \qquad \forall \tau \in [k_j - 1]$$

We will call this a uniform discretization.

When the conditions of the above theorem do not hold, the decomposition results from Section 4.3.6 and the recursive linearization (4.15) still apply. In the following two subsections, we leverage this fact.

Convex error functions As we pointed out above, taking the maximum, sum, and integral all preserve convexity. Thus, if the error

$$\left|\mu_z(a) - \mu_z(a')\right| \quad \text{or} \quad \left|\rho_s(a) - \rho(a')\right|$$

at each leaf or nature node can be represented by a convex function, the linear constraints in (4.15) can be used to represent the overall error as a convex function. As discussed in Section 4.3.6, this would, depending on the specific structure, allow the application of various efficient polynomial-time methods. We do not give specific algorithms here, but merely point out that optimal solutions can be found in polynomial time. In practice, the specific choice of which polynomial-time algorithm to apply should be informed by the exact structure of the error functions of the given game.

Piecewise linear error functions We now consider piecewise linear utility error functions and piecewise linear nature probability error functions for discretizing continuous player action intervals. We do not consider discretizing nature actions here because even with linear functions and a uniform nature distribution, discretizing nature intervals would lead to quadratic error, as shown in the proof of Theorem 15. Even if the actual error functions are not piecewise linear, this can be used for arbitrarily accurate approximation.

Finding a bound-minimizing discretization, subject to a limit on the number of discretization points, is NP-hard. This is easily seen from Theorem 9, and representing their discrete game-abstraction problem using a step function.

However, it can be represented by a MIP, where the number of binary variables is equal to the number of pieces summed over all functions. This number can be significantly decreased if the interval pieces over the different functions μ, ρ under some interval $j \in \mathbb{I}$ align. We use the same variable formulation $\vec{a}_j^l, g_j^l$ as defined in (4.13), and the feasible set $\mathcal{F}$ remains the same. Consider an interval $j \in \mathbb{I}$ and the set of points where some function in the subtrees at j changes piece. This set of points divides the interval $[\alpha_j, \beta_j]$ into pieces. Let $\mathbb{P}_j$ be an index set into these pieces. The size of $\mathbb{P}_j$ is clearly bounded by the sum of pieces in functions in subtrees at j. We introduce a Boolean variable $b_j^{\gamma,l}, c_j^{\gamma,\tau}$ for each $\gamma \in \mathbb{P}_j$, $l \in [k_j]$, and $\tau \in [k_j - 1]$, representing whether $\vec{a}_j^l, g_j^\tau$ fall into the interval representing piece $\gamma \in \mathbb{P}_j$, respectively. When $b_j^{\gamma,l}(c_j^{\gamma,\tau}) = 1$, we restrict $\vec{a}_j^l(g_j^\tau)$ as follows:

$$\alpha_j^\gamma \cdot b_j^{\gamma,l} \leq \vec{a}_j^l, \quad \vec{a}_j^l \leq \beta_j^\gamma + (\beta_j - \beta_j^\gamma) \cdot b_j^{\gamma,l}$$

The sum $\sum_{\gamma \in \mathbb{P}_j} b_j^{\gamma,l} = 1$ ensures that only one interval is chosen. The constraints for $c_j^{\gamma,\tau}, g_j^\tau$ are completely analogous. For each leaf node z with piecewise payoff function μ_z^γ, and $l \in [k_j]$, we can then introduce price

variables $p_z^l, p_z^l(\vec{a}_j^l), p_z^l(g_j^l) \in \mathbb{R}$, with the latter two for the interval $\left[\vec{a}_j^l, g_j^l\right]$, and constrain them linearly as follows:

$$\mathcal{P} = \left\{ \begin{array}{ll} p_z^l \geq p_z^l(g_j^l) - p_z^l(\vec{a}_j^l) & \\ p_z^l(g_j^l) \geq \mu_z^\gamma(g_j^l) - M \cdot c_j^{\gamma,\tau} & \forall \gamma \in \mathbb{P}_j \\ p_z^l(\vec{a}_j^l) \leq \mu_z^\gamma(\vec{a}_j^l) + M \cdot b_j^{\gamma,l} & \forall \gamma \in \mathbb{P}_j \end{array} \right\} \tag{4.16}$$

This is correct for sufficiently large $M \in \mathbb{R}$. We now have variables p_z^l representing each error function for the discrete points, and can apply the linear constraints from (4.15) to get a linear representation of the overall error. Thus we get the following MIP, where e_{r} is the objective value at the root according to (4.15):

$$\min \left\{ e_{\mathrm{r}} : e_{\mathrm{r}} \in \mathcal{E}, \left(\mathbb{A}'_v, g_v\right) \in \mathcal{F} \cap \mathcal{P}, b_j^{\gamma,l}, c_j^{\gamma,\tau} \in \{0,1\} \right\} \tag{4.17}$$

4.3.7 Applications

In this section, we discuss some applications of our results. We will focus on three recent problems that have included continuity in their problem formulation, or discretized away continuity: robust policy optimization under parameter uncertainty Chen and Bowling [2012], security games Yin and Tambe [2012], Kiekintveld et al. [2011], Marecki et al. [2012], and sequential wifi-jamming under battery constraints DeBruhl et al. [2014].

Chen and Bowling [2012] propose the use of zero-sum extensive-form game solving as a way of tractably computing optimally robust policies for Markov Decision Processes (MDPs) with parameter uncertainty. They design robustness criteria that can be implemented via nature first sampling parameter instances, and an opponent then choosing the worst of these. This sampling by nature was necessary in order to get games where the action space for the players is finite. Now, with our discretization-quality results, it is possible to use a broader class of robustness measures that allow continuous action spaces, while obtaining solution quality bounds.

Several papers have investigated continuous settings for security games. These have been for single-shot Kiekintveld et al. [2011], Yin and Tambe [2012] or repeated Bayesian Stackelberg games Marecki et al. [2012]. Since our framework is for the more general setting of extensive-form games, our solution-quality bounds apply to all these settings. Furthermore, they would also apply to more sophisticated models that include both sequential actions and imperfect information. Marecki et al. [2012] mention as future work the setting where the follower also behaves strategically. Our results immediately yield solution quality bounds for discretizations for this setting.

Another area with continuity is wifi jamming. In recent work, sequential-interaction models were introduced for this domain DeBruhl et al. [2014]. These models employ discretized action spaces, where both the jammer and transmitter have a (small) finite set of possible power levels to transmit at. However, this is an abstraction of reality, where software-defined radios mean that action spaces can be continuous (at least up to the bit-precision of the hardware). Using the techniques developed in this paper, we can give solution quality bounds on the utility loss obtained from considering only a discrete number of possible power levels (possibly by padding the game tree with dummy actions to satisfy Definition 15). DeBruhl et al. [2014] also mention that in a more realistic model, both transmitter and jammer would be modeled as observing only noisy signals of the actions taken by the other player. Since these observations would be of a continuum, the noise would likewise be continuous. The discretization quality bounds derived here would immediately apply to this setting.

4.3.8 Differences to abstraction practice in poker

We have already discussed how our framework can be used to give theoretical bounds on solution quality in practical scenarios. In particular, we showed that a uniform discretization is optimal for linear error functions (for discretizing nature this required a uniform distribution over the continuous action space). This stands somewhat in contrast to how practical abstractions are created for poker.

Consider no-limit Texas hold'em (NLHE). This game is the premier testbed for (discrete) extensive-form game solving algorithms [Sandholm, 2010]. Each year, the Annual Computer Poker Competition is held, where research groups submit highly-tuned poker-playing programs. The winning programs are based on computing Nash equilibrium approximations in abstractions of the full extensive-form game [Sandholm, 2010].

In NLHE, at each betting step, the acting player may bet any amount from the minimum bet to their entire stack of chips. To handle this action space, the top agents devise betting abstractions. These are completely analogous to the discretizations considered in this paper. The payoff functions under the subtrees are all linear in the specific actions chosen. At a cursory glance, one might say that Theorem 15 suggests that the optimal discretization would be uniform. However, the discretizations employed by the poker bots are more akin to a geometric progression. For example, Brown et al. [2015] describe using a betting abstraction consisting of $0.5, 0.75, 1, 1.5, 2$ and 5 times the pot size, as well as the all-in action. At the start of betting, all-in is approximately 133 times the pot size. Both examples and experiments by Ganzfried and Sandholm [2013] support the idea that the uniform mapping is not optimal.

A potential explanation for this is that the subtrees reached for different choices of bet size technically do not fall under the constraints of Definition 15. Consider a group of bets, say raising in the range of $[1, 2]$ (here we consider continuous bets in this small continuous range due to ease of exposition, one can construct similar examples with larger integer ranges) with a stack size of 2.5. The subtree where the player bets 2 exists as a subset of the subtree at every other betsize. These subsets all map to each other in a way that obeys Definition 15. However, if the player bets 1 instead, the opponent may reraise by 1, in which case the agent can call. This scenario does not exist when betting 2, as the player already bet her entire stack. For any two bet sizes $a_1 < a_2$, these discrepancies due to extra actions exist. To resolve this issue, one could pad the tree with extra actions such that Definition 15 is satisfied, but it is unclear how this would interact with the solution-quality bound, and could thus lead to nonoptimality of the uniform discretization.

4.3.9 Conclusions

We analyzed the problem of developing solution-quality bounds for discretization of extensive-form games with continuous action spaces. To our knowledge, we developed the first such bounds. We developed bounds for a very general class of continuous games: ones where there is a finite set of prototype trees, such that the instantiation of a given prototype has its nature probability outcomes and utilities parameterized by the choice on the continuous intervals. We developed bounds both where they depend on the specific Nash equilibrium (Theorem 14) and where they are independent of the Nash equilibrium chosen (Corollary 1).

We then considered the problem of computing bound-minimizing discretizations. First we developed very general bound-minimization formulations that allow a broad class of error-bounding functions. We discussed how such functions can be minimized in polynomial time when they are convex. We then considered the more specific problem of minimizing our bound developed for Corollary 1. For the case where all utility error and nature probability error functions are Lipschitz continuous (without additional structure), and nature chooses uniformly over each continuous interval, we showed that the bound-minimizing solution is to discretize uniformly. For the case where the error functions at individual nodes can be represented

by convex functions, we showed how to generate a convex optimization formulation of the overall problem. We also developed a MIP for piecewise linear error functions, which can also be used for arbitrarily accurate approximation. We also showed how the problem can be decomposed into separately optimizable components.

Ganzfried and Sandholm [2013] dicuss randomized mappings, where each real point has a probability distribution over which of its nearest discrete points it maps to. Their experiments strongly suggest that such randomization is desirable. Incorporating randomized mappings in our work could potentially lead to better discretizations, almost certainly in practice, and potentially also in theory. We leave this as future research.

4.4 Proposed work

While the above lines of work have provided much-need theoretical foundations for the area of abstraction in EFGs, there is still a lot of work to do in order to bridge the gap to practical abstraction work. All algorithms mentioned in the preceding section require either 1) solving an integer program whose size is on the order of magnitude of the game tree (this can be improved to just the signal tree for games of ordered signals) [Kroer and Sandholm, 2014], or 2) solving for abstractions level-by-level with prohibitively strong constraints on the type of abstraction allowed [Kroer and Sandholm, 2016a].

The main issue facing our theoretical framework (as well as that of Lanctot et al. [2012]) is that when we consider a pair of information sets I, I' for merging, we require some form of mapping between the nodes in the two information sets. In this mapping, we require that any pair of nodes $s \in I, s' \in I'$ that map to each other have the same path of information set-action pairs, over all players for perfect recall (this is, at least, the most obvious way to satisfy Definition 9), over the opponents in the past and over all players in the future for imperfect recall (explicitly required by Definition 13).

An example from a small poker game can elucidate this. Consider a poker game with three cards: king, queen, and jack. For an information set representing us holding a king, there will be nodes representing our opponent holding a queen or jack. Similarly if we are holding a queen, our opponent has to have a king or jack. If we want to merge our two information sets for king and queen, it is clear that the two jack nodes can map to each other. But if we want to satify the constraints in Definitions 9 and 13, the king and queen nodes cannot map to each other immediately, as they are on different information set-action pair paths for our opponent. Instead, we are then required to also perform this abstraction for our opponent. In a bigger game, these constraints end up making the space of feasible abstractions very small.

We propose developing practical methods for computing abstractions, while retaining bounds on the solution quality. We propose two different directions that could potentially achieve this:

- By allowing additional worst-case assumptions about paths such as the ones described above. This will allow a trade-off between 1) allowing more flexible abstraction that violates the prior constraints, and 2) deteriorating the solution quality bound, as these violations would lead to worse bounds. This will greatly expand the set of feasible abstractions, thereby potentially making the computational problem easier. In the best-case scenario, this new framework will allow analysis of a general level-by-level abstraction algorithm. Either this will be achieved by analyzing existing level-by-level algorithms such as Ganzfried and Sandholm [2014] or by developing new algorithms that explicitly use the bound quality as an objective.
- Making stronger assumptions about the game structure. This would be similar to the work on games of ordered signals [Gilpin and Sandholm, 2007b], but would require potentially slightly stronger assumptions (Section 4.1.5 suggests that stronger assumptions will be necessary). One concrete direction is to exploit just the private signals of the game for abstraction, since private signals are easier to prove

bounds for.

Depending on our success in developing such a scheme, we will attempt one or more of:

- Compute the first strategy with a meaningful bound on regret in NLTHE. Ideally, this would be achieved by first computing some abstraction with bounded solution quality, and then applying a scalable equilibrium-finding algorithm to the abstraction. Alternatively, we may focus on developing methodologies for ex-post bounded regret computation by developing a hybrid best-response and abstraction approach.
- Show that existing practical abstraction algorithms compute solutions with bounds on solution quality.
- If we are unable to develop a practical algorithm with bounds, we will also explore whether a solution-quality impossibility result can be proven about the type of abstraction that is employed in practice.

Chapter 5

Limited lookahead in sequential games

5.1 Introduction

Limited lookahead has been a central topic in AI game playing for decades. To date, it has been studied in single-agent settings and perfect-information games—specifically in well-known games such as chess, checkers, Go, etc., as well as in random game tree models [Korf, 1990, Pearl, 1981, 1983, Nau, 1983, Nau et al., 2010, Bouzy and Cazenave, 2001, Ramanujan et al., 2010, Ramanujan and Selman, 2011]. In this paper, we initiate the game-theoretic study of limited lookahead in imperfect-information games. Such games are more broadly applicable to practical settings—for example auctions, negotiations, security, cybersecurity, and medical settings—than perfect-information games. Mirrokni et al. [2012] conducted a game-theoretic analysis of lookahead, but they consider only perfect-information games, and the results are for four specific games rather than broad classes of games. Instead, we analyze the questions for imperfect information and general-sum extensive-form games.

As is typical in the literature on limited lookahead in perfect-information games, we derive our results for a two-agent setting. One agent is a rational player (Player R) trying to optimally exploit a limited-lookahead player (Player L). Our results extend immediately to one rational player and more than one limited-lookahead player, as long as the latter all break ties according to the same scheme (statically, favorably, or adversarially—as described later in the paper). This is because such a group of limited-lookahead players can be treated as one from the perspective of our results.

The type of limited-lookahead player we introduce is analogous to that in the literature on perfect-information games. Specifically, we let the limited-lookahead player L have a node evaluation function h that places numerical values on all nodes in the game tree. Given a strategy for the rational player, at each information set at some depth i, Player L picks an action that maximizes the expected value of the evaluation function at depth $i + k$, assuming optimal play between those levels. Our study is the game-theoretic, imperfect-information generalization of lookahead questions studied in the literature and we believe this makes it interesting in its own right. However, the model also has applications such as biological games, where the goal is to steer an evolution or adaptation process (which typically acts myopically with lookahead 1) [Sandholm, 2015] and security games where opponents are often assumed to be myopic (as makes sense when the number of adversaries is large [Yin et al., 2012]). Furthermore, investigating how well a rational player can exploit a limited-lookahead player lends insight into the limitations of using limited-lookahead algorithms in multiagent decision making.

We then design algorithms for finding an optimal strategy to commit to for the rational player. We focus on this rather than equilibrium computation because the latter seems nonsensical in this setting: the limited-

lookahead player determining a Nash equilibrium strategy would require her to reason about the whole game for the rational player's strategy, which rings contrary to the limited-lookahead assumption. Computing optimal strategies to commit to in standard rational settings has previously been studied in normal-form games [Conitzer and Sandholm, 2006] and extensive-form games [Letchford and Conitzer, 2010], the latter implying some complexity results for our setting as we will discuss.

As in the literature on lookahead in perfect-information games, a potential weakness of our approach is that we require knowing the evaluation function h (but make no other assumptions about what information h encodes). In practice, this function may not be known. As in the perfect-information setting, this can lead to the rational exploiter being exploited if their model of h is sufficiently wrong.

5.2 Model of limited lookahead

We now describe our model of limited lookahead. We use the term optimal *hypothetical* play to refer to the way the limited-lookahead agent thinks she will play when looking ahead from a given information set. In actual play part way down that plan, she may change her mind because she will then be able to see to a deeper level of the game tree.

Let k be the lookahead of Player L, and $S^k_{I,a}$ the nodes at lookahead depth k below information set I that are reachable (through some path) by action a. As in prior work in the perfect-information game setting, Player L has a node-evaluation function $h : S \to \mathbb{R}$ that assigns a heuristic numerical value to each node in the game tree.

Given a strategy σ_R for the other player and fixed action probabilities for Nature, Player L chooses, at any given information set $I \in \mathcal{I}_L$ at depth i, a (possibly mixed) strategy whose support is contained in the set of actions that maximize the expected value of the heuristic function at depth $i + k$, assuming optimal hypothetical play by her ($\max_{\sigma_l}$ in the formula below). We will denote this set by $A^*_I =$

$$\{a : a \in \arg\max_{a \in A_I} \max_{\sigma_L} \sum_{s \in I} \frac{\pi^{\sigma_{-L}}(s)}{\pi^{\sigma_{-L}}(I)} \sum_{s' \in S^k_{I,a}} \pi^{\sigma}(t^s_a, s')h(s')\},$$

where $\sigma = \{\sigma_L, \sigma_R\}$ is the strategy profile for the two players. Here moves by Nature are also counted toward the depth of the lookahead. The model is flexible as to how the rational player chooses σ_R and how the limited-lookahead player chooses a (possibly mixed) strategy with supports within the sets A^*_I. For one, we can have these choices be made for both players simultaneously according to the Nash equilibrium solution concept. As another example, we can ask how the players should make those choices if one of the players gets to make, and commit to, all her choices before the other.

5.3 Complexity

In this section we analyze the complexity of finding strategies according to these solution concepts.

5.3.1 Nash equilibrium

Finding a Nash equilibrium when Player L either has information sets containing more than one node, or has lookahead at least 2, is PPAD-hard [Papadimitriou, 1994]. This is because finding a Nash equilibrium in a 2-player general-sum normal-form game is PPAD-hard [Chen et al., 2009], and any such game can be

converted to a depth 2 extensive-form game (where the second player does not know what the first player moved), where the general-sum payoffs are the evaluation function values.

If the limited-lookahead player only has singleton information sets and lookahead 1, an optimal strategy can be trivially computed in polynomial time in the size of the game tree for the limited-lookahead player (without even knowing the other player's strategy σ_R) because for each of her information sets, we simply have to pick an action that has highest immediate heuristic value. To get a Nash equilibrium, what remains to be done is to compute a best response for the rational player, which can also be easily done in polynomial time [Johanson et al., 2011].

5.3.2 Commitment strategies

Next we study the complexity of finding commitment strategies. The complexity depends on whether the game has incomplete information (information sets that include more than one node) for the limited-lookahead player, how far that player can look ahead, and how she breaks ties in her action selection.

No information sets, lookahead 1, static tie-breaking As for the Nash equilibrium case, if the limited-lookahead player only has singleton information sets and lookahead 1, an optimal strategy can be trivially computed in polynomial time. We can use the same approach, except that the specific choice among the actions with highest immediate value is dictated by the tie-breaking rule. With this strategy in hand, finding a utility-maximizing strategy for Player R again consists of computing a best response.

No information sets, lookahead 1, adversarial tie-breaking When ties are broken adversarially, the choice of response depends on the choice of strategy for the rational player. The set of optimal actions A_s^* for any node $s \in S_L$ can be precomputed, since Player R does not affect which actions are optimal. Player L will then choose actions from these sets to minimize the utility of Player R. We can view the restriction to a subset of actions as a new game, where Player L is a rational player in a zero-sum game. An optimal strategy for Player R to commit to is then a Nash equilibrium in this smaller game. This is solvable in polynomial time by an LP that is linear in the size of the game tree [von Stengel, 1996], and algorithms have been developed for scaling to large games [Hoda et al., 2010, Zinkevich et al., 2007, Lanctot et al., 2009, Kroer and Sandholm, 2014, 2016a, Gilpin and Sandholm, 2007b, Ganzfried and Sandholm, 2014].

No information sets, lookahead 1, favorable tie-breaking In this case, Player L picks the action from A_s^* that maximizes the utility of Player R. Perhaps surprisingly, computing the optimal solution in this case is harder than when facing an adversarial opponent.

Theorem 16. *Computing a utility-maximizing strategy for the rational player to commit to is NP-hard if the limited-lookahead player breaks ties in favor of the rational player.*

No information sets, lookahead $>$ 1, favorable tie-breaking It is NP-hard to compute an optimal strategy to commit to in extensive-form games when both players are rational [Letchford and Conitzer, 2010]. That was proven by reducing from knapsack to a 2-player perfect-information game of depth 4. This immediately gives us two results: (1) finding an optimal strategy for Player R to commit to is NP-hard if Player L has lookahead at least 4, and (2) computing an optimal strategy to commit to for Player L is NP-hard even with lookahead 1. Their result also implies NP-hardness of computing a strategy to commit to for the rational player, if our L player has lookahead of at least 4. We tighten this to lookahead 2:

Theorem 17. *Computing a utility-maximizing strategy for the rational player to commit to is NP-hard if the limited lookahead player has lookahead at least* 2*.*

Limited-lookahead player has information sets When the limited lookahead player has information sets, we show that computing a strategy to commit to is NP-hard:

Theorem 18. *Computing a utility-maximizing strategy for the rational player to commit to is NP-hard if the limited lookahead player has information sets of at least size* 6.

5.4 Algorithms

In this section we will develop an algorithm for solving the hard commitment-strategy case. Naturally its worst-case runtime is exponential. As mentioned in the introduction, we focus on commitment strategies rather than Nash equilibria because Player L playing a Nash equilibrium strategy would require that player to reason about the whole game for the opponent's strategy. Further, optimal strategies to commit to are desirable for applications such as biological games [Sandholm, 2015] (because evolution is responding to what we are doing) and security games [Yin et al., 2012] (where the defender typically commits to a strategy).

Since the limited-lookahead player breaks ties adversarially, we wish to compute a strategy that maximizes the worst-case best response by the limited-lookahead player. For argument's sake, say that we were given $\mathcal{A}$, which is a fixed set of pairs, one for each information set I of the limited-lookahead player, consisting of a set of optimal actions A_I^* and one strategy for hypothetical play σ_l^I at I. Formally, $\mathcal{A} = \bigcup_{I \in \mathcal{I}_l} \langle A_I^*, \sigma_l^I \rangle$. To make these actions optimal for Player L, Player R must choose a strategy such that all actions in $\mathcal{A}$ are best responses according to the evaluation function of Player L. Formally, for all action triples $a, a^* \in \mathcal{A}, a' \notin \mathcal{A}$ (letting $\pi(s)$ denote probabilities induced by σ_L^I for the hypothetical play between I, a and s):

$$\sum_{s \in S_{I,a}^k} \pi(s) \cdot h(s) > \sum_{s \in S_{I,a'}^k} \pi(s) \cdot h(s) \tag{5.1}$$

$$\sum_{s \in S_{I,a}^k} \pi(s) \cdot h(s) = \sum_{s \in S_{I,a^*}^k} \pi(s) \cdot h(s) \tag{5.2}$$

Player R chooses a worst-case utility-maximizing strategy that satisfies (5.1) and (5.2), and Player L has to compute a (possibly mixed) strategy from $\mathcal{A}$ such that the utility of Player R is minimized. This can be solved by a linear program:

Theorem 19. *For some fixed choice of actions $\mathcal{A}$, Nash equilibria of the induced game can be computed in polynomial time by a linear program that has size $O(|S|) + O(\sum_{I \in \mathcal{I}_L} |A_I| \cdot \max_{s \in S} |A_s|^k)$.*

To prove this theorem, we first design a series of linear programs for computing best responses for the two players. We will then use duality to prove the theorem statement.

In the following, it will be convenient to change to matrix-vector notation, analogous to that of von Stengel [1996], with some extensions. Let $A = -B$ be matrices describing the utility function for Player R and the adversarial tie-breaking of Player L over $\mathcal{A}$, respectively. Rows are indexed by Player R sequences, and columns by Player L sequences. For sequence form vectors x, y, the objectives to be maximized for the players are then xAy, xBy, respectively. Matrices E, F are used to describe the sequence form constraints for Player R and L, respectively. Rows correspond to information sets, and columns correspond to sequences. Letting e, f be standard unit vectors of length $|\mathcal{I}_R|, |\mathcal{I}_L|$, respectively, the constraints $Ex = e, Fy = f$ describe the sequence form constraint for the respective players. Given a strategy x for Player R satisfying (5.1) and (5.2) for some $\mathcal{A}$, the optimization problem for Player L becomes choosing a vector of y' representing probabilities for all sequences in $\mathcal{A}$ that minimize the utility of Player R. Letting a prime superscript denote the restriction of each matrix and vector to sequences in $\mathcal{A}$, this gives the following primal (5.3) and dual (5.4) LPs:

$$\begin{aligned} \max_{y'} \quad & (x^T B')y' \\ & F'y' = f' \\ & y \geq 0 \end{aligned} \tag{5.3}$$

$$\begin{aligned} \min_{q'} \quad & q'^T f' \\ & q'^T F' \geq x^T B' \end{aligned} \tag{5.4}$$

where q' is a vector with $|\mathcal{A}| + 1$ dual variables. Given some strategy y' for Player L, Player R maximizes utility among strategies that induce $\mathcal{A}$. This gives the following best-response LP for Player R:

$$\begin{aligned} \max_{x} \quad & x^T(Ay') \\ & x^T E^T = e^T \\ & x \geq 0 \\ & x^T H_{\neg\mathcal{A}} - x^T H_{\mathcal{A}} \leq -\epsilon \\ & x^T G_{\mathcal{A}^*} = x^T G_{\mathcal{A}} \end{aligned} \tag{5.5}$$

where the last two constraints encode (5.1) and (5.2), respectively. The dual problem uses the unconstrained vectors p, v and constrained vector u and looks as follows

$$\begin{aligned} \min_{p,u,v} \quad & e^T p - \epsilon \cdot u \\ E^T p + (H_{\neg\mathcal{A}} - H_{\mathcal{A}})u + (G_{\mathcal{A}^*} - G_{\mathcal{A}})v & \geq A'y' \\ u & \geq 0 \end{aligned} \tag{5.6}$$

We can now merge the dual (5.4) with the constraints from the primal (5.5) to compute a minimax strategy: Player R chooses x, which she will choose to minimize the objective of (5.4),

$$\begin{aligned} \min_{x,q'} \quad & q'^T f' \\ q'^T F' - x^T B' & \geq 0 \\ -x^T E^T & = -e^T \\ x & \geq 0 \\ x^T H_{\mathcal{A}} - x^T H_{\neg\mathcal{A}} & \geq \epsilon \\ x^T G_{\mathcal{A}} - x^T G_{\mathcal{A}^*} & = 0 \end{aligned} \tag{5.7}$$

Taking the dual of this gives

$$\begin{aligned} \max_{y',p} \quad & -e^T p + \epsilon \cdot u \\ -E^T p + (H_{\mathcal{A}} - H_{\neg\mathcal{A}})u + (G_{\mathcal{A}} - G_{\mathcal{A}^*})v & \leq B'y' \\ F'y' & = f' \\ y, u & \geq 0 \end{aligned} \tag{5.8}$$

The theorem follows by a duality proof on (5.7) and (5.8).

In reality we are not given $\mathcal{A}$. To find a commitment strategy for Player R, we could loop through all possible structures $\mathcal{A}$, solve LP (5.7) for each one, and select the one that gives the highest value. We now introduce a mixed-integer program (MIP) that picks the optimal induced game $\mathcal{A}$ while avoiding enumeration. The MIP is given in (5.9). We introduce Boolean sequence-form variables that denote making

sequences suboptimal choices. These variables are then used to deactivate subsets of constraints, so that the MIP branches on formulations of LP (5.7), i.e., what goes into the structure $\mathcal{A}$. The size of the MIP is of the same order as that of LP (5.7).

$$\begin{aligned}
\min_{x,q,z} \quad & q^T f \\
& q^T F \geq x^T B - zM \\
& Ex = e \\
& x^T H_{\mathcal{A}} \geq x^T H_{\neg\mathcal{A}} + \epsilon - (1-z)M \\
& x^T G_{\mathcal{A}} = x^T G_{\mathcal{A}^*} \pm (1-z)M \\
& \sum_{a \in A_I} z_a \geq z_{a'} \\
& x \geq 0, \quad z \in \{0,1\}
\end{aligned} \tag{5.9}$$

The variable vector x contains the sequence form variables for Player R. The vector q is the set of dual variables for Player L. z is a vector of Boolean variables, one for each Player L sequence. Setting $z_a = 1$ denotes making the sequence a an inoptimal choice. The matrix M is a diagonal matrix with sufficiently large constants (e.g. the smallest value in B) such that setting $z_a = 1$ deactivates the corresponding constraint. Similar to the favorable-lookahead case, we introduce sequence form constraints $\sum_{a \in A_I} z_a \geq z_{a'}$ where a' is the parent sequence, to ensure that at least one action is picked when the parent sequence is active. We must also ensure that the incentivization constraints are only active for actions in $\mathcal{A}$:

$$\begin{aligned}
x^T H_{\mathcal{A}} - x^T H_{\neg\mathcal{A}} &\geq \epsilon - (1-z)M \\
x^T G_{\mathcal{A}} - x^T G_{\mathcal{A}^*} &= 0 \pm (1-z)M
\end{aligned} \tag{5.10}$$

for diagonal matrices M with sufficiently large entries. Equality is implemented with a pair of inequality constraints $\{\leq, \geq\}$, where $\pm$ denotes adding or subtracting, respectively.

The values of each column constraint in (5.10) is implemented by a series of constraints. We add Boolean variables $\sigma_L^I(I', a')$ for each information set action pair I', a' that is potentially chosen in hypothetical play at I. Using our regular notation, for each a, a' where a is the action to be made dominant, the constraint is implemented by:

$$\sum_{s \in S_{I,a}^k} v^i(s) \geq v_I^d(I), \quad v^i(s) \leq \sigma_L^I(I', a') \cdot M \tag{5.11}$$

where the latter ensures that $v^i(s)$ is only non-zero if chosen in hypothetical play. We further need the constraint $v^i(s) \leq \pi_{-L}^{\sigma}(s) h(s)$ to ensure that $v^i(s)$, for a node s at the lookahead depth, is at most the heuristic value weighted by the probability of reaching s.

5.5 Experiments

In this section we experimentally investigate how much utility can be gained by optimally exploiting a limited-lookahead player. We conduct experiments on Kuhn poker [Kuhn, 1950], a canonical testbed for game-theoretic algorithms, and a larger simplified poker game that we call KJ. Kuhn poker consists of a three-card deck: king, queen, and jack. Each player antes 1. Each player is then dealt one of the three cards, and the third is put aside unseen. A single round of betting ($p = 1$) then occurs. In KJ, the deck consists of

two kings and two jacks. Each player antes 1. A private card is dealt to each, followed by a betting round ($p = 2$), then a public card is dealt, follower by another betting round ($p = 4$). If no player has folded, a showdown occurs. For both games, each round of betting looks as follows:

- Player 1 can check or bet p.
  - If Player 1 checks Player 2 can check or raise p.
    - If Player 2 checks the betting round ends.
    - If Player 2 raises Player 1 can fold or call.
      - If Player 1 folds Player 2 takes the pot.
      - If Player 1 calls the betting round ends.
  - If Player 1 raises Player 2 can fold or call.
    - If Player 2 folds Player 1 takes the pot.
    - If Player 2 calls the betting round ends.

In Kuhn poker, the player with the higher card wins in a showdown. In KJ, showdowns have two possible outcomes: one player has a pair, or both players have the same private card. For the former, the player with the pair wins the pot. For the latter the pot is split. Kuhn poker has 55 nodes in the game tree and 13 sequences per player. The KJ game tree has 199 nodes, and 57 sequences per player.

To investigate the value that can be derived from exploiting a limited-lookahead opponent, a node evaluation heuristic is needed. In this work we consider heuristics derived from a Nash equilibrium. For a given node, the heuristic value of the node is simply the expected value of the node in (some chosen) equilibrium. This is arguably a conservative class of heuristics, as a limited-lookahead opponent would not be expected to know the value of the nodes in equilibrium. Even with this form of evaluation heuristic it is possible to exploit the limited-lookahead player, as we will show. We will also consider Gaussian noise being added to the node evaluation heuristic, more realistically modeling opponents who have vague ideas of the values of nodes in the game. Formally, let σ be an equilibrium, and i the limited-lookahead player. The heuristic value $h(s)$ of a node s is:

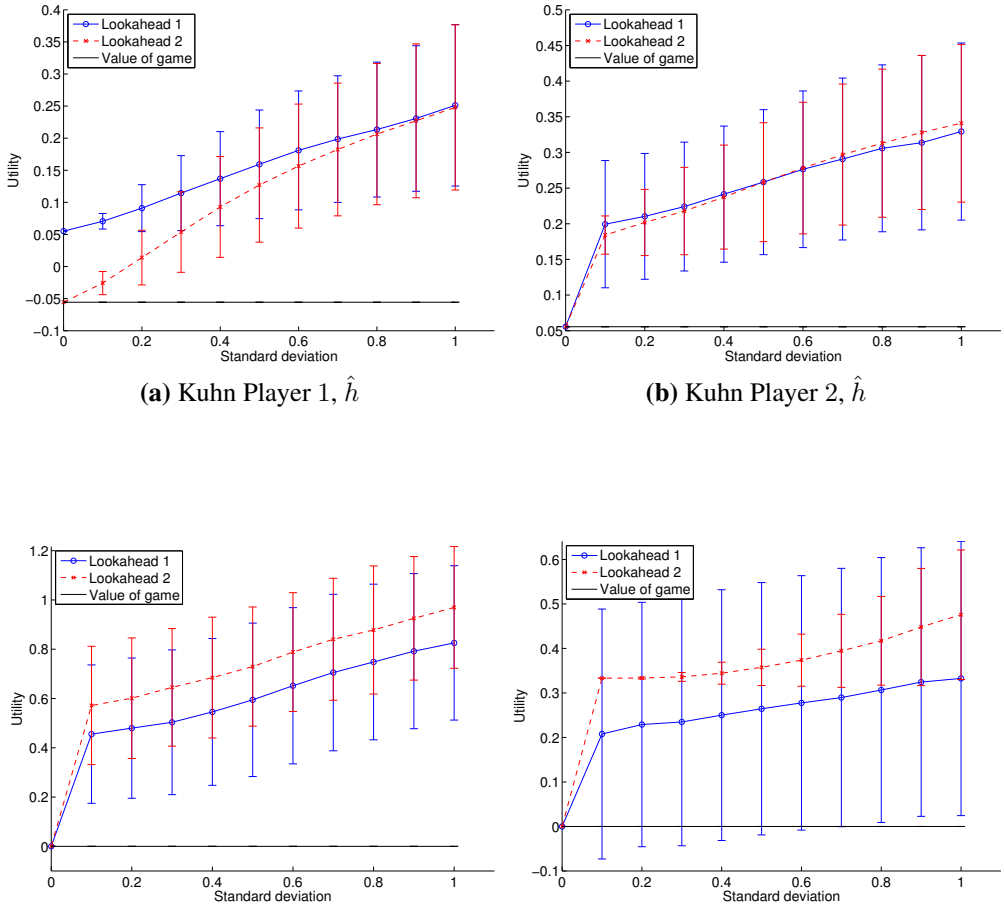
$$h(s) = \begin{cases} u_i(s) & \text{if } s \in Z \\ \sum_{a \in A_s} \sigma(s,a) h(t_a^s) & \text{otherwise} \end{cases} \tag{5.12}$$

We consider two different noise models. The first adds Gaussian noise with mean 0 and standard deviation γ independently to each node evaluation, including leaf nodes. Letting μ_s be a noise term drawn from $\mathcal{N}(0,\gamma)$: $\hat{h}(s) = h(s) + \mu_s$. The second, more realistic, model adds error cumulatively, with no error on leaf nodes:

$$\bar{h}(s) = \begin{cases} u_i(s) & \text{if } s \in Z \\ \left[\sum_{a \in A_s} \sigma(s,a) \bar{h}(t_a^s)\right] + \mu_s & \text{otherwise} \end{cases} \tag{5.13}$$

Using MIP (5.9), we computed optimal strategies for the rational player in Kuhn poker and KJ. The MIP models were solved by CPLEX version 12.5. The results are given in Figure 5.1. The x-axis is the noise parameter γ for $\hat{h}$ and $\bar{h}$. The y-axis is the corresponding utility for the rational player, averaged over at least 1000 runs per tuple $\langle$game, choice of rational player, lookahead, standard deviation$\rangle$. Each figure contains plots for the limited-lookahead player having lookahead 1 or 2, and a baseline for the value of the game in equilibrium without limited lookahead.

Figures 5.1a and b show the results for using evaluation function $\hat{h}$ in Kuhn poker, with the rational player in plot a and b being Player 1 and 2, respectively. For rational Player 1, we see that, even with no noise in the heuristic (i.e., the limited-lookahead player knows the value of each node in equilibrium), it is



(a) Kuhn Player 1, $\hat{h}$

(b) Kuhn Player 2, $\hat{h}$

(c) KJ Player 1, $\hat{h}$

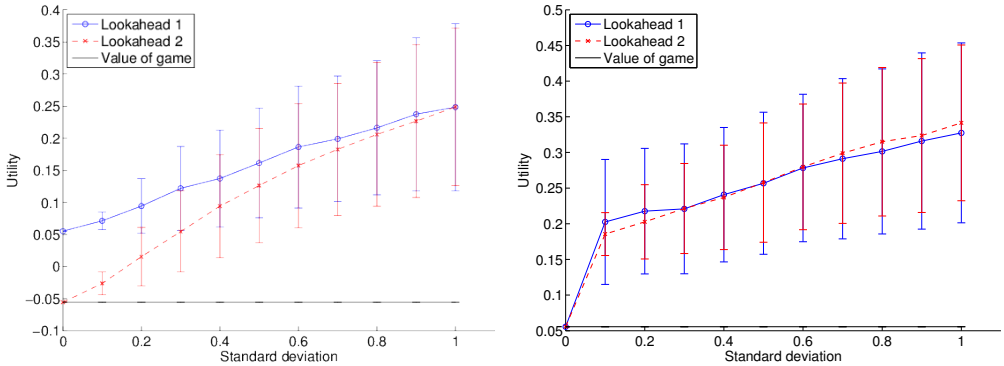
(d) KJ Player 2, $\hat{h}$

Figure 5.1: Winnings in Kuhn poker and KJ for the rational player as Player 1 and 2, respectively, for varying per-node independent evaluation function noise. Error bars show standard deviation.

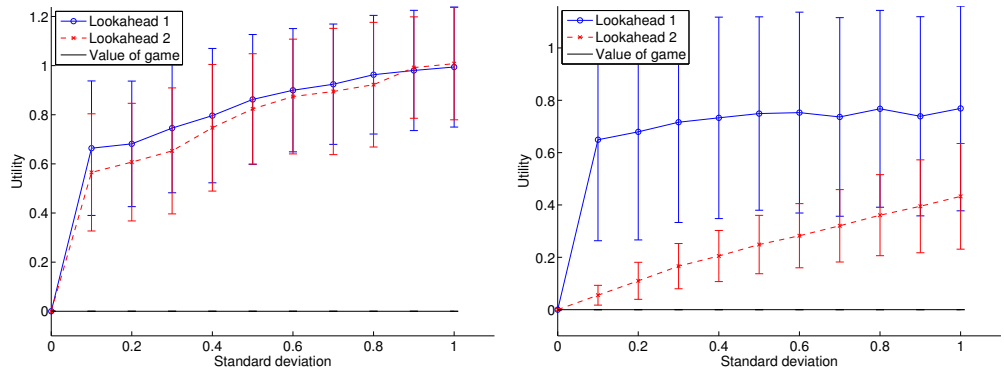
possible to exploit the limited-lookahead player if she has lookahead 1. (With lookahead 2 she achieves the value of the game.) For both amounts of lookahead, the exploitation potential steadily increases as noise is added.

Figures 5.1c and d show the same variants for KJ. Here, lookahead 2 is worse for the limited-lookahead player than lookahead 1. To our knowledge, this is the first known imperfect-information *lookahead pathology*. Such pathologies are well known in perfect-information games [Beal, 1980, Pearl, 1981, Nau, 1983], and understanding them remains an active area of research [Luštrek et al., 2006, Nau et al., 2010, Wilson et al., 2012]. This version of the node heuristic does not have increasing *visibility*: node evaluations do not get more accurate toward the end of the game. Our experiments on KJ with $\bar{h}$ in Figures 5.2 g and h do not have this pathology, and $\bar{h}$ does have increasing visibility.

Figure 5.3 shows a simple subtree (that could be attached to any game tree) where deeper lookahead can make the agent's decision arbitrarily bad, even when the node evaluation function is the exact expected



(a) Kuhn Player 1, $\bar{h}$ **(b)** Kuhn Player 2, $\bar{h}$



(c) KJ Player 1, $\bar{h}$ **(d)** KJ Player 2, $\bar{h}$

Figure 5.2: Winnings in Kuhn poker and KJ for the rational player as Player 1 and 2, respectively, for varying cumulative evaluation function noise. Error bars show standard deviation.

value of a node in equilibrium.

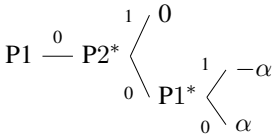


Figure 5.3: A subtree that exhibits lookahead pathology.

We now go over the example of Figure 5.3. Assume without loss of generality that all payoffs are positive in some game. We can then insert the subtree in Figure 5.3 as a subgame at any node belonging to P1, and it will be played with probability 0 in equilibrium, since it has expected value 0. Due to this, all strategies where Player 2 chooses up can be part of an equilibrium. Assuming that P2 is the limited-lookahead player

and minimizing, for large enough α, the node labeled P1* will be more desirable than any other node in the game, since it has expected value $-\alpha$ according to the evaluation function. A rational player P1 can use this to get P2 to go down at P2*, and then switch to the action that leads to α. This example is for lookahead 1, but we can generalize the example to work with any finite lookahead depth: the node P1* can be replaced by a subtree where every other leaf has payoff 2α, in which case P2 would be forced to go to the leaf with payoff α once down has been chosen at P2*.

Figures 5.2e and f show the results for Kuhn poker with $\bar{h}$. These are very similar to the results for $\hat{h}$, with almost identical expected utility for all scenarios. Figures 5.1g and h, as previously mentioned, show the results with $\bar{h}$ on KJ. Here we see no abstraction pathologies, and for the setting where Player 2 is the rational player we see the most pronounced difference in exploitability based on lookahead.

5.6 Conclusions and future work

This chapter initiated the study of limited lookahead in imperfect-information games. We characterized the complexity of finding a Nash equilibrium and optimal strategy to commit to for either player. Figure 5.4 summarizes those results, including the cases of favorable and static tie-breaking, the discussion of which we deferred to the extended online paper.

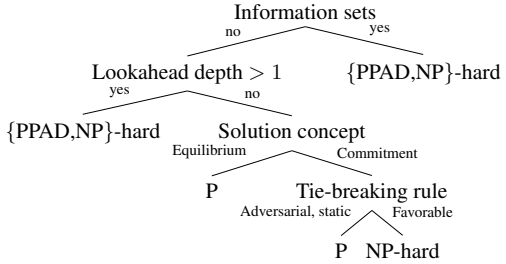


Figure 5.4: Our complexity results. {PPAD,NP}-hard indicates that finding a Nash equilibrium (optimal strategy to commit to) is PPAD-hard (NP-hard). P indicates polytime.

We then designed a MIP for computing optimal strategies to commit to for the rational player. The problem was shown to reduce to choosing the best among a set of two-player zero-sum games (the tie-breaking being the opponent), where the optimal strategy for any such game can be computed with an LP. We then introduced a MIP that finds the optimal solution by branching on these games.

We experimentally studied the impact of limited lookahead in two poker games. We demonstrated that it is possible to achieve large utility gains by exploiting a limited-lookahead opponent. As one would expect, the limited-lookahead player often obtains the value of the game if her heuristic node evaluation is exact (i.e., it gives the expected values of nodes in the game tree for some equilibrium)—but we provided a counterexample that shows that this is not sufficient in general. Finally, we studied the impact of noise in those estimates, and different lookahead depths. While lookahead 2 usually outperformed lookahead 1, we uncovered an imperfect-information game lookahead pathology: deeper lookahead can hurt the limited-lookahead player. We demonstrated how this can occur with any finite depth of lookahead, even if the limited-lookahead player's node evaluation heuristic returns exact values from an equilibrium.

Our algorithms in the NP-hard adversarial tie-breaking setting scaled to games with hundreds of nodes. For some practical settings more scalability will be needed. There are at least two exciting future directions toward achieving this. One is to design faster algorithms. The other is designing abstraction techniques for the limited-lookahead setting. As noted in Chapter 4, abstraction plays an important role in large-scale

game solving [Sandholm, 2010]. Limited-lookahead games have much stronger structure, especially locally around an information set, and it may be possible to utilize that to develop abstraction techniques with significantly stronger solution quality bounds. Also, leading practical game abstraction algorithms (e.g., [Ganzfried and Sandholm, 2014]), while theoretically unbounded, could immediately be used to investigate exploitation potential in larger games. Finally, uncertainty over h is an important future research direction. This would lead to more robust solution concepts, thereby alleviating the pitfalls involved with using an imperfect estimate.

5.7 Proposed work

The work developed in this chapter represents the first generalization of extensive-form game limited-lookahead research to the game-theoretic and imperfect-information setting. However, in order to be broadly applicable in practice, it is necessary to model uncertainty about the limited-lookahed behavior of the opponents. We propose extending the model developed in this chapter to include uncertainty over the node-evaluation function of the limited-lookahead player.

We will consider two types of uncertainty: uncertainty over a base set of functions $\{h_i : S \rightarrow \mathbb{R}\}$ sampled before the game begins, or a stochastic node-evaluation function $h : S \rightarrow X_s$, where X_s is an independent random variable representing the evaluation function at node s. The former is more likely to yield results; for example, we can model the uncertainty over the node-evaluation function as an initial chance-node appended to the game. The latter would lead to greater flexibility in modeling, and it could ideally be combined with the former. The latter, more general framework may be computationally difficult to handle if the goal is to maximize expected value. Expected value maximization has the drawback that it requires a distribution, and is therefore still unrealistic for some practical settings. Instead, we will investigate the use of robust optimization frameworks, which require fewer assumptions about the node-evaluation function, and are more likely to be practically computable. In particular, rather than try to maximize the expected payoff against some distributional assumption about the node-evaluation function, we will focus on showing robustness against any node-evaluation function from some uncertainty set. This will likely allow us to leverage algorithms for either robust convex or robust combinatorial optimization [Ben-Tal and Nemirovski, 2002, Ben-Tal et al., 2009, Mutapcic and Boyd, 2009, Bertsimas et al., 2011]

Chapter 6

Timeline

My proposed research will roughly be completed as follows:

January 2017: Proposal

January–February 2017: Large-scale equilibrium refinements (Section 3), submit to IJCAI

January–April 2017: Speeding up FOMs to be competitive with CFR at earlier stages of convergence (Section 3), submit first results to EC

May–July 2017: Show efficient implementation of FOMs on large games (Section 3)

June–August 2017: Robust optimization for uncertainty in limited lookahead (Section 5)

June–December 2017: Practical abstraction algorithms with solution-quality bounds (Section 4)

January–March 2018: Academic job interviews

March–April 2018: Writing my thesis document

May 2018: Thesis defense

Chapter 7

A brief overview of my non-thesis research

This chapter briefly surveys some of the research directions I have followed during my Ph.D. that are not included in my proposed thesis.

- **Prediction markets.** In my work on prediction markets [Kroer et al., 2016], we present a new combinatorial market maker that operates arbitrage-free combinatorial prediction markets specified by integer programs. Although the problem of arbitrage-free pricing, while maintaining a bound on the subsidy provided by the market maker, is #P-hard in the worst case, we posit that the typical case might be amenable to modern integer programming (IP) solvers. At the crux of our method is the Frank-Wolfe (conditional gradient) algorithm which is used to implement a Bregman projection aligned with the market maker's cost function, using an IP solver as an oracle. We demonstrate the tractability and improved accuracy of our approach on real-world prediction market data from combinatorial bets placed on the 2010 NCAA Men's Division I Basketball Tournament, where the outcome space is of size 2^{63}. To our knowledge, this is the first implementation and empirical evaluation of an arbitrage-free combinatorial prediction market on this scale.
- **Sequential planning for biological problems.** In Kroer and Sandholm [2016b] we study biological adaptation, a powerful mechanism that makes many disorders hard to combat. We study steering such adaptation through sequential planning. We propose a general approach where we leverage Monte Carlo tree search to compute a treatment plan, and the biological entity is modeled by a black-box simulator that the planner calls during planning. We show that the framework can be used to steer a biological entity modeled via a complex signaling pathway network that has numerous feedback loops that operate at different rates and have hard-to-understand aggregate behavior.
- **Computational bundling.** In Kroer and Sandholm [2015a], we study a common revenue-enhancement approach—bundling—in the context of the most commonly studied combinatorial auction mechanism, the Vickrey-Clarke-Groves (VCG) mechanism. A challenge in mechanism design for combinatorial auctions is that the prior distribution on each bidder's valuation can be doubly exponential. Such priors do not exist in most applications. Rather, in many applications (such as premium display advertising markets), there is essentially a point prior, which may not be accurate. We adopt the point prior model, and prove robustness to inaccuracy in the prior. Then, we present a branch-and-bound framework for finding the optimal bundling. We introduce several techniques for branching, upper bounding, lower bounding, and lazy bounding. Experiments on CATS distributions validate the approach and show that our techniques dramatically improve scalability over a leading general-purpose MIP solver.

Bibliography

ACPC. Annual Computer Poker Competition website. http://www.computerpokercompetition.org/, 2016. [Online; accessed 23-Feb-2016].

C. Archibald and Y. Shoham. Modeling billiards games. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, Budapest, Hungary, 2009.

Nicola Basilico and Nicola Gatti. Automated abstractions for patrolling security games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2011.

Donald F Beal. An analysis of minimax. *Advances in computer chess*, 2:103–109, 1980.

Amir Beck and Marc Teboulle. Smoothing and first order methods: A unified framework. *SIAM Journal on Optimization*, 22(2):557–580, 2012.

Aharon Ben-Tal and Arkadi Nemirovski. Robust optimization–methodology and applications. *Mathematical Programming*, 92(3):453–480, 2002.

Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*. Princeton University Press, 2009.

Ahron Ben-Tal and Arkadi Nemirovski. *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*, volume 2. Siam, 2001.

Dimitris Bertsimas, David B Brown, and Constantine Caramanis. Theory and applications of robust optimization. *SIAM review*, 53(3):464–501, 2011.

Darse Billings, Neil Burch, Aaron Davidson, Robert Holte, Jonathan Schaeffer, Terence Schauenberg, and Duane Szafron. Approximating game-theoretic optimal strategies for full-scale poker. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

Kellog S Booth and Charles J Colbourn. *Problems polynomially equivalent to graph isomorphism*. Computer Science Department, Univ., 1979.

B Bosansky, Christopher Kiekintveld, V 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.

Bruno Bouzy and Tristan Cazenave. Computer go: an ai oriented survey. *Artificial Intelligence*, 132(1): 39–103, 2001.

Michael Bowling, Neil Burch, Michael Johanson, and Oskari Tammelin. Heads-up limit hold'em poker is solved. *Science*, 347(6218):145–149, January 2015.

Noam Brown and Tuomas Sandholm. Regret transfer and parameter optimization. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2014.

Noam Brown and Tuomas Sandholm. Simultaneous abstraction and equilibrium finding in games. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.

Noam Brown, Sam Ganzfried, and Tuomas Sandholm. Hierarchical abstraction, distributed equilibrium computation, and post-processing, with application to a champion no-limit Texas Hold'em agent. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2015.

Neil Burch, Marc Lanctot, Duane Szafron, and Richard G Gibson. Efficient monte carlo counterfactual regret minimization in games with many player actions. In *Advances in Neural Information Processing Systems*, pages 1880–1888, 2012.

Katherine Chen and Michael Bowling. Tractable objectives for robust policy optimization. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2012.

Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of computing two-player Nash equilibria. *Journal of the ACM*, 2009.

Vincent Conitzer and Tuomas Sandholm. Computing the optimal strategy to commit to. In *Proceedings of the ACM Conference on Electronic Commerce (ACM-EC)*, Ann Arbor, MI, 2006.

Constantinos Daskalakis, Alan Deckelbaum, and Anthony Kim. Near-optimal no-regret algorithms for zero-sum games. *Games and Economic Behavior*, 92:327–348, 2015.

Bruce DeBruhl, Christian Kroer, Anupam Datta, Tuomas Sandholm, and Patrick Tague. Power napping with loud neighbors: optimal energy-constrained jamming and anti-jamming. In *Proceedings of the 2014 ACM conference on Security and privacy in wireless & mobile networks*, pages 117–128. ACM, 2014.

Gabriele Farina and Nicola Gatti. Extensive-form perfect equilibrium computation in two-player games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

Tomás Feder and Daniel Greene. Optimal algorithms for approximate clustering. In *Proceedings of the Annual Symposium on Theory of Computing (STOC)*, 1988.

Joaquim Gabarró, Alina García, and Maria Serna. On the complexity of game isomorphism. In *Mathematical Foundations of Computer Science 2007*, pages 559–571. Springer, 2007.

Sam Ganzfried and Tuomas Sandholm. Tartanian5: A heads-up no-limit Texas Hold'em poker-playing program. In *Computer Poker Symposium at the National Conference on Artificial Intelligence (AAAI)*, 2012.

Sam Ganzfried and Tuomas Sandholm. Action translation in extensive-form games with large action spaces: Axioms, paradoxes, and the pseudo-harmonic mapping. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2013.

Sam Ganzfried and Tuomas Sandholm. Potential-aware imperfect-recall abstraction with earth mover's distance in imperfect-information games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2014.

Richard Gibson, Marc Lanctot, Neil Burch, Duane Szafron, and Michael Bowling. Generalized sampling and variance in counterfactual regret minimization. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2012.

Andrew Gilpin and Tuomas Sandholm. A competitive Texas Hold'em poker player via automated abstraction and real-time equilibrium computation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 1007–1013, 2006.

Andrew Gilpin and Tuomas Sandholm. Better automated abstraction techniques for imperfect information games, with application to Texas Hold'em poker. In *International Conference on Autonomous Agents*

and Multi-Agent Systems (AAMAS), pages 1168–1175, 2007a.

Andrew Gilpin and Tuomas Sandholm. Lossless abstraction of imperfect information games. *Journal of the ACM*, 54(5), 2007b.

Andrew Gilpin and Tuomas Sandholm. Lossless abstraction of imperfect information games. *Journal of the ACM*, 54(5), 2007c. Early version 'Finding equilibria in large sequential games of imperfect information' appeared in the Proceedings of the ACM Conference on Electronic Commerce (EC), pages 160–169, 2006.

Andrew Gilpin and Tuomas Sandholm. Expectation-based versus potential-aware automated abstraction in imperfect information games: An experimental comparison using poker. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2008. Short paper.

Andrew Gilpin, Tuomas Sandholm, and Troels Bjerre Sørensen. Potential-aware automated abstraction of sequential games, and holistic equilibrium analysis of Texas Hold'em poker. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2007.

Andrew Gilpin, Tuomas Sandholm, and Troels Bjerre Sørensen. A heads-up no-limit Texas Hold'em poker player: Discretized betting models and automatically generated equilibrium-finding programs. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2008.

Andrew Gilpin, Javier Peña, and Tuomas Sandholm. First-order algorithm with $\mathcal{O}(\ln(1/\epsilon))$ convergence for ϵ-equilibrium in two-person zero-sum games. *Mathematical Programming*, 133(1–2):279–298, 2012. Conference version appeared in AAAI-08.

Teofilo F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38, 1985.

Anupam Gupta, Guru Guruganesh, and Melanie Schmidt. Approximation algorithms for aversion k-clustering via local k-median. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP*, pages 66:1–66:13, 2016.

Joseph Y Halpern and Rafael Pass. Sequential equilibrium in computational games. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 171–176. AAAI Press, 2013.

John Hawkin, Robert Holte, and Duane Szafron. Automated action abstraction of imperfect information extensive-form games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2011.

John Hawkin, Robert Holte, and Duane Szafron. Using sliding windows to generate action abstractions in extensive-form games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2012.

Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Fundamentals of convex analysis*. 2001.

Samid Hoda, Andrew Gilpin, Javier Peña, and Tuomas Sandholm. Smoothing techniques for computing Nash equilibria of sequential games. *Mathematics of Operations Research*, 35(2):494–512, 2010. Conference version appeared in WINE-07.

Albert Jiang and Kevin Leyton-Brown. Polynomial-time computation of exact correlated equilibrium in compact games. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, 2011.

Michael Johanson, Kevin Waugh, Michael Bowling, and Martin Zinkevich. Accelerating best response calculation in large extensive games. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

Michael Johanson, Nolan Bard, Neil Burch, and Michael Bowling. Finding optimal abstract strategies in extensive-form games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2012.

Michael Johanson, Neil Burch, Richard Valenzano, and Michael Bowling. Evaluating state-space abstractions in extensive-form games. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2013.

Anatoli Juditsky and Arkadi Nemirovski. First order methods for nonsmooth convex large-scale optimization, i: general purpose methods. *Optimization for Machine Learning*, pages 121–148, 2011a.

Anatoli Juditsky and Arkadi Nemirovski. First order methods for nonsmooth convex large-scale optimization, ii: utilizing problems structure. *Optimization for Machine Learning*, pages 149–183, 2011b.

Anatoli Juditsky, Arkadi Nemirovski, and Claire Tauvel. Solving variational inequalities with stochastic mirror-prox algorithm. *Stochastic Systems*, 1(1):17–58, 2011.

Christopher Kiekintveld, Janusz Marecki, and Milind Tambe. Approximation methods for infinite bayesian stackelberg games: modeling distributional payoff uncertainty. In *10th International Conference on Autonomous Agents and Multiagent Systems AAMAS 2011*, 2011.

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

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

Richard E Korf. Real-time heuristic search. *Artificial intelligence*, 42(2):189–211, 1990.

Christian Kroer and Tuomas Sandholm. Extensive-form game abstraction with bounds. In *Proceedings of the ACM Conference on Economics and Computation (EC)*, 2014.

Christian Kroer and Tuomas Sandholm. Computational bundling for auctions. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2015a. Extended version: Carnegie Mellon University, Computer Science Department technical report CMU-CS-13-111, 2013.

Christian Kroer and Tuomas Sandholm. Discretization of continuous action spaces in extensive-form games. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2015b.

Christian Kroer and Tuomas Sandholm. Limited lookahead in imperfect-information games. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2015c.

Christian Kroer and Tuomas Sandholm. Imperfect-recall abstractions with bounds in games. In *Proceedings of the ACM Conference on Economics and Computation (EC)*, 2016a.

Christian Kroer and Tuomas Sandholm. Sequential planning for steering immune system adaptation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2016b.

Christian Kroer, Kevin Waugh, Fatma Kılınç-Karzan, and Tuomas Sandholm. Faster first-order methods for extensive-form game solving. In *Proceedings of the ACM Conference on Economics and Computation (EC)*, 2015.

Christian Kroer, Miroslav Dudík, Sébastien Lahaie, and Sivaraman Balakrishnan. Arbitrage-free combinatorial market making via integer programming. In *Proceedings of the 2016 ACM Conference on Economics and Computation, EC '16, Maastricht, The Netherlands, July 24-28, 2016*, pages 161–178, 2016.

H. W. Kuhn. A simplified two-person poker. In H. W. Kuhn and A. W. Tucker, editors, *Contributions to the Theory of Games*, volume 1 of *Annals of Mathematics Studies, 24*, pages 97–103. Princeton University Press, Princeton, New Jersey, 1950.

Marc Lanctot, Kevin Waugh, Martin Zinkevich, and Michael Bowling. Monte Carlo sampling for regret

minimization in extensive games. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1078–1086, 2009.

Marc Lanctot, Richard Gibson, Neil Burch, Martin Zinkevich, and Michael Bowling. No-regret learning in extensive-form games with imperfect recall. In *International Conference on Machine Learning (ICML)*, 2012.

Joshua Letchford and Vincent Conitzer. Computing optimal strategies to commit to in extensive-form games. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, 2010.

Richard Lipton, Evangelos Markakis, and Aranyak Mehta. Playing large games using simple strategies. In *Proceedings of the ACM Conference on Electronic Commerce (ACM-EC)*, pages 36–41, San Diego, CA, 2003. ACM.

Viliam Lisy and Michael Bowling. Eqilibrium approximation quality of current no-limit poker bots. In *AAAI Computer Poker Workshop*, 2017.

Viliam Lisy, Trevor Davis, and Michael Bowling. Counterfactual regret minimization in sequential security games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2016.

Michael Littman and Peter Stone. A polynomial-time Nash equilibrium algorithm for repeated games. In *Proceedings of the ACM Conference on Electronic Commerce (ACM-EC)*, pages 48–54, San Diego, CA, 2003.

Mitja Luštrek, Matjaž Gams, and Ivan Bratko. Is real-valued minimax pathological? *Artificial Intelligence*, 170(6):620–642, 2006.

Janusz Marecki, Gerald Tesauro, and Richard Segal. Playing repeated stackelberg games with unknown opponents. In *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012*, 2012.

Peter Bro Miltersen and Troels Bjerre Sørensen. Fast algorithms for finding proper strategies in game ttrees. In *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2008.

Peter Bro Miltersen and Troels Bjerre Sørensen. Computing a quasi-perfect equilibrium of a two-player game. *Economic Theory*, 42(1):175–192, 2010.

Vahab Mirrokni, Nithum Thain, and Adrian Vetta. A theoretical examination of practical game playing: lookahead search. In *Algorithmic Game Theory*, pages 251–262. Springer, 2012.

Boris S Mordukhovich, Javier F Peña, and Vera Roshchina. Applying metric regularity to compute a condition measure of a smoothing algorithm for matrix games. *SIAM Journal on Optimization*, 20(6):3490–3511, 2010.

Enrique Munoz de Cote, Ruben Stranders, Nicola Basilico, Nicola Gatti, and Nick Jennings. Introducing alarms in adversarial patrolling games. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 1275–1276. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

Almir Mutapcic and Stephen Boyd. Cutting-set methods for robust convex optimization with pessimizing oracles. *Optimization Methods & Software*, 24(3):381–406, 2009.

Dana S Nau. Pathology on game trees revisited, and an alternative to minimaxing. *Artificial intelligence*, 21 (1):221–244, 1983.

Dana S Nau, Mitja Luštrek, Austin Parker, Ivan Bratko, and Matjaž Gams. When is it better not to look ahead? *Artificial Intelligence*, 174(16):1323–1338, 2010.

Arkadi Nemirovski. Prox-method with rate of convergence o (1/t) for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15(1):229–251, 2004.

Yurii Nesterov. Excessive gap technique in nonsmooth convex minimization. *SIAM Journal of Optimization*, 16(1):235–249, 2005a.

Yurii Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103:127–152, 2005b.

Christos H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and system Sciences*, 48(3):498–532, 1994.

François Pays. An interior point approach to large games of incomplete information. In *AAAI Computer Poker Workshop*, 2014.

Judea Pearl. Heuristic search theory: Survey of recent results. In *IJCAI*, volume 1, pages 554–562, 1981.

Judea Pearl. On the nature of pathology in game searching. *Artificial Intelligence*, 20(4):427–453, 1983.

Bezalel Peleg, Joachim Rosenmüller, and Peter Sudhölter. *The canonical extensive form of a game form: Symmetries*. Springer, 1999.

Raghuram Ramanujan and Bart Selman. Trade-offs in sampling-based adversarial planning. In *ICAPS*, pages 202–209, 2011.

Raghuram Ramanujan, Ashish Sabharwal, and Bart Selman. On adversarial search spaces and sampling-based planning. In *ICAPS*, volume 10, pages 242–245, 2010.

I. Romanovskii. Reduction of a game with complete memory to a matrix game. *Soviet Mathematics*, 3: 678–681, 1962.

Tuomas Sandholm. The state of solving large incomplete-information games, and application to poker. *AI Magazine*, pages 13–32, Winter 2010. Special issue on Algorithmic Game Theory.

Tuomas Sandholm. Steering evolution strategically: Computational game theory and opponent exploitation for treatment planning, drug design, and synthetic biology. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2015. Senior Member Track.

Tuomas Sandholm and Satinder Singh. Lossy stochastic game abstraction with bounds. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, 2012.

David Schnizlein, Michael Bowling, and Duane Szafron. Probabilistic state translation in extensive games with large action sets. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, 2009.

Jiefu Shi and Michael Littman. Abstraction methods for game theoretic poker. In *CG '00: Revised Papers from the Second International Conference on Computers and Games*, pages 333–345, London, UK, 2002. Springer-Verlag.

Finnegan Southey, Michael Bowling, Bryce Larson, Carmelo Piccione, Neil Burch, Darse Billings, and Chris Rayner. Bayes' bluff: Opponent modelling in poker. In *Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 550–558, July 2005.

Peter Sudhölter, Joachim Rosenmüller, and Bezalel Peleg. The canonical extensive form of a game form: Part II. representation. *Journal of Mathematical Economics*, 33(3):299–338, 2000.

Oskari Tammelin, Neil Burch, Michael Johanson, and Michael Bowling. Solving heads-up limit Texas

hold'em. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.

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

Kevin Waugh and Drew Bagnell. A unified view of large-scale zero-sum equilibrium computation. In *Computer Poker and Imperfect Information Workshop at the AAAI Conference on Artificial Intelligence (AAAI)*, 2015.

Kevin Waugh, David Schnizlein, Michael Bowling, and Duane Szafron. Abstraction pathologies in extensive games. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2009a.

Kevin Waugh, Martin Zinkevich, Michael Johanson, Morgan Kan, David Schnizlein, and Michael Bowling. A practical use of imperfect recall. In *Symposium on Abstraction, Reformulation and Approximation (SARA)*, 2009b.

Kevin Waugh, Dustin Morrill, Drew Bagnell, and Michael Bowling. Solving games with functional regret estimation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2015.

Michael P. Wellman, Daniel M. Reeves, Kevin M. Lochner, Shih-Fen Cheng, and Rahul Suri. Approximate strategic reasoning through hierarchical reduction of large symmetric games. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2005.

Brandon Wilson, Inon Zuckerman, Austin Parker, and Dana S Nau. Improving local decisions in adversarial search. In *ECAI*, pages 840–845, 2012.

Z Yin, A Jiang, M Tambe, C Kietkintveld, K Leyton-Brown, T Sandholm, and J Sullivan. TRUSTS: Scheduling randomized patrols for fare inspection in transit systems using game theory. *AI Magazine*, 2012.

Zhengyu Yin and Milind Tambe. A unified method for handling discrete and continuous uncertainty in bayesian stackelberg games. In *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012*, 2012.

Martin Zinkevich, Michael Bowling, Michael Johanson, and Carmelo Piccione. Regret minimization in games with incomplete information. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2007.