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

Christian Kroer

CMU-CS-18-118
September 14, 2018

School of Computer Science
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA

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

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

This research was sponsored by a Facebook Fellowship, Microsoft, the U.S. Army under grant numbers W911NF1610061 and W911NF1710082, and the National Science Foundation under grant numbers CCF-1101668, IIS-1320620, IIS-1546752, and IIS-1717590. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

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

To Janine, Niels, my parents, and my dog.

Abstract

Game-theoretic solution concepts provide a sound notion of rational behavior 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. Equilibrium computation in large-scale extensive-form games typically relies on two complementary methods: abstraction and iterative equilibrium-finding algorithms. We present new algorithmic and structural results for both methods.

For abstraction, we develop new theoretical guarantees on the solution quality of equilibria computed in abstractions. We establish 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 in the case of algorithm-specific guarantees, our approach leads to exponentially stronger bounds than prior results, and extend to more general games and abstractions.

For equilibrium computation, we focus on two-player zero-sum Nash equilibrium computation via convex optimization. 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, and more general, than bounds for prior smoothing methods. This leads to the state-of-the-art in convergence rate for iterative methods for computing a Nash equilibrium. In particular, we obtain the first convergence rate that generalizes the well-known logarithmic dependence on dimension in the matrix-game setting. In GPU-based experiments on large-scale real-world games we show that our methods lead to a convergence rate that beats all but the fastest practical method. We extend our smoothing approach to the computation of approximate Nash equilibrium refinements as well.

Finally, we investigate a number of new extensive-form game models. We develop new solution concepts and associated algorithmic results for games where opponents have limited lookahead. We then initiate the study of robust Stackelberg extensive-form games. We develop algorithms for computing solutions and classify the computational complexity of the problem.

Acknowledgments

First of all thanks to my advisor Tuomas Sandholm. Tuomas introduced me to the world of extensive-form games and shaped many of the types of questions that I am now fond of. At the same time he also gave me the freedom to pursue new research topics as my interests developed and matured over time. His marathon paper-writing sessions taught me most of what I know on how to write research papers. Tuomas has also been invaluable in helping me build my network and meet people in the field.

I am greatly indebted to Fatma Kılınç-Karzan, who had an enormous impact on both my thesis as well as my general research interests. Fatma spurred my interest in first-order methods, and taught me many of the things I know about mathematical optimization in general. Fatma is also an incredibly conscientious coauthor. Vince Conitzer has been incredibly helpful and friendly throughout my entire degree, as a source of advice, inspiration, and deep research discussions. Geoff Gordon has been helpful as one of the rare experts in both first-order methods and extensive-form games; I also enjoyed my occasional attendance at his research meetings. Finally, I would like to thank Yurii Nesterov not only for serving on my committee, but for inventing many of the foundations that my thesis rests on.

Beyond my work at CMU, I am fortunate to have spent excellent summers interning at industry research groups. Thanks to Sébastien Lahaie, Miro Dudík, and Dave Pennock for a great summer spent at MSR NYC. Thanks to Nico Stier and Eric Sodomka for a great internship at Facebook Core Data Science, but also for continued collaborations and support.

My interest in research, and algorithms, AI, and optimization more generally, was sparked by a number of excellent teachers, collaborators, and advisors during my years at the IT University of Copenhagen. Rune M. Jensen was instrumental both in me entering research, as well as my search for a doctoral program in the US. Thore Husfeldt sparked my love for algorithms and hardness proofs. Joe Kiniry gave me invaluable advice and help on getting into PhD programs. Kevin Tierney and Yuri Malitsky gave me guidance during my first forays into research. Finally Adam Britt, Niv Dayan, and Martin K. Svendsen deserve thanks for starting research projects with me during those early years.

Thanks to my collaborators during the Ph.D., without whom this whole doctorate journey would have been much less fun to to take on: Noam Brown, Vince Conitzer, Anupam Datta, Bruce DeBruhl, Miroslav Dudík, Nicola Gatti, Nam Ho-Nguyen, Fatma Kılınç-Karzan, Sébastien Lahaie, George Lu, Alberto Marchesi Tuomas Sandholm, Eric Sodomka, Nico Stier, Patrick Tague, and Kevin Waugh.

Beyond my collaborators and committee, David Parkes, Ariel Procaccia, and Michael Wellman have been great sources of support, advice, and discussion.

I thank my group members for all the shared paper-writing nights over the group-special pizza and research discussions. Noam Brown deserves huge thanks for repeatedly answering my questions about poker bots, and helping nurture my gambling habit; John Dickerson for countless research advice, “travel funding” and a contin-

ued commitment to discussing the possibility of collaborating someday; Gabriele Farina for great collaboration, helping me appreciate not working in coordinates, and sharing my unhealthy addiction to math-book purchases; Sam Ganzfried for poker discussions and recommending Hello Bistro, the best salad in the world.

Outside my group I would like to thank: Brandon Amos for deep-learning-as-a-service; Nika Haghtalab for Canadian Thanksgiving and sharing job-market anxiety with me; Sid Jain for not burning down our house when we lived together; David Kurokawa for the pool sessions; Jay-Yoon for good times in the office, even though you abandoned us; Jamie Morgenstern for holding down the 9221 fort and for the gainz; Kevin Waugh for great collaborations and discussions on convex optimization; Anders Øland for all the rugbrød.

I would like to thank the excellent administrative staff at CMU: Charlotte Yano for years of great help, advice, and great discussions with little-to-no snark, Jessica Packer for always being super helpful and attentive, and making sure that my recommendation letters actually arrived, and Deb Cavlovich for somehow managing to keep me from being dropped from the Ph.D. program every time I forgot to register for things, and more importantly helping me navigate the maze that is Ph.D. requirements.

The most important thanks goes to my family: My wonderful wife Janine for forcing me to go to on adventures and seeing the real world, and for attempting to make me a more well-rounded person; my parents for their constant support and kindness toward Janine and me while we are an ocean apart, and for being the best parents one could ask for; my brother for reminiscing with me about the old times in Aalborg and watching me play Final Fantasy on the couch, ork satme!

Contents

List of Figures

List of Tables

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 [42, 48, 49, 54, 68, 81, 93, 112, 114, 165, 179]. 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 [113, 125], cybersecurity games [43, 51], recreational games such as poker [148] and billiards [2], and certain medical treatment settings [41]

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 [1, 28, 148]. 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 when using an ϵ-Nash equilibrium from the abstraction. Previous 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 cannot be computed, even in the abstraction. Instead, fast iterative methods are applied, which converge to a Nash equilibrium in the limit, while avoiding having to write down a model linear in the size of the game tree.

This thesis explores and develops theoretical foundations for all steps of the solution process described above. Chapter 3 develops state-of-the-art convergence rate bounds for fast iterative solvers. Chapter 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.

In some contexts Nash equilibrium may not be a satisfactory solution concept, for example it may not be sequentially rational, and thus fail to capitalize when opponents make mistakes. This can be partially ameliorated by using Nash equilibrium refinements: refinements retain the guarantees of a Nash equilibrium while giving additional rationality properties in subtrees that are not reached in equilibrium. Chapters 5 develops scalable algorithms for computing

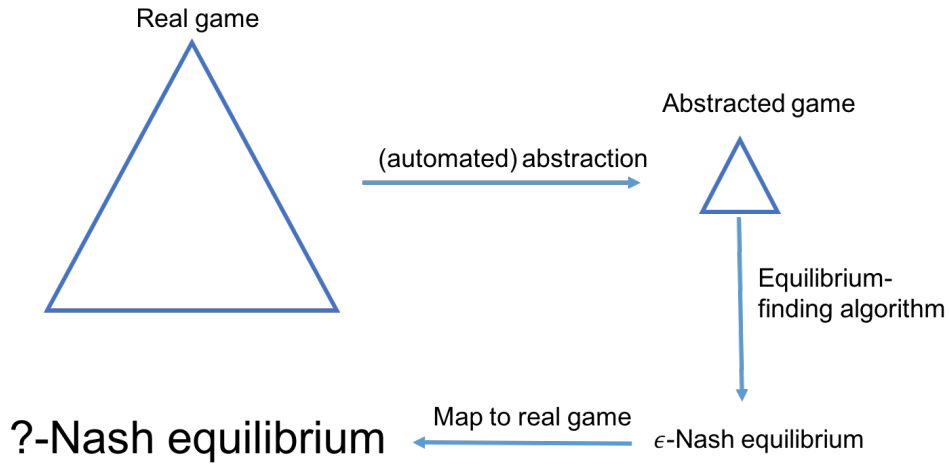


Figure 1.1: An overview of the general abstraction approach.

approximate variants of a particular type of Nash equilibrium refinement. Another context where Nash equilibrium is not ideal is when a rational player faces a myopic opponent, in which case Nash equilibrium is too pessimistic. Chapter 6 introduces the study of myopic opponents in EFGs and studies the problem of computing optimal strategies in such settings. *Stackelberg equilibria* are popular in settings where one player may credibly commit to a strategy, and let the opponent best respond to the strategy committed to. In such settings it is important the committing player optimizes against an accurate opponent model so that they do not overfit to an incorrect model. Chapter 7 initiates the study of robust opponent models in Stackelberg EFGs, and shows that robust solutions can be computed at no additional asymptotic cost. All alternative solution concepts are supported by experimental evidence investigating scalability and the performance of the solution concepts.

The next five sections will give a brief overview of the results obtained in each of the thesis chapters.

1.1 Algorithms for computing zero-sum Nash equilibria (Chapter 3)

This chapter focuses on developing iterative methods for large-scale equilibrium computation that have state-of-the-art theoretical convergence rate while being competitive with the best practical methods. These are methods that write down strategies for the players, while requiring only oracle access to the much larger game tree, usually for gradient 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 [165] for obtaining a bilinear saddle-point formulation, (2) convex minimization algorithms such as the excessive gap technique (EGT) [132] or mirror prox [130], and (3) smoothing techniques for the strategy spaces of the two players. The third part is cur-

rently not well understood.

A smoothing method in the context of an EFG can be thought of as a way to smoothly penalize moving away from some *center* strategy (usually the uniform strategy, but any other fully-mixed strategy can be used). Given a smoothing method, the idea is to define a *smoothed best-response function* (SBRF) where the opponent best responds to a given strategy, except that they optimize the sum of the expected payoff value plus a weighted penalty term. The SBRF smooths out the nonsmooth kinks associated with changes to the best response of the opponent. The general approach of smoothing out a bilinear saddle-point problem function was proposed by Nesterov [132, 133] and was since extended to other nonsmooth functions [8]. What we call a smoothing method can also be viewed as a *distance-generating function* (DGF); a distance measure between points can be obtained by taking the *Bregman divergence* between two points x_1, x_2: the error in the first-order approximation to the DGF at x_1 when taking the approximation at x_2. The DGF view was used by a number of later algorithms for solving BSPPs [39, 40, 44, 130, 134].

Hoda et al. [79] 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. [79] have a significantly worse dependence on the EFG size than counterfactual regret minimization (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 set. Our result leads to the strongest known convergence rate for iterative methods, achieving a dependence on game constants that is much stronger than that of Hoda et al. [79], while maintaining a $1/T$ convergence rate. The result can be informally stated as

Informal Theorem 1. *The dilated entropy function with simplex-weights chosen according a particular recurrence leads to a measure of distance for sequential action spaces whose strong-convexity parameter under the l_1 norm has no dependence on the branching factor of each decision point. This extends the suitability of the negative entropy function as a distance measure for simplexes to sequential action spaces.*

As a corollary of this theorem combined with, e.g., the EGT algorithm we get the state-of-the-art convergence rate for iterative methods on zero-sum perfect-recall EFGs.

While the chapter is on solving zero-sum EFGs the main theorem is more general than that: it can be used to smooth sequential decision spaces in a variety of settings, including games with more than two players or sequential decision making problems.

Furthermore, we introduce a new class of gradient estimators that allow instantiation of stochastic FOMs such as stochastic mirror prox [87]. Finally, we show experimentally that EGT instantiated with our smoothing technique leads to better practical convergence rates than CFR for medium-to-high accuracy solutions on medium-sized games. It is furthermore faster than CFR with every practical speedup modification, although CFR$^+$ is still faster. CFR$^+$ is a variant of the CFR algorithm where the *regret matching*$^+$ regret minimizer is used, a more aggressive stepsize scheme is invoked, and an alternation heuristic is used. The first two variations are guaranteed to preserve the $1/\sqrt{T}$ convergence rate of CFR [161], whereas the third variation may not be theoretically sound [59]. We introduce a new aggressive variant of EGT, and in-

vestigate the performance of EGT and our aggressive variant on large real-time subgames faced by a champion poker AI. These latter experiments represent the first comparison of FOMs and CFR methods on real large-scale problems. We find that for this setting EGT again outperforms all variants of CFR except CFR$^+$. Thus it outperforms all algorithms known to be theoretically sound, since we showed in Farina et al. [59] that CFR$^+$ is not proven to be theoretically sound due to its alternation heuristic.

The results on strong convexity and sampling were published as Kroer, Waugh, Kılınç-Karzan, and Sandholm [101], "Faster First-Order Methods for Extensive-Form Game Solving," at EC-2015, and Kroer, Waugh, Kılınç-Karzan, and Sandholm [103], "Theoretical and Practical Advances on Smoothing for Extensive-Form Games," at EC-2017. A journal version of these results is currently under submission. The aggressive EGT variant and GPU experiments will be published as Kroer, Farina, and Sandholm [105], "Solving Large Sequential Games with the Excessive Gap Technique" at NIPS-2018.

1.2 Abstraction for large general-sum 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 while retaining theoretical guarantees. Practical abstraction algorithms for EFGs have all been without any solution quality bounds [64, 66, 67, 69, 70]. This chapter ameliorates this fact by developing some of the first theoretical results on the quality of equilibria computed in abstractions of EFGs.

The central contribution of this chapter is a framework for measuring differences between a given abstraction and the original game. Based on this framework we prove two central theorems. The first relates ϵ-Nash equilibria in the abstraction to ϵ-Nash equilibria in the full game, and can be informally stated as:

Informal Theorem 2. *Given a perfect-recall EFG and a (potentially imperfect-recall) abstraction thereof, any ϵ-Nash equilibrium computed in the abstraction can be mapped to an ϵ'-Nash equilibrium in the original game. The equilibrium-approximation quality ϵ' can be decomposed into three terms: ϵ, a measure of how leaf-node payoffs differ, and a measure of how distributions over nodes differ.*

This theorem is the first of its kind in several ways. Most importantly, it is the first to establish an exact decomposition of abstraction error. Furthermore, for the first time in the literature, it gives abstraction-quality results for approximate Nash equilibria computed in lossy EFG abstractions.

Our second central theorem of the chapter gives a decomposition of the error associated with implementing a strategy profile computed in an abstraction of the full game, when that strategy profile has bounded counterfactual regret in the abstraction.

Informal Theorem 3. *Given a perfect-recall EFG and a (potentially imperfect-recall) abstraction thereof, any strategy profile with bounded counterfactual regret computed in the abstraction can be mapped to an ϵ'-Nash equilibrium in the original game. The equilibrium-approximation quality ϵ' can be decomposed into three terms: The counterfactual regrets in the abstraction, a*

measure of how leaf-node payoffs differ, and a measure of how distributions over nodes differ.

The only prior result that is somewhat comparable to our main theorems was for strategies with bounded counterfactual regret, as developed by Lanctot et al. [109]. That prior result does not give a decomposition, but rather a loose upper bound on the solution quality. In particular, we show that our decomposition results can be used to derive solution-quality bounds in the style of Lanctot et al. [109], but with a much stronger dependence on game constants. Their solution-quality bound has a linear dependence on the number of information sets in the game. Our result, depending on abstraction type, has either a logarithmic dependence on this quantity (more specifically a linear dependence on game height, which is logarithmic in the number of information sets), or no dependence at all. Our results are also very broad, while the prior result covers only a specific narrow class of abstractions.

Our decomposition result necessarily has a dependence on the specific strategy profile played in the abstraction as well as best responses in the full game. However, we show that for several specific game classes our decomposition yields bounds that are better-suited for ex-ante analysis.

This work will be published as Kroer and Sandholm [100], "A Unified Framework for Extensive-Form Game Abstraction with Bounds." at NIPS-2018.

We now describe our ex-ante oriented results for specific abstraction classes.

Perfect-recall abstraction. We develop the first algorithm-agnostic bounds on solution quality for equilibria computed in perfect-recall abstractions. The ex-ante bound that we derive has no dependence on game size when measuring payoff error, whereas the bound in Lanctot et al. [109] had a linear dependence on the number of information sets. 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.

The first version of this result was published as Kroer and Sandholm [95], "Extensive-form Game Abstraction with Bounds," at EC-2014; the version presented in this thesis is derived from our new decomposition result.

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 solutions computed in imperfect-recall abstractions, when implemented in the original (perfect-recall) game. Our ex-ante results are for a class of games that generalizes that of Lanctot et al. [109]. Further, our analysis is tighter in two ways, each of which can lead to an exponential reduction in the solution quality error

bound. The payoff error part of our ex ante bound for imperfect recall has a linear dependence on the depth of the game, whereas our ex ante bound for perfect recall had no such dependence. Thus imperfect-recall abstractions may have asymptotically worse error in the worst case, although it remains unclear whether this would occur in the types of abstractions computed in practice. Furthermore this part of our results may not be tight. 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. We also find that our bounds (when only performing single-level abstraction) are only about an order of magnitude off from the actual error when running CFR.

The first version of this result was published as Kroer and Sandholm [98], "Imperfect-Recall Abstractions with Bounds in Games," at EC-2016; the version presented in this thesis is derived from our new decomposition result.

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 any point in the continuous to its nearest point in the discretization, 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 as Kroer and Sandholm [96], "Discretization of Continuous Action Spaces in Extensive-Form Games."

1.3 Equilibrium refinement (Chapter 5)

In this chapter, we extend our smoothing method to computing Nash-equilibrium refinements in zero-sum EFGs. Polynomial-time algorithms are known for refinements such as normal-form proper and quasi-perfect equilibria [120, 121]. However, these algorithms rely on solving one or more linear programs, usually relying on modifications to the sequence-form LP of von Stengel [165]. As with Nash equilibria, such LP formulations are unlikely to be practical for large-scale games. Instead, we leverage a perturbed sequence-form polytope in the standard bilinear saddle-point formulation, and then focus on solving this saddle-point problem directly. To do this, we set up a perturbed variant of the dilated entropy distance function and show that our strong convexity results from Chapter 3 extend to perturbed sequence-form polytopes. For small-enough perturbations, this leads to an approximate Nash equilibrium refinement.

This work was published at IJCAI-2017 as Kroer, Farina, and Sandholm [102], "Smoothing Method for Approximate Extensive-Form Perfect Equilibrium." With the same authors I pub-

lished a related work on extending the CFR algorithm to a similar setting [58]; that work is not part of this thesis.

1.4 Limited lookahead (Chapter 6)

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) [150], and security games where opponents are often assumed to be myopic (as makes sense when the stakes are low such as in fare evasion [176]). 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.

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 as Kroer and Sandholm [97], "Limited Lookahead in Imperfect-Information Games."

1.5 Robust Stackelberg equilibria (Chapter 7)

Chapter 7 initiates the study of robust models in the context of Stackelberg EFGs. Stackelberg equilibria have gained importance as a solution concept in computational game theory, largely inspired by practical problems such as security settings (e.g. airport security or wildlife protection), where the leader is a defender who picks a mixed (i.e., potentially randomized) strategy first, and then the attacker, who is the follower, decides where to attack, if at all. Stackelberg games have been widely studied in single-shot settings, where uncertainty is considered an important problem [91, 136, 138, 160]. We extend the question of how to address uncertainty to the EFG setting and show that for the specific case of interval uncertainty around follower payoffs the problem of computing an optimal strategy for the leader can be solved with a MIP that is comparable in size to the MIPs used to compute leader strategies in the setting without follower

uncertainty. We show that this also holds experimentally: for several game classes we find that the runtime cost of taking uncertainty into account is only around one order of magnitude. We extend our result to the limited-lookahead setting as well, where we find that the addition of uncertainty limits the possibility for the leader to exploit a myopic adversary, although small amounts of uncertainty can still allow exploitation.

This work was published at AAAI-2018 as Kroer, Farina, and Sandholm [104], "Robust Stackelberg Equilibria in Extensive-Form Games and Extension to Limited Lookahead."

Chapter 2

Notation

This section introduces some general notation for EFGs that we will use throughout most of this proposal. Chapters 3 and 5 are largely independent of this notation, and so can safely be read in isolation. Chapters 4, 6, and 7 rely more heavily on the notation described here for formally describing the obtained results.

2.1 Extensive-form games

An *extensive-form game (EFG)* is a game tree, where each node in the tree corresponds to some history of actions taken by the players. Each node belongs to some player, and the actions available to the player at a given node are represented by the branches. Uncertainty is modeled by having a special player, *Chance*, that moves with some predefined fixed probability distribution over actions. EFGs model imperfect information by having groups of nodes in *information sets*, where an information set is a group of nodes all belonging to the same player such that the player cannot distinguish among them. In the original game that we are trying to solve, we assume *perfect recall*, which requires that no player forgets information they knew earlier in the game. This is a natural condition since you generally cannot force players to forget information, and it would not be in their interest to do so. Formally, an *extensive-form game* Γ is a tuple $(H, Z, A, P, \pi_0, \{\mathcal{I}_i\}, \{u_i\})$.

- H is the set of nodes in the game tree, corresponding to sequences (or histories) of actions. H_i is the subset of histories belonging to Player i.
- $Z \subseteq H$ is the set of terminal histories, or *leaves*.
- A is the set of actions in the game. A_I denotes the set of actions available at nodes in information set I.
- P, the player function, maps each non-terminal history $h \in H \setminus Z$ to $\{0, \ldots, n\}$, representing the player whose turn it is to move after history h. If $P(h) = 0$, the player is Chance.
- π_0 is a function that assigns to each $h \in H_0$ the probability of reaching h due to Chance (i.e., assuming that both players play to reach h). For any individual action $a \in A_h$ we let $\sigma_0(h, a)$ be the probability of Chance choosing this action ($\pi_0(h)$ is then the product of

action probabilities for Chance on the path to h).

- An information set $\mathcal{I}_i$, for $i \in \{1, \ldots, n\}$, is a partition of $\{h \in H : P(h) = i\}$.
- The utility function u_i maps $z \in Z$ to the utility obtained by player i when the terminal history is reached.

A *behavioral* strategy σ_i for a player i is a probability distribution over actions at each information set in $\mathcal{I}_i$. A *strategy profile* σ is a behavioral strategy for each player. The probability that σ puts on $a \in A_I$ is denoted $\sigma(I, a)$. We let $\pi^\sigma(z)$ and $\pi^\sigma(I)$ denote the probability of reaching z and I respectively, if players choose actions according to σ. We likewise let $\pi^\sigma(z|I)$ and $\pi^\sigma(\hat{I}|I)$ denote the reach probabilities conditioned on being at information set I. For a given strategy profile σ we let $\sigma_{I \to a}$ denote the same strategy except that $\sigma_{I \to a}(I, a) = 1$.

We will often quantify statements over the set of leaves or information sets that are reachable from some given information set I belonging to Player i, sometimes conditioned on taking a specific action $a \in A_I$. We let $\mathcal{Z}_I, \mathcal{D}_I \subset \mathcal{I}_i$ be the set of leaves and information sets reachable conditioned on being at information set I. We let Z_I and $\mathcal{C}_I \subset \mathcal{I}_i$ be the set of leaves and information sets that are reachable without Player i taking any further actions before reaching them. We let $\mathcal{Z}_I^a, \mathcal{D}_I^a, Z_I^a$ and $\mathcal{C}_I^a$ be defined analogously but conditioned on taking action $a \in A_I$.

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 $h_1, h_2 \in I$ if we let h_1^i, h_2^i denote the subset of the sequences of action that were taken by i then $h_1^i = h_2^i$. If perfect recall is not satisfied we say that 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. This leads to a host of computational problems, as evidenced by many problems such as best-response computation being significantly harder from a computational-complexity complexity perspective [92]. Most results in this thesis rely on the perfect-recall assumption (and indeed most natural game models would satisfy this). We will consider imperfect-recall *abstractions* of games in Section 4.6. Such abstraction can be advantageous for compactness of representation.

As is usual we use the subscript $-i$ to denote exclusion of Player i, for example, σ_{-i} is the set of behavioral strategies in σ except for the strategy of Player i, and $\pi_{-i}^\sigma(z)$ is the probability of reaching leaf node z disregarding actions taken by Player i, that is, assuming that Player i plays to reach z.

2.2 Equilibrium concepts

In this section we define two equilibrium concepts that we will use throughout the thesis. A few other solution concepts will be introduced when they are pertinent to a specific chapter. We start with the classic Nash equilibrium [127].

Definition 1 (Nash equilibrium)**.** *A Nash equilibrium in a game Γ with root node r is a strategy profile σ such that for each agent i, given σ_{-i}, σ_i is a utility maximizing strategy for i. In other words, for all i, $\bar{\sigma}_i$: $V_i^\sigma(r) \geq V_i^{\sigma_{-i}, \bar{\sigma}_i}(r)$.*

Intuitively, in a Nash equilibrium any individual player cannot gain any utility by unilaterally

deviating to a new strategy $\bar{\sigma}_i$, given the strategy of all the other players.

Sometimes Nash equilibria are too restrictive as a definition. This is particularly often true for algorithmic reasons, as the best practical algorithms only converge to a Nash equilibrium in the limit. Instead we need the following notion of an approximate Nash equilibrium:

Definition 2 (ϵ-Nash equilibrium)**.** *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*

In an ϵ-Nash equilibrium we have exactly the same property as in Nash equilibrium: if a player unilaterally deviates to another strategy then they can gain at most ϵ additional utility.

Chapter 3

Algorithms for computing zero-sum Nash equilibria

Practical EFG solving typically applies abstraction methods in order to get a smaller more manageable abstraction. Nonetheless, the resulting game is typically so large that exact approaches are impractical. This is because a larger (and thus more precise) abstraction coupled with an approximate solution typically performs better than a smaller (and thus less precise) abstraction with an exact solution. 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 [165]. 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 [23, 24, 65, 79, 83, 108, 179]. Recently, a sparse iterative solver, CFR+ [161], was used to practically solve limit texas holdem [21], 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 [179].[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 [83, 108, 161, 179]. In contrast to this, FOMs such as the excessive gap technique (EGT) [132] and mirror prox [130] achieve a convergence rate of $1/T$, with a cost per iteration that is only 2-3 times that of CFR, when instantiated with an appropriate smoothing technique for EFGs [79]. 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.

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. There are a number of efficient and well-known FOMs designed to solve BSPPs. The classical FOMs to solve BSPPs such as *mirror prox* (MP) [130] or

[1] Waugh and Bagnell [169] 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.

the *excessive gap technique* (EGT) [132] utilize *distance-generating functions* (DGFs) to measure appropriate notions of distances over the domains. Consequently, the convergence rate of these FOMs relies on the DGFs and their relation to the domains in three critical ways: Through the strong convexity parameters of the DGFs, the norm associated with the strong convexity parameter, and the set widths of the domains as measured by the DGFs.

Hoda et al. [79] introduced a general framework for constructing DGFs for *treeplexes*—a class of convex polytopes that generalize the domains associated with the strategy spaces of an EFG. While they also established lower bounds on the strong convexity parameter for their DGFs in some special cases, these lead to very weak bounds and result in slow convergence rates. In a work that is preliminary to the form presented here [101] we developed explicit strong convexity-parameter bounds for entropy-based DGFs (a particular subclass of DGFs) for general EFGs, and improved the bounds for the special cases considered by Hoda et al. [79]. Prior to this thesis there were no completely-developed bounds for the iteration complexity of FOMs using DGFs based on dilating simplex distance functions for general EFGs.

In this chapter 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. These bounds are simultaneously the first general bounds, and stronger than the only prior bounds which were for a narrower class of EFGs that have a particular uniform structure. Our new bounds are first-of-their kind as they have only a logarithmic dependence on the branching operation of the treeplex. In terms of their logarithmic dependence on the branching factor, our bounds parallel the simplex case for matrix games where the entropy function achieves a logarithmic dependence on the dimension of the simplex domain.

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 CFR$^+$ [161], which have a theoretical convergence rate of $O(\frac{1}{\epsilon^2})$. CFR$^+$ is the fastest prior algorithm for computing Nash equilibria in EFGs when the entire tree can be traversed (rather than sampled). Bowling et al. [21] used it to essentially solve the game limit Texas hold'em. CFR+ is also the algorithm used to accurately solve endgames in the Libratus agent, which showed superhuman performance against a team of top Heads-Up No-Limit Texas hold'em poker specialist professional players in the Brains vs AI event[2]. A slight variation[3] of CFR+ was used in the DeepStack agent Moravčík et al. [124], which beat a group of professional players.

We perform numerical experiments on scaled-up variants of *Leduc hold'em* [157], a poker game that has become a standard benchmark in the EFG-solving community, a security-inspired attacker/defender game played on a graph, and large-scale subgames encountered by the Libratus agent [27]. The performance we get from our FOM-based approach with EGT relative to CFR and CFR$^+$ is in contrast to the previous conventional practical wisdom in the field. Previously it was thought that FOM-based methods converged faster than CFR ultimately, but that CFR had a faster initial convergence and the cross-over point occurred later as games got larger, as we also found in our initial experiments on this topic [101]. Our experiments show that FOMs

[2]Confirmed through author communication

[3]This variation uses the current iterate rather than the average iterate due to decreased memory usage. It has inferior practical iteration complexity.

are substantially faster than CFR algorithms when using a practically-tuned variant of our DGF, even as the game-size is scaled up and when CFR is using the RM$^+$ regret minimizer. We find that CFR$^+$ is still faster in practice, but only due to its aggressive stepsize policy. This suggests that future work on FOMs could be coupled with our DGF to create state-of-the-art algorithms in practice. We also test the impact of stronger bounds on the strong convexity parameter: we instantiate EGT with the parameters developed in this chapter, and compare the performance to weaker parameters developed in a preliminary conference version of this work Kroer et al. [101]. These experiments illustrate that the tighter parameters developed here lead to better practical convergence rate.

The rest of the chapter is organized as follows. Section 3.1 discusses related literature. We present the general class of problems that we address—bilinear saddle-point problems—and describe how they relate to EFGs in Section 3.2. Then Section 3.3 describes our optimization framework. Section 3.4 introduces treeplexes, the class of convex polytopes that define our domains of the optimization problems. Our focus is on dilated entropy-based DGFs; we introduce these in Section 3.5 and present our main results—bounds on the associated strong convexity parameter and treeplex diameter. In Section 3.6 we demonstrate the use of our results on instantiating EGT. 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 3.6.1. Section 3.7 describes a closed-form solution for computing the smoothed-best response associated with our smoothing function. Sections 3.8 and 3.9 present 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 CFR$^+$ on medium-sized and large-scale games respectively. Section 3.10 describes a sampling approach for using stochastic FOMs. We close with a summary of our results and a few compelling further research directions in Section 3.11.

3.1 Related literature

Nash equilibrium computation has received extensive attention in the literature [48, 49, 68, 81, 95, 112, 114, 179]. 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. [93] present an LP which has size linear in the size of the game tree. This approach, coupled with lossless abstraction techniques, was used to solve Rhode-Island hold'em [68, 155], a game with 3.1 billion nodes (roughly size $5 \cdot 10^7$ after lossless abstraction). However, for very large games, the resulting LPs tend to not fit in the computer memory, and iterations of the simplex algorithm or interior-point methods tend to be too slow. Bošanskỳ et al. [17] extend the LP approach by considering a form of column-and-row-generation. This algorithm scales only for games where it can identify an equilibrium of small support, and thus suffers from the same performance issues as the general LP approach. The scalability issues of LP-based approaches thus require approximate solution techniques. These techniques fall into two categories: iterative ϵ-Nash equilibrium-finding algorithms and game abstraction techniques [148].

The most popular iterative Nash equilibrium algorithms are variants of the counterfactual-regret-minimization framework [108, 161, 179]. All these algorithms operate by defining a notion of regret local to each information set, called *counterfactual regret*. A simplex-based regret-

minimizing algorithm is then instantiated independently at each information set and given the counterfactual regrets for minimizing. It is shown that the per-information-set regret bounds obtained by the simplex-based regret minimizers lead to an overall bound on the convergence rate. Initially, the most practically popular variants were instantiated with *regret matching* (CFR) [179], and sometimes used Monte-Carlo methods for estimating regrets [108]. Recently a new regret-minimization technique called *regret matching plus* (RM$^+$) was shown to be practically superior when coupled with a more aggressive stepsizing strategy [21, 161]. The algorithm combining RM$^+$ and aggressive stepsizing is referred to as CFR$^+$. Despite their slow convergence rate of $O(\frac{1}{\epsilon^2})$, these regret-based algorithms perform very well in practice, especially CFR$^+$. Recently, Waugh and Bagnell [169] showed, with some caveats, an interpretation of CFR as a FOM with $O(\frac{1}{\epsilon^2})$ rate. In particular, they show that CFR is equivalent to *dual averaging* [134] (when using a particular set of distance functions that fall in the general class of dilated distance functions [79]) on the simplex (and thereby also on information sets with no child information sets), whereas on internal information sets these algorithms differ in how they aggregate utilities from child information sets. It is still unknown whether CFR is fully equivalent to a first-order method. Despite these similarities, in this chapter we make a distinction between regret-based methods and $O(\frac{1}{\epsilon})$ FOMs for ease of exposition when comparing algorithmic methodologies.

Hoda et al. [79] introduce DGFs for EFGs leading to $O(\frac{1}{\epsilon})$ convergence rate when used with EGT. In preliminary work, we improved the parameters associated with the dilated entropy function based DGFs [101]. While Gilpin et al. [72] give an algorithm with convergence rate $O(\ln(\frac{1}{\epsilon}))$, their bound has a dependence on a certain condition number of the payoff matrix, which is difficult to estimate, and their bound independent of the condition number has a $O(\frac{1}{\epsilon})$ convergence rate. We compare all three algorithms discussed here in detail in Section 3.6.1. Based on these results, the next chapter shows how to extend FOMs and our DGF to the computation of approximate Nash-equilibrium refinements.

In a work that is not part of this thesis, coauthors and I show experimentally that certain parts of the game tree can be pruned when computing gradients for EGT with our DGF [29]. In particular, when computing the gradient via tree traversal, we simply skip branches that have probability less than $O(1/\sqrt{t})$ where t is the current iteration. This lead to a speedup factor of about 2.

3.2 Problem setup

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

3.2.2 Sequence form

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 [93, 146, 165] in the EFG literature. 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. The matrix A encodes the reward structure associated with the game. For a complete treatment of this formulation, see von Stengel [165].

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 general BSPPs. We follow the presentation and notation of Juditsky and Nemirovski [85, 86] for BSPPs.

3.3 Optimization setup

In its most general form a BSPP is defined as

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

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:

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

with $\text{Opt}(P) = \text{Opt}(D) = \text{Opt}$. 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*

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

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

3.3.1 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 (we present components for $\mathcal{X}$, analogous components are used for $\mathcal{Y}$):

- *Vector norm*: $\|\cdot\|_{\mathcal{X}}$ on the Euclidean space $\mathbf{E}_x$ where the domain $\mathcal{X}$ of $(\mathcal{S})$ lives, along with its dual norm $\|\zeta\|_{\mathcal{X}}^* = \max_{\|x\|_{\mathcal{X}} \leq 1} \langle \zeta, x \rangle$.
- *Matrix norm*: $\|A\| = \max_y \{\|Ay\|_{\mathcal{X}}^* : \|y\|_{\mathcal{Y}} = 1\}$ based on the vector norms $\|\cdot\|_{\mathcal{X}}$ and $\|\cdot\|_{\mathcal{Y}}$.
- *Distance-Generating Function* (DGF): A function $\omega_{\mathcal{X}}(x) : \mathcal{X} \to \mathbb{R}$, which is convex and continuous on $\mathcal{X}$, and admits a continuous selection of subgradients $\omega'_{\mathcal{X}}(x)$ on the set $\mathcal{X}^\circ := \{x \in \mathcal{X} : \partial\omega_{\mathcal{X}}(x) \neq \emptyset\}$ (here $\partial\omega_{\mathcal{X}}(x)$ is a subdifferential of $\omega_{\mathcal{X}}$ taken at x), and is strongly convex with modulus $\varphi_{\mathcal{X}}$ w.r.t. the norm $\|\cdot\|_{\mathcal{X}}$:

$$\forall x', x'' \in \mathcal{X}^\circ : \ \langle \omega'_{\mathcal{X}}(x') - \omega'_{\mathcal{X}}(x''), x' - x'' \rangle \geq \varphi_{\mathcal{X}} \|x' - x''\|_{\mathcal{X}}^2. \tag{3.2}$$

- *Bregman distance*: $V(u\|x) := \omega_{\mathcal{X}}(u) - \omega_{\mathcal{X}}(x) - \langle \omega'_{\mathcal{X}}(x), u - x \rangle$ for all $x \in \mathcal{X}^\circ$ and $u \in \mathcal{X}$. The Bregman distance (or divergence) is how we construct a distance function from a DGF. Given two points in the polytope, one called the *center*, here x, we measure distance by taking the difference between two function values at u, the DGF value and the value of the linear approximation centered at x.
- *Prox-mapping*: Given a *prox center* $x \in \mathcal{X}^\circ$,

$$\text{Prox}_x(\xi) := \operatorname*{argmin}_{u \in \mathcal{X}} \{\langle \xi, u \rangle + V(u\|x)\} : \mathbf{E} \to \mathcal{X}^\circ.$$

  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 $x - \xi$ onto $\mathcal{X}$.
- *ω-center*: $x_\omega := \operatorname*{argmin}_{x \in \mathcal{X}} \omega_{\mathcal{X}}(x) \in \mathcal{X}^\circ$ of $\mathcal{X}$.
- *Set width*: $\Omega_x := \max_{x \in \mathcal{X}} V(x\|x_\omega) \leq \max_{x \in \mathcal{X}} \omega_{\mathcal{X}}(x) - \min_{x \in \mathcal{X}} \omega_{\mathcal{X}}(x)$.

In this chapter, for ease of exposition, we will introduce and work with the *Excessive Gap Technique* (EGT) of Nesterov [132] as the FOM. Next we introduce the related terminology and formally state the algorithm and the results from [132].

Nesterov [133] introduced *smoothed approximations* to the functions $\overline{\phi}$ and $\underline{\phi}$ via the distance-generating functions $\omega_{\mathcal{X}}, \omega_{\mathcal{Y}}$ as follows:

$$\overline{\phi}_{\mu_2}(x) = \max_{y \in \mathcal{Y}} \{\phi(x, y) - \mu_2 \omega_{\mathcal{Y}}(y)\}, \tag{3.3}$$

$$\underline{\phi}_{\mu_1}(y) = \min_{x \in \mathcal{X}} \{\phi(x, y) + \mu_1 \omega_{\mathcal{X}}(x)\}, \tag{3.4}$$

where $\mu_1, \mu_2 > 0$ are smoothness parameters denoting the amount of smoothing applied. Let $y_{\mu_2}(x)$ and $x_{\mu_1}(y)$ be the y and x solutions attaining the optima in (3.3) and (3.4). These can be thought of as *smoothed best responses*. Nesterov [133] also established that the gradients of the functions $\overline{\phi}_{\mu_2}(x)$ and $\underline{\phi}_{\mu_1}(y)$ exist and are Lipschitz continuous. The gradient operators and Lipschitz constants are given as follows

$$\nabla\overline{\phi}_{\mu_2}(x) = a_1 + Ay_{\mu_2}(x) \quad \text{and} \quad \nabla\underline{\phi}_{\mu_1}(y) = a_2 + A^\top x_{\mu_1}(y),$$
$$L_1\left(\overline{\phi}_{\mu_2}\right) = \frac{\|A\|^2}{\varphi_{\mathcal{Y}}\mu_2} \quad \text{and} \quad L_2\left(\underline{\phi}_{\mu_1}\right) = \frac{\|A\|^2}{\varphi_{\mathcal{X}}\mu_1}.$$

Based on this setup, we formally state the EGT of [132] in Algorithm 1.

ALGORITHM 1: EGT

input : ω-centers x_ω, y_ω, smoothness weights μ_1, μ_2, and desired accuracy ϵ
output: $z^t = [x^t, y^t]$
$t = 0,\ \mu_1^t = \mu_1,\ \mu_2^t = \mu_2$;
$x^t = \text{Prox}_{x_\omega}\left(\mu_1^{-1}\nabla\overline{\phi}_{\mu_2}(x_\omega)\right)$;
$y^t = y_{\mu_2}(x_\omega)$;
while $\epsilon_{\text{sad}}(z^t) > \epsilon$ **do**
  $\tau_t = \frac{2}{t+3}$;
  if *t is even* **then**
    $(\mu_1^{t+1}, x^{t+1}, y^{t+1}) = Step(\mu_1^t, \mu_2^t, x^t, y^t, \tau_t)$
  else
    $(\mu_2^{t+1}, y^{t+1}, x^{t+1}) = Step(\mu_2^t, \mu_1^t, y^t, x^t, \tau_t)$
  end
  $t = t + 1$;
end

ALGORITHM 2: Step

input : μ_1, μ_2, x, y, τ
output: μ_1^+, x_+, y_+
$\hat{x} = (1-\tau)\, x + \tau x_{\mu_1}(y)$;
$y_+ = (1-\tau)\, y + \tau y_{\mu_2}(\hat{x})$;
$\tilde{x} = \text{Prox}_{x_{\mu_1}(y)}\left(\frac{\tau}{(1-\tau)\mu_1}\nabla\overline{\phi}_{\mu_2}(\hat{x})\right)$;
$x_+ = (1-\tau)\, x + \tau\tilde{x}$;
$\mu_1^+ = (1-\tau)\,\mu_1$;

The EGT algorithm alternates between taking steps focused on $\mathcal{X}$ and $\mathcal{Y}$. Algorithm 2 shows a single step focused on $\mathcal{X}$. Steps focused on y are completely analogous. Algorithm 1 shows how initial points are selected and the alternating steps and stepsizes are computed. Nesterov [132] proves that the EGT algorithm converges in $O(\frac{1}{\epsilon})$ steps to an ϵ approximate saddle point of the BSPP $(\mathcal{S})$. More precisely, Nesterov [132, Theorem 6.3] states the following:

Theorem 1. *Suppose the input values μ_1, μ_2 in the EGT algorithm satisfy*

$$\mu_1 = \frac{\varphi_{\mathcal{X}}}{L_1(\overline{\phi}_{\mu_2})} = \frac{\varphi_{\mathcal{X}}\varphi_{\mathcal{Y}}}{\|A\|^2}\mu_2.$$

Then, at every iteration $t \geq 1$ of the EGT algorithm, the corresponding solution $z^t = [x^t; y^t]$ satisfies $x^t \in \mathcal{X}$, $y^t \in \mathcal{Y}$, and

$$\overline{\phi}(x^t) - \underline{\phi}(y^t) = \epsilon_{\text{sad}}(z^t) \leq \frac{4\|A\|}{t+1}\sqrt{\frac{\Omega_{\mathcal{X}}\Omega_{\mathcal{Y}}}{\varphi_{\mathcal{X}}\varphi_{\mathcal{Y}}}}.$$

3.4 Treeplexes

Hoda et al. [79] introduce the *treeplex*, a class of convex polytopes that encompass the sequence-form description of strategy spaces in perfect-recall EFGs. Understanding the treeplex structure is crucial because the proofs of our main results rely on induction over these structures.

In the context of EFGs a treeplex is used to model the strategy space of a single player in sequence form. A treeplex can be thought of as a tree of simplexes, where each simplex represents a decision point for the player. The simplexes in the tree are scaled by the parent variable leading to them, such that each simplex sums to the value of the parent variable. Thus the value in of a particular variable represents the probability of the player taking all actions on the path through the treeplex leading to that variable.

Definition 3. *Treeplexes are defined recursively as follows:*

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*

$$P\boxed{l}Q \coloneqq \left\{(u, q_1, \ldots, q_k) \in \mathbb{R}^{p+\sum_j n_j} : \; u \in P,\; q_1 \in u_{l_1} \cdot Q_1,\; \ldots, q_k \in u_{l_k} \cdot Q_k\right\}$$

is a treeplex. We say u_{l_j} is the branching variable for the treeplex Q_j.

A treeplex is a tree of simplexes 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. For EFGs, the simplexes 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 u represents the decision variables at certain stages, while the vectors q_j represent the decision variables at the k potential following stages, depending on external outcomes. Here $k \leq p$ since some variables in u may not have subsequent decisions. For treeplexes, von Stengel [165] has suggested a polyhedral representation of the form $Eu = e$ 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 simplexes 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 simplexes 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 depth of the treeplex rooted at Δ^j, that is, the maximum number of edges between Δ^j and any simplex beneath Δ^j plus one (that is, a lone simplex has depth 1, not 0). In an EFG the depth is the length of the longest sequence of actions starting at Δ^j. Then d_Q gives the depth of Q. We use b_Q^j to identify the number of branching operations preceding the j-th simplex in Q. We will say that a simplex j such that $b_Q^j = 0$ is a *root simplex*.

Figure 3.1 illustrates an example treeplex Q. This treeplex Q is constructed from nine two-to-three-dimensional simplexes $\Delta^1, \ldots, \Delta^9$. At level 1, we have two root simplexes, Δ^1, Δ^2, obtained by a Cartesian product operation (denoted by $\times$). We have maximum depths $d_1 = 2$, $d_2 = 1$ beneath them. Since there are no preceding branching operations, the parent variables for these simplexes 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 simplexes $\Delta^3, \ldots, \Delta^7$. 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 simplexes $\Delta^3, \ldots, \Delta^7$ 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 simplexes Δ^1, Δ^3 and Δ^8 is $b_Q^1 = 0, b_Q^3 = 1$ and $b_Q^8 = 2$, respectively.

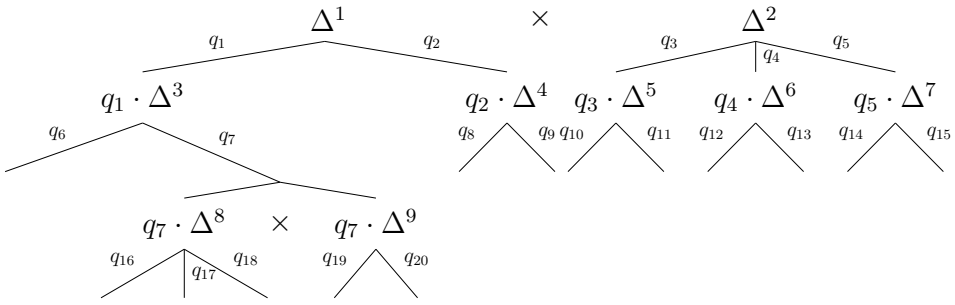


Figure 3.1: An example treeplex constructed from 9 simplexes. Cartesian product operation is denoted by $\times$.

The original formulation of Hoda et al. [79] allowed only two-way branches (although they also consider a uniform treeplex where all Cartesian products are k-way); we allow arbitrary branching, not necessarily two-way, and not necessarily uniform. As discussed in Hoda et al. [79], 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 associated 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.

Our analysis requires a measure of the size of a treeplex Q. For this purpose, we define $M_Q \coloneqq \max_{q \in Q} \|q\|_1$.

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

$$M_{Q,r} \coloneqq \max_{q \in Q} \sum_{j \in S_Q : b_Q^j \leq r} \|q^j\|_1. \tag{3.5}$$

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 simplexes that are not within r branching operations of the root of Q.

The quantities M_Q and $|S_Q|$ play an important role in the comparison of our bounds. We discuss them on an example next.

Example 1. *In order to illustrate M_Q and compare it to the size of $|S_Q|$, let us now consider an example of an EFG and its corresponding treeplexes. Consider a game where two players take turns choosing among k actions, and each player chooses actions d times before leaf nodes are reached. In the treeplex Q of Player 1, each time Player 1 chooses among k actions constitutes a size k branching operation, and every time Player 2 chooses among k actions constitutes a size k Cartesian product operation. The total dimensionality of the treeplex, $|S_Q|$, is k^{2d}, while the value of M_Q is k^d (since only Cartesian products blow up). Thus, M_Q is square root of the size of $|S_Q|$.*

3.5 Dilated entropy 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 (3.2)).

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 [85]). 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* [78] 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 4. *Given a function $f(z)$, the* dilation operation *is defined as the function $\bar{f} : \bar{K} \to \mathbb{R}$ given by*

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

The dilation operation preserves convexity, and thus the following function defined based on dilating the entropy function over the simplexes of a treeplex is convex:

Definition 5. *Given a treeplex Q and weights $\beta_j > 0$ for each $j \in S_Q$, we define the* dilated entropy function *as*

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

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. *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 weights β_j that scale appropriately with depth d_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.* ∎

We would also like the prox-mapping associated with our DGF to be efficiently computable. Hoda et al. [79] show that for any dilated function, its prox operator on a treeplex can be easily computed through a recursive bottom-up traversal involving the prox mappings associated with the function being dilated on individual simplexes. Since the entropy prox function can be computed in closed form on a simplex, the dilated entropy function can be computed by a single treeplex traversal involving closed-form expressions on each simplex.

Definition 5 above leads to a subset of the DGFs considered by Hoda et al. [79]. Hoda et al. [79] introduce the general class of treeplex DGFs obtained by dilating any simplex DGF, and show that any such DGF is guaranteed to have some lower bound on the strong-convexity parameter. They show explicit bounds for a special class of treeplexes called *uniform treeplexes*. Our main theoretical result shows that by selecting the weights β_j according to a particular recurrence, we can significantly improve the strong convexity bounds associated with the dilated entropy function, and we show that our analysis is tight.

We will consider weights that satisfy the following recurrence:

$$\begin{aligned}
&\alpha_j = 1 + \max_{i \in \mathbb{I}_j} \sum_{k \in \mathcal{D}_j^i} \frac{\alpha_k \beta_k}{\beta_k - \alpha_k}, && \forall j \in S_Q, \\
&\beta_j > \alpha_j, && \forall i \in \mathbb{I}_j \text{ and } \forall j \in S_Q \text{ s.t. } b_Q^j > 0, \\
&\beta_j = \alpha_j, && \forall i \in \mathbb{I}_j \text{ and } \forall j \in S_Q \text{ s.t. } b_Q^j = 0.
\end{aligned} \tag{3.6}$$

Intuitively, α_j represents the negative terms that the weight β_j has to cancel out: the constant 1 represents the negative term resulting from the squared norm in the strong convexity requirement; the summation term represents the amount of negative terms accumulated from the induction on simplexes descending from simplex j. The qualifications on β_j ensure that β_j is set such that it at least cancels out the negative terms; the difference $\beta_j - \alpha_j$ controls the amount of negative value the parent simplex has to make up. When $b_Q^j = 0$ there is no parent simplex, and so we set $\beta_j = \alpha_j$. The reason for our requirement of a strict inequality $\beta_j > \alpha_j$ for non-root simplexes becomes evident in the proof of Lemma 2.

Based on recurrence (3.6), our main results establish strong convexity of our dilated entropy DGF w.r.t. the ℓ_2 and ℓ_1 norms:

Theorem 2. *For a treeplex Q, the dilated entropy function with weights satisfying recurrence (3.6) is strongly convex with modulus 1 with respect to the ℓ_2 norm.*

Theorem 3. *For a treeplex Q, the dilated entropy function with weights satisfying recurrence (3.6) is strongly convex with modulus $\frac{1}{M_Q}$ with respect to the ℓ_1 norm.*

We give the proofs of Theorems 2 and 3 in Section 3.5.2. Based on Theorem 3, we get the following corollary:

Corollary 1. *For a 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.*

Corollary 1 follows easily from Theorem 3 and a recursive interpretation of the weights, which is presented as Fact 2 in the next section. In particular, a specific choice of weights in Fact 2 immediately satisfies the recurrence (3.6) and leads to Corollary 1.

This result is the first to show an explicit strong-convexity bound for general treeplexes. Even for the special case of uniform treeplexes considered by Hoda et al. [79], our bound is much stronger, and is the first to show no dependence on the branching operations in the treeplex construction, thus generalizing the similar property enjoyed by the entropy DGF for simplexes.

In Theorem 4 we use our strong convexity result to establish 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.

3.5.1 Preliminary results for the proofs of our main results

We start with some simple facts and a few technical lemmas that are used in our proofs.

Fact 1. *Given a treeplex Q, we have, respectively, for all $i \in \mathbb{I}_j, j \in S_Q$ and all $d = 1, \dots, d_Q, q \in Q$:*

$$(a) \quad M_{Q_j} \geq 1 + \sum_{l \in \mathcal{D}_j^i} M_{Q_l}, \qquad\qquad (b) \quad M_Q \geq \sum_{j \in S_Q : d_j = d} q_{p_j} M_{Q_j}.$$

Proof. The inequality follows from the definition of M_Q, that Δ^j is a simplex, and the fact that the maximum entry (as would be taken in the ℓ_1 norm) bounds the value of any individual entry i. The second follows by using $M_Q = \sum_j q_i$ for some q, and inductively replacing terms belonging to simplexes j at the bottom with M_{Q_j}. The result follows because branching operations cancel out by summing to 1. □

Our next observation follows from Fact 1(a) and is advantageous in suggesting a practically useful choice of the weights β_j that can be used for Theorem 3 to arrive at Corollary 1.

Fact 2. *Let Q be a treeplex and $\beta_j = 2 + \sum_{r=1}^{d_j} 2^r(M_{Q_j,r} - 1)$ for all $j \in S_Q$ as in Corollary 1. Then Fact 1(a) implies $\beta_j \geq 2 + \sum_{k \in \mathcal{D}_j^i} 2\beta_k, \ \ \forall i \in \mathbb{I}_j$ and $\forall j \in S_Q$.*

Consequently, by selecting $\beta_j = 2\alpha_j$, and $\alpha_j = 1 + \sum_{r=1}^{d_j} 2^{r-1}(M_{Q_j,r} - 1)$ for all $i \in \mathbb{I}_j$ and for all $j \in S_Q$ such that $b_Q^j > 0$, we immediately satisfy the conditions of the recurrence in (3.6).

Given a twice differentiable function f, we let $\nabla^2 f(z)$ denote its Hessian at z. Our analysis is based on the following sufficient condition for strong convexity of a twice differentiable function:

Fact 3. *A twice-differentiable function f is strongly convex with modulus φ with respect to a norm $\|\cdot\|$ on nonempty convex set $C \subset \mathbb{R}^n$ if $h^\top \nabla^2 f(z) h \geq \varphi \|h\|^2,\ \forall h \in \mathbb{R}^n, z \in C^\circ$.*

For simplexes Δ^j at depth 1, there is no preceding branching operation; so the variables h_{p_j}, q_{p_j} do not exist. We circumvent this with the convention $h_{p_j} = 0, q_{p_j} = 1$ for such $j \in S_Q$.

In our proofs we will use the expression derived in Lemma 1 for $h^\top \nabla^2 \omega(q) h$.

Lemma 1. *Given a treeplex Q and a dilated entropy function $\omega(\cdot)$ with weights $\beta_j > 0$, we have*

$$h^\top \nabla^2 \omega(q) h = \sum_{j \in S_Q} \beta_j \left[\sum_{i \in \mathbb{I}_j} \left(\frac{h_i^2}{q_i} - \frac{2 h_i h_{p_j}}{q_{p_j}} \right) + \frac{h_{p_j}^2}{q_{p_j}} \right] \quad \forall q \in \operatorname{ri}(Q) \text{ and } \forall h \in \mathbb{R}^n. \tag{3.7}$$

Proof. Consider $q \in \operatorname{ri}(Q)$ and any $h \in \mathbb{R}^n$. For each $j \in S_Q$ and $i \in \mathbb{I}_j$, the second-order partial derivates of $\omega(\cdot)$ w.r.t. q_i are:

$$\nabla^2_{q_i^2} \omega(q) = \frac{\beta_j}{q_i} + \sum_{k \in \mathcal{D}_j^i} \sum_{l \in \mathbb{I}_k} \frac{\beta_k q_l}{q_i^2} = \frac{\beta_j}{q_i} + \sum_{k \in \mathcal{D}_j^i} \frac{\beta_k}{q_i}, \tag{3.8}$$

where the last equality holds because $k \in \mathcal{D}_j^i$ and thus $\sum_{l \in \mathbb{I}_k} q_l = \|q^k\|_1 = q_{p_k} = q_i$. Also, for each $j \in S_Q, i \in \mathbb{I}_j, k \in \mathcal{D}_j^i$, and $l \in \mathbb{I}_k$, the second-order partial derivates w.r.t. q_i, q_l are given by:

$$\nabla^2_{q_i, q_l} \omega(q) = \nabla^2_{q_l, q_i} \omega(q) = -\frac{\beta_k}{q_i}. \tag{3.9}$$

Then equations (3.8) and (3.9) together imply

$$h^\top \nabla^2 \omega(q) h = \sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} \left[h_i^2 \left(\frac{\beta_j}{q_i} + \sum_{k \in \mathcal{D}_j^i} \frac{\beta_k}{q_i} \right) - \sum_{k \in \mathcal{D}_j^i} \sum_{l \in \mathbb{I}_k} h_i h_l \frac{2\beta_k}{q_i} \right]. \tag{3.10}$$

Given $j \in S_Q$ and $i \in \mathbb{I}_j$, we have $p_k = i$ for each $k \in \mathcal{D}_j^i$ and for any $k \in \mathcal{D}_j^i$, there exists some other $j' \in S_Q$ corresponding to k in the outermost summation. Then we can rearrange the following terms:

$$\sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} h_i^2 \sum_{k \in \mathcal{D}_j^i} \frac{\beta_k}{q_i} = \sum_{j \in S_Q} \beta_j \frac{h_{p_j}^2}{q_{p_j}} \quad \text{and} \quad \sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} \sum_{k \in \mathcal{D}_j^i} \sum_{l \in \mathbb{I}_k} h_i h_l \frac{2\beta_k}{q_i} = \sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} \beta_j \frac{2 h_i h_{p_j}}{q_{p_j}}.$$

Using these two equalities in the relation (3.10) leads to (3.7) and proves the lemma. □

3.5.2 Proofs of our main theorems

The majority of the work for our strong-convexity results is performed by the following lemma, from which our strong convexity results follow easily.

Lemma 2. *For any treeplex Q, the dilated entropy function with weights satisfying recurrence* (3.6) *satisfies the following inequality:*

$$h^\top \nabla^2 \omega(q) h \geq \sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} \frac{h_i^2}{q_i} \quad \forall q \in \operatorname{ri}(Q) \text{ and } \forall h \in \mathbb{R}^n. \tag{3.11}$$

Proof. We will prove this by induction. First we will prove an inductive hypothesis over the set of non-root simplexes $\widehat{S}_Q = \left\{ j \in S_Q \;\middle|\; b_Q^j > 0 \right\}$. In order to state our inductive hypothesis we will need a notion of the set of simplexes currently at the "top" of the recursion: for a given depth d, we let the set of simplexes at the top be $\widehat{S}_Q^d = \left\{ k \in \widehat{S}_Q \;\middle|\; d_k \leq d, \exists j, i \text{ s.t. } k \in \mathcal{D}_j^i, d_j > d \right\}$. This is simply the set of simplexes such that their depth is less than d and the depth of their parent simplex is strictly greater than d. The reader may wonder why we do not perform the induction over simplices such that $d_j = d$. This is in order to avoid some technicalities relating to cases where two child simplexes from the same parent have different depths. By using $\widehat{S}_Q^d$, we ensure that the right-hand side of the inductive hypothesis always consists of the simplexes that are at the top of the treeplex given the current induction depth. We now show the following inductive hypothesis for any depth $d \geq 1$:

$$\sum_{j \in \widehat{S}_Q : d_j \leq d} \beta_j \left[\sum_{i \in \mathbb{I}_j} \left(\frac{h_i^2}{q_i} - \frac{2 h_i h_{p_j}}{q_{p_j}} \right) + \frac{h_{p_j}^2}{q_{p_j}} \right] - \sum_{j \in \widehat{S}_Q : d_j \leq d} \sum_{i \in \mathbb{I}_j} \frac{h_i^2}{q_i} \geq - \sum_{j \in \widehat{S}_Q^d} \frac{\beta_j \alpha_j}{\beta_j - \alpha_j} \frac{h_{p_j}^2}{q_{p_j}}.$$

We first show the inductive step, as the base case will follow from the same logic. Consider a treeplex Q of depth $d > 1$. By applying the inductive hypothesis we have

$$\begin{aligned} &\sum_{j \in \widehat{S}_Q : d_j \leq d} \beta_j \left[\sum_{i \in \mathbb{I}_j} \left(\frac{h_i^2}{q_i} - \frac{2 h_i h_{p_j}}{q_{p_j}} \right) + \frac{h_{p_j}^2}{q_{p_j}} \right] - \sum_{j \in \widehat{S}_Q : d_j \leq d} \sum_{i \in \mathbb{I}_j} \frac{h_i^2}{q_i} \\ \geq &\sum_{j \in \widehat{S}_Q : d_j = d} \beta_j \left[\sum_{i \in \mathbb{I}_j} \left(\frac{h_i^2}{q_i} - \frac{2 h_i h_{p_j}}{q_{p_j}} \right) + \frac{h_{p_j}^2}{q_{p_j}} \right] - \sum_{j \in \widehat{S}_Q : d_j = d} \sum_{i \in \mathbb{I}_j} \frac{h_i^2}{q_i} - \sum_{j \in \widehat{S}_Q^{d-1}} \frac{\beta_j \alpha_j}{\beta_j - \alpha_j} \frac{h_{p_j}^2}{q_{p_j}}. \end{aligned} \tag{3.12}$$

Now we can rearrange terms: First we split $\widehat{S}_Q^{d-1}$ into two sets $\widehat{S}_Q^{d-1} \cap \widehat{S}_Q^d$ and $\widehat{S}_Q^{d-1} \setminus \widehat{S}_Q^d$. The sum over $j \in \widehat{S}_Q^{d-1} \setminus \widehat{S}_Q^d$ is equivalent to a sum over the immediate descendant information sets $k \in \mathcal{D}_j^i$ inside the square brackets since for each such $k \in \widehat{S}_Q^{d-1} \setminus \widehat{S}_Q^d$ there exists some $j \in \widehat{S}_Q^d$ such that $d_j = d$ (otherwise k would be in $\widehat{S}_Q^d$). Ignoring $\widehat{S}_Q^{d-1} \cap \widehat{S}_Q^d$ in (3.12) for now, we can write

$$\sum_{j \in \widehat{S}_Q : d_j = d} \beta_j \left[\sum_{i \in \mathbb{I}_j} \left(\frac{h_i^2}{q_i} - \frac{2 h_i h_{p_j}}{q_{p_j}} \right) + \frac{h_{p_j}^2}{q_{p_j}} \right] - \sum_{j \in \widehat{S}_Q : d_j = d} \sum_{i \in \mathbb{I}_j} \frac{h_i^2}{q_i} - \sum_{j \in \widehat{S}_Q^{d-1} \setminus \widehat{S}_Q^d} \frac{\beta_j \alpha_j}{\beta_j - \alpha_j} \frac{h_{p_j}^2}{q_{p_j}}. \tag{3.13}$$

$$= \sum_{j \in \widehat{S}_Q : d_j = d} \beta_j \left[\sum_{i \in \mathbb{I}_j} \left(\frac{h_i^2}{q_i} - \frac{2 h_i h_{p_j}}{q_{p_j}} - \sum_{k \in \mathcal{D}_j^i} \frac{\alpha_j}{\beta_j - \alpha_j} \frac{h_{p_j}^2}{q_{p_j}} \right) + \frac{h_{p_j}^2}{q_{p_j}} \right] - \sum_{j \in \widehat{S}_Q : d_j = d} \sum_{i \in \mathbb{I}_j} \frac{h_i^2}{q_i}.$$

Now we split the term $\frac{h_{p_j}^2}{q_{p_j}}$ into separate terms multiplied by $\frac{q_i}{q_{p_j}}$ and move it inside the parentheses

by using the fact that $\sum_{i\in\mathbb{I}_j} \frac{q_i}{q_{p_j}} = 1$, this gives

$$= \sum_{j\in\widehat{S}_Q:d_j=d} \beta_j \left[\sum_{i\in\mathbb{I}_j}\left(\frac{h_i^2}{q_i} - \frac{2h_ih_{p_j}}{q_{p_j}} - \sum_{k\in\mathcal{D}_j^i}\frac{\alpha_j}{\beta_j-\alpha_j}\frac{h_{p_j}^2}{q_{p_j}} + \frac{q_ih_{p_j}^2}{q_{p_j}^2}\right)\right] - \sum_{j\in\widehat{S}_Q:d_j=d}\sum_{i\in\mathbb{I}_j}\frac{h_i^2}{q_i}.$$

Now we can move the sum over $i \in \mathbb{I}_j$ outside the square brackets and consolidate the summation terms to get

$$\begin{aligned}
&= \sum_{j\in\widehat{S}_Q:d_j=d}\sum_{i\in\mathbb{I}_j}\left[\left(\beta_j - 1 - \sum_{k\in\mathcal{D}_j^i}\frac{\beta_k\alpha_k}{\beta_k-\alpha_k}\right)\frac{h_i^2}{q_i} - \left(\frac{2\beta_jh_ih_{p_j}}{q_{p_j}}\right) + \frac{q_i\beta_jh_{p_j}^2}{q_{p_j}^2}\right] \\
&\geq \sum_{j\in\widehat{S}_Q:d_j=d}\sum_{i\in\mathbb{I}_j}\left[(\beta_j-\alpha_j)\frac{h_i^2}{q_i} - \left(\frac{2\beta_jh_ih_{p_j}}{q_{p_j}}\right) + \frac{q_i\beta_jh_{p_j}^2}{q_{p_j}^2}\right], \qquad (3.14)
\end{aligned}$$

where the last inequality follows from the definition of α_j.

For indices $j \in S_Q$ such that $b_Q^j > 0$ and $i \in \mathbb{I}_j$, the relations in (3.6) imply $\beta_j > \alpha_j$, and so the expression inside the square brackets in (3.14) is a convex function of h_i. Taking its derivative w.r.t. h_i and setting it to zero gives $h_i = \frac{\beta_j}{\beta_j-\alpha_j}\frac{q_i}{q_{p_j}}h_{p_j}$. Thus, we arrive at

$$\begin{aligned}
(3.14) &\geq \sum_{j\in\widehat{S}_Q:d_j=d}\sum_{i\in\mathbb{I}_j}\left[\frac{\beta_j^2}{\beta_j-\alpha_j}\frac{q_ih_{p_j}^2}{q_{p_j}^2} - \frac{\beta_j^2}{\beta_j-\alpha_j}\frac{2q_ih_{p_j}^2}{q_{p_j}^2} + \frac{q_i\beta_jh_{p_j}^2}{q_{p_j}^2}\right] \\
&= \sum_{j\in\widehat{S}_Q:d_j=d}\frac{h_{p_j}^2}{q_{p_j}}\left[\left(\frac{-\beta_j^2}{\beta_j-\alpha_j}+\beta_j\right)\frac{\sum_{i\in\mathbb{I}_j}q_i}{q_{p_j}}\right] = -\sum_{j\in\widehat{S}_Q:d_j=d}\frac{\beta_j\alpha_j}{\beta_j-\alpha_j}\frac{h_{p_j}^2}{q_{p_j}}. \qquad (3.15)
\end{aligned}$$

Now we take our lower bound (3.15) on (3.13) and apply it to (3.12). Noting that $\widehat{S}_Q^d = \left\{\widehat{S}_Q : d_j = d\right\} \cup \left\{\widehat{S}_Q^{d-1} \cap \widehat{S}_Q^d\right\}$ we get

$$(3.12) \geq -\sum_{j\in\widehat{S}_Q:d_j=d}\frac{\beta_j\alpha_j}{\beta_j-\alpha_j}\frac{h_{p_j}^2}{q_{p_j}} - \sum_{j\in\widehat{S}_Q^{d-1}\cap\widehat{S}_Q^d}\frac{\beta_j\alpha_j}{\beta_j-\alpha_j}\frac{h_{p_j}^2}{q_{p_j}} = -\sum_{j\in\widehat{S}_Q^d}\frac{\beta_j\alpha_j}{\beta_j-\alpha_j}\frac{h_{p_j}^2}{q_{p_j}} \qquad (3.16)$$

Hence, the induction step is complete. For the base case $d = 0$ we do not need the inductive assumption: Because $\mathcal{D}_j^i = \emptyset$, $\alpha_j = 1$, and we get (3.14) by definition; we can then apply the same convexity argument. This proves our inductive hypothesis.

Then using Lemma 1, we now have

$$\begin{aligned}
h^\top\nabla^2\omega(q)h - \sum_{j\in S_Q}\sum_{i\in\mathbb{I}_j}\frac{h_i^2}{q_i} &= \sum_{j\in S_Q}\beta_j\left[\sum_{i\in\mathbb{I}_j}\left(\frac{h_i^2}{q_i} - \frac{2h_ih_{p_j}}{q_{p_j}}\right) + \frac{h_{p_j}^2}{q_{p_j}}\right] - \sum_{j\in S_Q}\sum_{i\in\mathbb{I}_j}\frac{h_i^2}{q_i} \\
&\geq \sum_{j\in S_Q:b_Q^j=0}\left[\sum_{i\in\mathbb{I}_j}\beta_j\frac{h_i^2}{q_i} - \sum_{k\in\mathcal{D}_j^i}\frac{\beta_k\alpha_k}{\beta_k-\alpha_k}\frac{h_i^2}{q_i} - \frac{h_i^2}{q_i}\right] \geq 0.
\end{aligned}$$

The first inequality follows from the fact that $h_{p_j} = 0$ for all $j \in S_Q$ such that $b_Q^j = 0$, and for all $j \in S_Q$ such that $b_Q^j > 0$, we used our induction. The last inequality follows from (3.6) and $q_i, h_i^2 \geq 0$. This then proves (3.11). □

We are now ready to prove our two main theorems, which we restate before proving them.

Theorem 2. *For a treeplex Q, the dilated entropy function with weights satisfying recurrence (3.6) is strongly convex with modulus 1 with respect to the ℓ_2 norm.*

Proof. Since $q_i \leq 1$, Lemma 2 implies $h^\top \nabla^2 \omega(q) h \geq \sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} h_i^2 = \|h\|_2^2$ for all $q \in \operatorname{ri}(Q)$ and for all $h \in \mathbb{R}^n$. Because the dilated entropy function $\omega(q)$ is twice differentiable on $\operatorname{ri}(Q)$, from Fact 3, we conclude that $\omega(\cdot)$ is strongly convex w.r.t. the ℓ_2 norm on Q with modulus 1. □

Remark 2. *The analysis in the proof of Theorem 2 is tight. By choosing a vector $q \in \{0,1\}^{|Q|}$ such that $\|q\|_1 = M_Q$, and setting $h_i = \frac{\beta_j}{\beta_j - \alpha_j} \frac{q_i}{q_{p_j}} h_{p_j}$ for all indices i such that $q_i = 1$ and $h_i = 0$ otherwise, every inequality in the proof of Lemma 2 becomes an equality.* ∎

Theorem 3. *For a treeplex Q, the dilated entropy function with weights satisfying recurrence (3.6) is strongly convex with modulus $\frac{1}{M_Q}$ with respect to the ℓ_1 norm.*

Proof. To show strong convexity with modulus 1 w.r.t. the ℓ_1 norm, we lower bound the right-hand side of (3.11) in Lemma 2:

$$\sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} \frac{h_i^2}{q_i} \geq \frac{1}{M_Q} \Bigg(\sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} q_i \Bigg) \sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} \frac{h_i^2}{q_i} \geq \frac{1}{M_Q} \Bigg(\sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} \frac{|h_i|}{\sqrt{q_i}} \sqrt{q_i} \Bigg)^2 = \frac{1}{M_Q} \|h\|_1^2,$$

where the first inequality follows from the fact that M_Q is an upper bound on $\|q\|_1$ for any $q \in Q$, and the second inequality follows from the Cauchy-Schwarz inequality.

Hence, we deduce $h^\top \nabla^2 \omega(q) h \geq \frac{1}{M_Q} \|h\|_1^2$ holds for all $q \in \operatorname{ri}(Q)$ and for all $h \in \mathbb{R}^n$. Because the dilated entropy function $\omega(q)$ is twice differentiable on $\operatorname{ri}(Q)$, from Fact 3, we conclude that $\omega(\cdot)$ is strongly convex w.r.t. the ℓ_1 norm on Q with modulus $\varphi = \frac{1}{M_Q}$. □

3.5.3 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 3.3.1. In order to establish full results on the convergence rates of these FOMs, we now bound this ratio using Corollary 1 scaled by M_Q.

Theorem 4. *For a treeplex Q, the dilated entropy function with simplex weights $\beta_j = M_Q(2 + \sum_{r=1}^{d_j} 2^r (M_{Q_j,r} - 1))$ 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.*

Proof. For our choice of scaled weights β_j, Corollary 1 implies that the resulting dilated entropy function is strongly convex with modulus $\varphi = 1$. Hence, we only need to bound Ω.

Any $q \in Q$ satisfying $q_i \in \{0, 1\}$ for all i maximizes $\omega(q)$ and results in $\max_{q \in Q} \omega(q) = 0$. For the minimum value, consider any $q \in \operatorname{ri}(Q)$. Applying the well-known lower bound of $-\log m$ for the negative entropy function on an m-dimensional simplex gives

$$\begin{aligned}
\omega(q) &= \sum_{j \in S_Q} \beta_j q_{p_j} \sum_{i \in \mathbb{I}_j} \frac{q_i}{q_{p_j}} \log \frac{q_i}{q_{p_j}} \geq -\sum_{j \in S_Q} \beta_j q_{p_j} \log m = -\sum_{d=0}^{d_Q} \sum_{j \in S_Q : d_j = d} \beta_j q_{p_j} \log m \\
&= -\sum_{d=1}^{d_Q} \sum_{j \in S_Q : d_j = d} \beta_j q_{p_j} \log m - \sum_{j \in S_Q : d_j = 0} \beta_j q_{p_j} \log m \\
&= -M_Q \log m \sum_{d=1}^{d_Q} \sum_{j \in S_Q : d_j = d} q_{p_j} \left(2 + \sum_{r=1}^{d} 2^r (M_{Q_j, r} - 1) \right) - M_Q \sum_{j \in S_Q : d_j = 0} 2 q_{p_j} \log m \\
&\geq -M_Q \log m \sum_{d=1}^{d_Q} \sum_{j \in S_Q : d_j = d} q_{p_j} M_{Q_j} \sum_{r=1}^{d} 2^r - 2 M_Q \log m \sum_{j \in S_Q : d_j = 0} q_{p_j}, \qquad (3.17)
\end{aligned}$$

where the last inequality follows because for each $j \in S_Q$ with $d_j = 0$, the definition of M_Q implies $\sum_{j \in S_Q : d_j = 0} q_{p_j} \leq M_Q$, and for each $j \in S_Q$ with $d_j = d \geq 1$, we have $2 + \sum_{r=1}^{d} 2^r (M_{Q_j, r} - 1) \leq \sum_{r=1}^{d} 2^r M_{Q_j, r} \leq \sum_{r=1}^{d} 2^r M_{Q_j}$ since $M_{Q_j, r} \leq M_{Q_j}$. Also, from Fact 1(b), we have $\sum_{j \in S_Q : d_j = d} q_{p_j} M_{Q_j} \leq M_Q$. Then we arrive at

$$\begin{aligned}
(3.17) &\geq -M_Q^2 \log m \left(2 + \sum_{d=1}^{d_Q} \sum_{r=1}^{d} 2^r \right) = -M_Q^2 \log m \left(2 + \sum_{d=1}^{d_Q} (2^{d+1} - 2) \right) \\
&= -M_Q^2 \log m \left(2 + \sum_{d=1}^{d_Q} 2^{d+1} - 2 d_Q \right) \geq -M_Q^2 (\log m) 2^{d_Q + 2},
\end{aligned}$$

where the last inequality follows because for $d_Q = 0$ we have $2^{d_Q + 2} = 4 > 2$ and for $d_Q \geq 1$ we have $2 d_Q \geq 2$.

This lower bound on the minimum value, i.e., $\min_{q \in Q} \omega(q) \geq -M_Q^2 (\log m) 2^{d_Q + 2}$, coupled with $\max_{q \in Q} \omega(q) \leq 0$, establishes the theorem. □

3.6 EGT for extensive-form game solving

We now describe how to instantiate EGT for solving two-player zero-sum EFGs of the form (3.1) with treeplex domains. Below we state the customization of all the definitions from Section 3.3 for our problem.

Let m be the size of the largest simplex in either of the treeplexes $\mathcal{X}, \mathcal{Y}$. Because $\mathcal{X}$ and $\mathcal{Y}$ are treeplexes, they are closed, convex, and bounded. We use the ℓ_1 norm on both of the embedding spaces $\mathbf{E}_x, \mathbf{E}_y$. As our DGFs for $\mathcal{X}, \mathcal{Y}$ compatible with the ℓ_1 norm, we use the dilated entropy DGF scaled with weights given in Theorem 4. Then Theorem 4 gives our bound on $\frac{\Omega_{\mathcal{X}}}{\varphi_{\mathcal{X}}}$ and $\frac{\Omega_{\mathcal{Y}}}{\varphi_{\mathcal{Y}}}$. Because the dual norm of the ℓ_1 norm is the ℓ_∞ norm, the matrix norm is given by $\|A\| = \max_{y \in \mathcal{Y}} \{\|Ay\|_1^* : \|y\|_1 = 1\} = \max_{i,j} |A_{i,j}|$.

Remark 3. *The matrix norm $\|A\|$ is not at the scale of the maximum payoff difference in the original game. The values in A are scaled by the probability of the observed nature outcomes on the path of each sequence. Thus, $\|A\|$ 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 4 immediately leads to the following convergence rate result for FOMs equipped with dilated entropy DGFs to solve EFGs (and more generally BSPPs over treeplex domains).

Theorem 5. *Consider a BSPP over treeplex domains $\mathcal{X}, \mathcal{Y}$. Then EGT algorithm equipped with the dilated entropy DGF with weights $\beta_j = 2 + \sum_{r=1}^{d_j} 2^r (M_{\mathcal{X}_j,r} - 1)$ for all $j \in S_{\mathcal{X}}$ and the corresponding setup for $\mathcal{Y}$ will return an ϵ-accurate solution to the BSPP in at most the following number of iterations:*

$$\frac{\max_{i,j} |A_{i,j}| \sqrt{M_{\mathcal{X}}^2 2^{d_{\mathcal{X}}+2} M_{\mathcal{Y}}^2 2^{d_{\mathcal{Y}}+2}} \log m}{\epsilon}.$$

This rate in Theorem 5, to our knowledge, establishes the state-of-the-art for FOMs with $O(\frac{1}{\epsilon})$ convergence rate for EFGs.

3.6.1 Improvements in extensive-form game convergence rate

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 EGT and MP. Compared to the rate obtained by [101], we get the following improvement: for simplicity, assume that the number of actions available at each information set is on average a, then our bound improves the convergence rate of [101] by a factor of $\Omega(d_{\mathcal{X}} \cdot a^{d_{\mathcal{X}}} + d_{\mathcal{Y}} \cdot a^{d_{\mathcal{Y}}})$.

As mentioned previously, Hoda et al. [79] proved only explicit bounds for the special case of uniform treeplexes that are constructed as follows: 1) A base treeplex Q_b 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. [79] for a uniform treeplex with d uniform treeplex levels (the total depth of the constructed treeplex is $d \cdot d_{Q_b}$, where d_{Q_b} is the depth of the base treeplex Q_b) 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 Q becomes

$$\frac{\Omega}{\varphi} \leq O\left(|S_Q|^2 d_Q^2 \log m\right).$$

Even for the special case of a uniform treeplex with a base simplex, comparing Theorem 4 to their bound, we see that our general bound improves the associated constants by exchanging $O(|S_Q|^2 d_Q^2)$ with $O(M_Q^2 2^{d_Q})$. Since M_Q does not depend on the branching operation in the

treeplex, whereas $|S_Q|$ does, our bounds are also the first bounds to remove an exponential dependence on the branching operation (we have only a logarithmic dependence). In Example 1 we showed that there exist games where $M_Q = \sqrt{|S_Q|}$, and in general M_Q is much smaller than $|S_Q|$. Consequently, our results establish the best known convergence results for all FOMs based on dilated entropy DGF such as EGT, MP, and stochastic variants of FOMs for BSPPs.

Gilpin et al. [72] 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 $\delta(A)$ is only shown to be finite for these games. Without any such unknown quantities based on condition numbers, Gilpin et al. [72] establish a convergence rate of $O(\frac{\|A\|_{2,2} \cdot D}{\epsilon})$. 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 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 $O(M_Q^2)$. In addition, as compared with previous work, the authors also only show experiments with their algorithm on instances with $9 \cdot 10^6$ leaf nodes [72], whereas the previous EGT algorithm showed experiments on instances with up to $4 \cdot 10^{12}$ leaf nodes.

CFR, CFR$^+$, and EGT all need to keep track of a constant number of current and/or average iterates, so the memory usage of all three algorithms is of the same order. When gradients are computed using tree traversal as opposed to storing the matrix A (or some decomposition thereof), each of these algorithms require a constant times the number of sequences in the sequence-form representation; in particular, each algorithm needs to keep track of some current $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ iterate, as well as a small number of gradients and intermediate solutions. Therefore, we compare mainly the number of iterations required by each algorithm. Since the theoretical properties of CFR and CFR$^+$ are comparable, we compare to CFR, with all statements being valid for CFR$^+$ as well.

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 [108]). Since our results utilizing EGT have a quadratic dependence on M_Q^2, CFR sometimes has a better dependence on game constants and can be more attractive for obtaining low-quality solutions quickly for games with many information sets. However, our theoretical results could be coupled with a $O(\frac{1}{\epsilon^2})$ convergence rate FOM such as mirror descent in order to achieve a similar dependence on game constants. In practice, a sampling-based variant of CFR called Monte-Carlo CFR (MCCFR) is preferred for certain applications [108]. MCCFR and CFR have a similar convergence rate, though MCCFR has cheaper iterations. Using a gradient-sampling method that we will describe in Section 3.10, our theoretical results can be utilized with the stochastic mirror prox algorithm [87] in order to achieve the same cost per iteration and convergence rate as MCCFR.

3.7 Smoothed best responses

We now show how to solve (3.3) and (3.4) for our DGF. While it is known that the more general class of dilated entropy DGFs has a closed-form solution, this is the first time the approach has been formally given. Furthermore, we believe that our particular solution is novel, and leads to better control over numerical issues. The problem we wish to solve is the following.

$$\operatorname{argmin} \sum_{j \in S_Q} \langle q^j, g_j \rangle + \beta_j q_{p_j} d_j(q^j / q_{p_j}) = \operatorname{argmin} \sum_{j \in S_Q} q_{p_j} (\langle \bar{q}^j, g_j \rangle + \beta_j d_j(\bar{q}^j)) \tag{3.18}$$

where the equality follows by the fact that $q_i = q_{p_j} \bar{q}_i$. For a leaf simplex j, its corresponding term in the summation has no dependence on any other part of the game tree except for the multiplication by x_{p_j} (because none of its variables are parent to any other simplex). Because of this lack of dependence, the expression

$$\langle \bar{q}^j / q_{p_j}, g_j \rangle + \beta_j d_j(q^j / q_{p_j})$$

can be minimized independently as if it were an optimization problem over a simplex with variables $\bar{q}^j = x^j / q_{p_j}$ (this was also pointed out in Proposition 3.4 in Hoda et al. [79]). We show how to solve the optimization problem at a leaf: $\min_{\bar{q}^j \in \Delta_j} \langle \bar{q}^j, g_j \rangle + \beta_j d_j(\bar{q}^j)$. Writing the Lagrangian with respect to the simplex constraint and taking the derivative wrt. $\bar{q}_i$ gives

$$\min_{\bar{q}^j} \langle \bar{q}^j, g_j \rangle + \beta_j d_j(\bar{q}^j) + \lambda(1 - \sum_{i \in \mathbb{I}_j} \bar{q}_i) \Rightarrow g_i + \beta_j(1 + \log \bar{q}_i) = \lambda \Rightarrow \bar{q}_i \propto e^{-g_i/\beta_j}$$

This shows how to solve the smoothed-best-response problem at a leaf. For an internal simplex j, Proposition 3.4 of Hoda et al. [79] says that we can simply compute the value at all simplexes below j, and propagate the value up into g_j (this is easily seen from (3.18); each q_i acts as a scalar on the value of all simplexes after i). Letting $|\mathbb{I}_j| = n$, we now simplify the objective function:

$$\begin{aligned} &\langle \bar{q}^j, g_j \rangle + \beta_j (\sum_{i \in \mathbb{I}_j} (\bar{q}_i \log \bar{q}_i) + \log n) = \sum_i (\bar{q}_i (g_i + \beta_j \log \bar{q}_i)) + \beta_j \log n \\ = &\sum_i (\bar{q}_i (\lambda - \beta_j)) + \beta_j \log n = \lambda - \beta_j + \beta_j \log n, \end{aligned}$$

where the last two equalities follow first by applying our derivation for λ and then the fact that $\bar{q}^j$ sums to one. This shows that we can choose an arbitrary index $i \in \mathbb{I}_j$ and propagate the value $g_i + \beta_j \log \bar{q}_i + \beta_j \log n$. In particular, for numerical reasons we choose the one that maximizes $\bar{q}_i$.

In addition to smoothed best responses, fast FOMs usually also require computation of proximal mappings, which are solutions to $\operatorname{argmin}_{q \in Q} \langle q, g \rangle + D(q \| q')$, where $D(q \| q') = d(q) - d(q') - \langle \nabla d(q'), q - q' \rangle$ is the Bregman divergence associated with the chosen DGF d. Unlike the smoothed best response, we are usually only interested in the minimizing solution and not the associated value. Therefore we can drop terms that do not depend on q and the problem reduces to $\operatorname{argmin}_{q \in Q} \langle q, g \rangle + d(q) - \langle \nabla d(q'), q \rangle$, which can be solved with our smoothed

best response approach by using the shifted gradient $\tilde{g} = g - \nabla d(q')$. This has one potential numerical pitfall: the DGF-gradient $\nabla d(q')$ may be unstable near the boundary of Q, for example because the entropy DGF-gradient requires taking logarithms. It is possible to derive a separate expression for the proximal mapping that is similar to what we did for the smoothed best response; this expression can help avoid this issue. However, because we only care about getting the optimal solution, not the value associated with it, this is not necessary. The large gradients near the boundary only affect the solution by setting bad actions too close to zero, which does not seem to affect performance.

3.8 Small and medium-scale numerical experiments

We carried out numerical experiments to investigate the practical performance of EGT on EFGs when instantiated with our DGF. We start out by comparing our DGF (henceforth referred to as new DGF) with that of Kroer et al. [101] (a preliminary version of the work presented here, henceforth referred to as old DGF) when used in the EGT algorithm, and then we compare EGT equipped with our DGF to CFR and CFR$^+$, the practical state-of-the-art EFG-solving algorithms.

We consider two variants of EGT: the original algorithm of Nesterov [132] as is, and a new variant where we incorporate several heuristics for speeding up practical convergence by avoiding overly pessimistic parameters. We demonstrate and discuss the practical convergence issues in comparison experiments later. First we describe our practical variant.

Our preliminary experiments for EGT demonstrated that the initial values for the smoothing parameters μ_1, μ_2 are much too conservative in practice. Instead, in our aggressive EGT we follow an automated tuning procedure. The goal of this procedure is to find a pair of initial smoothing parameters μ_1, μ_2 that fit the problem instance at hand. At the beginning of the algorithm we perform a binary search over 16 logarithmically-spaced numbers between $\frac{0.001}{\Omega_x}$ and 1.0 for μ_1. For each number, we binary search 3 multiples of μ_1 in order to choose μ_2: $0.75, 1$, and 1.25. We choose the smallest pair of parameters that give an excessive gap greater than 0.001 after computing x^0, y^0 in Algorithm 1.

Because we are choosing the smoothing parameters this way, we are no longer guaranteed convergence according to the EGT theory; however we can fix this problem by numerically checking the excessive gap condition at every iteration. We perform this check as a part of the following more aggressive step-sizing policy: instead of setting $\tau = \frac{2}{t+3}$ at every iteration t, we use the *aggressive μ reduction* heuristic of Hoda et al. [79], where a constant stepsize is used, and then whenever the post-step check of the excessive gap condition fails the step is retraced and τ is halved. We start τ at $\frac{1}{2}$. In addition to only decreasing τ when the excessive gap check fails, we also increase it by a factor of 1.11 whenever a step was successful. We also introduce a new heuristic for checking whether a step was successful: we check whether the saddle-point residual $\epsilon_{\text{sad}}(z)$ deteriorated by a factor of more than 1.1 after each step. If it did, we retrace and halve τ. This heuristic can be computed essentially for free (in particular using the gradients from the excessive gap check) and thus only requires two treeplex traversals.

We test the algorithms on two games. The first game is a scaled up variant of the poker game Leduc holdem [157], 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. We consider decks with 6, 30, and 70 cards. The smallest game has about 2000 nodes in the game tree, and the largest has about 3.2 million nodes in the game tree.

The second game is a zero-sum variant of a search-game played on the graph shown in Figure 3.2, we will refer to it as *Search*. Search is a simultaneous-move game (which can be modeled as a turn-taking EFG with appropriately chosen information sets). A defender controls two patrols that can each move within their respective shaded areas (labeled P1 and P2), and at each time step the controller chooses a move for both patrols. An attacker tries to move from the S node to one of the three payoff nodes. The attacker can move freely to any adjacent node (except at patrolled nodes, the attacker cannot move from a patrolled node to another patrolled node). The attacker can also choose to wait in place for a time step in order to clean up their traces. If a patrol visits a node that was previously visited by the attacker, and the attacker did not wait to clean up their traces, they can see that the attacker was there. If the attacker reaches any of the rightmost nodes they received the respective payoff at the node (5, 10, or 3, respectively) and the defender loses that amount. If the attacker and any patrol are on the same node at any time step, the attacker is captured, which leads to payoffs of 1 and -1 for the defender and attacker respectively. Finally, the game times out after 5 simultaneous moves, in which case both players receive a payoff 0. Search has 87,927 nodes and 11,830 and 69 defender and attacker sequences. General-sum variants of this game were studied by Bošanskỳ et al. [17], Bošanskỳ and Čermák [18]. One particularly noteworthy feature of this game is that the strategy spaces are extremely imbalanced: it is huge for one player and tiny for the other. This is in stark contrast with Leduc (and other poker games) where the strategy spaces are almost the same. In addition to our general results that we are about to present, we believe that our results are the first of their kind in demonstrating strong FOM performance on such imbalanced EFGs.

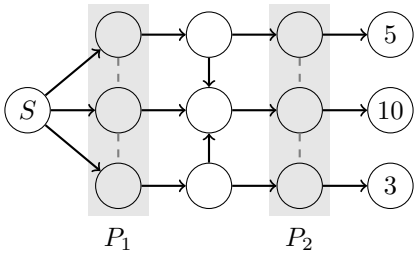


Figure 3.2: A search game between a defender and an attacker.

In our comparisons, we use loglog plots that show the results for a particular game. In each plot, we show the performance of the algorithms, with the x-axis showing the number of tree traversals, and the y-axis showing the solution accuracy, i.e. the saddle-point residual. Tree-traversals are a good proxy for overall computational effort because the majority of the time in the algorithms EGT and CFR algorithms for EFG solving is spent on gradient computations, which in our case directly translates into tree-traversals. All EGT experiments include the tree

traversals needed for automated parameter tuning described in the previous paragraph as part of the number of tree traversals performed by the algorithm.

First, we investigate the impact of applying the weights used in recurrence (3.6), as compared to our preliminary scheme introduced in Kroer et al. [101]. To instantiate recurrence (3.6) we have to choose a way to set β_j relative to α_j. We use the scheme of Corollary 1. This scheme will henceforth be referred to as new weights. We compare these new weights to the weights used in Kroer et al. [101] (henceforth referred to as old weights). Figure 3.3 shows the result of running EGT with the old and the new weights for Leduc with a 6-card (on the left) deck and Search (on the right). The top row shows results when instantiating EGT with the parameters dictated by the theory, whereas the bottom row shows EGT using all our heuristics. When instantiating parameters according to theory the most important observation is that the parameters are way too pessimistic, out of thousands of gradient computations, the majority are spent decreasing the smoothing parameters until they are small enough that the algorithms start to make progress. That said, this happens significantly faster for the new parameters than the old ones; for the search game $20,000$ gradient computations is not enough to start progressing with the old parameters. For our aggressive EGT variant we find that both DGFs perform much better, though our new weights still perform better on both games, significantly so for Search.

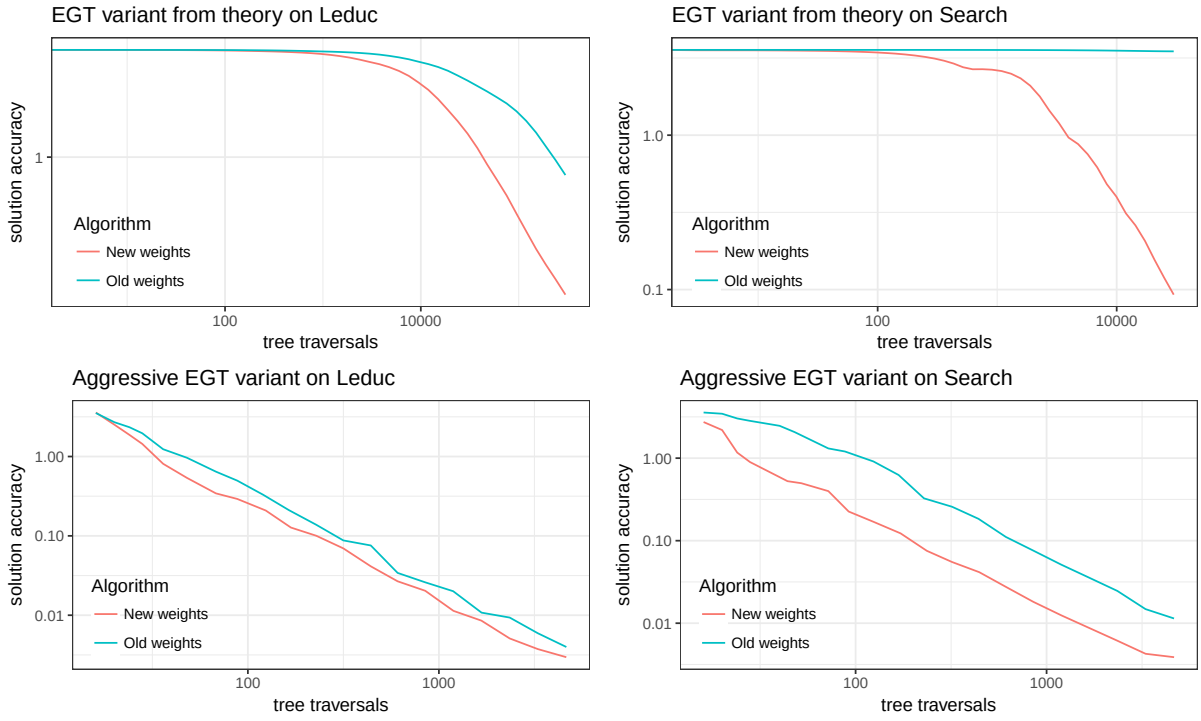


Figure 3.3: Solution accuracy as a function of the number of iterations for EGT with our weighting scheme (New weights) and with the weighting scheme from Kroer et al. [101] (Old weights). Both axes are on a log scale. The top row shows the effect of our weighting scheme when using EGT instantiated according to the original theory. The bottom row shows the effect when using our aggressive EGT variant.

We next compare the performance of EGT with our new weights to that of the CFR and CFR$^+$

algorithms on three Leduc variants (6, 30, and 70-card decks) and Search. For CFR we use two variants: the vanilla CFR algorithm with RM as the regret minimizer, and CFR with the RM$^+$ regret minimizer and alternating minimization. Finally we have CFR$^+$ which adds linear stepsizing on top of RM$^+$ and alternating minimization. The results are shown in Figure 3.4. We find that EGT instantiated with our DGF outperforms CFR with both RM and RM$^+$ and alternating minimization, whereas CFR$^+$ is slightly faster still. EGT maintains a stronger convergence rate across all iterations. It starts out slightly worse because its first iterate is shifted outward on the x-axis due to paying the upfront cost of our automated tuning based initialization. However, its convergence rate is immediately superior to CFR, and almost immediately overtakes both CFR algorithms.

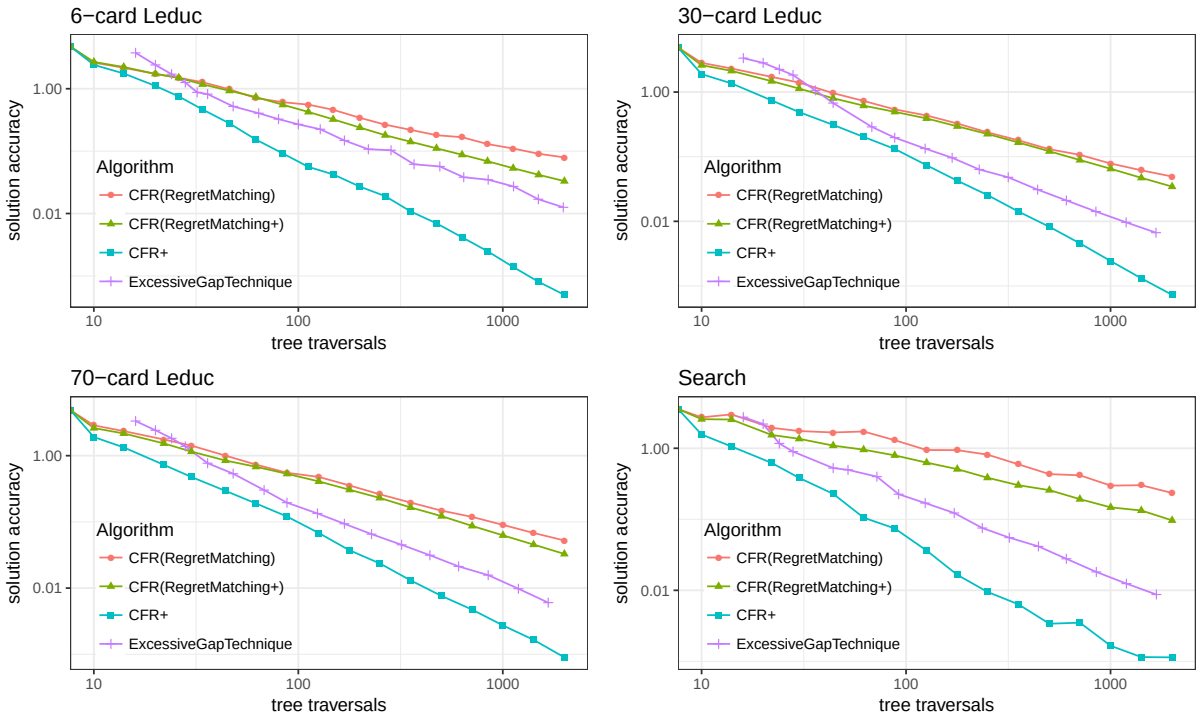


Figure 3.4: Solution accuracy as a function of the number of tree traversals in three different variants of Leduc hold'em and the Search game. Results are shown for CFR with regret mathing, CFR with regret mathing$^+$, CFR$^+$, and our aggressive EGT algorithm. Both axes are shown on a log scale.

The performance we get from EGT (with aggressive stepsizing) relative to CFR is in sharp contrast to the previous conventional practical wisdom in the field. In our preliminary conference paper [101] it was found that, while EGT has better asymptotic convergence rate, CFR had better initial convergence rate, and it was only after a certain number of iterations that EGT took over. Furthermore, the switching point where EGT is preferable was found to shift outward on the x-axis as the Leduc game size was increased. This sentiment has been mirrored by Brown and Sandholm [25]. In contrast to this, we find that our DGF along with proper initialization leads to EGT having a better convergence rate than not only CFR, but also CFR with RM$^+$. Furthermore, scaling up the game size does not seem to adversely affect this relationship.

The numerical results in Figure 3.4 suggest that our DGF may be useful in practice for solving large-scale zero-sum EFGs. Moreover, these results are with a particular FOM, EGT, and there is a myriad of possible ways that our DGF could be combined with other FOMs.

3.9 Large-scale numerical GPU experiments

We now present experimental results on running all the previously described algorithms on a GPU. All experiments were run on a Google Cloud instance with an NVIDIA Tesla K80 GPU with 12GB available. All code was implemented in C++ using CUDA for GPU operations, and cuSPARSE for the sparse payoff matrix. We compare against several CFR variants.[4] We run CFR with RM (CFR(RM)), RM$^+$ (CFR(RM$^+$)), and CFR$^+$ which is CFR with RM$^+$ and a linear averaging scheme. Detailed descriptions can also be found in Zinkevich et al. [179] and Tammelin et al. [161].

As in the previous section we use a practical EGT variant combining practical techniques from the previous section. The pseudocode is shown in Algorithm 1. As before we use a practically-tuned initial choice for the initial smoothing parameters μ. Furthermore, rather than alternating the steps on players 1 and 2, we always call STEP on the player with a higher μ value (this choice is somewhat reminiscent of the μ-balancing heuristic employed by Hoda et al. [79] although our approach avoids an additional fitting step). The EGT algorithm with a practically-tuned μ and this μ balancing heuristic will be denoted EGT in our experiments. In addition, we use an EGT variant that employs the *aggressive μ reduction* technique introduced by Hoda et al. [79]. Aggressive μ reduction uses the observation that the original EGT stepsize choices, which are $\tau = \frac{2}{3+t}$, are chosen to guarantee the excessive gap condition, but may be overly conservative. Instead, aggressive μ reduction simply maintains some current τ, initially set to 0.5, and tries to apply the same stepsize τ repeatedly. After every step, we check that the excessive gap condition still holds; if it does not hold then we backtrack, τ is decreased, and we repeat the process. A τ that maintains the condition is always guaranteed to exist by Theorem 2 of Nesterov [132]. The pseudocode for this is given in Algorithm 2. EGT with aggressive μ reduction, a practically tuned initial μ, and μ balancing, will be denoted EGT/AS in our experiments.

[4] All variants use the alternating updates scheme.

Algorithm 1 EGT/AS(DGF-center x_ω, DGF weights μ_x, μ_y, and $\epsilon > 0$)

$x^0 = \nabla d^*_{\mathcal{X}} \left(\mu_x^{-1} \nabla f_{\mu_y}(x_\omega)\right)$
$y^0 = y_{\mu_y}(x_\omega)$
$t = 0$
$\tau = \frac{1}{2}$
while $\epsilon_{\text{sad}}(x^t, y^t) > \epsilon$ **do**
  if $\mu_x > \mu_y$ **then**
    $(\mu_x^{t+1}, x^{t+1}, y^{t+1}, \tau) = \text{DECR}(\mu_x^t, \mu_y^t, x^t, y^t, \tau)$
  else
    $(\mu_y^{t+1}, y^{t+1}, x^{t+1}, \tau) = \text{DECR}(\mu_y^t, \mu_x^t, y^t, x^t, \tau)$
  $t = t + 1$
return x^t, y^t

Algorithm 2 DECR(μ_x, μ_y, x, y, τ)

$(\mu_x^+, x^+, y^+) = \text{STEP}(\mu_x, \mu_y, x, y, \tau)$
while $\text{EGV}(x, y) < 0$ **do**
  $\tau = \frac{1}{2}\tau$
  $(\mu_x^+, x^+, y^+) = \text{STEP}(\mu_x, \mu_y, x, y, \tau)$
return $\mu_x^+ x^t, y^t, \tau$

To compute smoothed best responses, we use a parallelization scheme. We parallelize across the initial Cartesian product of treeplexes at the root. As long as this Cartesian product is wide enough, the smoothed best response computation will take full advantage of parallelization. This is a common structure in real-world problems, for example representing the starting hand in poker, or some stochastic private state of each player in other applications. This parallelization scheme also works for gradient computation based on tree traversal. However, in these experiments we do gradient computation by writing down a sparse payoff matrix using CUDA's sparse library and let CUDA parallelize the gradient computation.

For poker-specific applications (and certain other games where utilities decompose nicely based on private information) it is possible to speed up the gradient computation substantially by employing the accelerated tree traversal of Johanson et al. [82]. We did not use this technique. In our experiments, the majority of time is spent in gradient computation, so this acceleration is likely to affect all the tested algorithms equally. Furthermore, since the technique is specific to games with certain structures, our experiments give a better estimate of general EFG-solving performance.

Our experiments are conducted on real large-scale "river" endgames faced by the *Libratus* AI [27]. *Libratus* was created for the game of heads-up no-limit Texas hold'em. *Libratus* was constructed by first computing a "blueprint" strategy for the whole game (based on abstraction and Monte-Carlo CFR [108]). Then, during play, *Libratus* would solve endgames that are reached using a significantly finer-grained abstraction. In particular, those endgames have no card abstraction, and they have a fine-grained betting abstraction. For the beginning of the subgame, the blueprint strategy gives a conditional distribution over hands for each player. The

subgame is constructed by having a Chance node deal out hands according to this conditional distribution.[5]

A subgame is structured and parameterized as follows. The game is parameterized by the conditional distribution over hands for each player, current pot size, board state (5 cards dealt to the board), and a betting abstraction. First, Chance deals out hands to the two players according to the conditional hand distribution. Then, Libratus has the choice of folding, checking, or betting by a number of multipliers of the pot size: 0.25x, 0.5x, 1x, 2x, 4x, 8x, and all-in. If Libratus checks and the other player bets then Libratus has the choice of folding, calling (i.e. matching the bet and ending the betting), or raising by pot multipliers 0.4x, 0.7x, 1.1x, 2x, and all-in. If Libratus bets and the other player raises Libratus can fold, call, or raise by 0.4x, 0.7x, 2x, and all-in. Finally when facing subsequent raises Libratus can fold, call, or raise by 0.7x and all-in. When faced with an initial check, the opponent can fold, check, or raise by 0.5x, 0.75x, 1x, and all-in. When faced with an initial bet the opponent can fold, call, or raise by 0.7x, 1.1x, and all-in. When faced with subsequent raises the opponent can fold, call, or raise by 0.7x and all-in. The game ends whenever a player folds (the other player wins all money in the pot), calls (a showdown occurs), or both players check as their first action of the game (a showdown occurs). In a showdown the player with the better hands wins the pot. The pot is split in case of a tie. (For our experiments we used endgames where it is *Libratus*'s turn to move first.)

We conducted experiments on two river endgames extracted from *Libratus* play: Endgame 2 and Endgame 7. Endgame 2 has a pot of size 2100 at the beginning of the river endgame. It has 140k and 144k sequences for *Libratus* and the opponent, respectively, and 176M leaves in the games tree. Endgame 7 has a pot of size $3750 at the beginning of the river subgame. It has 43k and 86k sequences for the players, and 54M leaves.

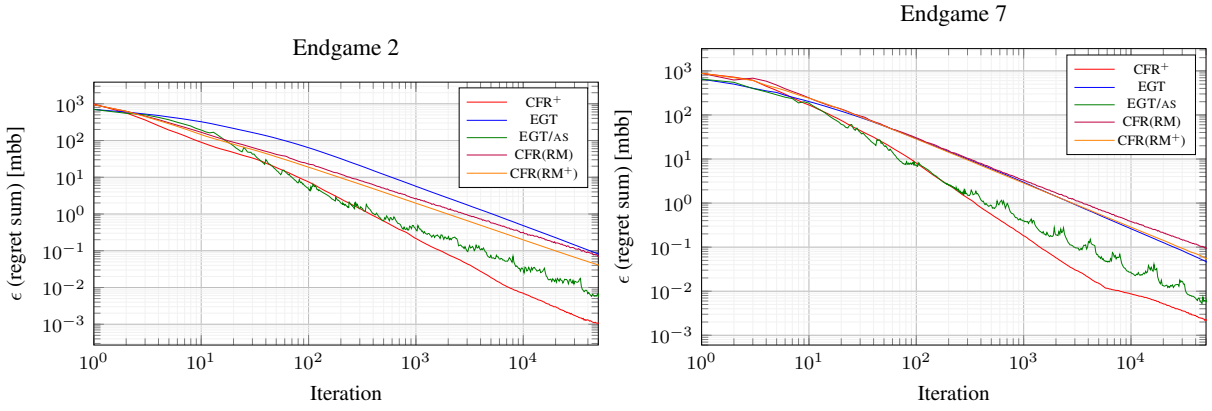


Figure 3.5: Solution quality as a function of the number of iterations for all algorithms on two river subgames. The solution quality is given as the sum of regrets for the players in milli-big-blinds.

In the first set of experiments we look at the per-iteration performance of each algorithm. The results are shown in Figure 3.5. The y-axis shows the sum of the regrets for each player, that is,

[5] *Libratus* used two different subgame-solving techniques, one "unsafe" and one "safe" [26]. The computational problem in the two is essentially identical. We experiment with the "unsafe" version, which uses the prior distributions described here.

how much utility they can gain by playing a best response rather than their current strategy. The unit is milli-big-blinds (mbb); at the beginning of the original poker game, *Libratus*, as the "big blind", put in \$100 and the opponent put in \$50, in order to induce betting. Mbb is a thousandth of the big blind value, that is, 10 cents. This is a standard unit used in research that uses poker games for evaluation. One mbb is often considered the convergence goal. CFR$^+$ and EGT/AS perform the best; both reach the goal of 1mbb after about 400 iterations in both Endgame 2 and 7. EGT, CFR(RM), and CFR(RM$^+$) all take about 3000 iterations to reach 1mbb in Endgame 7. In Endgame 2, EGT is slowest, although the slope is steeper than for CFR(RM) and CFR(RM$^+$). We suspect that better initialization of EGT could lead to it beating both algorithms. Note also that EGT was shown better than CFR(RM) and CFR(RM$^+$) by Kroer et al. [103] in the smaller game of Leduc hold'em with an automated μ-tuning approach. Their results further suggest that better initialization may help enhance converge speed significantly.

One issue with per-iteration convergence rates is that the algorithms do not perform the same amount of work per iteration. All CFR variants in our experiments compute 2 gradients per iteration, whereas EGT computes 3, and EGT/AS computes 4 (the additional gradient computation is needed in order to evaluate the excessive gap). Furthermore, EGT/AS may use additional gradient computations if the excessive gap check fails and a smaller τ is tried (in our experiments about 15 adjustments were needed). In our second set of plots, we show the convergence rate as a function of the total number of gradient computations performed by the algorithm. This is shown in Figure 3.6. By this measure, EGT/AS and EGT perform slightly worse relative to their performance as measured by iteration count. In particular, CFR$^+$ takes about 800 gradient computations in order to reach 1mbb in either game, whereas EGT/AS takes about 1800.

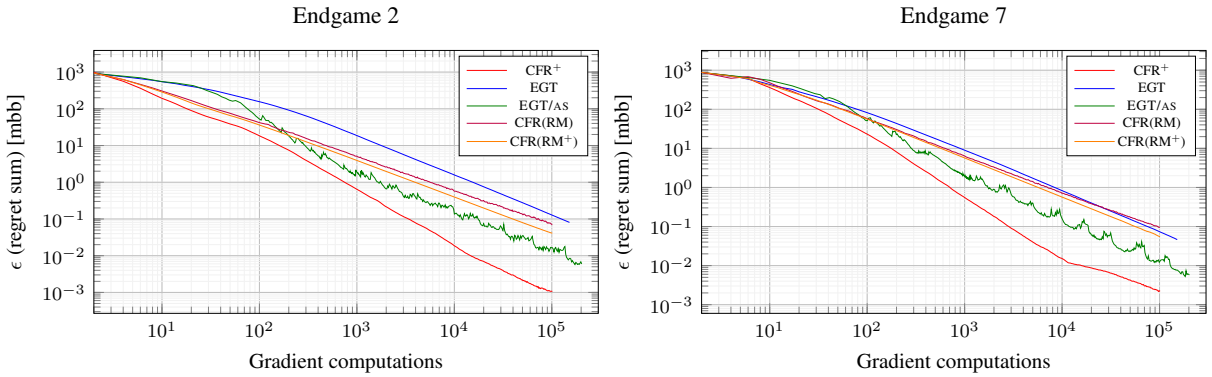


Figure 3.6: Solution quality as a function of the number of gradient computations for all algorithms on two river subgames. The solution quality is given as the sum of regrets for the players in milli-big-blinds.

In our experiments CFR$^+$ vastly outperforms its theoretical convergence rate (in fact, every CFR variant does significantly better than the theory predicts, but CFR$^+$ especially so). However, CFR$^+$ is known to eventually reach a point where it slows down and performs worse than $\frac{1}{T}$. In our experiments we start to see CFR$^+$ slowing down towards the end of Endgame 7. EGT, in contrast, is guaranteed to maintain a rate of $\frac{1}{T}$, and so may be preferable if a guarantee against slowdown is desired or high precision is needed.

In addition to the results presented here, we tried several variants of our weighting scheme,

using more or less aggressive weight increases as we move up the treeplex. Here we present only the results for the best variant. In future work it would be interesting to more thoroughly explore the space of possible instantiations of recurrence 3.6.

3.10 Sampling

In some practical settings each iteration of a FOM might be prohibitively expensive, usually because the gradient is expensive to compute. Instead it may be preferable to use a gradient estimator. For example, in the Libratus agent [27] a gradient estimator was used for constructing a precomputed base strategy (which was also computed in an abstraction), while full tree traversals were used when solving subgames in real time.

In order to facilitate gradient estimation we can use stochastic FOMs that work with unbiased estimates of the gradient. Juditsky et al. [87] discuss a stochastic variant of mirror prox for BSSP, namely *Stochastic mirror prox* (SMP). SMP does not use exact gradients when computing strategy updates, but rather unbiased estimators thereof. Specifically, for every $x \in \mathcal{X}$, we assume access to a probability distribution Π_x such that $\mathbb{E}_{\eta\sim\Pi_x}[\eta] = x$. Similarly, we assume access to P_y for $y \in \mathcal{Y}$ such that $\mathbb{E}_{\xi\sim P_y}[\xi] = y$. We define variance as follows:

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

Juditsky et al. [87] 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:

$$\mathbb{E}\left[\epsilon_{sad}(z^T)\right] \le \max\left[\frac{7\Omega\mathcal{L}}{2T\varphi}, 7\sqrt{\frac{2\Omega(\sigma_x^2+\sigma_y^2)}{3T\varphi}}\right]. \tag{3.19}$$

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 $\xi^T A$ 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. [108] 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

$$\mathcal{C}_{c,k}(h) = \begin{cases} k & \text{if } h \text{ is a chance node} \\ all & \text{else} \end{cases}. \tag{3.20}$$

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

Proof. We will show this for σ_x^2; the proof is analogous for σ_x^2. Our estimator can be thought of as $\eta = A^\top \xi_x$, where ξ_x is the vector that would be obtained by only writing down the probability at terminal nodes in the definition of $\mathbf{Sample}(h, \tau)$. Then using the definition of the conjugate norm, we have

$$\sigma_x^2 = \sup_{x \in \mathcal{X}} \mathbb{E}\left\{ \| A^\top \left[\xi_x - x\right] \|_{y,*}^2 \right\} \leq \sup_{x \in \mathcal{X}} \mathbb{E}\left\{ \max_{\|s\|_y \leq 1} \left\langle A^\top \left[\xi_x - x\right], s \right\rangle^2 \right\}. \tag{3.21}$$

Now we use the fact that $\mathbf{Sample}(h, \tau)$ never sets any entry in η to be greater than the largest payoff in the game, and likewise never lower than the smallest payoff in the game (we can assume without loss of generality that 0 is a payoff in the game): Because of perfect recall, the only way to reach the same index in η more than once is through one of the two bottom else statements

in the definition of **Sample**(h, τ), but both cases decrease τ such that the values sum to one. Likewise, $A^\top x$ is never component-wise larger or smaller than the maximum and minimum payoff in the game. Thus, we get that $A^\top [\xi_x - x]$ is component wise less than Γ_2. Letting $\mathbf{1}$ denote that all-ones vector of appropriate dimension, we get

$$(3.21) \leq \sup_{x \in \mathcal{X}} \mathbb{E} \left\{ \max_{\|s\|_y \leq 1} \langle \Gamma_2 \cdot \mathbf{1}, s \rangle^2 \right\} = \Gamma_2^2. \tag{3.22}$$

□

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

Theorem 7. *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*

$$\mathbb{E}\left[\epsilon_{sad}(z^T)\right] \leq \max \left[\frac{7\mathcal{L} M_{\mathcal{Z}}^2 2^{d_{\mathcal{Z}}+2} \log m}{2T}, \; 7\sqrt{\frac{2M_{\mathcal{Z}}^2 2^{d_{\mathcal{Z}}+2} \log m (\Gamma_1^2 + \Gamma_2^2)}{3T}} \right]. \tag{3.23}$$

This is the strongest known bound on algorithms for computing Nash equilibria in two-player zero-sum EFGs via stochastic methods. Unfortunately preliminary practical experiments found that SMP performed significantly worse than MCCFR in practice, and so more practical engineering work needs to be done to find a strong practical candidate for stochastic EFG solving via FOMs.

Unfortunately preliminary experiments found that the stochastic mirror prox method is not competitive with the MCCFR algorithm in spite of its superior theoretical properties. Thus more practical engineering work is needed in order to make stochastic FOMs useful. In future work it would be interesting to investigate whether other stochastic FOMs can be leveraged to get better practical performance [107, 131, 175].

3.11 Conclusions and future work

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 $O(|S_Q| M_Q d 2^d \log m)$ to $O(M_Q^2 2^{d_Q+2} \log m)$, thereby removing all but a logarithmic dependence on branching associated with the branching operator in the treeplex definition. These results lead to significant improvements in the theoretical convergence rates of FOMs that can be equipped with dilated entropy DGFs and used for EFG solving including, but not limited to, EGT, MP, and Stochastic MP.

We numerically investigated the performance of EGT and compared it to the practical state-of-the-art algorithms CFR and CFR$^+$. Our experiments showed that EGT equipped with the

dilated entropy DGF, when tuned with a proper scaling, has better practical, as well as theoretical, convergence rate than CFR even with RM$^+$, as opposed to the cross-over point phenomena based on the size of the games. While CFR$^+$ is still faster, our results are for a specific FOM instantiated with our DGF; it seems likely that future experimental work could lead to even faster algorithms based on our DGF, for example by incorporating randomization to reduce the cost of gradient computation, or other FOMs such as mirror prox [130] or the primal-dual algorithm by Chambolle and Pock [39].

Theorems 2 and 3 establish bounds for a general class of weights β_j satisfying the recurrence (3.6). Then in Corollary 1, we have selected a particular weighting scheme for β_j satisfying (3.6) and performed our numerical tests. There may be other interesting choices of β_j satisfying the recurrence (3.6). Thus, finding a way to optimally choose among the set of weights satisfying (3.6) to minimize the polytope diameter for specific games is appealing.

On a separate note, in practice CFR is often paired with an abstraction technique [148]. This is sometimes despite the lack of any theoretical justification. Effective ways to pair FOMs such as MP and EGT with practical abstraction techniques [28] or abstraction techniques that achieve solution-quality guarantees, such as those discussed in the abstraction section of this thesis, are also worth further consideration.

von Stengel and Forges [166] introduce an extension of the sequence-form LP to the computation of extensive-form correlated equilibria. The dilated entropy approach could potentially be extended to this setting, leading to a smoothing method for the correlated strategy space. In that case it would be interesting to find a formulation of the extensive-form correlated equilibrium problem that is amenable to FOMs.

The smoothing method presented in this chapter has no inherent dependence on being for two-player-zero-sum games; such games are simply the best current application of the smoothing method. For example, recent work has investigated the formulation of reinforcement learning as a first-order optimization problem amenable to methods such as mirror prox and other proximal algorithms [13, 115, 116, 118].

Chapter 4

Abstraction for large general-sum games

Initially, game abstractions were created by hand, using domain dependent knowledge [15, 155]. More recently, automated abstraction has taken over [66, 68, 179]. This has typically been used for information abstraction, whereas action abstraction is still largely done by hand [71, 153]. Recently, automated action abstraction approaches have also started to emerge [23, 24, 75, 76, 152].

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 [68] 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 [168]! Until recently, all lossy abstraction techniques for general games of imperfect information were without any solution quality bounds. Basilico and Gatti [6] give bounds for the special game class *Patrolling Security Games*. Johanson et al. [84] 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. [109] 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. [171] and Brown and Sandholm [24] 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 [152] 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 [109].

4.1 Introduction

Game-theoretic equilibria have played a key role in several recent advances in the ability to construct AIs with superhuman performance in games with imperfect information [21, 27, 124]. In particular these results rely on computing an approximate *Nash equilibrium* [126] for the game at hand. In typical real-world situations these games are so large that even approximate equilibria are intractable. Instead, the dominant paradigm has been to first construct some smaller *abstraction* of the game, apply an iterative algorithm for computing a Nash equilibrium in the abstraction, and map the resulting strategy back to the full game. This approach was used in the recent *Libratus* agent, which beat four top poker pros in the game of heads-ups no-limit Texas hold'em [27] (in addition to abstraction and equilibrium approximation the agent also utilized real-time subgame solving [26] and action abstraction refinement). Abstraction has also been used in trading-agent competitions [173] and security games [4, 5, 6].

In practice, abstractions are generated heuristically with no theoretical guarantees on solution quality [15, 28, 64, 66, 67, 69, 70, 71, 75, 76, 84, 149, 155]. Ideally, abstraction would be lossless, such that implementing an equilibrium from the abstract game results in an equilibrium in the full game. Gilpin and Sandholm [68] study lossless abstraction techniques for a structured class of games. Unfortunately, lossless abstraction often leads to games that are still too large to solve. Thus, one must turn to lossy abstraction. However, significant abstraction *pathologies* (*nonmonotonicities*) have been shown in games which cannot exist in single-agent settings: if an abstraction is refined, the equilibrium strategy from that new abstraction can be worse in the original game than the equilibrium strategy from a coarser abstraction [168]! Lossy abstraction remains poorly understood from a theoretical perspective. Results have been obtained only for various restricted models of abstraction. Basilico and Gatti [6] give bounds for the special game class called *patrolling security game*. Sandholm and Singh [152] provide lossy abstraction algorithms with bounds for stochastic games. Brown and Sandholm [23], Waugh et al. [171], Brown and Sandholm [24], and Čermák et al. [35, 37] develop iterative abstraction-refinement schemes that have various forms of converge guarantees but they do not give solution-quality guarantees for the original game for strategies computed in limited-size abstractions. Lanctot et al. [109] show that the *counterfactual regret minimization algorithm (CFR)* converges to an approximate NE when run on an imperfect-recall abstraction that is a *skew well-formed game* (SWF) with respect to the original game, where the error in the NE has a linear dependence on the number of information sets.

The work by Lanctot et al. [109] is most related to our work, as they also focus on abstraction of EFGs. However, their results are only for the narrow class of SWF games, and only for applying the CFR algorithm to the resulting abstraction. They assume that information sets (i.e., decision points) are aggregated into larger information sets. All pairs of information sets

that are aggregated together are compared by defining a mapping between subtrees under the information sets. This mapping then requires that the payoffs are similar, the distribution over chance outcomes is similar, and for pairs of leaves mapped to each other, the leaves have the same sequence of information-set-action pairs leading to them in the abstraction. Payoff and chance-outcome similarity is similar to what good practical abstraction algorithms seek to obtain. However, the requirement that information-set-action pairs are the same for leaf nodes mapped to each other is not satisfied by the best heuristic abstraction algorithms used in practice [28, 64, 84]. In this chapter we develop an exact decomposition of the solution-quality error that does not require any such assumption. This is the first decomposition of solution-quality error resulting from abstraction. This decomposition depends on several quantities that prior results did not (owing to its more general and exact nature). We then show that by making a weaker variant of previous assumptions, our decomposition can recover all previous solution-quality bounds. We show via counterexample that there exist games where the assumption on information-set-action pairs is, in a sense, necessary in order to avoid large abstraction error that is not measurable by the type of technique presented here and in prior work. We go on to define a generalization of SWF games called *chance-relaxed skew well-formed games* (CRSWF games) which, unlike SWF games, allows error in chance outcomes. We similarly define a class of perfect-recall abstractions that allow ex-post bounds on solution quality via our decomposition. For all our settings we prove bounds for exact Nash equilibria and strategies with bounded counterfactual regret. We also prove the first bounds for how ϵ-Nash equilibria computed in abstractions perform in the original game. This is important because often one cannot afford to compute an exact Nash equilibrium in the abstraction. All our results apply to general-sum n-player games. Finally we investigate the performance of our abstraction classes experimentally by computing bound-minimizing abstractions and their practical approximation to Nash equilibrium.

The rest of the chapter is structured as follows. Section 4.2 describes our overall model of abstraction, and the assumptions we make on how the abstraction and the original game are related. Section 4.3 describes how we measure differences between the original game and the abstraction. Section 4.4 gives our main result: an exact decomposition of the abstraction error based on our measures of difference. Section 4.5 specializes this decomposition result to the setting of perfect-recall abstraction and shows that under more stringent assumptions on the abstraction structure we can get ex-ante solution quality bounds. Section 4.6 similarly specializes to imperfect-recall abstraction. Finally Section 4.9 presents a model of EFGs with continuous action spaces and shows how our results on abstraction theory can be used to reason about discretization of continuous action spaces.

4.2 Game abstractions

We start by giving an intuitive description of how we model abstraction. We are given some perfect-recall EFG Γ for which we would like to compute a (possibly approximate) Nash equilibrium. Instead of solving Γ directly, we assume that we are given some *abstraction* of Γ called Γ'. Throughout we will assume that Γ' is itself an EFG, though it is allowed to be imperfect recall, unlike the original game. The high-level idea is to compute some approximate solution to Γ', and then use that approximate solution to construct a strategy for Γ. The type of approximate

solution computed for Γ' may vary. For example, computing an exact Nash equilibrium in Γ' may be overkill, since what we ultimately care about is how strong of a strategy profile we get in the original game. This is especially true when the abstraction has imperfect recall, in which case a Nash equilibrium is NP-hard to find, and it suffices to find a strategy with low counterfactual regret at every information set. We consider several notions of solution to the abstract game.

Once we have an abstraction and a solution thereof, the primary question that we ask in this chapter is whether we can construct a solution to the original game that is provably near-optimal. To answer this question we need a way to reason about the differences between the original game and the abstract game. We do this by setting up a mapping between the real game and the abstract game: every information set in the real game is assumed to map onto a specific abstract information set. The strategy that we construct for the real game is such that the distribution over actions at a given information set is constructed from the distribution over actions at the abstract information set that it maps onto. In order to analyze the quality of the obtained strategy we propose a two-step process for measuring differences between the real and abstract game: In the first step we think of the original game mapping onto an *information refinement* of the abstraction, where the refinement is the abstract game but with some abstract information sets refined into two or more new information sets. The information refinement has to be at least fine-grained enough to entail perfect recall, although it may be useful in practice to consider refinement even of perfect recall information sets. We set up measures of how different payoffs and probability distributions are in the original game versus in the refinement, where these measurements are based on how information sets and actions from the real game are mapped onto the refinement of the abstraction. In the second step, we measure the difference between the refinement and the abstract game. This is again done by measuring differences in payoffs and probability distributions, this time between each information set in the refinement and the larger abstract information set that it was refined from in the abstraction. This process is illustrated in Figure 4.1.

The step where the original game is mapped onto a refinement would typically be used to model action removal: say we have three actions a_1, a_2, a_3 available at an information set, in the abstraction we may want to have only a_1, a_2 and consider a_3 as mapped onto a_2. The refinement step can only model information coarsening, but is very powerful for modeling certain practical types of abstraction. As an example, in poker research cards have typically been abstracted via information coarsening, say treating a pair of aces and a pair of kings as the same hand in the abstraction. We can model this in the refinement step, where aces and kings would be refined into two separate information sets.

We now give a formal description of our framework. As noted above, we consider abstractions that are themselves EFGs, but we do not require abstractions to have perfect recall (the leading practical abstractions are of imperfect recall [28, 64, 84]). We will use *the original game* to refer to some perfect-recall game $\Gamma = (H, Z, A, P, \pi_0, \{\mathcal{I}_i\}, \{u_i\})$ that we would like to compute a Nash equilibrium for. We use *the abstract game* to refer to some other game $\Gamma' = (H', Z', A', P', \pi'_0, \{\mathcal{I}'_i\}, \{u'_i\})$ that is an abstraction of Γ. The goal is to compute a (possibly approximate) equilibrium in the abstraction, and map the resulting strategy profile to the full game. An example is shown in Figure 4.1. Figure 4.2 shows an example of going from the original game in Figure 4.1 to the abstraction.

We model abstraction as a two-stage process. First, the full game is mapped onto the abstract game, with every original information set $I \in \mathcal{I}_i$ mapping onto some *information-set partition* I'_I

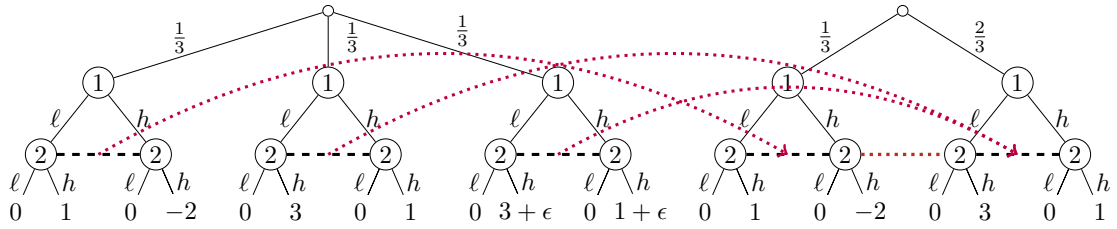


Figure 4.1: Abstraction example. Left: Original EFG. Right: Abstraction (which has perfect recall in this case). Dotted red arrows denote the mapping of information sets in the original game onto information set partitions in the abstract game. The dotted red line in the abstract game denotes an information set coarsening relative to $\mathcal{P}'$.

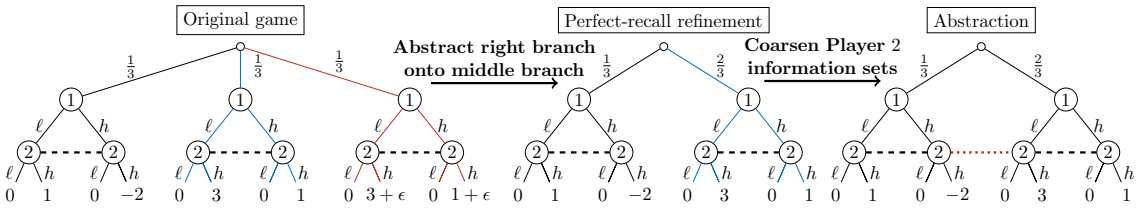


Figure 4.2: Example of how the abstraction in Figure 4.1 could be constructed. First the rightmost red branch is removed (which we would model as mapping it onto the middle blue branch). Second the information sets for Player 2 are coarsened as shown by the red dotted line.

in the abstraction via a function $f : \mathcal{I} \to \mathcal{P}'$ that maps $\mathcal{I}$ surjectively onto $\mathcal{P}'$. $\mathcal{P}'$ is assumed to be a partitioning of $H' \setminus Z'$ that refines $\mathcal{I}'$. Thus the information-set structure specified by $\mathcal{P}'$ can be thought of as specifying an intermediate game with (weakly) more information than Γ'; $\mathcal{P}'$ is assumed to induce a perfect-recall game[1]. In Figure 4.1, each of the three original information sets belonging to Player 2 map onto the same abstract information set, but the leftmost original information set maps onto the left partition, whereas the center and right information sets map onto the right partition. In the abstract game in Figure 4.1, Player 2 has two subsets in $\mathcal{P}'$: the left and right sides of their single information set. Actions are similarly mapped with an action mapping $g : A \to A'$ that maps each A_I surjectively onto $A_{f(I)}$. It is assumed that f respects the information-set tree structure by mapping $\mathcal{C}_I^a$ surjectively onto $\mathcal{C}_{I'_I}^{g(a)}$. The final part of the first step is a way to map leaf nodes under original information sets to leaf nodes under the corresponding abstract information set. For each information set I and action $a \in A_I$, we require a surjective leaf-node mapping ϕ_I from the set of leaf nodes reached below I, a before player i acts again, Z_I^a, onto $Z_{I'_I}^{a'}$.

The second step in our abstraction model captures the differences between the abstract game Γ' and the game induced by using the partitioning $\mathcal{P}'$ instead. This is done by comparing the distribution over leaf nodes conditioned on being at a given $I'_I \in \mathcal{P}'$ versus the distribution conditioned on being at the corresponding abstract information set I'. In Figure 4.1 this would

[1]Lanctot et al. [109] use the notion of a *perfect-recall refinement*, which is a partitioning of each imperfect-recall information set into several perfect-recall information sets. Our definition of $\mathcal{P}'$ can be thought of as specifying a perfect-recall refinement of the abstraction.

correspond to comparing the leaf nodes under e.g. the right pair of nodes in Player 2's information set in the abstraction to the leaf nodes in the overall information set for Player 2. For each partition I'_I this is done with a set-valued map $\phi_{I'_I}$ that maps the set of leaf nodes $\mathcal{Z}_{I'_I}^{a'}$ onto $\mathcal{Z}_{I'}^{a'}$ for each a' in a way such that $\{\phi_{I'_I}(z') : z' \in \mathcal{Z}_{I'_I}^{a'}\}$ specifies a partitioning of $\mathcal{Z}_{I'}^{a'}$. For a given partition I'_I, we let $\mathcal{D}_{I'_I}$ and $\mathcal{C}_{I'_I}$ be the set of descendant and child partitions, respectively, that can be reached from I'_I.

For a strategy profile σ' computed in Γ' we need a way to interpret it as strategy profiles in Γ. We present the natural extension of a *lifted strategy*, originally developed by Sandholm and Singh [152] for stochastic games, to EFGs. Intuitively, a lifted strategy $\sigma^{\uparrow\sigma'}$ is a strategy where for any abstract action a', the sum of probabilities in $\sigma^{\uparrow\sigma'}$ assigned to actions that map to a' is equal to the probability placed on a' in σ'.

Definition 6 (Strategy lifting)**.** *Given an abstract strategy profile σ', a lifted strategy profile is any strategy profile $\sigma^{\uparrow\sigma'}$ such that for all I, all $a' \in A_{f(I)}$: $\sum_{a \in g^{-1}(a')} \sigma^{\uparrow\sigma'}(I, a) = \sigma'(f(I), a')$.*

We use the definition of counterfactual value of an information set, introduced by Zinkevich et al. [179], 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. It is defined as $V_i^\sigma(I) = \sum_{z \in \mathcal{Z}_I} \pi^\sigma(z|I) u_i(z)$ when $\pi^\sigma_{-i}(I) > 0$; otherwise it is 0. Analogously, $W_i^{\sigma'} : \mathcal{I}'_i \to \mathbb{R}$ is the corresponding function for the abstract game. For the information set I_r that contains just the root node r, we have $V_i^\sigma(I_r) = V_i^\sigma(r)$, which is the value of playing the game with strategy profile σ. We assume that at the root node it is not Chance's turn to move. This is without loss of generality since we can insert dummy player nodes above a root node belonging to Chance.

We show that for an information set I, $V_i^\sigma(I)$ can be written as a sum over descendant information sets. The proof is straightforward.

Lemma 3. *For any strategy profile σ or abstract strategy profile σ', the counterfactual value of an information set I, or abstract information-set partition I'_I, can respectively be written as*

$$\begin{aligned} V_i^\sigma(I) &= \sum_{a \in A_I} \sigma(I, a) \Bigg[\sum_{\hat{I} \in \mathcal{C}_I^a} \pi^\sigma_{-i}(\hat{I}|I) V_i^\sigma(\hat{I}) + \sum_{z \in Z_I^a} \pi^\sigma_{-i}(z|I) u_i(z) \Bigg], \\ W_i^{\sigma'}(I'_I) &= \sum_{a' \in A_{I'}} \sigma'(I', a') \Bigg[\sum_{\hat{I}'_{\hat{I}} \in \mathcal{C}_{I'_I}^{a'}} \pi^{\sigma'}_{-i}(\hat{I}'_{\hat{I}}|I'_I) W_i^{\sigma'}(\hat{I}'_{\hat{I}}) + \sum_{z' \in Z_{I'_I}^{a'}} \pi^{\sigma'}_{-i}(z'|I'_I) u_i(z') \Bigg]. \end{aligned} \tag{4.1}$$

Proof. We show the statement for $V_i^\sigma(I)$, the result for $W_i^{\sigma'}(I'_I)$ follows by viewing the partitioning as a perfect-recall game. We have

$$V_i^\sigma(I) = \sum_{z \in \mathcal{Z}_I} \pi^\sigma(z|I) u_i(z) = \sum_{\hat{I} \in \mathcal{C}_I} \sum_{z \in \mathcal{Z}_{\hat{I}}} \pi^\sigma(z|I) u_i(z) + \sum_{z \in Z_I} \pi^\sigma(z|I) u_i(z) \tag{4.2}$$

Now for any $\hat{I} \in \mathcal{C}_I$ we have

$$\sum_{z \in \mathcal{Z}_{\hat{I}}} \pi^\sigma(z|I) u_i(z) = \sum_{h \in \hat{I}} \pi^\sigma(h|I) \sum_{z \in \mathcal{Z}_{\hat{I}}} \pi^\sigma(z|h) u_i(z) = \pi^\sigma(\hat{I}|I) \sum_{h \in \hat{I}} \pi^\sigma(h|\hat{I}) \sum_{z \in \mathcal{Z}_{\hat{I}}} \pi^\sigma(z|h) u_i(z)$$
$$= \pi^\sigma(\hat{I}|I) \sum_{z \in \mathcal{Z}_{\hat{I}}} \pi^\sigma(z|\hat{I}) u_i(z) = \pi^\sigma(\hat{I}|I) V_i^\sigma(\hat{i}),$$

where the second equality follows from $\pi^\sigma(h|I) = \pi^\sigma(h|\hat{I})\pi^\sigma(\hat{I}|I)$ and the third equality follows from $\pi^\sigma(z|\hat{I}) = \pi^\sigma(h|\hat{I})\pi^\sigma(z|h)$. Plugging this into (4.2) gives the result. □

We will show results for three different solution concepts that come up in practice. An ϵ-*Nash equilibrium* is a strategy profile σ such that $V_i^\sigma(r) \geq V_i^{\sigma'}(r) - \epsilon$ for all players i and $\sigma' = (\sigma_{-i}, \sigma_i')$. In other words, each player can gain at most ϵ by deviating to any other strategy σ_i'. This is what is computed by approaches based on first-order methods [79, 101, 103]. A *Nash equilibrium* is an ϵ-Nash equilibrium where $\epsilon = 0$. Finally, a strategy profile σ has bounded counterfactual regret if for all $i, I \in \mathcal{I}$, and $a \in A_I$, $V_i^{\sigma_I \to a}(I) \leq V_i^\sigma(I) + r(I)$. Strategy profiles with bounded counterfactual regret are important because regret minimization algorithms for EFGs converge by producing strategies with low $r(I)$ [27, 31, 58, 108, 179].

4.3 Measuring differences between the original game and the abstract game

Our goal is to show a decomposition of the utility difference, or abstraction error, between the original game and the abstract game when using a lifted strategy. This will consist of showing that the ϵ in an ϵ-Nash equilibrium for the original game constructed from an ϵ'-Nash equilibrium in the abstraction can be decomposed into three types of error: ϵ', a measure of how payoffs differ between the games, and a measure of how node probabilities differ. In order to do this, we now define measures of differences between the original and abstract game. We measure payoff differences between real and abstract nodes z, z' as

$$\Delta_i^R(z, z') = u_i(z) - u_i(z').$$

We measure leaf-node reach-probability differences conditioned on reaching a given information set I versus its corresponding abstract information set-partition I_I' as follows

$$\Delta_{-i}^P(z'|I, a, \sigma, \sigma') = \sum_{z \in \phi_I^{-1}(z') : z \in Z_I^a} \pi_{-i}^\sigma(z|I) - \pi_{-i}^{\sigma'}(z'|I_I'), \quad \text{for } z' \in Z_{I'}^{a'}.$$

We will also need to measure the difference in probability of reaching information set partitions, conditioned on being at the preceding information set partition belonging to the same player,

$$\Delta_{-i}^P(\hat{I}_{\hat{I}}'|I, a, \sigma, \sigma') = \sum_{\tilde{I} \in f^{-1}(\hat{I}_{\hat{I}}')} \pi_{-i}^\sigma(\tilde{I}|I, a) - \pi_{-i}^{\sigma'}(\hat{I}_{\hat{I}}'|I_I').$$

While the set $f^{-1}(\hat{I}'_{\hat{I}})$ can include information sets $\tilde{I}$ that do not come after I, a, such information sets are irrelevant since $\pi^{\sigma}_{-i}(\tilde{I}|I,a) = 0$.

Proposition 1. *For any player i, abstract strategy σ', real strategy σ, information sets I and $I' = f(I)$, and actions a and $a' = g(a)$*

$$\sum_{z \in Z^a_I} \pi^{\sigma}_{-i}(z|I) u_i(z) - \sum_{z' \in Z^{a'}_{I'_I}} \pi^{\sigma'}_{-i}(z'|I'_I) u_i(z') = \sum_{z \in Z^a_I} \pi^{\sigma}_{-i}(z|I) \Delta^R(z, \phi_I(z)) + \sum_{z' \in Z^{a'}_{I'_I}} \Delta^P_{-i}(z'|I, a, \sigma, \sigma') u_i(z')$$

Proof. We have

$$\begin{aligned}
&\sum_{z \in Z^a_I} \pi^{\sigma}_{-i}(z|I) u_i(z) = \sum_{z' \in Z^{a'}_{I'_I}} \sum_{z \in \phi_I^{-1}(z') : z \in Z^a_I} \pi^{\sigma}_{-i}(z|I) u_i(z) \\
&= \sum_{z' \in Z^{a'}_{I'_I}} \sum_{z \in \phi_I^{-1}(z') : z \in Z^a_I} \pi^{\sigma}_{-i}(z|I) (u_i(z') + \Delta^R(z, \phi_I(z))) \\
&= \sum_{z' \in Z^{a'}_{I'_I}} \left[\pi^{\sigma'}_{-i}(z'|I'_I) + \Delta^P_{-i}(z'|I, a, \sigma, \sigma') \right] u_i(z') + \sum_{z \in Z^a_I} \pi^{\sigma}_{-i}(z|I) \Delta^R(z, \phi_I(z)) \\
&= \sum_{z' \in Z^{a'}_{I'_I}} \pi^{\sigma'}_{-i}(z'|I'_I) u_i(z') + \sum_{z \in Z^a_I} \pi^{\sigma}_{-i}(z|I) \Delta^R(z, \phi_I(z)) + \sum_{z' \in Z^{a'}_{I'_I}} \Delta^P_{-i}(z'|I, a, \sigma, \sigma') u_i(z').
\end{aligned}$$

The first equality follows from the fact that every leaf node in Z^a_I maps onto some leaf node in $Z^{a'}_{I'}$, the second from the definition of Δ^R, the third by rearranging terms and the definition of Δ^P, and the fourth by rearranging terms. □

We now prove a technical lemma that will be used as the primary tool for inductively proving that strategies from abstractions have bounded regret.

Lemma 4. *For any information set $I, I' = f(I)$ and pair of strategy profiles σ, σ', assume there is a bound $\Delta(\hat{I}, f(\hat{I}))$ such that $V^{\sigma}_i(\hat{I}) - W^{\sigma'}_i(\hat{I}'_{\hat{I}}) \leq \Delta(\hat{I}, f(\hat{I}))$ for all $\hat{I} \in \mathcal{C}^a_I, a \in A_I$, and $\sigma'(I', a') = \sum_{a \in g^{-1}(a')} \sigma(I, a)$. Then*

$$\begin{aligned}
V^{\sigma}_i(I) - W^{\sigma'}_i(I'_I) \leq \sum_{a \in A_I} \sigma(I, a) \Bigg[&\sum_{z \in Z^a_I} \pi^{\sigma}_{-i}(z|I) \Delta^R_i(z, \phi_I(z)) \\
&+ \sum_{z' \in Z^{g(a)}_{I'_I}} \Delta^P_{-i}(z'|I, a, \sigma, \sigma') u_i(z') + \sum_{\hat{I} \in \mathcal{C}^a_I} \pi^{\sigma}_{-i}(\hat{I}|I) \Delta(\hat{I}, f(\hat{I})) + \sum_{\hat{I}'_{\hat{I}} \in \mathcal{C}^{g(a)}_{I'_I}} \Delta^P_{-i}(\hat{I}'_{\hat{I}}|I, a, \sigma, \sigma') W^{\sigma'}_i(\hat{I}'_{\hat{I}}) \Bigg]
\end{aligned}$$

The above holds with equality if $V^{\sigma}_i(\hat{I}) - W^{\sigma'}_i(\hat{I}'_{\hat{I}}) = \Delta(\hat{I}, f(\hat{I}'))$ for all $\hat{I} \in \mathcal{C}^a_I$ and $a \in A_I$.

Proof. By Lemma 4.1 we have

$$V^{\sigma}_i(I) = \sum_{a \in A_I} \sigma(I, a) \left[\sum_{\hat{I} \in \mathcal{C}^a_I} \pi^{\sigma}_{-i}(\hat{I}|I) V^{\sigma}_i(\hat{I}) + \sum_{z \in Z^a_I} \pi^{\sigma}_{-i}(z|I) u_i(z) \right]$$

We show the result by separately rewriting the two summation terms. For the summation over information sets we have

$$\sum_{\hat{I}\in\mathcal{C}_I^a} \pi_{-i}^{\sigma}(\hat{I}|I)V_i^{\sigma}(\hat{I}) \leq \sum_{\hat{I}\in\mathcal{C}_I^a} \pi_{-i}^{\sigma}(\hat{I}|I)\left[W_i^{\sigma'}(\hat{I}'_{\hat{I}}) + \Delta(\hat{I}, f(\hat{I}))\right]$$

$$= \sum_{\hat{I}'\in\mathcal{C}_{I'}^{a'}} \left[\pi_{-i}^{\sigma'}(\hat{I}'_I|I', a') + \Delta_{-i}^{P}(\hat{I}'_I|I, a, \sigma, \sigma')\right] W_i^{\sigma'}(\hat{I}'_{\hat{I}}) + \sum_{\hat{I}\in\mathcal{C}_I^a} \pi_{-i}^{\sigma}(\hat{I}|I)\Delta(\hat{I}, f(\hat{I}))$$

Where the last step follows by the definition of $\Delta_{-i}^{P}(\hat{I}'_I|I, a, \sigma, \sigma')$. For the summation over $\sum_{z\in Z_I^a}$ we can apply Proposition 1. Adding up terms and using the condition $\sigma'(I', a') = \sum_{a\in g^{-1}(a')} \sigma(I, a)$ then gives the result.

The inequality holds with equality when the bounds on child information sets are equalities, because the only inequality introduced in the proof comes from applying the bound on the child information sets. □

We now introduce a shorthand for denoting the utility difference attributable to differences between a given information set I and its abstract counterpart $f(I)$. This is the utility difference that would arise from recursively applying Lemma 4 to information sets.

$$\begin{aligned}\mathrm{Mdiff}(I, \sigma, \sigma'_{-i}) \overset{\text{def}}{=} \sum_{a\in A_I} \sigma(I, a)\Bigg[&\sum_{z\in Z_I^a} \pi_{-i}^{\sigma}(z|I)\Delta_i^R(z, \phi_I(z)) + \sum_{z'\in Z_{I'_I}^{g(a)}} \Delta_{-i}^{P}(z'|I, a, \sigma, \sigma')u_i(z') \\ &+ \sum_{\hat{I}\in\mathcal{C}_I^a} \pi_{-i}^{\sigma}(\hat{I}|I)\,\mathrm{Mdiff}(\hat{I}, \sigma, \sigma'_{-i}) + \sum_{\hat{I}'_{\hat{I}}\in\mathcal{C}_{I'_I}^{g(a)}} \Delta_{-i}^{P}(\hat{I}'_{\hat{I}}|I, a, \sigma, \sigma')W_i^{\sigma'}(\hat{I}'_{\hat{I}})\Bigg]\end{aligned}$$

It follows from Lemma 4 that the players' values in any lifted strategy profile in the original game are close to the players' values of the corresponding abstract strategy profile:

Lemma 5. *Given any abstract strategy profile σ', any lifted strategy profile $\sigma^{\uparrow\sigma'}$ achieves utility*

$$W_i^{\sigma'}(r') = V_i^{\sigma^{\uparrow\sigma'}}(r) - \mathrm{Mdiff}(r, \sigma^{\uparrow\sigma'}, \sigma'_{-i})$$

Next we derive an expression for the difference between an abstract information set and any subset in its partitioning. We will need a way to measure the difference between an information set I' and any partition I'_I. For reach probability, we let

$$\Delta^P(z'|I'_I, \sigma') = \pi^{\sigma'}(z'|I'_I) - \sum_{\hat{z}'\in\phi_{I'_I}(z')} \pi^{\sigma'}(\hat{z}'|I') \tag{4.3}$$

be the difference between the probability of arriving at z' conditioned on a strategy σ' and being in partition I'_I of I' and the probability of arriving at any leaf node $z' \in \phi_{I'_I}^{-1}(z')$ conditioned on the same strategy σ' and being in I'. For reward differences we let the utility difference between a leaf node $\hat{z}' \in Z_{I'}$ and its corresponding leaf node $z' = \phi_{I'_I}^{-1}(z')$ in $Z_{I'_I}$ be

$$\Delta_i^R(\hat{z}'|I'_I) = u_i(z') - \delta_{I'_I} u_i(\hat{z}') \tag{4.4}$$

These terms allow us to measure the difference between the value $W_i^{\sigma'}(I')$ and $W_i^{\sigma'}(I'_I)$ for any information set I' and any I'_I in its partition. We let $\mathrm{Pdiff}(I'_I, \sigma')$ denote this difference.

Lemma 6. *For any player i, abstract strategy profile σ', information set I' and any I'_I in its partition,*

$$W_i^{\sigma'}(I'_I) - \delta_{I'_I} W_i^{\sigma'}(I') = \sum_{\hat{z}' \in \mathcal{Z}_{I'}} \pi^{\sigma'}(\hat{z}'|I') \Delta_i^R(\hat{z}'|I'_I) + \sum_{z' \in \mathcal{Z}_{I'_I}} \Delta^P(z'|I'_I, \sigma') u_i(z') \stackrel{\text{def}}{=} \text{Pdiff}(I'_I, \sigma')$$

Proof. Using the definition of the value of a partition and applying (4.3), rearranging, applying (4.4), rearranging again and using the definition of $W_i^{\sigma'}(I')$ gives

$$\begin{aligned}
W_i^{\sigma'}(I'_I) &= \sum_{z' \in \mathcal{Z}_{I'_I}} \pi^{\sigma'}(z'|I'_I) u_i(z') = \sum_{z' \in \mathcal{Z}_{I'_I}} \left[\sum_{\hat{z}' \in \phi_{I'_I}(z')} \pi^{\sigma'}(\hat{z}'|I') + \Delta^P(z'|I'_I, \sigma') \right] u_i(z') \\
&= \sum_{z' \in \mathcal{Z}_{I'_I}} \sum_{\hat{z}' \in \phi_{I'_I}(z')} \pi^{\sigma'}(\hat{z}'|I') u_i(z') + \sum_{z' \in \mathcal{Z}_{I'_I}} \Delta^P(z'|I'_I, \sigma') u_i(z')) \\
&= \sum_{z' \in \mathcal{Z}_{I'_I}} \sum_{\hat{z}' \in \phi_{I'_I}(z')} \pi^{\sigma'}(\hat{z}'|I') \left[\delta_{I'_I} u_i(\hat{z}') + \Delta^R(\hat{z}'|I'_I) \right] + \sum_{z' \in \mathcal{Z}_{I'_I}} \Delta^P(z'|I'_I, \sigma') u_i(z')) \\
&= \delta_{I'_I} W_i^{\sigma'}(I') + \sum_{\hat{z}' \in \mathcal{Z}_{I'}} \pi^{\sigma'}(\hat{z}'|I') \Delta^R(\hat{z}'|I'_I) + \sum_{z' \in \mathcal{Z}_{I'_I}} \Delta^P(z'|I'_I, \sigma') u_i(z'))
\end{aligned}$$

where the last step follows because $\phi_{I'_I}(\cdot)$ specifies a partitioning of the leaves under I'. □

4.4 An exact decomposition of abstraction error

Our first theorem shows that an ϵ-Nash equilibrium in the abstract game maps to an ϵ'-Nash equilibrium in the original game, where ϵ' depends on the difference terms introduced in the previous section. We say that the abstract game has a *cycle* if there exists a sequence of information sets $I'_1, \ldots, I'_k$ such that for all $j \neq k$ there exist nodes $h'_j \in I'_j, h'_{j+1} \in I'_{j+1}$ such that h'_j is an ancestor of h'_{j+1}, and I'_1 is equal to I'_k. The next theorem assumes the abstract game is acyclic. This enables induction over information sets.

Theorem 8. *Given an ϵ-Nash equilibrium σ' for an acyclic abstract game, any lifted strategy profile $\sigma^{\uparrow\sigma'}$ is an ϵ'-Nash equilibrium in the original game where $\epsilon' = \max_{i \in N} \epsilon_i$ and*

$$\epsilon'_i = \epsilon + \text{Mdiff}(r, \sigma^*, \sigma'_{-i}) - \text{Mdiff}(r, \sigma^{\uparrow\sigma'}, \sigma'_{-i}) + \sum_{I \in \mathcal{I}_i} \pi^{\sigma^*}(I) \left[\text{Pdiff}(I'_I, \sigma^{*\prime}_{I' \to I}) - \text{Pdiff}(I'_I, \sigma^{*\prime}) \right]$$

here $\sigma^* = (\sigma_i^*, \sigma_{-i}^{\uparrow\sigma'})$ is $\sigma^{\uparrow\sigma'}$ except Player i plays any best response strategy for the original game, $\sigma^{*\prime} = (\sigma_i^{*\prime}, \sigma'_{-i})$ is such that $\sigma^{*\prime}(I', a') = \sum_{g^{-1}(a')} \sigma^*(I, a)$ where $I \in f^{-1}(I')$ is chosen for each I' in order to maximize $W_i^{\sigma^{*\prime}}(r)$, and $\sigma^{*\prime}_{I' \to I}$ is $\sigma^{*\prime}$ except that at I' we set the strategy according to I, i.e. $\sigma^{*\prime}(I', a') = \sum_{g^{-1}(a')} \sigma^*(I, a)$.

Proof. The proof consists of showing that $V_i^{\sigma^*}(r)$ can be bounded by $W_i^{\sigma^{*\prime}}(r')$ and some difference terms, where $\sigma^{*\prime}$ is a (intuitively speaking) reversely lifted strategy from σ^*. We can then

use the fact that σ' is an ϵ-Nash equilibrium to bound $W_i^{\sigma^{*\prime}}(r')$ in terms of $W_i^{\sigma'}(r')$ and finally show that $W_i^{\sigma'}(r')$ is close to $V_i^{\sigma^{\uparrow\sigma'}}(r)$.

To rewrite $V_i^{\sigma^*}(r)$ in terms of $W_i^{\sigma^{*\prime}}(r')$ we first prove the following inductive statement for $I \in \mathcal{I}_i, I' = f(I)$ and letting $\hat{I}' = f(\hat{I})$ for each $\hat{I}$:

$$V_i^{\sigma^*}(I) \leq W_i^{\sigma^{*\prime}}(I'_I) + \text{Mdiff}(I, \sigma^*, \sigma'_{-i}) + \sum_{\hat{I} \in \mathcal{D}_I \cup \{I\}} \pi^{\sigma^*}(\hat{I}|I) \left[\text{Pdiff}(\hat{I}'_{\hat{I}}, \sigma^{*\prime}_{\hat{I}' \to \hat{I}}) - \text{Pdiff}(\hat{I}'_{\hat{I}}, \sigma^{*\prime})\right]$$

We will start by showing the inductive step, as the base case is the special case of information sets that only have leaves beneath them (i.e. information sets I such that for all a, $\mathcal{C}_I^a = \emptyset$).

Let $I, I' = f(I)$ be such that the inductive statement holds for all $a \in A_I$ and $\hat{I} \in \mathcal{C}_I^a$. The inductive assumption then gives a bound for each $\hat{I} \in \mathcal{D}_I$ as required by Lemma 4. We have

$$V_i^{\sigma^*}(I) \leq W_i^{\sigma^{*\prime}_{I' \to I}}(I'_I) + \text{Mdiff}(I, \sigma^*, \sigma'_{-i}) + \sum_{\hat{I} \in \mathcal{D}_I} \pi^{\sigma^*}(\hat{I}|I) \left[\text{Pdiff}(\hat{I}'_{\hat{I}}, \sigma^{*\prime}_{\hat{I}' \to \hat{I}}) - \text{Pdiff}(\hat{I}'_{\hat{I}}, \sigma^{*\prime})\right], \tag{4.5}$$

where the result follows by collecting terms that arise from Lemma 4 into the three separate terms above.

Now we can bound $W_i^{\sigma^{*\prime}_{I' \to I}}(I'_I)$

$$\begin{aligned} W_i^{\sigma^{*\prime}_{I' \to I}}(I'_I) =& \delta_{I'_I} W_i^{\sigma^{*\prime}_{I' \to I}}(I') + \text{Pdiff}(I'_I, \sigma^{*\prime}_{I' \to I}); \text{ by Lemma 6} \\ \leq& \delta_{I'_I} W_i^{\sigma^{*\prime}}(I') + \text{Pdiff}(I'_I, \sigma^{*\prime}_{I' \to I}); \text{ because } \sigma^{*\prime} \text{ maximizes } W_i^{\sigma^{*\prime}}(r) \\ =& W_i^{\sigma^{*\prime}}(I'_I) + \text{Pdiff}(I'_I, \sigma^{*\prime}_{I' \to I}) - \text{Pdiff}(I'_I, \sigma^{*\prime}); \text{ by Lemma 6.} \end{aligned} \tag{4.6}$$

Putting (4.5) and (4.6) together gives the inductive statement.

For the base case, it follows from the exact same logic but where there are no descendant information sets. Applying the induction to the whole game we get

$$V_i^{\sigma^*}(r) \leq W_i^{\sigma^{*\prime}}(r') + \text{Mdiff}(r, \sigma^*, \sigma'_{-i}) + \sum_{I \in \mathcal{I}_i} \pi^{\sigma^*}(I) \left[\text{Pdiff}(I'_I, \sigma^{*\prime}_{I' \to I}) - \text{Pdiff}(I'_I, \sigma^{*\prime})\right] \tag{4.7}$$

Now we can bound $W_i^{\sigma^{*\prime}}(r')$ by using the fact that σ' is an ϵ-Nash equilibrium in the abstract game:

$$(4.7) \leq W_i^{\sigma'}(I') + \epsilon + \text{Mdiff}(r, \sigma^*, \sigma'_{-i}) + \sum_{I \in \mathcal{I}_i} \pi^{\sigma^*}(I) \left[\text{Pdiff}(I'_I, \sigma^{*\prime}_{I' \to I}) - \text{Pdiff}(I'_I, \sigma^{*\prime})\right] \tag{4.8}$$

Finally we can apply Lemma 5 to get the theorem statement. □

This theorem is the first to show results for mapping an ϵ'-Nash equilibrium in the abstract game to an ϵ-Nash equilibrium in the original game. Prior results have been for abstract strategies that are either exact Nash equilibria [95] or with bounded counterfactual regret [98, 109]. That is because all prior proofs were based on applying a worst-case counterfactual regret bound as part of the inductive step (which works for exact Nash equilibrium or strategies with bounded

counterfactual regret but not ϵ-Nash equilibrium); our proof instead constructs an expression for $W_i^{\sigma^{*\prime}}(r')$ (i.e., for the value of the whole abstract game) before using the fact that σ' is an ϵ-Nash equilibrium. We next show that our framework can also measure differences for strategies with bounded counterfactual regret.

Theorem 9. *For an abstract strategy profile σ' with bounded counterfactual regret $r(I')$ at every information set $I' \in \mathcal{I}'$, any lifted strategy profile $\sigma^{\uparrow\sigma'}$ is an ϵ-Nash equilibrium where*

$$\epsilon = \max_{i \in N} \epsilon_i, \qquad \epsilon_i \leq \sum_{I \in \mathcal{I}_i} \pi^{\sigma^*}(I) \left[\delta_{f(I)_I} r(f(I)) + \text{Pdiff}(I'_I, \sigma'_{I \to \sigma^{*\prime}}) - \text{Pdiff}(I'_I, \sigma') \right] + \text{Mdiff}(r, \sigma^*, \sigma'_{-i}) - \text{Mdiff}(r, \sigma^{\uparrow\sigma'}, \sigma'_{-i})$$

where $\sigma^* = (\sigma_i^*, {\sigma^{\uparrow\sigma'}}_{-i})$ is $\sigma^{\uparrow\sigma'}$ except for Player i best responding, and each $\sigma'_{I \to \sigma^*}$ is equal to σ' except that $\sigma'_{I \to \sigma^*}(f(I), a') = \sum_{a \in g^{-1}(a')} \sigma^*(I, a)$ for all $a' \in A_{f(I)}$.

Proof. Consider some best-response profile $\sigma^* = (\sigma_i^*, {\sigma^{\uparrow\sigma'}}_{-i})$. We will show that it satisfies this bound. First we show that the following holds by induction for every $I, I' = f(I)$:

$$V_i^{\sigma^*}(I) - W_i^{\sigma'}(I'_I) \leq \sum_{\hat{I} \in \mathcal{D}_I \cup \{I\}} \pi^{\sigma^*}(\hat{I}|I) \left[\delta_{\hat{I}'_{\hat{I}}} r(f(\hat{I})) + \text{Pdiff}(\hat{I}'_{\hat{I}}, \sigma'_{I \to \sigma^*}) - \text{Pdiff}(\hat{I}'_{\hat{I}}, \sigma') \right] + \text{Mdiff}(I, \sigma^*, \sigma'_{-i})$$

We start by proving the inductive case. The base case follows by being a special case with no descendant information sets.

We will use the inductive assumption to apply Lemma 4 to the strategy pair $\sigma^*, \sigma'_{I \to \sigma^*}$, which satisfies the condition in the Lemma since $\sigma'_{I \to \sigma^*}$ plays according to σ^* at I'. Lemma 4 requires a bound for each $\hat{I} \in \mathcal{C}_I^a$: we let $\Delta(\hat{I}, f(\hat{I}))$ be equal to the right-hand side terms in the inductive assumption. Lemma 4 then gives

$$V_i^{\sigma^*}(I) \leq W_i^{\sigma'_{I \to \sigma^*}}(I'_I) + \sum_{a \in A_I} \sigma^*(I, a) \left[\sum_{z \in Z_I^a} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(z|I) \Delta^R(z, \phi_I(z)) \right. \tag{4.9}$$

$$+ \sum_{z' \in Z_{I'_I}^{a'}} \Delta_{-i}^P(z'|I, a, \sigma^{\uparrow\sigma'}, \sigma') u_i(z')$$

$$\left. + \sum_{\hat{I} \in \mathcal{C}_I^a} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(\hat{I}|I) \Delta(\hat{I}, f(\hat{I})) + \sum_{\hat{I}'_{\hat{I}} \in \mathcal{C}_{I'_I}^{a'}} \Delta_{-i}^P(\hat{I}'_{\hat{I}}|I, a, \sigma^{\uparrow\sigma'}, \sigma') W_i^{\sigma'}(\hat{I}'_{\hat{I}}) \right] \tag{4.10}$$

For some quantities that do not depend on player i, we have substituted $\sigma^{\uparrow\sigma'}$ for σ^*, which is valid because ${\sigma^{\uparrow\sigma'}}_{-i} = \sigma^*_{-i}$. The same applies to σ' and $\sigma'_{I \to \sigma^*}$ for quantities that do not depend on I'. We now show how to convert $W_i^{\sigma'_{I \to \sigma^*}}(I'_I)$ into $W_i^{\sigma'}(I'_I)$. First we apply Lemma 6 followed by the bound on immediate regret to get

$$W_i^{\sigma'_{I \to \sigma^*}}(I'_I) = \delta_{I'_I} W_i^{\sigma'_{I \to \sigma^*}}(I') + \text{Pdiff}(I'_I, \sigma'_{I \to \sigma^*}) \leq \delta_{I'_I} \left[W_i^{\sigma'}(I') + r(I') \right] + \text{Pdiff}(I'_I, \sigma'_{I \to \sigma^*}) \tag{4.11}$$

Now we can apply Lemma 6 again to get

$$(4.11) = W_i^{\sigma'}(I'_I) + \delta_{I'_I} r(I') + \text{Pdiff}(I'_I, \sigma'_{I \to \sigma^*}) - \text{Pdiff}(I'_I, \sigma')$$

This proves the inductive step: expanding the $\Delta(\hat{I}f(\hat{I}))$ and $\Delta(\hat{I}f(\hat{I}))$ terms in (4.10), using the above equality, and collecting terms gives the inductive assumption.

Applying the inductive statement to the whole game almost gives the theorem statement, we only need to convert $W_i^{\sigma'}(r)$ into $V_i^{\sigma^{\uparrow\sigma'}}(r)$ and acquire a negative term $\text{Mdiff}(r, \sigma^{\uparrow\sigma'}, \sigma'_{-i})$. This is exactly what we get if we apply Lemma 5 to $W_i^{\sigma'}(r)$, and so the proof is done. □

We will show in the next sections that our two main theorems generalize prior results. In addition, our theorems are the first to give an exact expression for the abstraction error; the inequalities arise only from inexactly solving the abstract game.

4.5 Perfect-recall abstraction

In this section we investigate how our decomposition results can be used to reason about perfect-recall abstractions. Perfect-recall abstraction was the dominant paradigm for a long time [66, 69, 70]. Perfect recall is attractive from an algorithmic perspective because a Nash equilibrium can be computed in the resulting abstraction in polynomial time in the two-player zero-sum case via linear programming [165], and there are highly successful fast iterative algorithms [79, 101, 103, 108, 179]. This is in contrast to imperfect-recall abstractions, which are known to be algorithmically intractable. We explore imperfect recall in Section 4.6.

4.5.1 Removing probability-error dependence on strategies

We now show that if the reach of leaf nodes and child information sets in the original and abstract game are the same (without considering Chance moves), the exact results from the previous section can be used to obtain solution-quality bounds where the probability error depends only on the probability difference between chance outcomes. These results generalize the previous solution-quality results of Sandholm and Singh [152], which were specific to finite-depth stochastic games.

For the following theorem we define measures of how well Chance outcomes are approximated in the abstraction:

$$\Delta_0(h, a') = \sum_{a \in g^{-1}(a')} \sigma_0(h, a) - \sigma_0(h', a'), \qquad \Delta_0(h) = \sum_{a' \in A_h} \Delta_0(h, a')$$

Similarly for nodes in infosets belonging to Player i we have the following error

$$\Delta_0(h'|I) = \sum_{h \in I_{h'}} \pi_0(h|I) - \pi_0(h'|I'_I)$$

where $I_{h'}$ is the set of nodes in h that precede leaf nodes mapping to z'.

First we prove that our condition gives a outcome-probability difference that is a probability distribution over differences in actions taken by Chance.

Lemma 7. *Let $\sigma^{\uparrow\sigma'}$ be strategy profile such that $\pi_{-i,-0}(z|h) = \pi_{-i,-0}(z'|h')$ for all $I \in \mathcal{I}_i, z \in Z_I$ with h and h' being the nodes preceding z and $z' = \phi_I(z)$ in I and $f(I)$, and let I_h be the set of node $h \in I$ such that $\pi_{-i}^{\sigma^{\uparrow\sigma'}}(h|I) > 0$. We then have:*

$$\Delta_{-i}^{P}(z'|I'_I, \vec{a}, \sigma^{\uparrow\sigma'}, \sigma') = \Delta_0(z'[I'_I]|I)\pi_{-i}^{\sigma'}(z'|z'[I'_I]) + \sum_{h_0 \in \mathcal{H}_0 : h \sqsubseteq h_0} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|I)\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(z'|t_{a'}^{h'_0}).$$

Proof. In this proof, for readability we always let h and h' be corresponding real and abstract nodes. In particular, for a given $h \in I$, h' will be the node in I'_I that leads to $\phi_I(z)$. First we show the following inductive statement for any node $\hat{h} \in \mathcal{H}_h$ in the subtree at h where $h \in I$, where the induction is on the length of path from the history to the leaf:

$$\sum_{z \in \phi_I^{-1}(z') : \hat{h} \sqsubseteq z} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(z|\hat{h}) = \pi_{-i}^{\sigma'}(z'|\hat{h}') + \sum_{h_0 \in \mathcal{H}_0 : \hat{h} \sqsubseteq h_0} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|\hat{h})\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(z'|t_{a'}^{h'_0}) \tag{4.12}$$

The base case is trivial since a leaf node z and z' each have probability 1 of being reached given that you start out at that given node. Assume the induction holds for paths of length l and that the path length from h' to z' is $l + 1$. We then have:

$$\begin{aligned}
&\sum_{z \in \phi_I^{-1}(z') : \hat{h} \sqsubseteq z} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(z|\hat{h}) = \sum_{a \in A_{\hat{h}}} \sigma^{\uparrow\sigma'}(\hat{h}, a) \sum_{z \in \phi_I^{-1}(z') : \hat{h} \sqsubseteq z} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(z|t_a^{\hat{h}}) \\
&= \sum_{a \in A_{\hat{h}}} \sigma^{\uparrow\sigma'}(\hat{h}, a) \sum_{z \in \phi_I^{-1}(z') : \hat{h} \sqsubseteq z} \left[\pi_{-i}^{\sigma'}(z'|t_a^{\hat{h}'}) + \sum_{h_0 \in \mathcal{H}_0 : t_a^{\hat{h}} \sqsubseteq h_0} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|t_a^{\hat{h}})\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(z'|t_{a'}^{h'_0}) \right].
\end{aligned} \tag{4.13}$$

If $\hat{h}$ is a non-Chance node then we can use the definition of a lifted strategy to get

$$(4.13) = \pi_{-i}^{\sigma'}(z'|\hat{h}') + \sum_{h_0 \in \mathcal{H}_0 : \hat{h} \sqsubseteq h_0} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|\hat{h})\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(z'|t_{a'}^{h'_0}).$$

If $\hat{h}$ is a Chance node then we can apply the definition of $\Delta_0^A(\hat{h})$ to get

$$\begin{aligned}
(4.13) &= \sum_{a' \in A_{I'}} \left(\sigma_0(\hat{h}', a') + \Delta_0^A(\hat{h}, a') \right) \pi_{-i}^{\sigma'}(z'|t_{a'}^{\hat{h}'}) + \sum_{h_0 \in \mathcal{H}_0 : t_a^{\hat{h}} \sqsubseteq h_0} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|\hat{h})\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(z'|t_{a'}^{h'_0}) \\
&= \pi_{-i}^{\sigma'}(z'|\hat{h}') + \sum_{h_0 \in \mathcal{H}_0 : \hat{h} \sqsubseteq h_0} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|\hat{h})\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(z'|t_{a'}^{h'_0}).
\end{aligned}$$

This proves the inductive statement.

By our definitions and (4.12) we have:

$$
\begin{aligned}
&\sum_{z\in\phi_I^{-1}(z'):z\in Z_I^{\vec{a}}} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(z|I) \\
&= \sum_{h\in I_{h'}} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h|I) \sum_{z\in Z_I^{\vec{a}}:h\sqsubseteq z} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(z|h) \\
&= \sum_{h\in I_{h'}} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h|I) \left(\pi_{-i}^{\sigma'}(z'|h') + \sum_{h_0\in\mathcal{H}_0:h\sqsubseteq h_0} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|h)\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(z'|t_{a'}^{h_0'}) \right) \\
&= \left(\pi_{-i}^{\sigma'}(h'|I_I') + \Delta_0(h'|I)\right)\pi_{-i}^{\sigma'}(z'|h') + \sum_{h\in I_{h'}} \sum_{h_0\in\mathcal{H}_0:h\sqsubseteq h_0} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|I)\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(z'|t_{a'}^{h_0'}) \\
&= \pi_{-i}^{\sigma'}(z'|I_I') + \Delta_0(h'|I)\pi_{-i}^{\sigma'}(z'|h') + \sum_{h\in I_{h'}} \sum_{h_0\in\mathcal{H}_0:h\sqsubseteq h_0} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|I)\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(z'|t_{a'}^{h_0'}).
\end{aligned}
$$

Where the third equality follows from

$$
\sum_{h\in I_{h'}} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h|I) = \sum_{h\in I_{h'}} \pi_{-i,-0}^{\sigma^{\uparrow\sigma'}}(h|I)\pi_0(h|I) = \sum_{h\in I_{h'}} \pi_{-i,-0}^{\sigma^{\uparrow\sigma'}}(h'|I_I')\pi_0(h|I) = \pi_{-i}^{\sigma'}(h'|I_I') + \Delta_0(h'|I)
$$

This proves the lemma. □

By exactly the same proof, but where we treat nodes in $\hat{I}_{\hat{I}}'$ the way we treated leaf nodes above, we get

Lemma 8.

$$
\Delta_{-i}^P(\hat{I}_{\hat{I}}'|I,\vec{a},\sigma^{\uparrow\sigma'},\sigma') = \sum_{\hat{h}'\in\hat{I}_{\hat{I}}'} \left[\Delta_0(h'|I)\pi_{-i}^{\sigma'}(\hat{h}'|h') + \sum_{h\in I_{h'}} \sum_{h_0\in\mathcal{H}_0:h\sqsubseteq h_0} \pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|I)\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(\hat{h}'|t_{a'}^{h_0'}) \right].
$$

In order to state our result, we let χ_i be the set of pure strategies belonging to Player i. We will often use $\vec{a}$ as an index where a single action a would go according to our definitions; in such cases $\vec{a}$ should be interpreted as the specific action in $\vec{a}$ that pertains to the definition, usually the action at a given information set I prescribed by $\vec{a}$. In a slight abuse of notation, we let $g(\vec{a})$ denote the pure strategy in the abstract game corresponding to $\vec{a}$ when applying g.

Proposition 2. *If an abstract strategy profile σ' and a lifted strategy profile $\sigma^{\uparrow\sigma'}$ are such that for all $i, I \in \mathcal{I}$, $\Delta_{-i,-0}^P(z'|I,a,\sigma,\sigma') = 0$, $\Delta_{-i,-0}^P(z|I,\sigma,\sigma') = 0$, and $\Delta_{-i,-0}^P(\hat{I}_{\hat{I}}'|I,a,\sigma,\sigma') = 0$ then for all players i and $\sigma = (\sigma_i, \sigma_{-i}^{\uparrow\sigma'})$ we have*

$$\mathrm{Mdiff}(r,\sigma,\sigma'_{-i}) - \mathrm{Mdiff}(r,\sigma^{\uparrow\sigma'},\sigma'_{-i}) \le 2\max_{\vec{a}\in\chi_i}\sum_{I\in\mathcal{I}_i}\pi^{\sigma^{\uparrow\sigma'}}(I|\vec{a})\Bigg[\sum_{z\in Z_I^{\vec{a}}}\pi_{-i}^{\sigma^{\uparrow\sigma'}}(z|I)\Delta^R(z,\phi_I(z))$$

$$+\sum_{z'\in Z_{I'_I}^{g(\vec{a})}}\left[\Delta_0(z'[I]|I)\pi_{-i}^{\sigma'}(z'|z'[I]) + \sum_{h\in I_{h'}}\sum_{h_0\in\mathcal{H}_0:h\sqsubseteq h_0}\pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|I)\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(z'|t_{a'}^{h'_0})u_i(z')\right]$$

$$\sum_{\hat{I}\in\mathcal{C}_I^{\vec{a}}}\sum_{\hat{h}'\in\hat{I}'_{\hat{I}}}\left[\Delta_0(\hat{h}'[I'_I]|I)\pi_{-i}^{\sigma'}(\hat{h}'|\hat{h}'[I'_I]) + \sum_{h\in I_{h'}}\sum_{h_0\in\mathcal{H}_0:h\sqsubseteq h_0}\pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|I)\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(\hat{h}'|t_{a'}^{h'_0})\right]W_i^{\sigma'}(\hat{I}'_{\hat{I}})\Bigg]$$

$$\stackrel{\text{def}}{=}\overline{\mathrm{Mdiff_i}}(\sigma^{\uparrow\sigma'},\sigma')$$

Proof. If we unroll the recursive definition of Mdiff and use the fact that $\sigma_{-i} = \sigma_{-i}^{\uparrow\sigma'}$ we get

$$\begin{aligned}
&\mathrm{Mdiff}(r,\sigma,\sigma'_{-i}) - \mathrm{Mdiff}(r,\sigma^{\uparrow\sigma'},\sigma'_{-i})\\
=&\sum_{I\in\mathcal{I}_i}\sum_{a\in A_I}\left[\pi_i^{\sigma}(I)\sigma(I,a) - \pi_i^{\sigma^{\uparrow\sigma'}}(I)\sigma^{\uparrow\sigma'}(I,a)\right]\pi_{-i}^{\sigma^{\uparrow\sigma'}}(I)\left[\sum_{z\in Z_I^a}\pi_{-i}^{\sigma^{\uparrow\sigma'}}(z|I)\Delta^R(z,\phi_I(z))\right.\\
&\left.+\sum_{z'\in Z_{I'_I}^{g(a)}}\Delta_{-i}^P(z'|I,a,\sigma^{\uparrow\sigma'},\sigma')u_i(z') + \sum_{\hat{I}'_{\hat{I}}\in\mathcal{C}_{I'_I}^{g(a)}}\Delta_{-i}^P(\hat{I}'_{\hat{I}}|I,a,\sigma^{\uparrow\sigma'},\sigma')W_i^{\sigma'}(\hat{I}'_{\hat{I}})\right]\\
\le&2\max_{\vec{a}\in\chi_i}\sum_{I\in\mathcal{I}_i}\pi^{\sigma^{\uparrow\sigma'}}(I|\vec{a})\left[\sum_{z\in Z_I^{\vec{a}}}\pi_{-i}^{\sigma^{\uparrow\sigma'}}(z|I)\Delta^R(z,\phi_I(z)) + \sum_{z'\in Z_{I'_I}^{g(\vec{a})}}\Delta_{-i}^P(z'|I,\vec{a},\sigma^{\uparrow\sigma'},\sigma')u_i(z')\right.\\
&\left.+\sum_{\hat{I}'_{\hat{I}}\in\mathcal{C}_{I'_I}^{g(\vec{a})}}\Delta_{-i}^P(\hat{I}'_{\hat{I}}|I,\vec{a},\sigma^{\uparrow\sigma'},\sigma')W_i^{\sigma'}(\hat{I}'_{\hat{I}})\right] \qquad (4.14)
\end{aligned}$$

Now by Lemmas 7 and 8 we have

$$2\max_{\vec{a}\in\chi_i}\sum_{I\in\mathcal{I}_i}\pi^{\sigma^{\uparrow\sigma'}}(I|\vec{a})\Bigg[\sum_{z\in Z_I^{\vec{a}}}\pi_{-i}^{\sigma^{\uparrow\sigma'}}(z|I)\Delta^R(z,\phi_I(z))$$

$$+\sum_{z'\in Z_{I'_I}^{g(\vec{a})}}\left[\Delta_0(z'[I]|I)\pi_{-i}^{\sigma'}(z'|z'[I]) + \sum_{h\in I_{h'}}\sum_{h_0\in\mathcal{H}_0:h\sqsubseteq h_0}\pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|I)\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(z'|t_{a'}^{h'_0})u_i(z')\right]$$

$$\sum_{\hat{I}\in\mathcal{C}_I^{\vec{a}}}\sum_{\hat{h}'\in\hat{I}'_{\hat{I}}}\left[\Delta_0(\hat{h}'[I'_I]|I)\pi_{-i}^{\sigma'}(\hat{h}'|\hat{h}'[I'_I]) + \sum_{h\in I_{h'}}\sum_{h_0\in\mathcal{H}_0:h\sqsubseteq h_0}\pi_{-i}^{\sigma^{\uparrow\sigma'}}(h_0|I)\Delta_0^A(h_0)\pi_{-i}^{\sigma'}(\hat{h}'|t_{a'}^{h'_0})\right]W_i^{\sigma'}(\hat{I}'_{\hat{I}})\Bigg]$$

□

We can combine Proposition 2 with Theorem 8 to get a bound that is independent of the best-response strategy:

Corollary 2. *If σ' is an abstract ϵ'-Nash equilibrium, satisfies the condition of Proposition 2, and* Pdiff *is zero everywhere, then any lifted strategy profile $\sigma^{\uparrow\sigma'}$ is an ϵ-Nash equilibrium where ϵ is less than* $\max_{i \in N} \overline{\mathrm{Mdiff}_{\mathrm{i}}}(\sigma^{\uparrow\sigma'}, \sigma') + \epsilon'$

This bound generalizes the bound of Sandholm and Singh [152] while simultaneously tightening their bound.

Corollary 2 shows a result for ϵ-Nash equilibrium computed in the abstraction. An analogous corollary for abstract strategies with bounded immediate regret can easily be obtained by combining Proposition 2 with Theorem 9.

In the next sections we will discuss a particular assumption that, together with choosing a lifted strategy that assigns all probability to a single action in each $g^{-1}(a')$, implies the conditions given in Proposition 2. This assumption is analogous to the assumption of passing through the same information-set-action pairs for nodes that map to each other given by [109], which they use in order to get a condition similar to that of Proposition 2.

Assumption 1. *For every player i, information set $I \in \mathcal{I}_i$ and leaf $z \in Z_I$ the sequence obtained by applying g to each non-Chance action on the path to z yields the abstract action sequence from $f(I)$ to $\phi_I(z)$ across all players, and the path from the root to z' for all non-Chance players except i.*

4.5.2 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. [64, 66, 67, 68, 70, 71, 84, 152, 179]. 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 APPROXIMATELYISOMORPHIC, which computes whether the actions can be merged and at what cost, for each node $h \in S$. 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.[2]

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

[2]The theory from earlier in this chapter 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 3 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'

$A_{Iso} \leftarrow \emptyset$

for *each action pair* $a_1, a_2 \in A_S$ **do**

  for *each node* $h \in S$ **do** Test APPROXIMATELYISOMORPHIC($t^h_{a_1}, t^h_{a_2}$) ;

  Let c be the cost of merging a_1, a_2 over all $h \in S$

  if *isomorphic for all* $s \in I$ **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.

end

Extensive-form game tree isomorphism

In order to determine whether two given actions can be merged at some information set (the function APPROXIMATELYISOMORPHIC 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 h_1, h_2. Since we are assuming that abstraction is conducted level by level, the algorithmic problem amounts to finding the cheapest onto mapping of h_1 onto h_2. To reason about this problem, we introduce the *extensive-form game tree isomorphism* problem.

First we introduce a simple notion of isomorphism. It does not capture everything we need for the APPROXIMATELYISOMORPHIC 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 7. *(Extensive form game tree topology isomorphism) A topology isomorphism (TI) between extensive form games* Γ, Γ' *is an onto mapping* ϕ *such that*

1. *For nodes* $h_1, h_2 \in \mathcal{H}$ *belonging to the same information set* I*,* $\phi(h_1)$ *and* $\phi(h_2)$ *belong to the same information set in* Γ'*.*
2. *For nodes* $h_1 \in I_1$ *and* $h_2 \in I_2$*, where* $I_1 \neq I_2$*,* $\phi(h_1)$ *and* $\phi(h_2)$ *belong to different information sets in* Γ'*.*
3. *For nodes* h *and* h_c*, where* h_c *is a child of* h*,* $\phi(h_c)$ *is a child of* $\phi(h)$*.*

We give an example game in Figure 4.3. The games have different values at the leaves, but are topologically equivalent. We can map the two nodes in the information sets any way we want, and the lone nodes at level 2 must map to each other. 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 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 must map to the root node in Γ', since

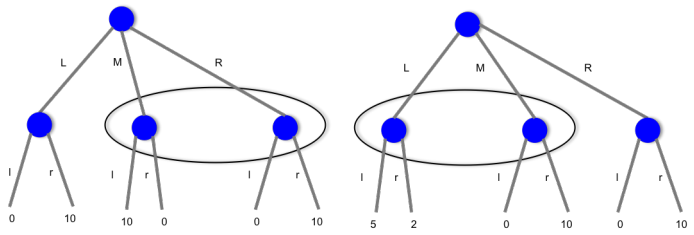


Figure 4.3: Two topologically isomorphic extensive-form games.

they are each the unqiue parent-less node in the respective game trees. Now consider some node h at height k, at depth d. $\phi(h)$ must be at height k in order to be isomorphic, and the shortest path to the root node must be of length d.

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 APPROXIMATELYISOMORPHIC subroutine. For subtrees where this isomorphism does not exist, we want to compute the mapping that minimizes the distance to strategic isomorphism.

Definition 8. *(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 chance nodes $h \in \mathcal{H}_0$ and children h_c, $\pi_0(h, h_c) = \pi_0(\phi(h), \phi(h_c)$.*
2. *For all leaf nodes $z \in Z$ and players $i \in N$, $V_i(z) = V_i(\phi(z))$.*

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 [16].

Theorem 10. *Deciding whether two game trees are extensive-form game tree topologically isomorphic or extensive-form game tree strategically isomorphic is graph isomorphism-complete.*

Proof. First we show a reduction to GI. For TI, consider a given extensive form game Γ. We modify the game tree such that it forms a graph G_Γ in the following way. For each information set, we make the nodes in the information set a clique. For nature nodes, we leave them the way they are, and ignore the probability distributions. For leaf nodes, since we are only concerned with topological isomorphism, we just convert them to an unlabeled node with no value. Finally, a unique clique size is chosen for the roots to be converted to. An example is given in Figure 4.4. Here we have a single information set containing all the nodes at height 2. We turn those nodes into a clique in the transformation, and convert the utilities at leaf nodes to regular leaf nodes. First we show that any GI mapping ϕ computed on $G_\Gamma, G_{\Gamma'}$ corresponds to a TI on Γ, Γ'. We will do this by induction on the height k.

Consider leaf nodes at height $k = 1$. Leaf nodes are the only nodes with degree 1, and thus for any leaf node $z \in Z_\Gamma$, $\phi(z)$ must be a leaf node in Γ', and vice versa. There are no information sets to respect at the leaf level.

Now consider height k, and assume that all nodes at heights below k are correctly mapped. For any node h at height k, we have that $\phi(h)$ must also be at height k in Γ', and they must have

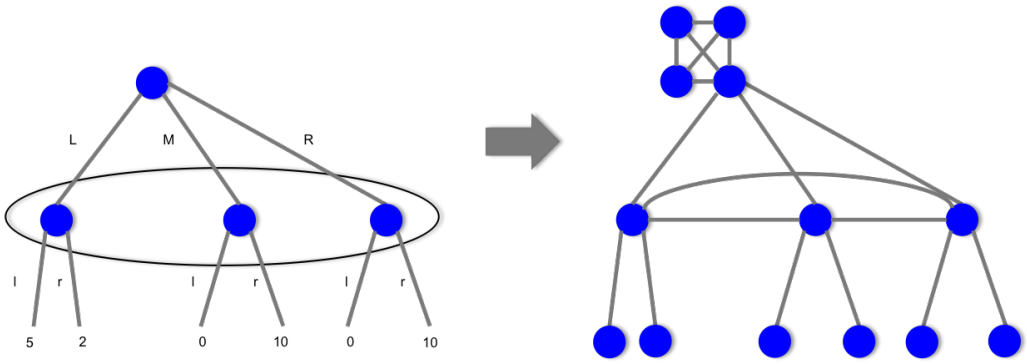


Figure 4.4: On the left is an extensive form game. On the right is the resulting graph after turning the information set in to a clique, and converting utilities at leaf nodes to regular leaf nodes.

the same length path to $r, \phi(r)$ respectively, since tree isomorphism preserves levels, as described above when we introduced our isomorphism definitions. Our added edges do not change this, since they only connect vertices within the level. Now consider some information set I at height k, and $h_1, h_2 \in I$. h_1, h_2 are connected by information set edges, and thus $\phi(h_1), \phi(h_2)$ must also be connected, which is only the case if they're in an information set together. Conversely, if $h_1 \in I_1, h_2 \in I_2, I_1 \neq I_2$, then they do not have an edge between them, and neither does $\phi(h_1), \phi(h_2)$, which is only the case if $\phi(h_1), \phi(h_2)$ are in separate information sets.

Now, we show that if ϕ is an isomorphic mapping for Γ, Γ' then it also defines an isomorphic mapping for $G_\Gamma, G_{\Gamma'}$. By the same reasoning as above, we have that all nodes must be at their respective levels, and leaf nodes must be correct. For $h_1, h_2 \in I$ at some level k, we have that $\phi(h_1), \phi(h_2)$ share an information set as well, and thus the nodes are connected by an edge in both augmented graphs. Similarly, for $h_1 \in I_1, h_2 \in I_2, I_1 \neq I_2$, $\phi(h_1), \phi(h_2)$ do not share an information set, and thus the nodes are not connected by an edge in either of the augmented graphs.

For SI, we perform the same transformation as above, except for leaf nodes and nature nodes. For every nature node type, we insert a separate unique structure. Then only nature nodes of the same type can map to each other. Further, for nature nodes that have uniform distribution over their actions, we can treat them as normal nodes, since they will only be merging with other nature nodes at the same level with uniform distribution, and the same number of actions. This can be any unique structure, one option is to use cliques of varying sizes, one size for each type, where the path from the root enters by some node in the clique, and leaves from some other node. This is a separate unqiue structure from the information set cliques, since those had edges going in and out of the clique from every node in the clique. Further, to make sure that we map the correct probability edges to each other, we can insert another unique structure along each unique edge probability, for example a circle, with circles of length $\{3, \ldots, 3 + |\text{nature edge types}|\}$.

Now consider some leaf node $z \in Z_\Gamma$, with values $v_1, \ldots, v_n$ for the players. From the node z, we insert a clique of size $i + 2$ for each player i, and add an edge from some node u in the clique to z. We then construct a cycle graph of size $c(v_i)$, and connect u to some node in the cycle graph. $c(v_i)$ is the smallest unique cycle size. This type of clique is again unique, since edges

enter and leave the clique from the same node u, which is different from the nature cliques. Now, for two leaf nodes $z \in \Gamma, z' \in \Gamma'$, we have that they are only isomorphic to each other if they have the same payoffs for each player, since each player has a uniquely sized clique, and the circles at the cliques must be of the same size. An example is given in Figure 4.5. On the left is a leaf node with payoffs $[2, 2, 3]$ for players $1, 2, 3$ respectively. On the right is the resulting subgraph that replaces the leaf node. At the top is the leaf node. Connected to the leaf are cliques, one for each player, of size $2 + i$ for Player i. For each clique, we have the value that the player attains as a cycle graph connected to that player's clique.

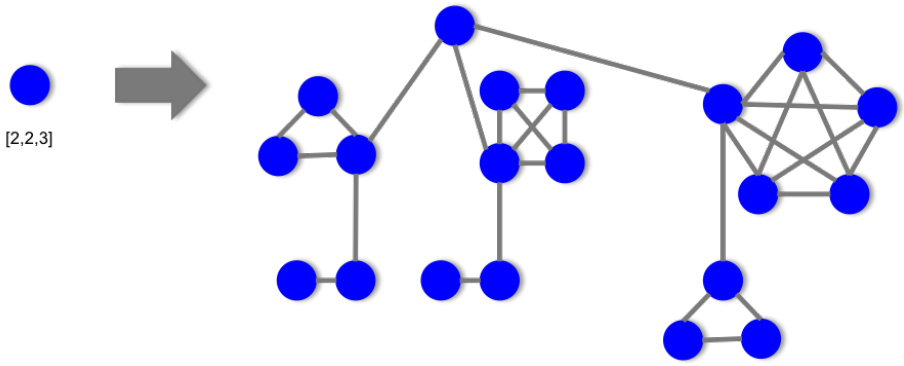


Figure 4.5: On the left is a leaf node with payoffs $[2, 2, 3]$ for players $1, 2, 3$ respectively. On the right is the subgraph that replaces the lead node in the reduction.

Now we show that GI reduces to TI, which suffices to show GI completeness, since TI is a special case of SI. Given a graph $G = \{V, E\}$, we construct the following extensive form game. First we add a root node. For each vertex $v \in V$, we add a node h_v descending from the root node, in an information set by itself. Thus, at depth two, we have $|V|$ nodes. Now, for any two nodes $v, \hat{h} \in V$ that share an edge, we create nodes $h_{v,\hat{v}}, h_{\hat{v},v}$ at depth three, each descending from their respective nodes $h_v, h_{\hat{v}}$, and put them in an information set together.

Now consider determining isomorphism between graphs $G = \{V, E\}, G' = \{V', E'\}$ given an isomorphic mapping ϕ between their game representation. If nodes $h_v, h_{\hat{v}}$ share an information set at their child nodes, then $\phi(h_v), \phi(h_{\hat{v}})$ must share an information set at their child nodes. However, this can only happen if both sets share an edge. Conversely, if nodes $h_v, h_{\hat{v}}$ do not share an information set at their child nodes, then $\phi(h_v), \phi(h_{\hat{v}})$ also do not share an information set at their child nodes. This implies that neither of the sets share an edge in their respective graphs. Thus, we get that any game isomorphism ϕ defines a graph isomorphism as well. We can apply the exact same logic to achieve the converse result.

We have shown that both TI and SI can be reduced to GI, and that GI reduces to TI. Thus, we have that TI and SI are GI-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. [62]. Peleg et al. [142] and Sudhölter et al. [159] study extensive form game tree isomorphism in the context of strategic equivalence, but do not consider the computational problem.

To our knowledge, ours is the first work 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 9. *(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 11. *The action subset selection problem is NP-complete.*[3]

Proof. The problem is easily seen to be in NP. To show NP-hardness, we reduce from SAT. Consider some SAT problem consisting of variables V and clauses C. For each variable $v \in V$, we construct an action for each of the literals $v, \neg v$. We let the cost of merging these actions be 0, and the cost of merging two literals that are not from the same variable be M. Here M is some large number. Now, for each clause, we make an action. We make the cost of merging this action with any other clause or any literal not in the clause M, and 0 if merged with a literal in the clause. We also add add an extra dummy action for each variable, which has cost 0 of merging with either of the two literals in the clause, and M for everything else. Now we ask if there is some abstraction with $|V|$ actions and cost zero. Clearly, the clauses cannot be chosen as prototypes without incurring a loss of M, because of the dummy actions. Instead, for each variable one of the two literals must be chosen as a prototype, and if there is such a set, it means that all clauses can map onto some literal, and must therefore be satisfied. Conversely, if there is a satisfying assignment, then we can pick the literals in the assignment as prototypes, and the clauses can be mapped onto the chosen literals with 0 loss. □

The problem of computing an extensive form game abstraction that minimizes any bound via abstracting actions, including minimizing Mdiff error, is easily seen to be NP-complete as well since a special case is a single player and the action subset selection problem. The action subset selection problem 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. Thus it holds for zero-sum games also.

[3] Sandholm and Singh [152] 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.

4.5.3 Level-by-level impossibility

We now show a a non-computational 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. This result is motivated by the GameShrink algorithm [68], which performed abstraction in the way that the below theorem proves is not always valid.

Theorem 12. *Any abstraction technique that computes the abstraction by merging subtrees must satisfy at least one of the following properties.*

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 [68], 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 she beats a private J1, and otherwise Player 1 wins. Both variants are zero-sum games of ordered signals. Figure 4.6 shows the possible sequences of cards dealt. The two variants are obtained by setting δ equal to 1 or -1 in Figure 4.6, respectively. If Player 1 always wins ($\delta = 1$), the whole

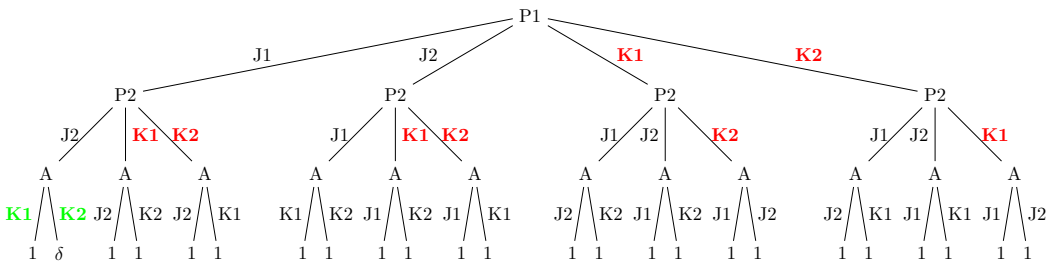


Figure 4.6: A signal tree for a simple poker game. Nodes labeled P1 or P2 denote the card being dealt privately to player 1 or 2, respectively. Nodes labeled A denote a public card. A leaf node labeled 1 indicates Player 1 winning.

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 [68] does not work correctly for all games of ordered signals. A proof that the game in Figure 4.6 is a game of ordered signals and that merging K1 and K2 constitutes an abstraction transformation of GameShrink can be found in the appendix.

4.5.4 Generating abstractions by considering all levels at once

Motivated by the problems described in Section 4.5.2, we developed an integer program (IP) for computing abstractions with bounded loss according to Corollary 2 that operates on all levels at once (we aim to minimize an ex-post bound, so we take the maximum over all expectations that depend on the choice of strategy profile). 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. The IP is shown in the appendix.

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 [68] 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 poker the signal tree is the tree of possible card dealings throughout the game.

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 additional bookkeeping variables) 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 4.7 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 the model where

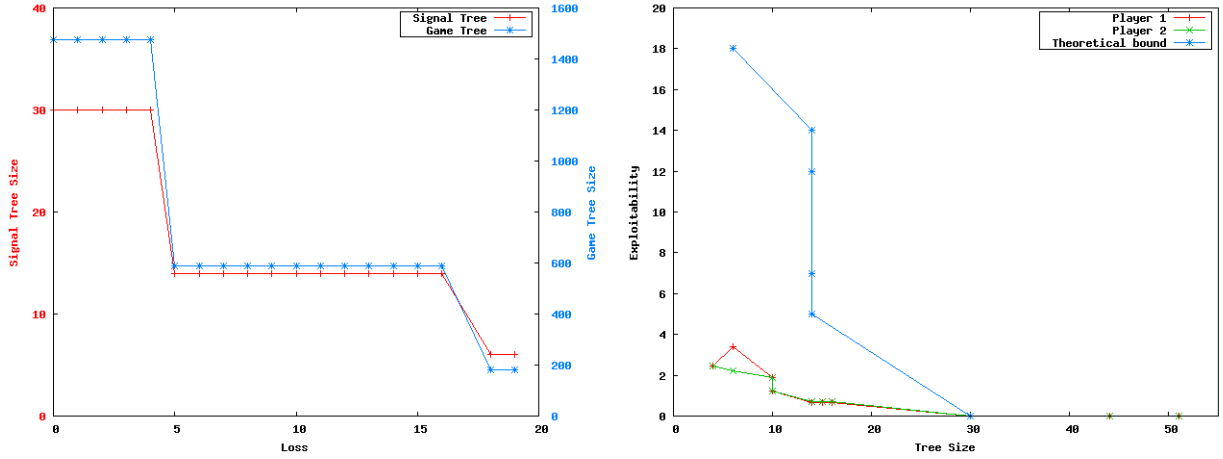


Figure 4.7: Left: Number of nodes in the signal tree (left y-axis) and in the game tree (right y-axis) as a function of the allowed loss in the IP model when minimizing tree size. Right: Exploitability as a function of the allowed signal tree size. Exploitability for both players is shown, along with our theoretical bound.

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 4.7 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 Corollary 2. 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 [168]. when we allow a tree size of 30, the lossless abstraction is found, and beyond that all experiments show zero exploitability.

4.6 Imperfect-recall abstraction

Imperfect-recall abstraction is currently the most important type of abstraction employed in practice [27, 28, 170]. This is in spite of the fact that computing a Nash equilibrium in an imperfect-recall zero-sum game is NP-hard [92][4], and the CFR algorithm is known to cycle on certain imperfect-recall games [109]. For practical purposes, imperfect-recall abstractions are generated via clustering, and regret-minimization algorithms perform well. We now show how our decomposition results can be used to reason about particular classes of imperfect-recall abstractions.

[4]That said, the type of game used in showing hardness is most likely substantially more complicated than the types of imperfect-recall abstractions encountered in practice

4.6.1 Imperfect-recall abstraction without probability-error dependence on strategies

We now show that, similar to mapping error, if the reach of leaf nodes in the original and abstract game are the same without considering Chance moves, we can bound partitioning error with an expression that does not depend on the best response σ_i^* of Player i.

Proposition 3. *If σ' is such that $\pi_{-0}^{\sigma'}(z'|I'_I, a') = \pi_{-0}^{\sigma'}(\hat{z}'|I', a')$ for all $I'_I, a', z', \hat{z}' \in \phi_{I'_I}(z')$, then*

$$\begin{aligned}\mathrm{Pdiff}(I'_I, \sigma'_{I\to\sigma^*}) - \mathrm{Pdiff}(I'_I, \sigma') \leq 2 \max_{a'\in A_{I'}} \Bigg[\sum_{z'\in\mathcal{Z}_{I'}^{a'}} \pi^{\sigma'}(z'|I', a')\Delta_i^R(z'|I'_I) \\ + \pi_{-0}^{\sigma'}(z'|I'_I, a') \sum_{\hat{z}'\in\phi_{I'_I}(z')} \left[\pi_0^{\sigma'}(\hat{z}'|I', a')) - \pi_0^{\sigma'}(z'|I'_I, a'\right] \Bigg] \overset{\text{def}}{=} \overline{\mathrm{Pdiff}}(I'_I, \sigma'_{I\to\sigma^*}, \sigma'), \qquad \forall\sigma'_{I\to\sigma^*}\end{aligned}$$

Proof. Since $\pi^{\sigma'_{I\to\sigma}}(z'|I') = \pi^{\sigma'}(z'|I', a')\sigma'_{I\to\sigma}(I', a')$ we can write the difference as

$$\begin{aligned}\mathrm{Pdiff}(I'_I, \sigma'_{I\to\sigma^*}) - \mathrm{Pdiff}(I'_I, \sigma') = \sum_{a'\in A_{I'}} [\sigma'_{I\to\sigma^*}(I', a') - \sigma'(I', a')] \Bigg[\\ \sum_{z'\in\mathcal{Z}_{I'}^{a'}} \pi^{\sigma'}(z'|I', a')\Delta^R(z'|I'_I) + \sum_{z'\in\mathcal{Z}_{I'_I}^{a'}} \Delta^P(z'|I'_I, \sigma', a')u_i(z') \Bigg] \qquad (4.15)\end{aligned}$$

We can bound this by two times the maximum value of the expression in the second brackets, where the maximum is over all $a' \in A_{I'}$ to get

$$(4.15) \leq 2 \max_{a'\in A_{I'}} \left[\sum_{z'\in\mathcal{Z}_{I'}^{a'}} \pi^{\sigma'}(z'|I', a')\Delta^R(z'|I'_I) + \sum_{z'\in\mathcal{Z}_{I'_I}^{a'}} \Delta^P(z'|I'_I, \sigma', a')u_i(z') \right]$$

Now by our condition $\pi_{-0}^{\sigma'}(z'|I'_I, a') = \pi_{-0}^{\sigma'}(\hat{z}'|I', a')$ we have

$$\begin{aligned}\Delta^P(z'|I'_I, \sigma', a') &= \pi^{\sigma'}(z'|I'_I, a') - \sum_{\hat{z}'\in\phi_{I'_I}(z')} \pi^{\sigma'}(\hat{z}'|I', a') \\ &= \pi^{\sigma'}(z'|I'_I, a') - \sum_{\hat{z}'\in\phi_{I'_I}(z')} \pi_{-0}^{\sigma'}(z'|I'_I, a')\pi_0^{\sigma'}(\hat{z}'|I', a') \\ &= \pi^{\sigma'}(z'|I'_I, a') - \sum_{\hat{z}'\in\phi_{I'_I}(z')} \pi_{-0}^{\sigma'}(z'|I'_I, a') \left[\pi_0^{\sigma'}(z'|I'_I, a') + \left(\pi_0^{\sigma'}(\hat{z}'|I', a')) - \pi_0^{\sigma'}(z'|I'_I, a'\right)\right] \\ &= \pi_{-0}^{\sigma'}(z'|I'_I, a') \sum_{\hat{z}'\in\phi_{I'_I}(z')} \left[\pi_0^{\sigma'}(\hat{z}'|I', a')) - \pi_0^{\sigma'}(z'|I'_I, a'\right]\end{aligned}$$

which proves the theorem. □

This can be combined with our main theorems in order to get results for ϵ-Nash equilibrium or strategies with bounded regret where the partition error does not depend on the best response.

Corollary 3. *If σ' has bounded counterfactual regret $r(I')$ at every information set $I' \in \mathcal{I}'$, satisfies the condition of Proposition 3, and* Mdiff *is zero everywhere, then any lifted strategy $\sigma^{\uparrow\sigma'}$ is an ϵ-Nash equilibrium where $\epsilon = \max_{i \in N} \epsilon_i$ and*

$$\epsilon_i \leq \sum_{I \in \mathcal{I}_i} \pi^{\sigma^*}(I) \left[\delta_{f(I)_I} r(f(I)) + \overline{\text{Pdiff}}(I'_I, \sigma'_{I \to \sigma^*}, \sigma') \right]$$

An analogue to Corollary 3 but for ϵ-Nash equilibrium can be obtained by combining Theorem 8 with Proposition 3.

Corollary 4. *If σ' is an abstract ϵ'-Nash equilibrium, satisfies the condition of Proposition 2, and* Mdiff *is zero everywhere, then any lifted strategy profile $\sigma^{\uparrow\sigma'}$ is an ϵ-Nash equilibrium where $\epsilon = \max_{i \in N} \epsilon_i$ and*

$$\epsilon_i \leq \epsilon' + \sum_{I \in \mathcal{I}_i} \pi^{\sigma^*}(I) \overline{\text{Pdiff}}(I'_I, \sigma'_{I \to \sigma^*}, \sigma')$$

For practical game solving, Corollary 3 has an advantage over Corollary 4: 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 an imperfect-recall abstraction, and achieve the bound in Corollary 3, with the information set regrets $\pi^{\sigma^*}(I) r(f(I))$ decreasing at a rate of $O(\sqrt{(T)})$. Conversely, no good algorithms are known for computing Nash equilibria in imperfect-recall games.

4.6.2 Chance-relaxed skew-well-formed (CRSWF) games

Now we introduce a class of imperfect-recall games that we consider as potential abstractions of a perfect-recall game. This class of games is guaranteed to satisfy the condition in Proposition 3 and thus we can bound the regret of strategies computed in abstractions of this game class. 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.

We will say that a perfect-recall game Γ *refines* Γ' if the games are identical, except that the information sets in Γ are such that they partition each of the information sets in Γ'. This is completely analogous to our notion of Γ' being an abstraction of Γ, where the information sets in Γ correspond to a partitioning of each information set in Γ'. We let $\mathcal{P}(I')$ be the set of information sets in Γ that refine the particular information set I' of Γ'.

Definition 10. *For an EFG Γ' and a 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 $\phi_{I,\breve{I}} : Z_I \to Z_{\breve{I}}$ such that for all $z \in Z_I$:*

1. *For leaf nodes z and $\phi_{I,\breve{I}}(z)$ mapped to each other, the action sequences of all players except i and Chance on those two paths must be the same in the abstraction.*[5]
2. *For leaf nodes z and $\phi_{I,\breve{I}}(z)$ mapped to to each other, the action sequence of Player i from I to z and from $\breve{I}$ to $\phi_{I,\breve{I}}(z)$ 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.

We can think of an CRSWF game as an abstraction of any of its refinements. Thus we can apply our general theory. Since a CRSWF game Γ' and a refinement Γ are identical except for the information-set structure, actions and leaf nodes in either game uniquely identify actions and leaf nodes in the other game. Thus we refer to actions and leaf nodes in either game as convenient. If we view $\mathcal{P}$ as a partitioning of Γ' information sets then it satisfies the theory of Section 4.6.1. Thus Corollaries 3 and 4 apply to CRSWF games and their refinements, and we get that strategies computed in Γ' have bounded regret in any refinement.

Instead of our probability error terms, Lanctot et al. [109] require $\pi_0(z|I) = l_{I,\breve{I}}\pi_0(\phi_I(z)|\breve{I})$, 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 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. The solution quality bounds we get are exponentially stronger than those of Lanctot et al. [109], 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.

Figure 4.8 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 probability 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

[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.

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.

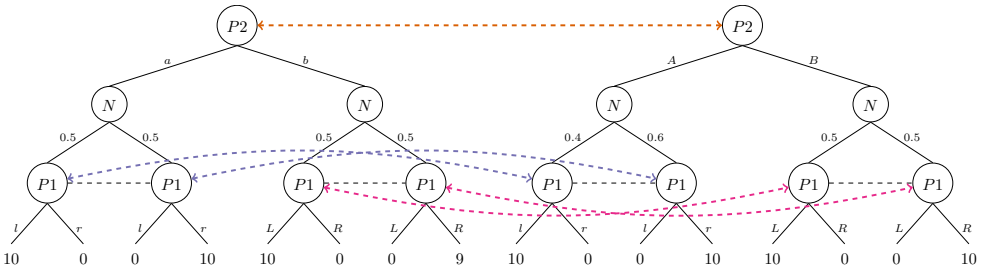


Figure 4.8: Two subtrees of a game tree. Information sets are denoted by dotted lines. A CRSWF abstraction is shown, with merged information sets and their node mapping denoted by dotted lines with arrowheads. All actions are mapped to their corresponding upper/lower-case actions in the merged information sets.

4.6.3 Complexity and algorithms

We now investigate the problem of computing CRSWF abstractions with minimal ex-ante error bounds. By ex-ante we mean that we take maxima over all probabilities relating to the choice of player strategies in the 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 ex-ante bound as in Corollary 3 or 4 exists is NP-complete. It follows that it is NP-complete to compute the minimum number of information sets given a maximum bound. This holds even if there is only a single player, and the game tree has height two.*

Proof. Consider the *two-dimensional k-center clustering decision problem* with the L_q distance metric. It is defined as follows: given a set $P = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ of n points in the plane, and an integer k, does there exist a partition of P into k clusters $\mathcal{C} = \{c_1, \ldots, c_k\}$ such that the maximum distance $\|p - p'\|_q \leq c$ between any pair of points p, p' in the same cluster is minimized. This problem is NP-hard to approximate within a factor of 2 for $q = \infty$, amongst others [61].

Given such a problem, we construct a perfect-recall game as follows. For each point $p \in P$, we construct an information set I_p. We insert two nodes h_p^x, h_p^y in each information set I_p,

[6] Sandholm and Singh [152] already showed hardness of computing an optimal abstraction when minimizing the actual loss of a unique equilibrium.

representing the dimensions x, y respectively. All these nodes descend directly from the root node r, where Player 1 acts. At each information set we have two actions, a_c, a_v. For any point p, we add leaf nodes at the branch a_c with payoff $M, 2M$ at the nodes h_p^x, h_p^y respectively. If we pick a sufficiently large M, this ensures that for any two points p, p', their nodes $h_p^x, h_{p'}^x$ will map to each other, and similarly for y. This also ensures that the scaling variable has to be set to 1 for all information set mappings. For the branches a_v, we add leaf nodes with utility equal to the x, y coordinate of p at the h_p^x, h_p^y nodes respectively.

There is a one-to-one mapping between clusterings of the points P and partitions of the information sets $\{I_p : p \in P\}$. The quality of a clustering is

$$\max_{z \in \{x,y\}} \max_{j=1,\ldots,k} \max_{p,p' \in c_j} |p(z) - p'(z)| .$$

Since Player 1 acts at r, the abstraction quality bound is equal to the maximum difference over any two leaf nodes mapped to each other, as there are no chance outcome differences. This is the same as the quality measure of the clustering. Thus, an optimal k size clustering is equivalent to an optimal k information set abstraction.

Given some CRSWF abstraction, verifying the solution is easy to do: in one top-down traversal of the game tree, compute the node distributions at each information set. For each full-game information set, this gives the distribution-approximation error. For each information set pair mapped to each other, the transition- and reward-approximation error can now be computed by a single traversal of the two. Thus the problem is in NP. □

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 the conditions of Definition 10. 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 Figure 4.6). 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 [41], 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. [109] as a game abstraction benchmark is amenable to this approach.

We now show that single-level abstraction problems (SLAPs) where the conditions of Definition 10 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 [61, 73].

Definition 11. *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 4. *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.*

Proof. The first condition follows from the other three. Condition b, identity of indiscernibles, does not hold for information sets. However, any pair of information sets with distance zero can be merged losslessly in preprocessing, thus rendering the condition true (having distance zero is transitive, so the minimal preprocessing solution is unique). Condition c, symmetry, holds by definition, since our distance metric is defined as the error incurred from merging two information sets, which considers the error from both directions of the mapping.

Finally, we show that Condition d, the triangle inequality holds. Consider any three information sets $I_1, I_2, I_3 \in \mathcal{I}^m$. We need to show that $d(I_1, I_3) \leq d(I_1, I_2) + d(I_2, I_3)$. Let $\phi_{I_1,I_2}, \phi_{I_2,I_3}$ be the mappings for I_1, I_2 and I_2, I_3 respectively. We construct a mapping $\phi_{I_1,I_3} = \phi_{I_2,I_3} \circ \phi_{I_1,I_2}$ and show that it satisfies the triangle inequality. For the leaf payoff error, since $\delta_{I_1,I_2} = \delta_{I_2,I_3} = 1$, at any leaf $z \in Z_{I_1}$ we get:

$$u_i(z) \leq u_i(\phi_{I_1,I_2}(z)) + \epsilon_{I_1,I_2}(z) \leq u_i(\phi_{I_2,I_3}(\phi_{I_1,I_2}(z))) + \epsilon_{I_2,I_3}(\phi_{I_1,I_2}(z)) + \epsilon_{I_1,I_2}(z)$$

For the nature leaf probability error we can apply the same reasoning:

$$\begin{aligned} & \pi_0^\sigma(z[I_1], z) \\ \leq\ & \pi_0^\sigma(\phi_{I_1,I_2}(z[I_1]), \phi_{I_1,I_2}(z)) + \epsilon^0_{I_1,I_2} \\ \leq\ & \pi_0^\sigma(\phi_{I_2,I_3}(\phi_{I_1,I_2}(z[I_1])), \phi_{I_2,I_3}(\phi_{I_1,I_2}(z))) + \epsilon^0_{I_2,I_3}(\phi_{I_1,I_2}(z)) + \epsilon^0_{I_1,I_2}(z) \end{aligned}$$

Again, we derive the distribution error using a similar approach:

$$\begin{aligned} & \frac{\pi_0(z[I_1])}{\pi_0(I_1)} \\ \leq\ & \frac{\pi_0(\phi_{I_1,I_2}(z[I_1]))}{\pi_0(I_2)} + \epsilon^D_{I_1,I_2}(z[I_1]) \\ \leq\ & \frac{\pi_0(\phi_{I_2,I_3}(\phi_{I_1,I_2}(z[I_1])))}{\pi_0(I_3)} + \epsilon^D_{I_2,I_3}(\phi_{I_2,I_3}(z[I_1])) + \epsilon^D_{I_1,I_2}(z[I_1]) \end{aligned}$$

This completes the proof. □

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 [73] gave a 2-approximation algorithm when the distance function satisfies the triangle inequality. Coupled with Proposition 4 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. [74] developed approximation algorithms for this problem based on a pre-publication manuscript of our work.

Generally, 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 [67, 70] (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 4 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.

4.6.4 Experimental performance of CRSWF abstractions

We now investigate what the optimal single-level CRSWF abstraction bounds (in terms of Corollary 3) look like for the *die roll poker (DRP)* game, a benchmark game for testing abstraction [109]. 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 10. 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 section we will focus on abstraction at the bottom level of the tree. We use CPLEX to solve an IP encoding the single-level abstraction problem of minimizing our bound given a limit on the number of abstract information sets. The results are shown in Figure 4.9. For one or two clusters, the bound is bigger than the

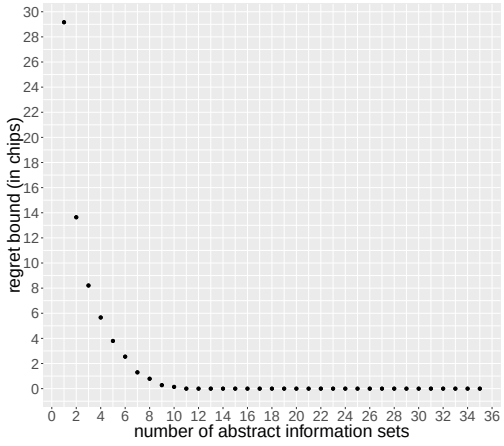


Figure 4.9: Regret bounds for varying single-level abstraction problem sizes in DRP. The x-axis shows the number of information sets in the abstraction, and the y-axis shows the theoretical bound on solution quality. The total number of information sets in the original game is 36

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. [109] conducted experiments to investigate the performance of CFR strategies computed in imperfect-recall abstractions of several games: DRP, Phantom 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. scheme ensures that we compute an imperfect-recall abstraction that falls under the CRSWF game class. 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 4.10 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 Corollary 3 for an exact Nash equilibrium. On the left in Figure 4.10 is shown the results for the

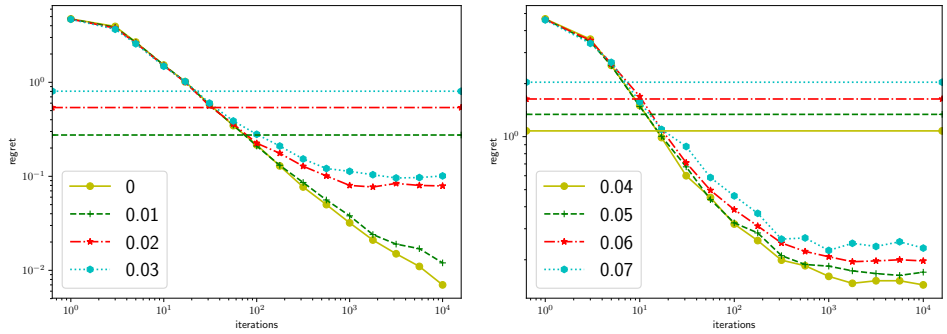


Figure 4.10: Log-log plots of the sum of the two players' regrets as a function of CFR iterations on the bound-minimizing abstraction of CDRP. The legends give the amount of correlation in the die rolls of the different CDRP games on which we ran experiments. The horizontal lines show the respective ex-ante regret bound of Corollary 3 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.)

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.

4.7 Neccessity of distributional similarity of reach probabilities

We now show that the style of bound given by Lanctot et al. [109] as well as our corrolaries 2 and 3 cannot generalize to games where opponents do not have the same sequence of information-set-action pairs, or in our case the slightly weaker requirements in Propositions 2 and 3, for game nodes that map to each other in the abstraction. The two games that we will use as counterexamples are shown in Figure 4.11. From the perspective of our results, the usefulness of assuming the same sequence of information-set-action pairs is that it implies the condition used in Propositions 2 and 3; the following counterexamples thus also show that this assumption is a useful way to disallow bad abstractions such as the ones presented here (although overly restrictive from a practical perspective). Contrary to the prior results, our Theorems 8 and 9 still apply to the games below. Our two theorems would give weak bounds commensurate with the large error in the abstract equilibrium; this error is contained in the terms that depend on Δ^P.

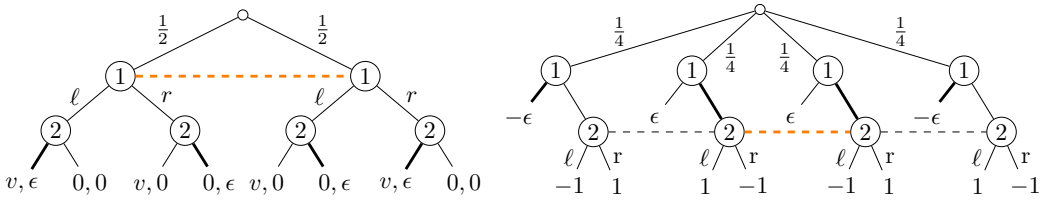


Figure 4.11: Left: General-sum EFG with abstraction. Right: zero-sum EFG with abstraction where Player 1 wants to minimize. Orange dashed lines denote information sets joined in the abstraction. Bold edges denote actions taken with probability 1 in the abstracted equilibrium.

On the left in Figure 4.11 is a general-sum game where the two nodes belonging to Player 1 are abstracted into a single information set. If we map ℓ onto ℓ and r onto r we get an abstraction with low payoff error: ϵ at every node. At a high level, the idea in this counterexample is that Player 2, because their nodes are not abstracted, can play opposite actions in the left and right subtrees, thus changing whether Player 1 prefers going left or right. In the original game Player 1 can react to this by choosing different actions, but not in the abstraction. Formally: Let $\epsilon > 0$. Player 2 plays the bolded edges at nodes with non-zero probability of being reached. In the abstraction, Player 1 gets $\frac{v}{2}$ for every strategy. In the full game, Player 1 can choose ℓ in the left subtree and r in the right subtree for a payoff of v. Thus in every equilibrium where Player 2 plays according to the bolded edges (which includes all equilibrium refinements) Player 1 loses $\frac{v}{2}$ from abstracting, despite the payoff error being arbitrarily small. If we set $\epsilon = 0$, equilibria where Player 2 plays the bolded edges still have high loss—despite zero payoff error. This example showed that information-set-action structure has to be taken into account in order to get satisfying bounds in general. While the example is very simple (and can thus easily occur

in the context of a larger game), it does exploit the fact that Player 1 utility is discontinuous in Player 2 utility. We next show that a more intricate counterexample can avoid relying on this discontinuity.

On the right in Figure 4.11 is a zero-sum game where the two bottom information sets belonging to Player 2 have been abstracted. Consider the following abstract equilibrium: Player 1 plays the bolded edges with probability 1, and Player 2 plays ℓ, r with equal probability. Player 2 gets expected utility $-\frac{\epsilon}{2}$, but in the full game Player 2 can choose ℓ (r) in the left (right) information set to get utility $\frac{1-\epsilon}{2}$. Thus Player 2 has a utility loss of $\frac{1}{2}$ despite a payoff error of 0. The idea in this example is that, because Player 1 is not abstracted, they control the distribution over nodes in Player 2's information set in the abstraction in a way that is inconsistent with Player 2's original-game information sets: in the abstraction they get an equal distribution over nodes where ℓ or r is the preferred action, whereas in the original game the corresponding strategy for Player 1 means that they know exactly which node they are at.

4.8 Conclusions and future work

We showed an exact decomposition of the error resulting from solving an abstraction of an EFG rather than the original game. This decomposition had a dependence on the strategies played and best responses, and thus it is not immediately helpful for ex-ante analysis of abstractions. However, we went on to show that this decomposition can be combined with assumptions on the information-set-action structure of the game, in order to get state-of-the-art ex-ante bounds on solution quality. Our ex-ante bounds generalize the only class of abstractions where bounds were previously known, while being exponentially better. We showed results for both perfect- and imperfect-recall abstractions. Our results were also the first algorithm-agnostic bounds, and the first to allow ϵ-Nash equilibria computed in the abstraction. We then showed that computing optimal abstractions is computationally hard for several different variants of the problem, and that it is generally impossible to give bounds on solution quality purely based on payoff and chance-outcome error.

In spite of our hardness and impossibility results abstraction is employed to great success in practice (e.g. the Libratus agent). The computational issues are circumvented by reducing the problem to clustering, and employing off-the-shelf clustering algorithms that perform well in spite of the theoretical hardness of the problem. One way forward would be to analyze particular game classes where one could potentially prove that our impossibility result does not apply. This is complicated by the fact that our zero-sum EFG counterexample probably applies to certain strategy profiles even in poker. Thus any such approach would most likely need to argue that this does not happen in strategies that matter, for example ones near equilibrium.

In work subsequent to this thesis work, Čermák et al. [36] investigate a particular class of imperfect-recall abstraction, called *A-loss recall games*, and show that some computational problems are tractable in this class while being hard in general imperfect-recall games. It would be interesting to relate that game class to abstraction-quality results.

From a theoretical perspective it would also be interesting to understand how abstraction relates to timeability issues [80].

4.9 Discretizing continuous action spaces

This thesis, and the computational EFG literature at large, has 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. [79, 108, 139, 165, 179]). (One notable exception to this was introduced by Johanson et al. [83], 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. This section extends our work on abstraction to certain types of continuous 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 4.5 (though the proofs presented here use an earlier version of our results presented in Kroer and Sandholm [95]), 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 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.

4.9.1 Continuous action spaces

We will assume that we are dealing with a game Γ, where one or more nodes $h \in H$ have one or more continuous action intervals $A_{h,c} = [\alpha_{h,c}, \beta_{h,c}] \subseteq A_h$ for $c \in C_h$, where C_h is an index set of the continuous action spaces at h. We also assume that each layer of the game tree belongs to a single player. The set of heights in the game tree will be denoted $\mathcal{H}$ and $\mathcal{H}_i$ is the set of heights belonging to Player i.

Let H_a be the set of nodes in the subtree reached by taking action $a \in A_{h,c}$ at node h. We assume there is a one-to-one mapping $\phi_{a,\hat{a}} = \phi_{\hat{a},a}$ between H_a and $H_{\hat{a}}$ for any two actions $a, \hat{a} \in A_{h,c}$, where nodes at a given height map onto nodes at the same height, and the condition of Definition 12 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 $A_{h,c}$ at h, 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 12. *For any node $\hat{h} \in H_a$ in the subtree at $a \in A_{h,c}$, we require one of the two following conditions to hold for all a':*

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

For each leaf node $z \in Z_{t_a^h}$ for some $a \in A_{h,c}$, we assume that the payoff to Player i can be described as $u_i(z) = \mu_z^i(a)$, where $\mu_z^i = \mu^i_{\phi_{a,\hat{a}}(z)}$ for all $\hat{a} \in A_{h,c}$. 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 h' in the subtree rooted at t_a^h is described by a function $\rho_{h'}(a)$ where $\rho_{h'} = \rho_{\phi_{a,\hat{a}}}(h')$ for all $\hat{a} \in A_{h,c}$.

Since we require a bijection mapping $\phi_{a,\hat{a}}$ between the subtrees for any two actions $a, \hat{a} \in [\alpha_{h,c}, \beta_{h,c}]$ for each node h 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 $a \in [\alpha_{h,c}, \beta_{h,c}]$, and the information set topology of the instantiations satisfy Definition 12.

An example game is shown in Figure 4.12. 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 12, while the one that does not span across the trees satisfies Condition 2.

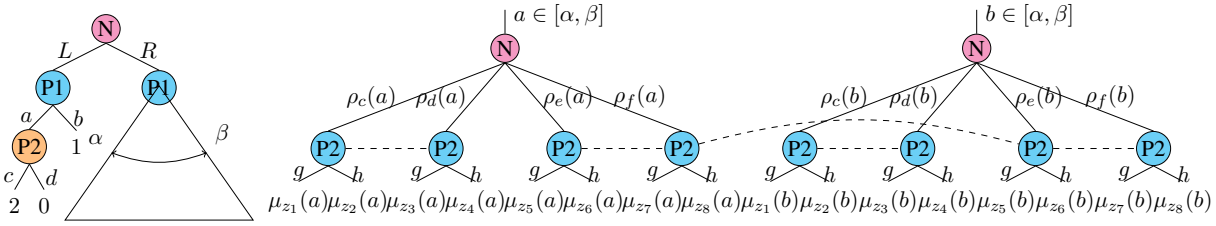


Figure 4.12: An example of a game tree with the triangle representing a subtree with a continuous action space at the node.

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 h that have one or more continuous action intervals, and h is the first node on the path from the root to h with a continuous action space. Let the set of all such nodes be $\mathcal{S}^1 \subseteq H$. These nodes are handled exactly as described above. If there are

additional continuous action spaces in the subtree after taking some action $a \in A_{h,c}$ for some h, c, then the bijections between subtrees simply map uncountably many nodes.

The second group is the set of all nodes h' such that there is at least one node-action pair h, a on the path from the root to h', where h has a continuous action space $A_{h,c}$ such that $a \in A_{h,c}$. Let this set of nodes be called $\mathcal{S}^2 \subset H$. Let $\vec{a} \in \mathbb{R}^n$, where n is the number of continuous action spaces leading to h' and including the one at h', 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 h'. 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 $\mathcal{A} = \times_{h \in \mathcal{S}^1 \cup \mathcal{S}^2, c \in C_h} A_{h,c}$ 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 $\vec{a} \in \mathcal{A}$ to denote elements of this set. A *discretization* is a finite set of points $\mathcal{A}' \subset \mathcal{A}$. The size $|\mathcal{A}'| = m$ is the number of discrete points chosen for each interval. If fewer than m points are desired for some interval $A_{h,c}$ with index $j \in \mathbb{I}$, we can simply let $\vec{a}_j = \vec{a}'_j$ for distinct points $\vec{a}, \vec{a}' \in \mathcal{A}'$. For a node h, we will use $\vec{a}^h$ to refer to the subset of actions taken on the path to h 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 h, they take elements $\vec{a} \in \mathcal{A}$ as input, with the implicit understanding that the value of the function depends only on $\vec{a}^h$. 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 $\Gamma' = \langle H', Z', A', \mathcal{H}, \sigma_0, \{\mathcal{I}'\}, \{u_i\} \rangle$ is the discrete extensive-form game obtained by restricting players to selecting actions that are part of the discretization $\mathcal{A}'$.

4.9.2 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 $\mathcal{A}'$. To do this we will require a mapping of the continuous action space onto the discretization:

Definition 13. *A* discretization mapping *is a surjective function $g : \mathcal{A} \to \mathcal{A}'$ that maps the continuous action space $\mathcal{A}$ onto the discretization $\mathcal{A}'$. We require that g is decomposable, so for all $\vec{a} \in \mathcal{A}$, $g(\vec{a})_j$ depends only on $\vec{a}_j$. $\mathcal{G}_{\mathcal{A}'}$ denotes the set of legal discretization maps for a given discretization $\mathcal{A}'$.*

The discretization mapping along with the bijections $\phi_{a,a'}$ for all $h \in H, c \in C_h, a, a' \in A_{h,c}$ immediately defines a mapping of the nodes H onto the nodes H'. Denote this node mapping function by $\phi : H \to H'$. For any node $h \in H \cap H'$, $\phi(h) = h$. For $h \in H, h \notin H'$, $\phi(h)$ is the node in H' reached by inductively applying the maps g and $\phi_{\vec{a}^h_j, g(\vec{a}^h)_j}$ at each continuous action space on the path to h.

Due to the constraints in Definition 12, g also leads to an information set mapping, as any two nodes $h_1, h_2 \in I$ for some I must map to the same information set: $\phi(h_1), \phi(h_2) \in I'$ for some I'. We let $f : \mathcal{I} \to \mathcal{I}'$ be this information set mapping.

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

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

$$\epsilon^R_{h,i} = \begin{cases} \max_{a \in A_h} \epsilon^R_{t^h_a,i} & \text{if } h \text{ is a player node} \\ \int_{a \in A_h} \sigma_0(h,a) \epsilon^R_{t^h_a,i} & \text{if } h \text{ is a nature node} \\ |\mu^i_h(\vec{a}) - \mu^i_h(g(\vec{a}))| & \text{if } h \text{ is a leaf node} \end{cases}$$

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

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

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

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

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

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

The nature error at height k is

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

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 discretization problem in Section 4.9.1) 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 $s \in H$, we set $\sigma(h, a) = \sigma'(\phi(h), a')$, where a' is the action at h' that leads to $\phi(t_a^h)$. 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 5. *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 $\sigma_i(I) > 0$,*

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

This follows immediately from how we defined the mapping.

4.9.3 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 4.13 illustrates the approach.

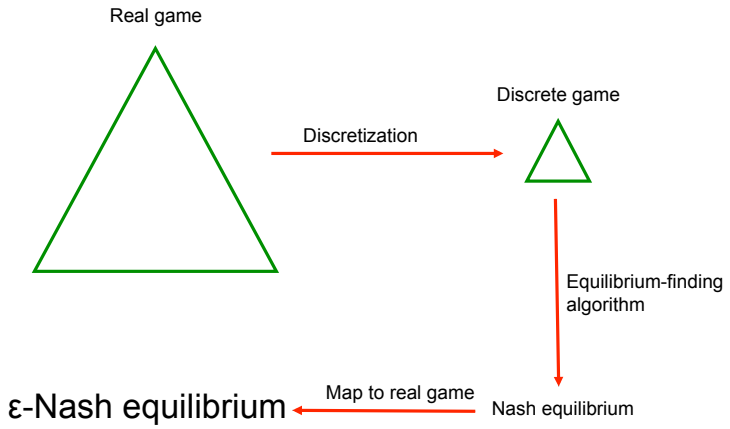


Figure 4.13: An overview of our discretization approach.

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 4.5.

4.9.4 Discretization quality bounds

We start by showing an error bound on solution quality for any given discretization and discretization mapping, leveraging our result on bounded solution quality for perfect-recall abstraction.

Theorem 14. *For any game Γ with continuous action spaces $\mathcal{A}$ that satisfy the constraint given in Definition 12, discretization $\mathcal{A}' \subset \mathcal{A}$, and discretization mapping $g : \mathcal{A} \to \mathcal{A}'$, any Nash equilibrium σ computed in Γ' constitutes an ϵ-Nash equilibrium when implemented in Γ, where*

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

Proof. We can view the game Γ' obtained by this discretization as an abstraction, where g defines a mapping of each real action a to some abstract action a'. Coupled with the bijection $\phi_{a,a'}$, this induces a node mapping h and information set mapping f, as argued in Section 4.9.2.

Theorem 4.2 of Kroer and Sandholm [95] gives error bound results for abstractions like this. To see that f indeed forms a surjective function that respects h, consider Conditions 1 and 2 of Definition 12, which ensure that any discretization induces an information set mapping. First consider when Condition 1 is satisfied. In this case the information set mapping is obviously respected, as the nodes that map to each other were already in the same information set.

Now consider any information set I such that a node $s \in I$ is mapped onto another node $\hat{h} \in \hat{I}, \hat{I} \neq I$. Condition 2 ensures that this only happens for information sets completely contained in the subtrees rooted at the continuous interval in question. We have to show that I is surjectively mapped onto $\hat{I}$. Condition 2 states that for any other node $\bar{h} \in I$, $\bar{h}$ must map to a node in the same information set as $\hat{h}$. Thus we just have to verify that every node $h^* \in \hat{I}$ has a node from I mapped onto it. This is immediately seen by applying Definition 12 to the bijection from the perspective if $\hat{I}$, as the bijection has to be the same when applying the definition in both directions.

Theorem 4.2 of Kroer and Sandholm [95] also requires that the strategy implemented in the full game is an *undivided lifted strategy*. We did not quite guarantee this property with our strategy mapping described in Section 4.9.2. However, this property is only used in the very last step of the proof of Theorem 4.2 in their paper, where they use it to apply Proposition 3.1 of their paper. Instead, we can apply our Proposition 5, which achieves the same effect.

The result as presented by Kroer and Sandholm [95] takes the maximum nature outcome error for each height. Their proof is easily modified to take the expected value, ϵ^0_k, instead, as we do in the theorem. (This approach was also taken by Kroer and Sandholm [98].) □

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_h 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:

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

$$\epsilon_k^0 \leq \int_{h \in H_k} \pi(h) \sum_{a' \in A_{h'}} \left| \int_{a \in g_I^{-1}(a')} \sigma_0(h,a) - \sigma_0(h',a') \right| = \xi_k^0$$

This gives the following corollary:

Corollary 5. *For any game Γ with continuous action spaces $\mathcal{A}$ that satisfy the constraint given in Definition 12, discretization $\mathcal{A}' \subset \mathcal{A}$, and discretization mapping $g : \mathcal{A} \to \mathcal{A}'$, any Nash equilibrium σ computed in Γ' constitutes an ϵ-Nash equilibrium when implemented in Γ, where*

$$\epsilon = \max_i \left\{ 2\epsilon_{h,i}^R + \sum_{k \in \mathcal{H}_i} \xi_k^0 \overline{W} \right\} + 2 \sum_{k \in \mathcal{H}_0} \xi_k^0 \overline{W}$$

The ξ_k^0 terms do not diverge. While an infinite sum is taken in both cases, the terms are probability weighted, and $\int_{I \in \mathcal{I}_k} \pi(I) = 1, \int_{h \in H_k} \pi(h) = 1$. Since we are dealing with probability distributions, we can take the maximum over $\mathcal{I}_k$ or H_k. In practical settings, it may be desireable to take several maxima. First, in the current form of both Theorem 14 and Corollary 5, 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 H_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.

4.9.5 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 4.9.4, 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 4.9.5.

Optimal discretization as an optimization problem

We consider a more general class of problems than those described in Section 4.9.1. We consider a game Γ where one or more $h \in H$ has a continuous action space. We again let $\mathcal{A}$ 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 $\Psi : \mathcal{A}^k \times \mathcal{G}_{\mathcal{A}} \to \mathbb{R}$ that takes as input a discretization $\mathcal{A}'$ and discretization map g and returns a real-valued error bound $\Psi(\mathcal{A}', g)$. 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 $\mathcal{A}'$, 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 $\vec{a}_j^l \in \mathcal{A}_j, l \in [k_j]$ and $g_{h,c}^\tau, \tau \in [k_j - 1]$, where k_j is the desired number of discrete points for interval $\mathcal{A}_j$. 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 $\mathcal{A}'_v, g_v$ denote the discretization and mapping obtained by a solution. The feasible set of this problem is

$$\mathcal{F} = \left\{ \mathcal{A}'_v, g_v : \begin{array}{ll} \vec{a}_j^l \le \vec{a}_j^{l+1} & \forall j \in \mathbb{I}, l \in [k_j - 1] \\ \vec{a}_j^\tau \le g_j^\tau \le \vec{a}_j^{\tau+1} & \forall j \in \mathbb{I}, \tau \in [k_j - 1] \\ \vec{a}_j^l \in \mathcal{A}_j & \forall j \in \mathbb{I}, l \in [k_j] \\ g_j^\tau \in \mathcal{A}_j & \forall j \in \mathbb{I}, \tau \in [k_j - 1] \end{array} \right\} \tag{4.16}$$

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

$$\min \left\{ \Psi(\mathcal{A}'_v, g_v) : (\mathcal{A}'_v, g_v) \in \mathcal{F} \right\} \tag{4.17}$$

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 (4.16) are all linear. Thus, if Ψ is convex, solving (4.17) becomes a convex minimization problem over linear constraints. These are solvable in polynomial time under mild assumptions [11][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 [11] give a thorough discussion of such representability), smooth, or non-smooth with certain structure [8, 133].

Decomposable error function

In Section 4.9.4, 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

[7] A heavily updated version of this great book can be found at `http://www2.isye.gatech.edu/~nemirovs/`.

now consider how to (recursively) represent such error functions using linear inequalities, for the purpose of computation.

Let $\xi_j : \mathcal{A}_j \times g_j \to \mathbb{R}$ be an error function that gives the error incurred at interval $j \in \mathbb{I}$ when choosing discretization $\mathcal{A}'_j$ and mapping g_j at $\mathcal{A}_j$. 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:

$$\mathcal{E} = \left\{ \begin{array}{ll} e_h \geq \sum_{a \in A_h} \sigma_0(s,a) e_{t_a^h} \\ e_h \geq \max_{a \in A_h} e_{t_a^h} \\ e_h \geq \xi_j \end{array} \right\} \tag{4.18}$$

The same linear formulation can also be used to formulate the error bound in the case where $\xi_h : \mathcal{A} \times g$ 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 (4.18) by $\mathcal{E}$.

If each error function ξ_j depends only on the subset of $\mathcal{A}$ 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 12 and minimization of the bound given in Corollary 5. 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_h, at all $z \in Z, h \in H$ that are descendants of a continuous interval, are Lipschitz continuous with Lipschitz constant $L_{z/h}$. 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 $\mathcal{A}$, where each utility and nature outcome distribution function is Lipschitz continuous with constant $L_{z/h}$ for each leaf z or nature node h, and all continuous nature intervals are uniformly distributed, the bound-minimizing k_j-point discretization at each interval $\mathcal{A}'_j$ is:*

$$\begin{aligned} \bar{a}_j^l &= \alpha_j + \left(l - \frac{1}{2}\right) \cdot \left(\frac{\beta_j - \alpha_j}{k_j}\right) && \forall l \in [k_j] \\ g_j^\tau &= \alpha_j + \tau \cdot \left(\frac{\beta_j - \alpha_j}{k_j}\right) && \forall \tau \in [k_j - 1] \end{aligned}$$

We will call this a uniform discretization.

Proof. Consider such a discretization and mapping. We will show that any other discretization or mapping can not be better. For an interval $j \in \mathbb{I}$ at a player node, this is easily seen. The error of the interval is the maximum error over the interval. Since all functions are linear, this is simply the point $a \in \mathcal{A}_j$ with the largest distance to its discrete point. For a uniform discretization, this is either endpoint α_j, β_j or some inner point g_j^τ, which all have distance $\frac{\beta_j - \alpha_j}{2k_j}$. For any other discretization, some point must have distance strictly greater than this to its discrete point, thus worsening the bound.

For an interval $j \in \mathbb{I}$ at some nature node h, we first observe that the error function is convex. The error at each leaf or nature node in the subtrees is linear (and thus convex). The error at each other node is the finite sum or maximum over descendant errors. The error over the interval is the integral over error at each subtree. Since taking finite sums, maxima, and integrals all preserve convexity (see Boyd and Vandenberghe [22] for a calculus of convex functions), we get that the error over the interval is convex. For convex functions, local minimizers are global minimizers. Thus, it is sufficient to show that the derivative is zero at the uniform discretization. Since the subtrees have the same structure, by the conditions given in Definition 12, and the fact that the error functions depends only on the distance, we get that the error between any two points $a_1, a_2 \in \mathcal{A}_j$ can be represented by some function $\Delta(|a_1 - a_2|)$. Using this representation, we consider the error of each interval $\left[\vec{a}_j^l, g_j^l\right]$:

$$\int_{\vec{a}_j^l}^{g_j^l} \frac{1}{\beta_j - \alpha_j} \Delta\left(\left|\vec{a}_j^l - t\right|\right) \mathrm{d}t = \frac{1}{\beta_j - \alpha_j} \int_0^{g_j^l - \vec{a}_j^l} \Delta\left(t\right) \mathrm{d}t \tag{4.19}$$

Using exactly the same approach, we can also get the error of the interval $\left[g_j^l, \vec{a}_j^{l+1}\right]$:

$$\frac{1}{\beta_j - \alpha_j} \int_0^{\vec{a}_j^{l+1} - g_j^l} \Delta\left(t\right) \mathrm{d}t \tag{4.20}$$

We see that for any subgradient x of (4.19), $-x$ is a subgradient of (4.20). Using additivity of subdifferentials (see Rockafellar [145] chapter 23), we get that 0 is a subgradient of g_j^l. We can apply exactly the same approach to each $\vec{a}_j^l$ to see that 0 is a subgradient there as well. □

When the conditions of the above theorem do not hold, the decomposition results from Section 4.9.5 and the recursive linearization (4.18) 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

$$|\mu_z(a) - \mu_z(a')| \quad \text{or} \quad |\rho_h(a) - \rho(a')|$$

at each leaf or nature node can be represented by a convex function, the linear constraints in (4.18) can be used to represent the overall error as a convex function. As discussed in Section 4.9.5, 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 11, 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 (4.16), 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:

$$\mathcal{P} = \left\{ \begin{array}{rl} p_z^l \geq & p_z^l(g_j^l) - p_z^l(\vec{a}_j^l) \\ p_z^l(g_j^l) \geq & \mu_z^\gamma(g_j^l) - M \cdot c_j^{\gamma,\tau} \quad \forall \gamma \in \mathbb{P}_j \\ p_z^l(\vec{a}_j^l) \leq & \mu_z^\gamma(\vec{a}_j^l) + M \cdot b_j^{\gamma,l} \quad \forall \gamma \in \mathbb{P}_j \end{array} \right\} \tag{4.21}$$

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 (4.18) 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 (4.18):

$$\min \left\{ e_{\mathrm{r}} : e_{\mathrm{r}} \in \mathcal{E}, (\mathcal{A}'_v, g_v) \in \mathcal{F} \cap \mathcal{P}, b_j^{\gamma,l}, c_j^{\gamma,\tau} \in \{0,1\} \right\} \tag{4.22}$$

4.9.6 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 [41], security games [90, 119, 177], and sequential wifi-jamming under battery constraints [51].

Chen and Bowling [41] 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 [90, 177] or repeated Bayesian Stackelberg games [119]. 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. [119] 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 [51]. 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 section, 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 12). DeBruhl et al. [51] 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. A subsequent paper to this work investigates the use of game abstraction in wireless resource scheduling [43], though they focus on stochastic games. It would be interesting to consider more sophisticated models with imperfect information.

4.9.7 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 [148]. 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 [148].

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 chapter. 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. [28] 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 [63] 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 12. 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 12. 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 12 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.

4.9.8 Conclusions and future work

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 5).

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 5. 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 [63] 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.

Here we showed that our results on perfect-recall abstraction can be used to prove bounds for EFGs with continuous action spaces. It would be interesting to show similar results for the imperfect-recall setting as well.

Chapter 5

Equilibrium refinement

In spite of their popularity, Nash equilibria suffer from a potential deficiency: they might not play reasonably in parts of the game tree that are reached with zero probability in equilibrium. In particular, the only guarantee that Nash equilibrium gives in these parts of the game tree is that it does not give up more utility than the value of the game. Thus, if the opponent makes a big mistake, Nash equilibrium might give back all the utility gained from the opponent making that mistake, since it is only maintaining the value of the game (Miltersen and Sørensen [121] show nice examples of such behavior).

The above shows that Nash equilibrium is not always satisfactory in extensive-form games, and is the motivation for equilibrium refinements [154]. When information is perfect, the classical solution concept of *subgame-perfect equilibrium (SPE)* can be satisfactory, while it is not when information is imperfect. In this latter case, refinements are usually based on the idea of perturbations representing mistakes of the players. In a *quasi-perfect equilibrium (QPE)* [162], a player maximizes her utility in each decision node taking into account the future mistakes of the opponents only, whereas, in an *extensive-form perfect equilibrium (EFPE)*, players maximize their utility in each decision node taking into account the future mistakes of both themselves and their opponents [77, 154].

Computation of Nash equilibrium refinements in EFGs has received some attention in the literature. Von Stengel et. al. [164] give a pivoting algorithm for computing normal-form-perfect equilibria in EFGs. Miltersen and Sørensen [121] give an algorithm for computing quasi-perfect equilibria. Miltersen and Sørensen [120] show how to compute a normal-form-proper equilibrium. Farina and Gatti [57] give an algorithm for computing extensive-form perfect equilibria. All these results rely on linear programming (LP) (in the zero-sum case) or linear complementary programming (LCP). In zero-sum games, several of these solution concepts can be computed in polynomial time using an LP or a series of LPs. However, as previously argued, even for the easier case of Nash equilibria, the LP approach is not scalable for large games (beyond roughly 10^8 nodes in the game tree [68]). Each iteration of an LP-solving algorithm is expensive, and the LP might even be too large to fit in memory. Thus, as with Nash equilibria, we would like to use faster iterative algorithms for computing equilibrium refinements.

In this chapter, we show how to extend the FOMs results from the previous chapter to the computation of an approximate variant of EFPE. Miltersen and Sørensen [121] and Farina and Gatti [57] presented perturbed polytopes of EFGs that capture equilibrium refinements where

each action has to be played with positive probability. We prove that recent results on smoothing techniques for EFGs based on dilating the entropy function can be modified to provide smoothing for such perturbed games, where the perturbations are with respect to *behavioral strategies*. We then instantiate this method for the perturbed game of Farina and Gatti, which leads to our approximate EFPE.

We then experimentally validate our method. We show that it is effective at obtaining low maximum regret at each information set of the game—even ones that have low probability of being reached—while simultaneously achieving the same practical convergence rate that FOMs and the best CFR variants traditionally achieve for just Nash equilibrium. This has benefits both in approximate Nash equilibrium finding (such approximation is necessary in practice in large games) where some probabilities are low while possibly heading toward zero in the limit, and exact Nash equilibrium computation where the low probabilities are actually zero.

5.1 Preliminaries

In the previous chapter we showed how to efficiently compute Nash equilibrium approximations for large-scale games. We will rely on the treeplex and convex optimization concepts and notation developed in that chapter.

We now show an example of why Nash equilibrium may not be satisfactory when dealing with EFGs, independently of whether the game has perfect or imperfect information, and whether it is general- or zero-sum. A Nash equilibrium σ might prescribe irrational play in those information sets that are visited with zero probability when playing according to σ (e.g., [120]). In the general-sum case, consider the left example of Figure 5.1: the strategy profile (σ_1, σ_2) where player 1 always chooses action *x* and player 2 always chooses action y is a NE. However, this strategy profile is irrational: Player 2 is "threatening" to play a suboptimal action, and Player 1 is caving in to the threat. Yet, the threat is not credible: if Player 1 were to actually play action y, it would be irrational for Player 2 to honor the threat.

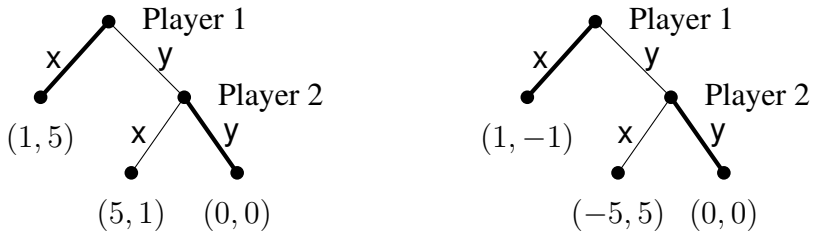


Figure 5.1: General-sum (left) and zero-sum (right) games where Nash equilibrium prescribes irrational play. Numbers in parentheses denote the payoffs to Players 1 and 2.

The right example in Figure 5.1 shows that even in zero-sum games, a NE can fail to capture (sequential) rationality. In this game, the same strategy profile as in the previous game is again a NE. If Player 2 plays according to this profile, she gives up a potential payoff of 5 if Player 1 plays action y.

5.1.1 Perturbations and Extensive-Form Perfection

A way to mend the issue just described is to introduce the idea of "trembling hands": each player cannot fully commit to a pure strategy, and ends up making mistakes with a small (yet strictly positive) probability. This guarantees that the whole game tree gets visited. More formally, let $l(I, a)$ be the *perturbation* of the game, a (positive) function defining the minimum amount of probability mass with which the player playing at information set I in the game will select action a when playing in I. Let Γ_l be the game where players are subject to such perturbation: an extensive-form perfect equilibrium of the game Γ is any limit point of the sequence of Nash equilibria of the game Γ_l, as l vanishes [154]. In this chapter, we deal with the simplest form of perturbation – a uniform perturbation l_ϵ for $\epsilon > 0$, defined as $l_\epsilon(I, a) = \epsilon$ for all a and I. We will denote the game Γ_{l_ϵ} as Γ_ϵ.

We let Q^ϵ refer to a ξ-perturbed variant of a treeplex Q, for the perturbed game Γ_ξ. Q^ϵ is the intersection of Q with the set of constraints $q^j \geq \epsilon q_{p_j}$ for all $j \in S_Q$. By constructing perturbed polytopes $\mathcal{X}^\epsilon, \mathcal{Y}^\epsilon$ and using these rather than $\mathcal{X}, \mathcal{Y}$ in (3.1), we get an approximate variant of EFPEs.

5.2 Distance-generating functions for the ξ-perturbed game

Let d_s be a DGF for the n-dimensional simplex Δ_n. As in the previous chapter we construct a DGF for Q by *dilating* d_s for each simplex in S_Q and taking their sum: $d(q) = \sum_{j \in S_Q} \beta_j q_{p_j} d_s(\frac{q^j}{q_{p_j}})$. We show that d_s and d can be used to implement a smoothing function for Q^ϵ and reason about its properties. To construct a smoothing function for Q^ϵ, we first construct a smoothing function for an *ϵ-perturbed simplex* $\Delta_n^\epsilon = \{q^s : \|q^s\|_1 = 1, q^s \geq \epsilon\}$, with $\epsilon > 0$. We construct a smoothing function for Δ_n^ϵ by composing d_s with a simple affine mapping $\phi(\tilde{q}^s) = \frac{\tilde{q}^s - \epsilon}{1 - n\epsilon}$, which sets up a one-to-one mapping between Δ_n and Δ_n^ϵ. The inverse of this function is $\phi^{-1}(q^s) = (1 - n\epsilon)q^s + \epsilon$. We let $d_s^\epsilon = d_s(\phi(\tilde{q}^s))$. We will show that d_s^ϵ retains all nice DGF properties of d_s.

Since d_s is continuously differentiable, we can apply the chain rule to get

$$\nabla d_s^\epsilon(q^s) = (1 - n\epsilon)^{-1} \nabla d_s(\tilde{q}^s). \tag{5.1}$$

For our new DGF to be practical we need the conjugate and its gradient to be easily computable. We show that this reduces to a simple transformation of the conjugate of d_s:

Lemma 9. *For a simplex DGF d_s and its ϵ-perturbed variant d_s^ϵ, the convex conjugate and its gradient for d_s^ϵ can be computed as*

$$d_s^{\epsilon,*}(g) = d_s^*((1 - n\epsilon)g) + \langle g, \epsilon \rangle$$

$$\nabla d_s^{\epsilon,*}(g) = (1 - n\epsilon)\nabla d_s^*((1 - n\epsilon)g) + \epsilon$$

Proof. Follows by the definition of conjugate and the chain rule for gradients. □

Thus computing our conjugate reduces to computing the conjugate for d_s coupled with simple linear transformations. Hoda et. al. [79] showed that the conjugate for a treeplex based on a sum over dilated simplex DGFs is easy to compute. Combined with Lemma 9, their result shows that

the conjugate of a treeplex DGF consisting of a sum over dilated perturbed simplex DGFs is easy to compute, as long as the same holds for the individual conjugates.

We now focus on the case where d_s is the entropy DGF for a simplex, that is, $d_s(q^s) = \sum_i q_i^s \log(q_i^s)$. Formally, we get the following DGF for a perturbed treeplex:

$$d_Q^\epsilon(q) = \sum_{j \in S_Q} \beta_j q_{p_j} \sum_{i \in \mathbb{I}_j} \frac{q_i/q_{p_j} - \epsilon}{1 - n_j\epsilon_j} \log\left(\frac{q_i/q_{p_j} - \epsilon}{1 - n_j\epsilon_j}\right)$$

In the previous chapter we showed strong convexity and convergence results for the class of dilated entropy functions for treeplexes. We now show how that result can be leveraged to prove strong convexity bounds for the perturbed entropy DGF.

Theorem 16. *The dilated perturbed entropy DGF on a treeplex with weights that satisfy the following recurrence*

$$\begin{aligned}
&\alpha_j = 1 + \max_{i \in \mathbb{I}_j} \sum_{k \in \mathcal{D}_j^i} \frac{\alpha_k \beta_k}{\beta_k - \alpha_k}, && \forall j \in S_Q, \\
&\beta_j > \alpha_j, && \forall i \in \mathbb{I}_j \text{ and } \forall j \in S_Q \text{ s.t. } b_Q^j > 0, \\
&\beta_j = \alpha_j, && \forall i \in \mathbb{I}_j \text{ and } \forall j \in S_Q \text{ s.t. } b_Q^j = 0.
\end{aligned}$$

is strongly convex modulus 1 *with respect to the* ℓ_2 *norm and modulus* $\frac{1}{M_Q}$ *with respect to the* ℓ_1 *norm.*

Proof. We will show that the quadratic over the Hessian of d_Q^ϵ can be expressed as a constant times the quadratic over the unperturbed dilated entropy DGF for Q. This will allow us to invoke the strong convexity results in Theorems 2 and 3.

Consider $q \in \operatorname{ri}(Q^\epsilon)$ and any $h \in \mathbb{R}^n$. For each $j \in S_Q$ and $i \in \mathbb{I}_j$, the second-order partial derivates of $d_Q^\epsilon(\cdot)$ with respect to q_i are:

$$\nabla^2_{q_i^2} d_s^\epsilon(q) = \frac{\beta_j}{(1 - n_j\epsilon_j)(q_i - \epsilon q_{p_j})} + \sum_{k \in \mathcal{D}_j^i} \sum_{l \in \mathbb{I}_k} \frac{\beta_k q_l^2}{(1 - n_k\epsilon_k)(q_l - \epsilon q_i) q_i^2} \tag{5.2}$$

Also, for each $j \in S_Q, i \in \mathbb{I}_j$, the second-order partial derivates with respect to q_i, q_{p_j} are given by:

$$\nabla^2_{q_i, q_{p_j}} d_s^\epsilon(q) = \nabla^2_{q_{p_j}, q_i} d_s^\epsilon(q) = -\frac{\beta_j q_i}{(1 - n_j\epsilon)(q_i - \epsilon q_{p_j}) q_{p_j}}. \tag{5.3}$$

Then equations (5.2) and (5.3) together imply

$$\begin{aligned}
h^\top \nabla^2 \omega(q) h = \sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} \Bigg[h_i^2 \Bigg(&\frac{\beta_j}{(1 - n_j\epsilon_j)(q_i - \epsilon q_{p_j})} \\
&+ \sum_{k \in \mathcal{D}_j^i} \sum_{l \in \mathbb{I}_k} \frac{\beta_k q_l^2}{(1 - n_k\epsilon_k)(q_l - \epsilon q_i) q_i^2} \Bigg) \\
&- h_i h_{p_j} \frac{2\beta_j q_i}{(1 - n_j\epsilon)(q_i - \epsilon q_{p_j}) q_{p_j}} \Bigg].
\end{aligned} \tag{5.4}$$

Given $j \in S_Q$ and $i \in \mathbb{I}_j$, we have $p_k = i$ for each $k \in \mathcal{D}_j^i$ and for any $k \in \mathcal{D}_j^i$, there exists some other $j' \in S_Q$ corresponding to k in the outermost summation. Then we can rearrange the following terms:

$$\sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} h_i^2 \sum_{k \in \mathcal{D}_j^i} \sum_{l \in \mathbb{I}_k} \frac{\beta_k q_l^2}{(1 - n_k \epsilon_k)(q_l - \epsilon q_i) q_i^2}$$
$$= \sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} \beta_j \frac{h_{p_j}^2 q_i^2}{(1 - n_j \epsilon_j)(q_i - \epsilon q_{p_j}) q_{p_j}^2}.$$

Using this equality in (5.4) leads to

$$(5.4) = \sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} \left[\frac{\beta_j h_i^2}{(1 - n_j \epsilon_j)(q_i - \epsilon q_{p_j})} + \frac{\beta_j h_{p_j}^2 q_i^2}{(1 - n_j \epsilon_j)(q_i - \epsilon q_{p_j}) q_{p_j}^2} - \frac{2 \beta_j h_i h_{p_j} q_i}{(1 - n_j \epsilon)(q_i - \epsilon q_{p_j}) q_{p_j}} \right]$$
$$= \sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} \frac{\beta_j q_i \left(\frac{h_i^2}{q_i} + \frac{h_{p_j}^2 q_i}{q_{p_j}^2} - \frac{2 h_i h_{p_j}}{q_{p_j}} \right)}{(1 - n_j \epsilon_j)(q_i - \epsilon q_{p_j})} \tag{5.5}$$

Now we can view the three terms inside the brackets as a convex function of h_i. First-order optimality implies that this function is nonnegative. Furthermore, since $q_i \geq \epsilon q_{p_j}$ we have $\frac{q_i}{q_i - \epsilon q_{p_j}} \geq 1$. Combined, this gives

$$(5.5) \geq \sum_{j \in S_Q} \sum_{i \in \mathbb{I}_j} \frac{\beta_j}{(1 - n_j \epsilon_j)} \left(\frac{h_i^2}{q_i} + \frac{h_{p_j}^2 q_i}{q_{p_j}^2} - \frac{2 h_i h_{p_j}}{q_{p_j}} \right)$$
$$\geq \sum_{j \in S_Q} \beta_j \left[\sum_{i \in \mathbb{I}_j} \left(\frac{h_i^2}{q_i} - \frac{2 h_i h_{p_j}}{q_{p_j}} \right) + \frac{h_{p_j}^2}{q_{p_j}} \right] \tag{5.6}$$

The last step follows because $\frac{q_i}{q_{p_j}}$ form simplex weights. By Lemma 1 in Kroer et. al. [103] this is exactly the expression for the quadratic of the Hessian of the unperturbed dilated entropy function on Q with weights β_j. Since our weights satisfy the requirements in Theorems 1 and 2 of Kroer et. al., the unperturbed dilated entropy function with these weights is strongly convex on Q, and thus we get (5.6) $\geq c\|h\|^2$ where $c = 1$ when $\|\cdot\|$ is the l_2 norm (by Theorem 1 of Kroer et. al.) and $c = \frac{1}{M_Q}$ when $\|\cdot\|$ is the l_1 norm (by Theorem 2 of Kroer et. al.). By Fact 3 this proves our theorem. □

Using Theorem 16 we can use the perturbed dilated entropy function to instantiate EGT. Since the value of the perturbed entropy on Δ_n^ϵ can be lower-bounded by $\log(n)$ exactly the same way as with the unperturbed entropy, we can apply Theorem 3 of Kroer et. al. [103], to bound EGT convergence rate as follows:

Theorem 17. *For a perturbed treeplex Q^ϵ, the dilated perturbed entropy function with simplex weights $\beta_j = M_Q(2 + \sum_{r=1}^{d_j} 2^r (M_{Q_j,r} - 1))$ 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.*

Theorem 17 immediately leads to the following convergence rate result for EGT equipped with dilated perturbed entropy DGFs to solve perturbed EFGs.

Theorem 18. *The EGT algorithm equipped with the dilated perturbed entropy DGF with weights* $\beta_j = 2 + \sum_{r=1}^{d_j} 2^r (M_{\mathcal{X}_j,r} - 1)$ *for all* $j \in S_{\mathcal{X}}$ *and the corresponding setup for* $\mathcal{Y}$ *will return a* ϵ*-accurate solution to the perturbed variant of* (3.1) *in at most the following number of iterations:*

$$\left(\max_{i,j} |A_{i,j}| \sqrt{M_{\mathcal{X}}^2 2^{d_{\mathcal{X}}+2} M_{\mathcal{Y}}^2 2^{d_{\mathcal{Y}}+2}} \log m \right) / \epsilon,$$

where the matrix norm is given by:

$$\|A\| = \max_{y \in \mathcal{Y}} \{ \|Ay\|_1^* : \|y\|_1 = 1 \} = \max_{i,j} |A_{i,j}|.$$

To our knowledge, this is the first result for FOMs that compute an approximate Nash equilibrium refinement.

5.3 Experiments

We conducted experiments to investigate the practical performance of our smoothing approach when used to instantiate the EGT algorithm. We compare EGT with our smoothing approach to EGT on an unperturbed polytope using the smoothing technique by Kroer et. al. [103] and CFR+ [161]. We conducted the experiments on Leduc hold'em poker [157], a widely-used benchmark in the imperfect-information game-solving community, except we tested on a larger variant of the game in order to better test scalability. In our enlarged version, *Leduc 5*, the deck consists of 5 pairs of cards $1 \ldots 5$, for a total deck size of 10. 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 that both players have the same private card, they draw and split the pot. Kroer et. al. [103] point out that the theoretically sound scale at which the overall weight on the DGF should be set is too conservative. We tune an overall weight on each DGF by choosing the weight that performs best with EGT and $\epsilon = 0$ among $1, 0.1, 0.05, 0.01, 0.005$ on the first 20 iterations. We test our approach on ϵ-perturbed polytopes of the strategy spaces for $\epsilon \in \{0.1, 0.05, 0.01, 0.005, 0.001\}$.

The first experiment measures convergence to Nash equilibrium (Figure 5.2). The x-axis shows the number of tree traversals performed per algorithm[1]. The y-axis shows the sum of player regrets in the full (unperturbed) game. We find that the ϵ perturbations have almost no effect on overall convergence rate until convergence within the perturbed polytope, at which point the regret in the unperturbed game stops decreasing, as expected. This shows that our approach can be utilized in practice: there is no substantial loss of convergence rate. Later in the run once the perturbed algorithms have bottomed out, there is a tradeoff between exploitability in the full game and refinement (i.e., better performance in low-probability information sets).

[1]Game tree traversals are equally expensive for all the algorithms studied. Treeplex traversal for each player is slower in EGT than CFR due to requiring exponentiation $\exp(\cdot)$, but the algorithms spend significantly less time on treeplex traversals than tree traversals, so this difference between the algorithms is insignificant.

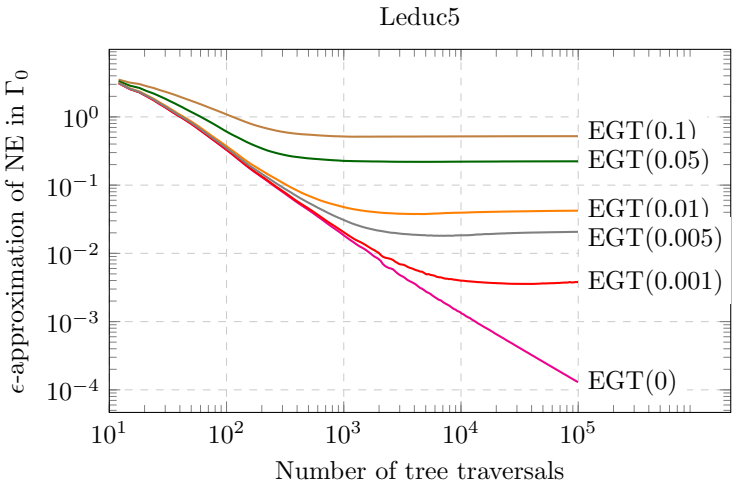


Figure 5.2: Regret as a function of the number of iterations for EGT with various ϵ perturbations (denoted in parentheses) and CFR+. Both axes are on a log scale.

The second experiment shows a measure of refinement convergence (Figure 5.3). The x-axis shows the number of tree traversals performed. The y-axis shows the maximum regret at any individual information set. Information set regret is calculated assuming that the information set is reached with probability one and applying Bayes' rule to get a distribution over nodes at the information set; the regret is the increase in expected utility from best-responding throughout all information sets in the subtrees rooted at the information set. Both CFR+ and unperturbed EGT perform badly with respect to this measure of refinement. Both have maximum regret two orders of magnitude worse than the perturbed approach. The maximum regret one can possibly cause in an information set in Leduc 5 is 22, so CFR+ and unperturbed EGT also do poorly in that sense. In contrast to this, we find that our ϵ-perturbed solution concepts converge to a strategy with low regret at every information set. The choice of ϵ is important: for $\epsilon = 0.001$, the smallest perturbation, we see that it takes a long time to converge at low-probability information sets, whereas we converge reasonably quickly for $\epsilon = 0.01$ or $\epsilon = 0.005$; for $\epsilon = 0.1$ and $\epsilon = 0.05$ the perturbations are too large, and we end up converging with relatively high regret (due to being forced to play every action with probability ϵ). Thus, within this set of experiments, $\epsilon \in [0.005, 0.01]$ seems to be the ideal amount of perturbation.

5.4 Conclusions and future research

We studied the extension of FOMs to the computation of Nash-equilibrium refinements. We developed a smoothing scheme based on perturbations of smoothing schemes for standard EFG solving, and proved that the convergence rate is comparable to that of solving the original game for Nash equilibrium. We performed numerical simulations where we showed that our approach has an overall convergence rate that is comparable to that of state-of-the-art Nash equilibrium methods. At the same time, we showed that our approach leads to solutions that have substantially better performance in subsets of the game tree that are reached with low probability. This

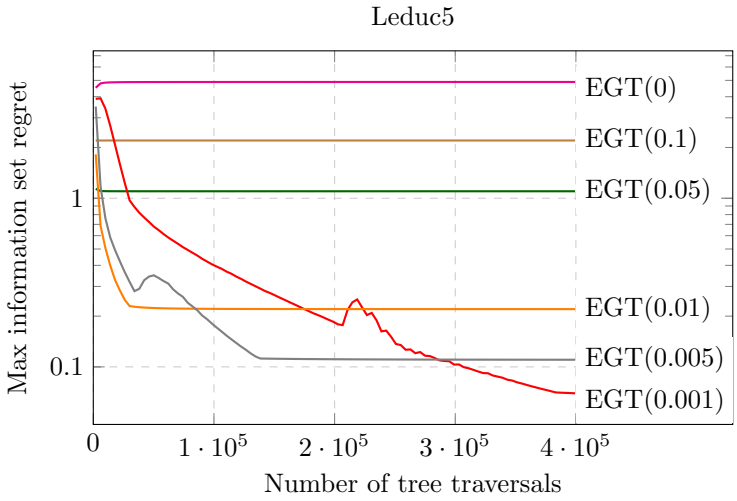


Figure 5.3: Maximum regret at any individual information set, as a function of the number of iterations.

has benefits both in approximate Nash equilibrium finding (such approximation is necessary in practice in large games) where some probabilities are low while possibly heading toward zero in the limit, and exact Nash equilibrium computation where the low probabilities are actually zero.

Our work suggests several research directions. It would be interesting to find a way to systematically decrease the ϵ-perturbations over time, so that we eventually converge to an exact Nash equilibrium in the full game. This requires at least two extensions. First, FOMs usually assume static domains, whereas this would involve a slowly expanding domain. Second, the ϵ would need to be decreased at a rate that is simultaneously fast enough that it converges at a reasonable rate, and slow enough that we actually converge to a refinement.

We showed how to compute approximate EFPE refinements using methods that scale to large games. It would be interesting to find a way to instantiate scalable methods such as FOMs or CFR+ for other equilibrium refinement concepts as well. The perturbed polytope due to Miltersen and Sørensen could be used to construct a notion of approximate QPE that would lead to an optimization setup similar to ours. However, this will require constructing a DGF for the perturbed-QPE polytope, which has ϵ-perturbations on the realization plans. Our approach relied on ϵ-perturbations to the behavioral strategies, and so it is likely that a different DGF class is needed to handle approximate QPE.

Chapter 6

Limited lookahead in sequential games

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 [20, 94, 128, 129, 140, 141, 143, 144]. In this chapter, 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. [122] 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 chapter). 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) [150] and security games where opponents are often assumed to be myopic (as makes sense when the number of adversaries is large [176]). 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 [45] and extensive-form games [110], 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.

6.1 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.

6.2 Complexity

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

6.2.1 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 [137]. This is because finding a Nash equilibrium in a 2-player general-sum normal-form game is PPAD-hard [42, 48], 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 [82].

6.2.2 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 [165], and algorithms have been developed for scaling to large games [64, 68, 79, 95, 98, 108, 179].

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 19. *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.*

Proof. We reduce from 3SAT. A picture illustrating our reduction is given in Figure 6.1, and a description is given below.

Let the root node be a chance node. It chooses with equal probability between $|C|$ child nodes, each representing a clause. Each such descendant clause node is a singleton information set belonging to Player L. Each clause node has three actions, representing the three literals in the clause. Each such action leads to a node representing that literal. Player L gets the same value from each action and is therefore indifferent. Player R acts at each literal node, with all literal nodes representing the same variable being in an information set together. Thus, Player R has an information set for each variable. At each variable information set, there is a true and false action. For a given literal node in some variable information set, the true action leads a payoff of 1 if the literal requires the variable to be true, and 0 otherwise. Similarly, the false action leads to a payoff of 1 if the literal requires the variable to be false, and 0 otherwise.

The decision problem is then: does there exist a strategy for Player R with expected payoff 1? This is the case if and only if the strategy for Player R represents a satisfying assignment to V, C, as each clause must have some action available where a satisfying assignment for the literal is chosen with probability 1. □

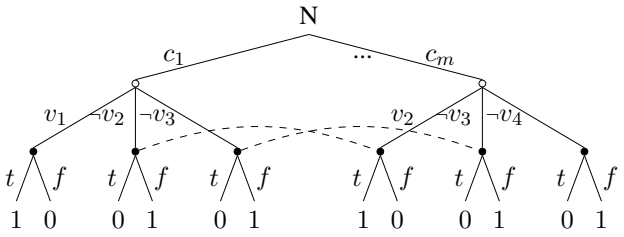


Figure 6.1: The game tree in our proof of Theorem 19. Dashed lines denote information sets.

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 [110]. 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 20. *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*.*

Proof. We reduce from 3SAT. We use the same reduction as for Theorem 19, except that at each clause node, we also add an "unsatisfied" action that leads directly to a leaf node with payoff 0 for Player R and payoff $\frac{2}{3}$ for Player L.

For all leaf nodes under a variable node, we set the payoff to 1 for Player R, and 0 or 1 for Player L, for when the leaf represents the ancestor literal being unsatisfied or satisfied, respectively. The modifications are shown for a single clause in Figure 6.2.

The question is whether Player R can compute a strategy such that Player L selects a literal action for each clause, assuming that Player L breaks ties such that the unsatisfied action is least

preferred. For a given variable, choosing a strategy strictly between $0, \frac{2}{3}$ for the two actions leads to zero utility gain, since Player 2 will then always prefer the unsatisfied actions over any literal belonging to the variable. Thus we can assume that Player R plays a pure strategy, since at most one action can have its probability set high enough to yield utility gain. Now, for each clause, Player L will only choose a literal action if that variable is set to the correct value to satisfy the clause. Thus, if Player R can compute a strategy that gives expected utility 1, each clause node must have at least one variable with a satisfying assignment. □

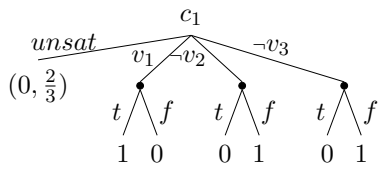


Figure 6.2: The clause modification in our proof of Theorem 20.

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 21. *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*.*

Proof. We reduce from 3SAT. Let the root node be a chance node. It chooses with equal probability between all variable and clause pairs v, c such that $v \in c$. Player R acts at each child node, being able to distinguish only which variable was chosen. For each information set, Player 1 can choose between a true and a false action, representing setting the associated variable to true or false, respectively. At the next level where Player L is active. The information sets at the level are constructed as follows. For each $c \in C$ an information set is constructed, containing all nodes representing Player R choosing both true and false for each $v \in c$. For each information set representing some clause c, Player L has 4 actions. First is an unsat action, leading to payoff 0 for Player 1 and payoff 1 for Player L, no matter which node in the information set play has reached. Second, an action for each variable $v \in c$ leading to payoff 1 for Player R, and payoff 3 to Player L if play reached a node representing v with true or false chosen such that it satisfies c, and payoff 0 for all other nodes in the information set.

We claim that there is a satisfying assignment if and only if Player 1 can commit to a strategy with expected payoff 1. Let $\phi : V \rightarrow \{true, false\}$ be a satisfying assignment to V, C. Let Player R deterministically pick actions at each variable information set according to ϕ. If play reaches a singleton node, Player L has only one action available, guarateeing payoff 1. If play reaches some information set representing a clause c, Player L has expected payoff of $3 \cdot \frac{1}{3}$ when picking any action representing a satisfied literal $l \in c$, as the conditional probability of being at a node representing $v(l)$ is $\frac{1}{3}$, and Player R chooses the satisfying action with probability 1. Since Player L breaks ties such that unsatisfied actions are least preferred, she will pick an action representing a variable for each information set, yielding payoff 1 to Player R. This covers all possible outcomes, giving an expected payoff of 1 to Player R.

Given some strategy for Player R that gives payoff 1 in expectation, we show how to construct a satisfying assignment to V, C. For a strategy to have payoff 1, Player L must be choosing variable actions at each information set for some clause c. This is the case if and only if Player R selects the satisfying truth value with probability 1 for some $v \in c$, since the expected payoff of taking a variable action is otherwise strictly smaller than the unsatisfied action. This leads directly to a satisfying assignment, by choosing the corresponding value assignment for each action that is selected with probability 1, and choosing an arbitrary assignment for every other variable. □

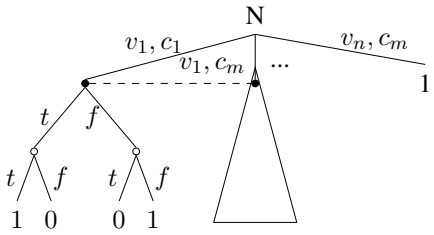


Figure 6.3: The game tree for our proof of Theorem 21.

6.3 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 [150] (because evolution is responding to what we are doing) and security games [176] (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):

$$\sum_{s \in S_{I,a}^k} \pi(s) \cdot h(s) > \sum_{s \in S_{I,a'}^k} \pi(s) \cdot h(s) \tag{6.1}$$

$$\sum_{s \in S_{I,a}^k} \pi(s) \cdot h(s) = \sum_{s \in S_{I,a^*}^k} \pi(s) \cdot h(s) \tag{6.2}$$

Player R chooses a worst-case utility-maximizing strategy that satisfies (6.1) and (6.2), 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 22. *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 [165], 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 (6.1) and (6.2) 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 (6.3) and dual (6.4) LPs:

$$\begin{aligned} \max_{y'} \quad & (x^T B')y' \\ & F'y' = f' \\ & y \geq 0 \end{aligned} \tag{6.3}$$

$$\begin{aligned} \min_{q'} \quad & q'^T f' \\ & q'^T F' \geq x^T B' \end{aligned} \tag{6.4}$$

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:

$$\begin{aligned} \max_{x} \quad & x^T(Ay') \\ & x^T E^T = e^T \\ & x \geq 0 \\ & x^T H_{\neg\mathcal{A}} - x^T H_{\mathcal{A}} \leq -\epsilon \\ & x^T G_{\mathcal{A}^*} = x^T G_{\mathcal{A}} \end{aligned} \tag{6.5}$$

where the last two constraints encode (6.1) and (6.2), respectively. The dual problem uses the unconstrained vectors p, v and constrained vector u and looks as follows

$$\begin{aligned} \min_{p,u,v} \quad & e^T p - \epsilon \cdot u \\ E^T p + (H_{\neg\mathcal{A}} - H_{\mathcal{A}})u + (G_{\mathcal{A}^*} - G_{\mathcal{A}})v & \geq A'y' \\ u & \geq 0 \end{aligned} \tag{6.6}$$

We can now merge the dual (6.4) with the constraints from the primal (6.5) to compute a minimax strategy: Player R chooses x, which she will choose to minimize the objective of (6.4),

$$\begin{aligned} \min_{x,q'} \quad & q'^T f' \\ q'^T F' - x^T B' &\geq 0 \\ -x^T E^T &= -e^T \\ x &\geq 0 \\ x^T H_{\mathcal{A}} - x^T H_{\neg\mathcal{A}} &\geq \epsilon \\ x^T G_{\mathcal{A}} - x^T G_{\mathcal{A}^*} &= 0 \end{aligned} \tag{6.7}$$

Taking the dual of this gives

$$\begin{aligned} \max_{y',p} \quad & -e^T p + \epsilon \cdot u \\ -E^T p + (H_{\mathcal{A}} - H_{\neg\mathcal{A}})u + (G_{\mathcal{A}} - G_{\mathcal{A}^*})v &\leq B'y' \\ F'y' &= f' \\ y, u &\geq 0 \end{aligned} \tag{6.8}$$

We are now ready to prove Theorem 22.

Proof. The LPs in Theorem 22 are (6.7) and (6.8). We will use duality to show that they provide optimal solutions to each of the best response LPs. Since $A = -B$, the first constraint in (6.8) can be multiplied by -1 to obtain the first constraint in (6.6) and the objective function can be transformed to that of (6.6) by making it a minimization. By the weak duality theorem, we get the following inequalities

$$\begin{aligned} q'^T f' &\geq x^T B' y'; \text{ by LPs (6.3) and (6.4)} \\ e^T p - \epsilon \cdot u &\geq x^T A' y'; \text{ by LPs (6.5) and (6.6)} \end{aligned}$$

We can multiply the last inequality by -1 to get:

$$q'^T f' \geq x^T B' y' = -x^T A' y' \geq -e^T p + \epsilon \cdot u \tag{6.9}$$

By the strong duality theorem, for optimal solutions to LPs (6.7) and (6.8) we have equality in the objective functions $q'^T f' = -e^T p + \epsilon u$ which yields equality in (6.9), and thereby equality for the objective functions in LPs (6.3), (6.4) and for (6.5), (6.6). By strong duality, this implies that any primal solution x, q' and dual solution y', p to LPs (6.7) and (6.8) yields optimal solutions to the LPs (6.3) and (6.5). Both players are thus best responding to the strategy of the other agent, yielding a Nash equilibrium. Conversely, any Nash equilibrium gives optimal solutions x, y' for LPs (6.3) and (6.5). With corresponding dual solutions p, q', equality is achieved in (6.9), meaning that LPs (6.7) and (6.8) are solved optimally.

It remains to show the size bound for LP (6.7). Using sparse representation, the number of non-zero entries in the matrices A, B, E, F is linear in the size of the game tree. The constraint set $x^T H_{\mathcal{A}} - x^T H_{\neg\mathcal{A}} \geq \epsilon$, when naively implemented, is not. The value of a sequence $a \notin$

A_I^* is dependent on the choice among the cartesian product of choices at each information set I' encountered in hypothetical play below it. In practice we can avoid this by having a real-valued variable $v_I^d(I')$ representing the value of I' in lookahead from I, and constraints $v_I^d(I') \geq v_I^d(I', a)$ for each $a \in I'$, where $v_I^d(I', a)$ is a variable representing the value of taking a at I'. If there are more information sets below I' where Player L plays, before the lookahead depth is reached, we recursively constrain $v_I^d(I', a)$ to be:

$$v_I^d(I', a) \geq \sum_{\check{I} \in \mathcal{D}} v_I^d(\check{I}) \tag{6.10}$$

where $\mathcal{D}$ is the set of information sets at the next level where Player L plays. If there are no more information sets where Player L acts, then we constrain $v_I^d(I', a)$:

$$v_I^d(I', a) \geq \sum_{s \in S^k_{I',a}} \pi^{\sigma}_{-\langle} h(s) \tag{6.11}$$

Setting it to the probability-weighted heuristic value of the nodes reached below it. Using this, we can now write the constraint that a dominates all $a' \in I, a' \notin \mathcal{A}$ as:

$$\sum_{s \in S^k_{I,a}} \pi^{\sigma}(s) h(s) \geq v_I^d(I)$$

There can at most be $O(\sum_{I \in \mathcal{I}_L} |A_I|)$ actions to be made dominant. For each action at some information set I, there can be at most $O(\max_{s \in S} |A_s|^{\min\{k,k'\}})$ entries over all the constraints, where k' is the maximum depth of the subtrees rooted at I, since each node at the depth the player looks ahead to has its heuristic value added to at most one expression. For the constraint set $x^T G_{\mathcal{A}} - x^T G_{\mathcal{A}^*} = 0$, the choice of hypothetical plays has already been made for both expressions, and so we have the constraint

$$\sum_{s \in S^k_{I,a}} \pi^{\sigma}(s) h(s) = \sum_{s \in S^k_{I,a'}} \pi^{\sigma'}(s) h(s)$$

for all $I \in \mathcal{I}_L, a, a' \in I, \{a, \sigma^L\}, \{a', \sigma^{\langle,\prime}\} \in \mathcal{A}$, where

$$\sigma = \{\sigma_{-L}, \sigma^L\}, \sigma' = \{\sigma_{-L}, \sigma^{\langle,\prime}\}$$

There can at most be $\sum_{I \in \mathcal{I}_L} |A_I|^2$ such constraints, which is dominated by the size of the previous constraint set.

Summing up gives the desired bound. □

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 (6.7) 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 (6.12). 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 (6.7), i.e., what goes into the structure $\mathcal{A}$. The size of the MIP is of the same order as that of LP (6.7).

$$\begin{aligned}
\min_{x,q,z} \quad & q^T f \\
& q^T F \geq x^T B - zM \\
& Ex = e \\
& x^T H_{\mathcal{A}} \geq x^T H_{\neg\mathcal{A}} + \epsilon - (1-z)M \\
& x^T G_{\mathcal{A}} = x^T G_{\mathcal{A}^*} \pm (1-z)M \\
& \sum_{a \in A_I} z_a \geq z_{a'} \\
& x \geq 0, \quad z \in \{0,1\}
\end{aligned} \tag{6.12}$$

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 $\mathcal{A}$:

$$\begin{aligned}
x^T H_{\mathcal{A}} - x^T H_{\neg\mathcal{A}} &\geq \epsilon - (1-z)M \\
x^T G_{\mathcal{A}} - x^T G_{\mathcal{A}^*} &= 0 \pm (1-z)M
\end{aligned} \tag{6.13}$$

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 (6.13) 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:

$$\sum_{s \in S_{I,a}^k} v^i(s) \geq v_I^d(I), \quad v^i(s) \leq \sigma_L^I(I', a') \cdot M \tag{6.14}$$

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.

6.4 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 [106], 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:

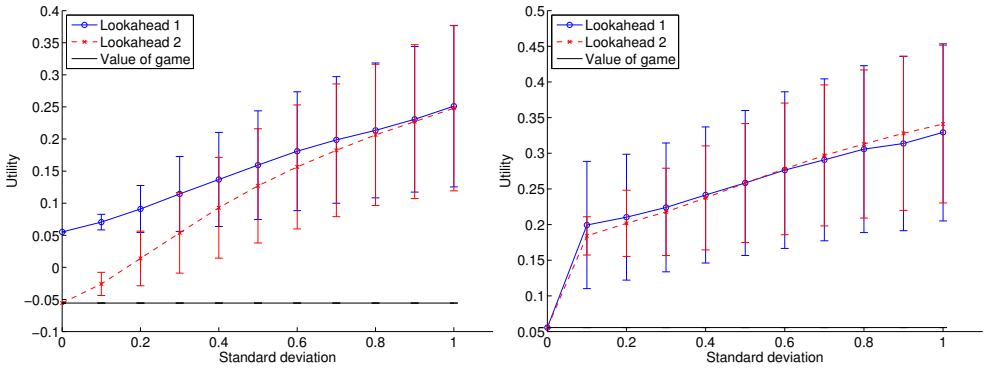
$$h(s) = \begin{cases} u_i(s) & \text{if } s \in Z \\ \sum_{a \in A_s} \sigma(s,a) h(t_a^s) & \text{otherwise} \end{cases} \tag{6.15}$$

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:

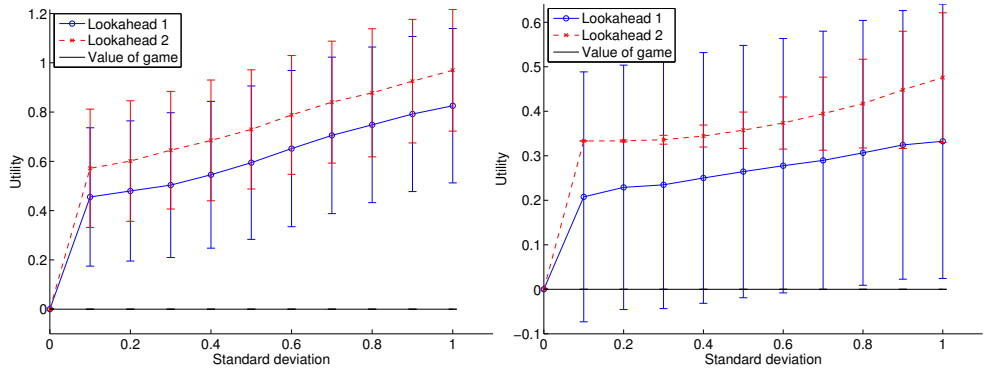
$$\bar{h}(s) = \begin{cases} u_i(s) & \text{if } s \in Z \\ \left[\sum_{a \in A_s} \sigma(s,a) \bar{h}(t_a^s)\right] + \mu_s & \text{otherwise} \end{cases} \tag{6.16}$$

Using MIP (6.12), 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 6.4.

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.



(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 6.4: 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.

Figures 6.4a 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 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 6.4c 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 [7, 128, 140], and understanding them remains an active area of research [117, 129, 174]. 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 6.6 g and h do not have this pathology, and $\bar{h}$ does have increasing visibility.

Figure 6.5 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 value of a node in equilibrium.

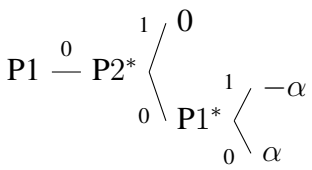


Figure 6.5: A subtree that exhibits lookahead pathology.

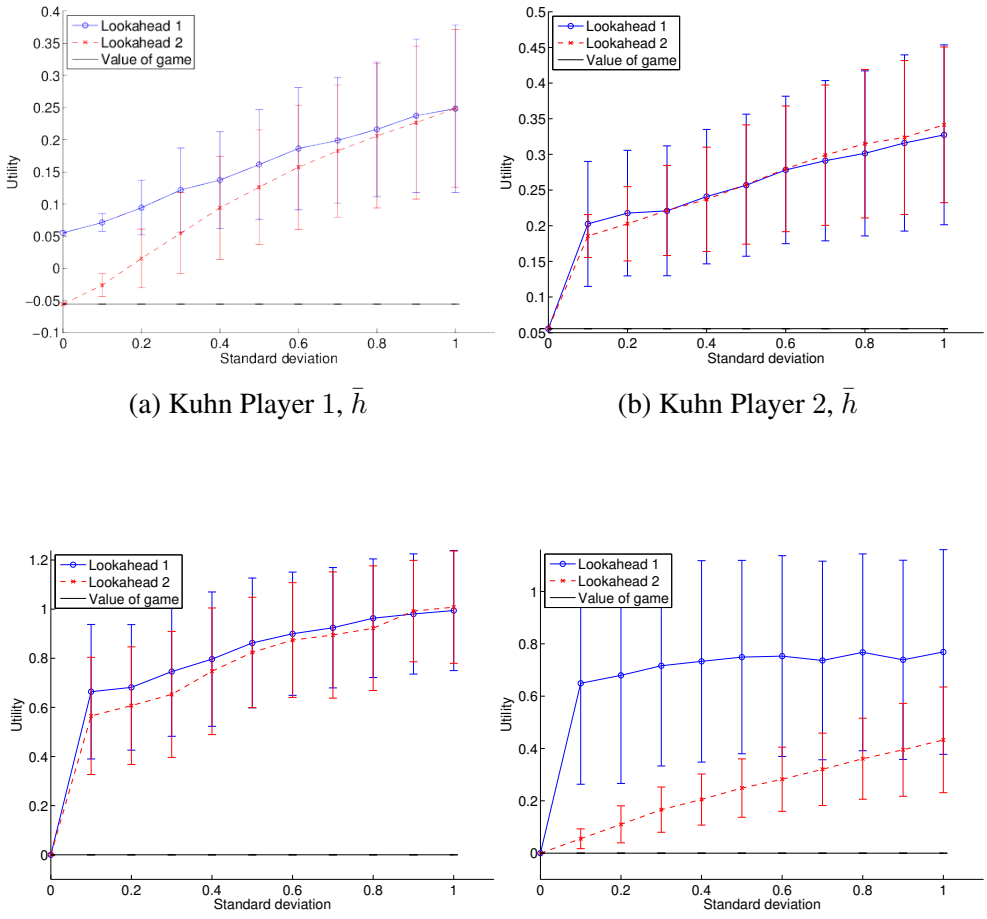
We now go over the example of Figure 6.5. Assume without loss of generality that all payoffs are positive in some game. We can then insert the subtree in Figure 6.5 as a subgame at any node belonging to P1, and it will be played with probability 0 in equilibrium, since it has expected value 0. Due to 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 6.6e 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 6.4g 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.

6.5 Conclusions and future research

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 6.7 summarizes those results.

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



(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 6.6: 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.

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-

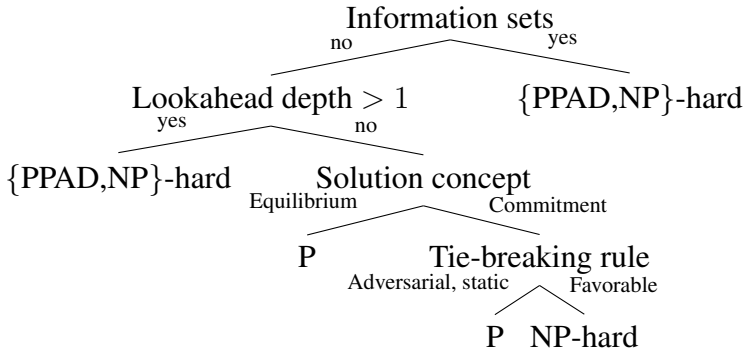


Figure 6.7: Our complexity results. {PPAD,NP}-hard indicates that finding a Nash equilibrium (optimal strategy to commit to) is PPAD-hard (NP-hard). P indicates polytime.

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 [148]. 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., [64]), while theoretically unbounded, could immediately be used to investigate exploitation potential in larger games.

In work that was subsequent to the work presented in this chapter, Brown et al. [30] develop an approach for depth-limited subgame solving. It would be interesting to relate our results on limited lookahead to that paper.

In the next section we present results on Robust Stackelberg equilibria both with and without limited lookahead, thus extending the results of this section to a robust setting.

Chapter 7

Robust Stackelberg equilibria

In a Stackelberg equilibrium, a *leader* commits to a strategy first, and then a *follower* chooses a strategy for herself. By committing to a strategy the leader may gain utility by guiding the outcome, in spite of the follower choosing a best response. This was originally studied in the context of using pure strategies to commit to quantity [47] or price [12] in deterring market entry [163]. Further power can be gained by committing to a mixed (i.e., randomized) strategy [45, 167]. Stackelberg equilibria have become important as a solution concept in computational game theory, largely inspired by practical problems such as security settings, where the leader is a defender who picks a mixed (i.e., potentially randomized) strategy first, and then the follower who is the attacker decides where to attack, if at all.

Most work on Stackelberg equilibria has focused on *normal-form* (aka. matrix-form) games. Conitzer and Sandholm [46] studied the problem of computing an optimal strategy to commit to in normal-form games. That line of work has been extended to many security-game applications. In practice, there is typically uncertainty about the opponent's payoffs. In normal-form games this has been studied as *Bayesian Stackelberg games* where the players have private information about their own payoffs, and there is common knowledge of the prior distribution over the payoffs [45, 138, 160]. As an alternative, the *robust* (distribution-free) approach has been suggested for security games: bounds are assumed on the follower's payoffs [91, 136].

Extensive-form games (EFGs)—i.e., tree-form games—are a very general game representation language. EFGs are exponentially more compact and also more expressive than normal-form games. Letchford and Conitzer [110] study how to compute an optimal strategy to commit to in EFGs and prove hardness results under several assumptions about the game structure. Generally, they show that finding an SSE in an EFG is NP-hard. Furthermore, because Bayesian games are a special case of EFGs with Nature moves the results of De Nittis et al. [50] imply Poly-APX hardness for EFGs with Nature moves as well. In robust settings, such as ours, computing an optimal strategy to commit to is already NP-hard for normal-form games with interval uncertainty and inapproximable with a discrete uncertainty set [111]. Bošanskỳ et al. [19] provide further results specifically for perfect-information EFGs. Bošanskỳ and Čermák [18] develop a *mixed-integer program (MIP)* for computing a Stackelberg strategy, and Čermák [33] develop an iterative approach based on upper-bounding solutions from extensive-form *correlated* Stackelberg equilibria.

To our knowledge, we are the first to consider uncertainty about the opponent in Stackelberg

strategies for EFGs. This is important because EFGs are a powerful representation language and because in practice there is typically uncertainty about the opponent. We take a robust approach to modeling this uncertainty. We introduce robust Stackelberg equilibria for EFGs, where the uncertainty is about the opponent's payoffs, as well as ones where the opponent has limited lookahead and the uncertainty is about the opponent's node evaluation function.

A robust Stackelberg EFG with uncertainty can be equivalently represented by a robust normal-form game. Letchford et al. [111] studied robust normal-form games from the perspective of optimizing against a worst-case type of the follower, as well as interval uncertainty on payoffs. They show hardness results and develop a MIP for computing optimal leader strategies. However, this equivalent representation requires an exponential blowup in problem size, and is thus not appropriate for most EFGs (their hardness results mentioned previously show hardness of our setting however). Instead we tackle the EFG directly.

We develop a new MIP for the deterministic limited-lookahead setting. We then extend the MIP to the robust setting for Stackelberg equilibrium under unlimited and under limited lookahead by the opponent. We show that for the specific case of interval uncertainty about the opponent's payoffs (or about the opponent's node evaluations in the case of limited lookahead), robust Stackelberg equilibria can be computed with a MIP that is of the same asymptotic size as that for the deterministic setting.

Our results for robust Stackelberg equilibria in EFGs are relevant to security-game settings with sequential interactions, where EFG models can more compactly represent certain games, as compared to a normal-form representation [18]. Robust models are important in security games, where opponent models often have uncertainty, both in standard security games [89, 91, 136], and *green security games* [135].

Our limited-lookahead results are useful for settings where it is not always desirable to model adversaries as fully rational, but as having limited lookahead capability. This includes settings such as biological games, where the goal is to steer an evolutionary process or an adaptation process which typically acts myopically without lookahead [99, 150] and security games where opponents are often assumed to be myopic (which can be especially well motivated when the stakes are low such as in fare evasion [176] or in the case of *opportunistic criminals* [147, 178]). Our model of limited lookahead is an extension of that of Kroer and Sandholm [97] to a robust setting. Kroer and Sandholm [97] gave a MIP for computing an optimal strategy to commit to in the deterministic setting. We show an alternative MIP for computing such a strategy to commit to, which we then extend to the robust setting.

Finally, the question of robust variants of optimization problems has been studied extensively in the optimization literature [9, 10, 14]. In that literature, the assumption is that we are given some *nominal* mathematical program, and then the robust variant requires that each constraint in the nominal program holds with respect to every instantiation of a set of uncertainty parameters. This makes the setting substantially different from our setting, where there is no nominal program: the best response of the follower does not need to be a best response for every uncertainty instantiation (this would be the equivalent to robust optimization, and often infeasible), but rather the best response is chosen after the uncertainty parameters are chosen.

7.1 Stackelberg setting

In this section we focus on *Stackelberg EFGs*. These are two-player games where there is a designated *leader* and a designated *follower* rather than the usual two players. We will focus on settings where the leader first commits to a strategy that the follower observes. The follower then plays a best response to the leader strategy. A *strong Stackelberg equilibrium* (SSE) is a pair of strategies r_l, r_f such that r_f is a best response to r_l and r_l is a solution to the optimization problem of maximizing $u(r_l, r_f)$ over r_l and r_f, subject to the constraint that r_f is a best response to r_l. This definition implies the common assumption that the follower breaks ties in favor of Player l [46, 138, 160]. A *weak Stackelberg equilibrium* assumes minimization over the set of optimal best responses.

In this section we will investigate settings where there is uncertainty about the follower's utility function u_f. Specifically, the follower's utility can be any function from some given *uncertainty set* U_f consisting of functions that map from the set of leaf nodes to $\mathbb{R}$. We leave the exact structure of U_f undefined for now; in our algorithmic section we show that the case where each leaf has independent interval uncertainty can be solved using a MIP.

It will be convenient to have function expressing expected values for a given pair of sequences. Given two sequences σ_l and σ_f, we let

$$g_l(\sigma_l, \sigma_f) = \sum_{h \in Z; \sigma_f(h) = \sigma_f; \sigma_l(h) = \sigma_l} \pi_0(h) u_l(h),$$

$$g_f^{u_f}(\sigma_l, \sigma_f) = \sum_{h \in Z; \sigma_f(h) = \sigma_f; \sigma_l(h) = \sigma_l} \pi_0(h) u_f(h)$$

be the expected utilities, for the leader and follower respectively, over leaf nodes that are reached with σ_f, and σ_l as the corresponding last player sequences. The function for the follower $g_f^{u_f}$ depends on the choice of utility function u_f, whereas we always know the utility function for the leader.[1] Given two realization plans r_l, r_f and a utility function u_i, we overload notation slightly and let the expected value for Player i induced by the realization plans be denoted by

$$u_i(r_l, r_f) = \sum_{\sigma_l \in \Sigma_l, \sigma_f \in \Sigma_f} r_l(\sigma_l) r_f(\sigma_f) g_i(\sigma_l, \sigma_f).$$

7.2 Limited-lookahead model

We will also consider a limited-lookahead variant of EFGs. There has been a significant amount of work on limited lookahead in perfect-information games (such as chess and checkers) in the AI community. Modeling limited lookahead in imperfect-information games (that have information sets) is more intricate. A model for that was presented recently [97], and we use that model. In that model, the follower can only look ahead k steps. He uses a *node-evaluation function* $\tilde{u}$:

[1] In Stackelberg equilibrium, the follower does not have to be concerned about the leader's utility function because the leader commits to his strategy and declares his strategy to the follower.

$H \to \mathbb{R}$ that associates a heuristic utility with any node in the game tree. At any information set $I \in \mathcal{I}_f$, the follower has a set of nodes $\tilde{H}_I \subset H$ called the *lookahead frontier*. When choosing his action at information set I, the follower chooses an action that maximizes the expected value of $\tilde{u}$, assuming that they choose actions so as to maximize $\tilde{u}$ at any follower information sets reached before $\tilde{H}_I$. We let $g_I(\sigma_l, \sigma_f)$ be the expected value over lookahead-frontier nodes according to the node-evaluation function (analogous to g_i for the setting without limited lookahead). We assume that for any information set $I' \in \mathcal{I}_f$ that comes after I, all the nodes of I' are entirely contained in the set of nodes that precede $\tilde{H}_I$, or entirely disjoint with the set of preceding nodes (this is in order to avoid any information sets belonging to the follower being only partially contained in the hypothetical decision making under I). We let the set of information sets that come after I such that their nodes are all preceding $\tilde{H}_I$ be denoted by $\mathcal{I}_I$. We $\Sigma_f^I \subseteq \Sigma_f$ denote the set of all sequences beneath a given information set I that are within the lookahead frontier.

In the prior chapter on limited lookahead in imperfect-information games [97] it was assumed that the leader knows the follower's node evaluation function exactly. That seems quite unrealistic. Therefore, we will extend the work to the case where the leader has uncertainty about the follower's node evaluation function.

7.3 Best responses and how to compute them

Our solution concept will depend on the notion of a *best response* for the follower. For a given leader strategy r_l and utility function $u_f \in U_f$, the set of best responses is

$$BR(r_l, u_f) = \{r_f : u_f(r_l, r_f) = \max_{r'_f} u_f(r_l, r'_f)\}.$$

Given a strategy r_l for the leader and a utility function u_f, the value of each information set can be computed with the following feasibility program (this holds outside of a leader-follower setting as well):

$$v_{\inf_f(\sigma_f)} = s_{\sigma_f} + \sum_{\substack{I' \in \mathcal{I}_f \\ \sigma_f(I') = \sigma_f}} v_{I'} + \sum_{\sigma_l \in \Sigma} r_l(\sigma_l) g_f^{u_f}(\sigma_l, \sigma_f) \qquad \forall I \in \mathcal{I}_f, \sigma_f = \sigma_f(I) \tag{7.1}$$

$$0 \leq s_{\sigma_f} \leq M(1 - b_f(\sigma_f)) \qquad \forall \sigma_f \in \Sigma_f \tag{7.2}$$

$$\textstyle\sum_{a \in A(I)} b_f(\sigma a) = 1 \qquad \forall I \in \mathcal{I}_f, \sigma_f = \sigma_f(I) \tag{7.3}$$

$$b_f(\sigma_f) \in \{0, 1\} \qquad \forall \sigma_f \in \Sigma_f \tag{7.4}$$

The variables v_I represent the value of a given information set I, $b_f(\sigma_f)$ represents whether σ_f is a best response at its respective information set, and s_{σ_f} represents how much less utility the follower gets by following the sequence σ_f rather than the optimal action at $\inf(\sigma_f)$. It is easy to show via induction that the feasibility MIP given in equations (7.1-7.4) computes a best response to r_l and the variables v_I represent the values of information sets I when best-responding to r_l: For the base case of an information set with no future information sets belonging to the follower, disregarding s_{σ_f}, the RHS of (7.1) clearly represents the value of choosing σ_f at the information set. Now, since all s_{σ_f} are nonnegative and (7.1) is an equality, it follows that v_I upper bounds

the value of each individual sequence at I. But since $s_{\sigma_f} = 0$ for some σ_f, it must be an equality for said σ_f. Thus v_I upper bounds the value of all sequences at I, but is also equal to the value of some sequence, and therefore it represents the value when best responding. Applying the inductive hypothesis to any information set I that has future information sets belonging to the follower reduces the expression for v_I to one that is equivalent to the base case.

7.4 Extension to uncertainty about the opponent

We now extend the EFG model to incorporate uncertainty about the follower's utility function. We will take a robustness approach, where we care about the worst-case instantiation of the uncertainty set U_f. For limited-lookahead EFGs we will analogously consider uncertainty over the node-evaluation function.

Due to the uncertainty (represented by the uncertainty set U_f), defining a Stackelberg equilibrium is not straightforward. We take the perspective that a robust Stackelberg solution is a strategy for the leader that maximizes the leader utility in the worst-case instantiation of U_f:

Definition 14. *A* robust strong Stackelberg solution *(RSSS) is a realization plan r_l such that*

$$r_l \in \arg\max_{r'_l \in R_l} \inf_{u_f \in U_f} \max_{r'_f \in BR(r_l, u_f)} u_l(r_l, r'_f).$$

The robustness is represented by the minimization over U_f. Intuitively, if the actual instantiation of u_f does not take on the minimizer over U_f, the leader can only receive better utility, so we are computing the maximin utility against the robustness. Typically one is interested in finding an RSSS strategy for the leader, but we nonetheless define the entire equilibrium concept as well:

Definition 15. *A* robust strong Stackelberg equilibrium *(RSSE) is a realization plan r_l and a (potentially uncountably large) set of realization plans $\{r_f^{u_f} : \forall u_f \in U_f\}$ such that r_l is an RSSS and $r_f^{u_f} \in BR(r_l, u_f)$ for all $u_f \in U_f$.*

Whether an RSSE is even practical to represent is highly dependent on the structure of the specific game and uncertainty sets at hand, as it would frequently need to be represented parametrically. On the other hand, once we have r_l, the best response for a specific u_f can easily be computed. One method for doing this is to first compute the follower value u^* under u_f when best responding to r_l (e.g., via a single tree traversal), and then solving the *linear program (LP)* that consists of maximizing the leader's utility over the set of follower strategies that achieve u^* (this can be done by adding a single constraint to the sequence-form best-response LP given by von Stengel [165]).

One might consider applying the robustness after the follower chooses her strategy (in a sense, swapping the inner max and min). In this case, we cannot represent this as a minimization on the inside since the set of best responses is defined with respect to the choice of u_f. Arguably the most natural way to apply robustness after the best response of the follower would be to ask for a pair of strategies r_l, r_f such that r_f is a best response no matter the instantiation of u_f. This definition of robustness would allow us to apply standard robust optimization techniques to any Stackelberg MIP. However, this definition has several drawbacks. First, if we are applying a robust model, we are often interested in maximizing our worst-case utility. By applying robustness

after choosing r_f, we would not be doing that, but instead would be maximizing utility subject to the constraint that we want to be sure what the follower response is. Second, a robust Stackelberg equilibrium defined that way would not necessarily exist: if there is overlap between the range of possible utilities associated with a pair of actions at some information set, there would be no way to guarantee that a single action will always be a best response.

7.5 MIP for full-certainty setting

We now give a MIP for computing a Stackelberg equilibrium in a game where the follower has limited lookahead.

$$\max_{p,r,v,s} \sum_{z \in Z} p(z) u_l(z) \pi_0(z) \tag{7.5}$$

$$v_I = s_{\sigma_f} + \sum_{I' \in \mathcal{I}_f : \sigma_f(I') = \sigma_f} v_{I,I'} + \sum_{\sigma_l \in \Sigma_l} r_l(\sigma_l) g_I(\sigma_l, \sigma_f) \quad \forall \sigma_f \in \Sigma_f, I = \inf_f(\sigma_f) \tag{7.6}$$

$$v_{I,\inf(\sigma_f)} = s^I_{\sigma_f} + \sum_{I' \in \mathcal{I}_f : \sigma_f(I') = \sigma_f} v_{I,I'} + \sum_{\sigma_l \in \Sigma_l} r_l(\sigma_l) g_I(\sigma_l, \sigma_f) \quad \forall I \in \mathcal{I}, \sigma_f \in \Sigma^I_f \tag{7.7}$$

$$r_i(\emptyset) = 1 \qquad \forall i \in \{l, f\} \tag{7.8}$$

$$r_i(\sigma_i) = \textstyle\sum_{a \in A(I_i)} r_i(\sigma_i a) \qquad \forall i \in \{l, f\}, I_i \in \mathcal{I}_i \tag{7.9}$$

$$0 \le s_{\sigma_f} \le (1 - r_f(\sigma_f))\, M \qquad \forall \sigma_f \in \Sigma_f \tag{7.10}$$

$$0 \le s^I_{\sigma_f} \le \left(1 - r^I_f(\sigma_f)\right) M \qquad \forall I \in \mathcal{I}_f, \sigma^I_f \in \Sigma^I_f \tag{7.11}$$

$$r_f(\sigma_f) \in \{0, 1\} \qquad \forall \sigma_f \in \Sigma_f \tag{7.12}$$

$$r^I_f(\sigma_f) \in \{0, 1\} \qquad \forall I \in \mathcal{I}_f \sigma_f \in \Sigma^I_f \tag{7.13}$$

$$0 \le p(z) \le r_i(\sigma_i(z)) \qquad \forall i \in \{l, f\}, z \in Z \tag{7.14}$$

$$1 = \textstyle\sum_{z \in Z} p(z) \pi_0(z) \tag{7.15}$$

$$0 \le r_l(\sigma_l) \le 1 \qquad \forall \sigma_l \in \Sigma_l \tag{7.16}$$

This MIP is an extension of the MIP given by Bošanskỳ and Čermák [18] to the limited-lookahead setting of Kroer and Sandholm [97]. Eq. (7.5) is the expected leader value over leaf nodes. Equations (7.6) to (7.13) set up best-response constraints for each follower information set, as well as for each pair of information sets I, I' such that $I' \in \mathcal{I}_I$ (these constraints are completely analogous to (7.1)-(7.4) except that the constraints involving $v_{I,I'}$ must be set up for each I in order to represent best responses when applying the lookahead evaluation function at I). Equations (7.14) and (7.15) ensure that the probabilities over leaves are correct. Finally (7.8), (7.9), and (7.16) ensure that r_l is a valid leader strategy.

7.6 MIP with uncertainty about follower payoff

We now move to the computation of RSSS for the setting with uncertainty about follower payoff but no limited lookahead. We will consider a particular class of uncertainty functions: interval uncertainty on each leaf payoff. More concretely, the uncertainty set will be

$$U_f = \{u_f : u_f(h) \in [L(h), U(h)], \forall h \in Z\},$$

where $L(h), U(h)$ are given upper and lower bounds on the interval that the payoff for leaf node h must be chosen from.

One issue that now arises is that we may not be able to make a single action optimal: if the maximum-to-minimum utility intervals for two sequences are guaranteed to overlap we cannot make either sequence the optimal choice for the follower player. Instead, we allow choosing both sequences, and we then assume that the leader player receives the minimum over the two. Intuitively, this can be thought of as a zero-sum game played within the space of actions made optimal for the follower player (a similar technique was used in Kroer and Sandholm [97]). More generally, we may have $k > 1$ actions at a given information set that can all be made optimal under various instantiations of the utility function. We now introduce a set-valued function that, under some given strategy for the follower r_f, returns the set of actions at a given follower information set that can be made optimal under some instantiation of the utility function, given the tie-breaking rule,

$$A_I(r_f) = \{a \in A_I : \nexists\, a' \in A_I,\ v_I^L(a') \geq v_I^U(a)\}.$$

For any $a \in A_I(r_f)$, the minimization over the uncertainty can choose an instantiation making a the only best-response action at I. Conversely, for $a \notin A_I(r_f)$, even if the utility function is chosen to maximize the value of action a, there exists some other action a' whose worst-case instantiation is at least as good; if a leads to better leader utility than a' then the minimization over utility functions will not allow them to be tied, and if a leads to worse utility than a', then even if a utility function causing a tie is chosen, the best-response tie-breaking in favor of the leader means that a' will be chosen. Thus, a' (or some other action) is always chosen over a.[2]

The function $A_I(r_f)$ is illustrated in Figure 7.1. The general intuition can be seen from the figure: the dotted line denotes the split between potentially optimal actions (black bars) and actions that cannot be made optimal through any utility-function choice (opaque bars). The dotted line is touched by interval end-points from both sets: this means that the two actions could be tied, but the lower-value would never be chosen, since it is either worse for the leader, in which case the tie-breaking does not choose it even in case of a tie, or if it is better then the minimization over the intervals will break the tie and make it inoptimal.

The intuition behind our robust MIP consists of three components: 1) the best-response feasibility MIP described in (7.1)-(7.4), instantiated independently for both the set of maximal and minimal valuation functions, 2) a set of constraints for computing the set $A_I(r_f)$ for a given r_f via best-response values for the maximal and minimal utility functions, and 3) a minimization similar to the dual best-response LP from the standard sequence-form LP [165].

[2]Here we rely on the assumption that every action has a strict inequality $v_I^U(a) > v_I^L(a)$. Without this assumption our MIP still works, but the math becomes more cumbersome.

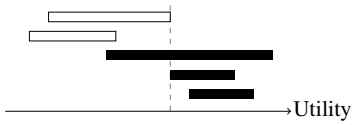


Figure 7.1: A set of action value-uncertainty intervals.

In the robust MIP given below, g_f^U, g_f^L are the functions giving the expected value over leaf nodes consistent with a pair of sequences, when every node has its payoff set to the maximal (g_f^U) and minimal (g_f^U) payoff, respectively.

$$\min_{r,v,s,y} y_0 \tag{7.17}$$

$$y_{\inf_f(\sigma_f)} \geq \sum_{\substack{I' \in \mathcal{I}_f \\ \sigma_f(I')=\sigma_f}} y_{I'} - \sum_{\sigma_l \in \Sigma_l} g_l(\sigma_l, \sigma_f) r_l(\sigma_l) - M(1 - r_f(\sigma_f)), \qquad \forall \sigma_f \in \Sigma_f \tag{7.18}$$

$$v^q_{\inf_f(\sigma_f)} = s^q_{\sigma_f} + \sum_{\substack{I' \in \mathcal{I}_f \\ \sigma_f(I')=\sigma_f}} v^q_{I'} + \sum_{\sigma_l \in \Sigma} r_l(\sigma_l) g^q_f(\sigma_l, \sigma_f), \qquad \forall \sigma_f \in \Sigma_f, q \in \{U, L\} \tag{7.19}$$

$$0 \leq s^q_{\sigma_f} \leq M(1 - b^q_f(\sigma_f)), \qquad \forall \sigma_f \in \Sigma_f, q \in \{U, L\} \tag{7.20}$$

$$\textstyle\sum_{a \in A(I)} b^q_f(\sigma_f(I)a) = 1, \qquad \forall q \in \{U, L\}, I \in \mathcal{I}_f \tag{7.21}$$

$$b^q_f(\sigma_f) \in \{0, 1\}, \qquad \forall \sigma_f \in \Sigma_f, q \in \{U, L\} \tag{7.22}$$

$$v^U_I - s^U_{\sigma_f} \geq v^L_I - M(1 - r_f(\sigma_f)), \qquad \forall \sigma_f \in \Sigma_f \tag{7.23}$$

$$v^U_I - s^U_{\sigma_f} \leq v^L_I + M r_f(\sigma_f), \qquad \forall \sigma_f \in \Sigma_f \tag{7.24}$$

$$r_i(\emptyset) = 1, \qquad \forall i \in \{l, f\} \tag{7.25}$$

$$r_l(\sigma) = \sum_{a \in A(I)} r_l(\sigma a), \qquad \forall I \in \mathcal{I}_l, \sigma = \sigma_l(I) \tag{7.26}$$

$$r_f(\sigma) \leq \sum_{a \in A(I)} r_f(\sigma a), \qquad \forall I \in \mathcal{I}_f, \sigma = \sigma_f(I) \tag{7.27}$$

$$r_f(\sigma_f) \in \{0, 1\}, \qquad \forall \sigma_f \in \Sigma_f \tag{7.28}$$

$$0 \leq r_l(\sigma_l) \leq 1, \qquad \forall \sigma_l \in \Sigma_l \tag{7.29}$$

Equations (7.17) and (7.18) implement the minimization over the set of potentially optimal actions $A_I(r_l)$ at a given information set I. Equations (7.19) to (7.22) ensure that v_I^U, v_I^L represent the correct value of each information set under the maximal and minimal utility function. Equations (7.23) and (7.24) ensure that actions in or not in $A_I(r_l)$ can potentially be made optimal (7.23) or cannot be made optimal (7.24). Equations (7.25) to (7.29) ensure that r_l is a valid sequence-form leader strategy and that one more pure strategies are active for the follower. We prove that this MIP computes a RSSS.

Theorem 23. *The interval uncertainty MIP computes a robust Stackelberg equilibrium for an*

EFG with interval uncertainty on each leaf payoff for the follower.

Proof. First we show that a robust Stackelberg equilibrium corresponds to a solution to the MIP. Let r_l, r_f be a robust Stackelberg equilibrium. Without loss of generality, assume that r_f is a pure strategy (for any mixed-strategy best response, by the assumption of tie-breaking in favor of the leader, a pure best response exists yielding the same utility for the leader). Set all MIP variables $r_l(\sigma_l)$ according to the equilibrium strategies. For all $\sigma_f \in A_{\inf(\sigma_f)}(r_l)$ set the corresponding MIP variable $r_f(\sigma_f) = 1$, and for all $\sigma_f \notin A_{\inf(\sigma_f)}(r_l)$ set $r_f = 0$. For any information set I, the action a played in the pure strategy r_f must be in $A_I(\sigma_l)$: if not it could not be made optimal by any utility function. Conversely, it must be the action providing the lowest utility to the leader among actions in $A_I(\sigma_l)$, or the minimization over utility functions would not have made it optimal. Choose y so as to minimize (7.17) subject to (7.18). This corresponds exactly to minimizing the leader utility over the set of follower sequences such that $r_f(\sigma_f) = 1$, and thus the objective is equal to the value of the RSSE. This can also be seen by realizing that (7.17, 7.18) correspond to the dual sequence-form best-response LP of an opponent wishing to minimize the leader utility over $A_I(r_l)$. Set $v_I^U, b_U, s_{\sigma_f}^U$ and $v_I^L, b_L, s_{\sigma_f}^L$ equal to the values obtained by arbitrarily chosen best responses according to the maximal and minimal utility functions respectively. By our choices for MIP variables it is clear that (7.18), (7.19), and (7.25)-(7.29) are satisfied. For (7.23) we set $r_f(\sigma_f) = 1$ only for variables in $A_{\inf(\sigma_f)}(\sigma_l)$, that is, sequences where their upper-bound value is greater than every lower-bound value, and thus $v_I^U - s_{\sigma_f}^U$, which is exactly the upper-bound value associated with σ_f, is greater than v_I^L. Conversely, for σ_f such that $r_f(\sigma_f) = 0$ we know that their upper-bound value is less than some lower-bound value, and thus $v_I^U - s_{\sigma_f}^U \leq v_I^L$.

Now consider an optimal solution to the MIP. The leader strategy for an RSSE is exactly the values computed for $r_l(\sigma_l)$ for all σ_l. By the same logic as for the standard Stackelberg MIP, v_I^U, v_I^L represent the information-set values according to the maximal and minimal utility functions for the follower [18, 156]. Since $v_I^U - s_\sigma^U$ corresponds exactly to the information-set value associated with a given sequence $\sigma \in \Sigma_I$, (7.23) implies that $r_f(\sigma) = 1$ if and only if $v_I^U(\sigma) \geq v_I^L(\sigma')$ for all $\sigma' \in \Sigma_I$. In other words, $\sigma \in A_I(r_l)$. Conversely $r_f(\sigma) = 0$ implies $\sigma \notin A_I(r_l)$. Thus the set of active sequences is exactly the set of sequences that can be made optimal for some choice of utility function for the follower. Because y is chosen to minimize the utility over active variables, this corresponds to the utility achieved when committing to r_l.

Since every RSSE is a solution of the MIP, and the optimal solution to the MIP corresponds to the payoff received by the leader if they were to commit to the strategy computed by the MIP, we conclude that the MIP computes an RSSS. If not, there would exist some RSSE which achieves better utility than what is computed by the MIP. This would be a contradiction, since such an RSSE would also be feasible and its objective would be equal to the RSSE value. □

7.6.1 MIP for Limited-Lookahead Interval Uncertainty

We also present an extension of the full-certainty MIP for limited lookahead to a setting with uncertainty about the limited-lookahead node-evaluation function. That MIP joins the ideas from both the full-certainty MIP with limited lookahead ((7.5)-(7.16)) and the robust MIP ((7.17)-(7.29)) and is thus the most comprehensive, but it combines the novel ideas from the former two MIPs in a fairly straightforward way.

$$\min_{r,v,s,y} y_0 \tag{7.30}$$

$$y_{\inf_f(\sigma_f)} \geq \sum_{\substack{I' \in \mathcal{I}_f \\ \sigma_f(I') = \sigma_f}} y_{I'} - \sum_{\sigma_l \in \Sigma_l} g_l(\sigma_l, \sigma_f) r_l(\sigma_l) - M(1 - r_f(\sigma_f)) \qquad \forall \sigma_f \in \Sigma_f \tag{7.31}$$

$$v^q_{I,\inf_f(\sigma_f)} = s^q_{I,\sigma_f} + \sum_{\substack{I' \in \mathcal{I}_I \\ \sigma_f(I') = \sigma_f}} v^q_{I,I'} + \sum_{\sigma_l \in \Sigma} r_l(\sigma_l) g^q_I(\sigma_l, \sigma_f), \quad \forall I \in \mathcal{I}_f, \sigma_f \in \Sigma_I, q \in \{U, L\} \tag{7.32}$$

$$0 \leq s^q_{I,\sigma_f} \leq M(1 - b^q_I(\sigma_f)) \qquad \forall I \in \mathcal{I}_f, \sigma_f \in \Sigma_I, q \in \{U, L\} \tag{7.33}$$

$$\sum_{a \in A(I')} b_I(\sigma_f(I')a) = 1 \qquad \forall I \in \mathcal{I}_f, q \in \{U, L\}, I' \in \mathcal{I}_I \tag{7.34}$$

$$b^q_I(\sigma_f) \in \{0, 1\} \qquad \forall I \in \mathcal{I}_f, \sigma_f \in \Sigma_I, q \in \{U, L\} \tag{7.35}$$

$$v^U_{I,I} - s^U_{I,\sigma} \geq v^L_{I,I} - M(1 - r_f(\sigma)) \tag{7.36}$$

$$v^U_{I,I} - s^U_{I,\sigma} \leq v^L_{I,I} + M r_f(\sigma) \tag{7.37}$$

$$r_i(\emptyset) = 1, \qquad \forall i \in \{l, f\} \tag{7.38}$$

$$r_l(\sigma) = \sum_{a \in A(I)} r_l(\sigma a) \qquad \forall I \in \mathcal{I}_l, \sigma = \sigma_l(I) \tag{7.39}$$

$$r_f(\sigma) <= \sum_{a \in A(I)} r_f(\sigma a) \qquad \forall I \in \mathcal{I}_f, \sigma = \sigma_f(I) \tag{7.40}$$

$$r_f(\sigma_f) \in \{0, 1\} \qquad \sigma_f \in \Sigma_f \tag{7.41}$$

$$0 \leq r_l(\sigma_l) \leq 1 \qquad \sigma_l \in \Sigma_l \tag{7.42}$$

7.7 Experiments

Using our MIPs presented in the previous section we investigated the scalability and qualitative properties of RSSS solutions. We experimented with three kinds of EFG: Kuhn poker (Kuhn) [106], a 2-card poker variant (2-card), and a parameterized security-inspired search game (Search). The search game is similar to games considered by Bošanskỳ et al. [17] and Bošanskỳ and Čermák [18]).

Kuhn consists of a three-card deck: king, queen, and jack. Each player first has to put a payment of 1 into the pot. Each player is then dealt one of the three cards, and the third is put aside unseen. A single round of betting then occurs (with betting parameter $p = 1$, explained below).

In 2-card, the deck consists of two kings and two jacks. Each player first has to put a payment of 1 into the pot. A private card is dealt to each, followed by a betting round (with betting parameter $p = 2$), then a public card is dealt, followed by another betting round (with $p = 4$).

In both games, each round of betting goes 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.

If no player has folded, a showdown occurs. In Kuhn poker, the player with the higher card wins in a showdown. In 2-card, 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 2-card game tree has 199 nodes, and 57 sequences per player.

We also perform experiments on the Search game used previously in Section 3.9. It is restated here for ease of reading. The search game is played on the graph shown in Figure 7.2. It is a simultaneous-move game (which can be modeled as a turn-taking EFG with appropriately chosen information sets). The leader controls two patrols that can each move within their respective shaded areas (labeled P1 and P2), and at each time step the controller chooses a move for both patrols. The follower is always at a single node on the graph, initially the leftmost node labeled S and can move freely to any adjacent node (except at patrolled nodes, the follower cannot move from a patrolled node to another patrolled node). The follower can also choose to wait in place for a time step in order to clean up their traces. If a patrol visits a node that was previously visited by the follower, and the follower did not wait to clean up their traces, they can see that the follower was there. If the follower reaches any of the rightmost nodes they received the respective payoff at the node (5, 10, or 3, respectively). If the follower and any patrol are on the same node at any time step, the follower is captured, which leads to a payoff of 0 for the follower and a payoff of 1 for the leader. Finally, the game times out after k simultaneous moves, in which case the leader receives payoff 0 and the follower receives $-\infty$ (because we are interested in games where the follower attempts to reach an end node). We consider games with k being 5 and 6. We will denote these by Search-5 and Search-6. Search-5 (Search-6) has 87,927 (194,105) nodes and 11,830 and 69 (68,951 and 78) leader and follower sequences.

All experiments were conducted using Gurobi 7.5.1 to solve MIPs, on a cluster with 8 Intel Xeon E5607 2.2Ghz cores and 47 GB RAM per experiment.

In the first set of experiments we investigate the impact on runtime caused by uncertainty intervals in each of the four games, without considering limited lookahead. We compare the MIP by Bošanskỳ and Čermák [18] (B&C) for the full-certainty setting to our robust MIP ((7.17)-(7.29)) with an uncertainty interval of diameter d at each node in the game, for 6 different values of d. The results are shown in Table 7.1. Interestingly, our robust MIP with interval 0 is sig-

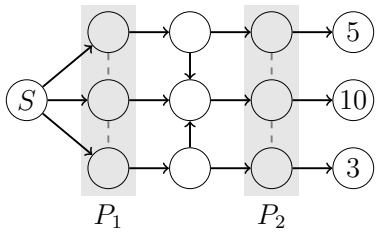


Figure 7.2: The graph on which the search game is played.

	Kuhn	2-card	Search-5	Search-6
B&C	0	1	13	190
R-0	0	26	1	40
R-0.01	0	548	24	683
R-0.05	0	616	30	910
R-0.1	0	209	36	955
R-0.5	0	365	64	1648
R-1	0	42	41	395

Table 7.1: Runtime experiments for the MIP by Bošanskỳ and Čermák [18] (B& C) and our robust Stackelberg MIP for increasing uniform uncertainty intervals (R-c where c is the interval radius). All runtimes are in seconds.

nificantly faster than the B&C MIP for the Search games. (We do not specialize our robust MIP to the full-certainty setting but instead let Gurobi presolve away most redundant variables and constraints. One could easily specialize it and potentially make it even faster.) Once we add uncertainty, the MIP gets harder to solve, with the runtime increasing for larger uncertainty intervals—except for the largest uncertainty interval where the problem starts to get easier again.

In the second set of experiments, we investigate the cost of computing an RSSS against a follower utility function that is different from the one actually employed by the follower. These experiments were conducted on the Search-5 game. On Search-6 it would take prohibitively long to conduct all the experiments, and the experiments would not be interesting on Kuhn and 2-card because they are zero-sum games (the leader will end up getting the value of the game as long as the correct utility function is contained in the uncertainty intervals). The setup is as follows. We use our robust MIP to compute a leader strategy for the original payoffs in Search-5. We instantiate the MIP with several different uncertainty-interval widths (given in the leftmost column in Table 7.2). For each leader strategy, we then conduct a grid search over triplets of numbers in $\{\pm 0.1, \pm 0.5, \pm 1, \pm 2, \pm 3\}^3$, where the three numbers correspond to a change in utility being added to each of the three rightmost payoff nodes in Figure 3.2. For each payoff change, we compute the follower's best response (breaking ties in favor of the leader) to the leader strategy under the new game and the resulting leader utility. The second column in Table 7.2 (EV) denotes the value that the leader is expected to get if he were solving the correct game. The following three columns, labeled $\leq 1, \leq 2, \leq 3$, show the worst utility achieved by the leader when the grid search is restricted to payoff changes of at most 1, 2, and 3, respectively. For example, in the case ≤ 1 we only do the grid search over $\{\pm 0.1, \pm 0.5, \pm 1\}^3$ The experiment shows that when uncertainty is not taken into account, all amounts of perturbation leads to a large decrease in leader utility. Conversely, taking uncertainty into account leads to much better utility in almost

	EV	≤ 1	≤ 2	≤ 3
0	0.842	0.474	0.474	0.474
0.1	0.834	0.479	0.479	0.479
0.5	0.800	0.500	0.500	0.500
1	0.758	0.616	0.526	0.526
2	0.688	0.688	0.417	0.417
4	0.667	0.667	0.667	0.667
6	0.667	0.667	0.667	0.667
40	0.500	0.500	0.500	0.500

Table 7.2: Leader utility when maximizing utility against an incorrect utility function. Each row corresponds to a different size of uncertainty interval used for computing the leader strategy (interval size is given in the leftmost column). The columns are ordered in increasing amounts of incorrectness allowed in the follower utility function.

every case.

In the third set of experiments, we investigate the cost to the leader from having to take uncertainty into account against a limited-lookahead follower. We perform this experiment on Kuhn and 2-card, both zero-sum games, which allows us to apply the same node-evaluation scheme as in Kroer and Sandholm [97]. In order to construct the limited-lookahead evaluation function, we first compute a Nash equilibrium of the game. We then recursively define the value of each node to be the weighted sum over the values of nodes beneath it, where the weights are the probabilities of each action in the Nash equilibrium, and then add Gaussian noise to the computed value (we do not add any noise to leaf nodes). Since the value of a node is based on the noisy value of nodes beneath it, the farther away from leaf nodes a node is, the noisier the estimate of the node's value (from Nash equilibrium) is. We then use our robust limited-lookahead MIP to solve the limited-lookahead game resulting from having the follower apply this node-evaluation function. We consider lookahead depths of 1 and 2. The results for 2-card are shown in Table 7.3 and the results for Kuhn are shown in Table 7.4. The different rows in the tables correspond to varying standard deviations in the Gaussian noise, and columns correspond to increasing sizes of uncertainty intervals. For all games, lookahead depths, and noise levels, we see that the amount that the leader can exploit the follower goes down as uncertainty intervals get larger. However, we also see that for most noise amounts, some amount of robustness can be added without losing substantial leader utility. Coupled with our results from the second set of experiments, which showed that uncertainty intervals are necessary if there is mis-specification in the model, this suggests that uncertainty intervals can lead to substantially more robust outcomes, potentially at a small cost to optimality even if the initial model turns out to be correct.

7.8 Conclusions and future research

We initiated the study of robustness in the context of Stackelberg EFGs. We introduced MIPs that can compute the optimal strategy to commit for the leader in several settings, including standard Stackelberg and against a limited-lookahead opponent. The MIP relies on interval uncertainty around payoffs. It would be interesting to find mathematical programs for more general uncertainty sets, ideally while staying within a class of practically-solvable mathematical programs.

Lookahead depth: 1

Noise σ	0.01	0.05	0.1	0.5	1	2
0.1	1.35	1.33	1.33	1.15	0.25	0.00
0.5	1.42	1.41	1.33	1.33	0.61	0.00
1	1.50	1.50	1.50	1.33	1.33	0.34
2	1.50	1.50	1.50	1.44	1.33	1.33

Lookahead depth: 2

Noise σ	0.01	0.05	0.1	0.5	1	2
0.1	0.67	0.43	0.05	0.00	0.00	0.00
0.5	0.69	0.69	0.68	0.05	0.00	0.00
1	0.73	0.72	0.71	0.48	0.08	0.00
2	0.80	0.79	0.78	0.71	0.59	0.19

Table 7.3: Limited-lookahead with depth 1 and 2 in 2-card.

Lookahead depth: 1

Noise σ	0.01	0.05	0.1	0.5	1	2
0.1	0.33	0.33	0.33	0.25	-0.06	-0.06
0.5	0.33	0.33	0.33	0.33	-0.06	-0.06
1	0.33	0.33	0.33	0.33	0.33	0.16
2	0.87	0.87	0.87	0.86	0.84	0.22

Lookahead depth: 2

Noise σ	0.01	0.05	0.1	0.5	1	2
0.1	-0.03	-0.05	-0.06	-0.06	-0.06	-0.06
0.5	0.29	0.28	0.28	0.16	-0.06	-0.06
1	0.41	0.41	0.40	0.30	0.22	0.16
2	0.87	0.87	0.87	0.86	0.84	0.22

Table 7.4: Limited-lookahead with depth 1 and 2 in Kuhn.

We showed experimentally that the loss due to not considering uncertainty can be quite large, and that this can be ameliorated with our approach. We also showed that robust models can be solved at a modest increase in runtime, about a factor of 10.

While we showed that our technique scales to medium-size games, in practice we would often like to scale to even larger games. The iterative LP-based approach of Čermák [33] could potentially be extended to the robust setting. Černỳ et al. [38] extend the iterative LP approach to incorporate strategy-generation, in order to increase scalability since many strategies do not need to be generated. It would be interesting to develop a strategy-generation approach for our robust setting, which is likely to yield greater scalability. It would also be beneficial to develop a more traditional column-generation approach in order to understand what type of incremental approach is better.

Abstraction methods have dramatically increased the scalability of Nash equilibrium finding in EFGs (e.g., [28, 68, 95, 98, 109]) and could potentially be adapted to the robust Stackelberg setting as well. This could be done while giving guarantees on follower behavior by only abstracting the strategy space of the leader.

In recent work that is not part of this thesis, coauthors and I studied equilibrium refinement in Stackelberg EFGs [60]. The robust framework presented in this chapter could potentially be combined with some notion of Stackelberg equilibrium refinement. Questions include whether there is a benefit to refinement when using a robust follower model, whether the problem becomes algorithmically harder, and whether scalable techniques such as MIP can still be leveraged.

Chapter 8

Concluding remarks and further thoughts

We showed how FOMs can be made state-of-the-art for solving large-scale EFGs by constructing a dilated entropy function. There is some reason to think that our DGF construction is the "right" way to measure distances over treeplexes (and thus EFG strategy spaces): we showed that our analysis is tight, and most importantly we showed that it generalizes the well-known dimension independence of the entropy DGF on a simplex. From a practical perspective we showed that our approach beats all the best recent algorithms for solving zero-sum EFGs except for the CFR$^+$ algorithm. This is especially encouraging because our smoothing approach is a general technique that can be combined with many different FOMs. There is a large, and increasing, number of algorithms for solving saddle-point problems, and so an important future direction is to find the best FOM for solving EFGs in practice. Since this approach automatically benefits from future developments in FOMs it seems plausible that our DGF can eventually be combined with some FOM that can beat CFR$^+$. In a similar vein, the space of stochastic FOMs combined with our DGF remains largely unexplored from a practical perspective. Here it again seems plausible that some stochastic FOM variant combined with our DGF could eventually lead to practical performance gains over the MCCFR approach.

Poker AIs like Libratus [27] and DeepStack [124] have shown that Nash equilibria combined with methods for scaling up to very large games allows creating superhuman AIs. Creating strong AIs for general-sum, and potentially n-player, EFGs remains a large unsolved problem. Both practical abstraction approaches and our theoretical abstraction framework developed in this thesis extend to such games. However, we do not have scalable algorithms for computing Nash equilibria in the resulting abstractions, and more fundamentally it is unclear whether Nash equilibrium is even the right solution concept: there are equilibrium selection problems, and so even if some Nash equilibrium is known to perform well, we also need to guarantee that our algorithm finds that Nash equilibrium, and not some other bad equilibrium. Furthermore, in zero-sum games we know that if the opponent does not play a best response to our strategy we can only be better off. This is not so in general, and so deploying a Nash equilibrium in a poker game with 5 other players is not guaranteed to be a strong strategy if they do not play according to our equilibrium. Čermák et al. [34] show that strategies computed by CFR perform well in small general-sum poker games and random EFGs when compared to various other solution concepts when playing against a number of non-equilibrium strategies in randomly generated games. One could similarly explore the solution quality from smoothed-best-response iteration

that would arise from applying a FOM with the dilated entropy. One potentially more rigorous approach would be to find a way to exploit near-zero-sum properties of certain games, and use regularization to compute an approximate equilibrium that does not skew too heavily towards favoring or disfavoring any players.

For abstraction, the most significant future work is to find a way to leverage our decomposition result to give ex-ante solution-quality bounds for the types of abstraction employed in practice. As evidenced by our counterexamples this will not be possible for general abstractions. Instead, it may be possible to leverage Theorem 8 or 9 for special game classes that can capture practical abstractions. In the two recent results on strong poker AIs [27, 124] only Libratus [27] applied abstraction in order to scale up the equilibrium finding, whereas DeepStack [124] used deep learning in order to estimate values of actions (both approaches relied on iterative equilibrium-finding algorithms for solving subgames in realtime). Since the DeepStack approach is not guaranteed to learn a correct estimator this begs the obvious question of whether we can give a characterization of what sort of solution-quality can be expected given a certain accuracy in the estimator. Similarly, since both agents employed real-time near-exact subgame solving it would be useful to understand how such re-solving interacts with the overall solution-quality bounds for the abstract game originally solved (or in the case of DeepStack, the game solved via estimated payoffs).

In chapter 5 on equilibrium refinement we showed that our DGF approach can be extended to the computation of approximate Nash equilibrium refinements via a perturbation to the entropy DGF employed at each simplex in the treeplex. This shows that iterative methods can be extended to computing refinements, and provides hope that it may be possible to employ equilibrium refinements in practice (since exact approaches rely on LPs that are not scalable to large EFGs). However, the approach relies on a fixed perturbation, which may be either too large and cause too much strategy perturbation, or it may be so small that it is impractical for converging to a solution. An obvious remedy would be to find a way to adaptively decrease the perturbation in sync with our convergence to a solution. This would require developing a FOM variant that handles the slowly-changing strategy polytope, or proving that this is implicitly handled by some existing FOM. In the vein of scaling up equilibrium-refinement computation there is also the question of whether the abstraction framework developed in this thesis can be modified to give some form of approximation quality toward equilibrium refinement.

Another potentially-exciting application of our work would be to apply our smoothing approach to the computation of Stackelberg equilibria. For example, it may be possible to smooth the strategy space of the leader, while performing the follower best-responses computations as part of each iteration that incrementally updates the leader strategy space. An obvious caveat to this approach is that the utility for the leader is non-convex due to the discontinuity of best responses. It would be interesting to investigate whether there are convex relaxations that can be handled efficiently.

From a practical perspective there are many interesting problems for which EFGs hold promise. One area of application that has already started to see some work is security settings, where EFGs can capture temporal interactions and imperfect information. Such models are being explored in *green security games*, where rangers in parks with endangered wildlife wish to protect animals from poaching [55, 56, 88]. Green security games are naturally modeled as EFGs because rangers and poachers move around the park, but have imperfect information about the adver-

sary's movements: when they observe tracks, old campsites, and similar signals they can adapt to try to track down the poachers (or avoid the rangers if the poachers observe such signals). Such games are extremely large, and so scalability techniques play a key role. Cybersecurity is another emerging topic being modeled via game theory. Examples include modeling increased threat risk and adaptive defense due to anomalies such as crashes (which can be from an attacker probing the system) [172], adaptive defense against distributed denial-of-service attacks [158], strategic honeypot placement [32], network hardening [52, 53], and defensive and attacking use of vulnerability discovery and exploitation [3, 123]. Sequential decision making and EFGs are also being explored for steering evolutionary biological adaptation processes [99, 151]. However, there are many other interesting applications waiting for further study: high-level strategy selection in real-time games, negotiation, strategic pricing, financial strategy, sequential auctions, electricity-market strategy, and many more.

Bibliography

[1] ACPC. Annual Computer Poker Competition website. `http://www.computerpokercompetition.org/`, 2016. [Online; accessed 23-Feb-2016]. 1

[2] C. Archibald and Y. Shoham. Modeling billiards games. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, Budapest, Hungary, 2009. 1

[3] Tiffany Bao, Yan Shoshitaishvili, Ruoyu Wang, Christopher Kruegel, Giovanni Vigna, and David Brumley. How shall we play a game?: A game-theoretical model for cyber-warfare games. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 7–21. IEEE, 2017. 8

[4] Anjon Basak, Fei Fang, Thanh Hong Nguyen, and Christopher Kiekintveld. Abstraction methods for solving graph-based security games. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 13–33. Springer, 2016. 4.1

[5] Anjon Basak, Fei Fang, Thanh Hong Nguyen, and Christopher Kiekintveld. Combining graph contraction and strategy generation for green security games. In *International Conference on Decision and Game Theory for Security*, pages 251–271. Springer, 2016. 4.1

[6] Nicola Basilico and Nicola Gatti. Automated abstractions for patrolling security games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2011. 4, 4.1

[7] Donald F Beal. An analysis of minimax. *Advances in computer chess*, 2:103–109, 1980. 6.4

[8] Amir Beck and Marc Teboulle. Smoothing and first order methods: A unified framework. *SIAM Journal on Optimization*, 22(2):557–580, 2012. 1.1, 4.9.5

[9] Aharon Ben-Tal and Arkadi Nemirovski. Robust optimization–methodology and applications. *Mathematical Programming*, 92(3):453–480, 2002. 7

[10] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*. Princeton University Press, 2009. 7

[11] Ahron Ben-Tal and Arkadi Nemirovski. *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*, volume 2. Siam, 2001. 4.9.5

[12] Joseph Louis François Bertrand. Theorie mathematique de la richesse sociale. *Journal des Savants*, pages 499–508, 1883. 7

[13] Dimitri P Bertsekas. Proximal algorithms and temporal difference methods for solving

fixed point problems. *Computational Optimization and Applications*, 70(3):709–736, 2018. 3.11

[14] Dimitris Bertsimas, David B Brown, and Constantine Caramanis. Theory and applications of robust optimization. *SIAM review*, 53(3):464–501, 2011. 7

[15] 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 International Joint Conference on Artificial Intelligence (IJCAI)*, 2003. 4, 4.1

[16] Kellog S Booth and Charles J Colbourn. *Problems polynomially equivalent to graph isomorphism*. Computer Science Department, Univ., 1979. 4.5.2

[17] B Bošanskỳ, Christopher Kiekintveld, V Lisý, and Michal Pěchouček. An exact double-oracle algorithm for zero-sum extensive-form games with imperfect information. *Journal of Artificial Intelligence Research*, pages 829–866, 2014. 3.1, 3.8, 7.7

[18] Branislav Bošanskỳ and Jiří Čermák. Sequence-form algorithm for computing Stackelberg equilibria in extensive-form games. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. (document), 3.8, 7, 7.5, 7.6, 7.7, 7.7, 7.1

[19] Branislav Bošanskỳ, Simina Brânzei, Kristoffer Arnsfelt Hansen, Troels Bjerre Lund, and Peter Bro Miltersen. Computation of Stackelberg equilibria of finite sequential games. *ACM Transaction on Economics and Computation (TEAC)*, 5(4):23:1–23:24, December 2017. ISSN 2167-8375. 7

[20] Bruno Bouzy and Tristan Cazenave. Computer go: an AI oriented survey. *Artificial Intelligence*, 132(1):39–103, 2001. 6

[21] Michael Bowling, Neil Burch, Michael Johanson, and Oskari Tammelin. Heads-up limit hold'em poker is solved. *Science*, 347(6218), January 2015. 3, 3.1, 4.1

[22] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. 4.9.5

[23] Noam Brown and Tuomas Sandholm. Regret transfer and parameter optimization. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2014. 3, 4, 4.1

[24] Noam Brown and Tuomas Sandholm. Simultaneous abstraction and equilibrium finding in games. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2015. 3, 4, 4.1

[25] Noam Brown and Tuomas Sandholm. Strategy-based warm starting for regret minimization in games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2016. 3.8

[26] Noam Brown and Tuomas Sandholm. Safe and nested subgame solving for imperfect-information games. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, pages 689–699, 2017. 5, 4.1

[27] Noam Brown and Tuomas Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, page eaao1733, Dec. 2017. 3, 3.9, 3.10, 4.1, 4.2, 4.6, 8

[28] 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. 1, 3.11, 4.1, 4.2, 4.6, 4.9.7, 7.8

[29] Noam Brown, Christian Kroer, and Tuomas Sandholm. Dynamic thresholding and pruning for regret minimization. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2017. 3.1

[30] Noam Brown, Tuomas Sandholm, and Brandon Amos. Depth-limited solving for imperfect-information games. *arXiv preprint arXiv:1805.08195*, 2018. 6.5

[31] Neil Burch, Marc Lanctot, Duane Szafron, and Richard G Gibson. Efficient Monte Carlo counterfactual regret minimization in games with many player actions. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, pages 1880–1888, 2012. 4.2

[32] Thomas E Carroll and Daniel Grosu. A game theoretic investigation of deception in network security. *Security and Communication Networks*, 4(10):1162–1172, 2011. 8

[33] Jiří Čermák. Using correlated strategies for computing Stackelberg equilibria in extensive-form games. 7, 7.8

[34] Jiří Čermák, Branislav Bošanskỳ, and Nicola Gatti. Strategy effectiveness of game-theoretical solution concepts in extensive-form general-sum games. In *Autonomous Agents and Multi-Agent Systems*, pages 1813–1814. International Foundation for Autonomous Agents and Multiagent Systems, 2015. 8

[35] Jiří Čermák, Branislav Bošansky, and Viliam Lisý. An algorithm for constructing and solving imperfect recall abstractions of large extensive-form games. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 936–942, 2017. 4.1

[36] Jiří Čermák, Branislav Bošanskỳ, Karel Horák, Viliam Lisỳ, and Michal Pěchouček. Approximating maxmin strategies in imperfect recall games using a-loss recall property. *International Journal of Approximate Reasoning*, 93:290–326, 2018. 4.8

[37] Jiří Čermák, Viliam Lisý, and Branislav Bošansky. Constructing imperfect recall abstractions to solve large extensive-form games. *arXiv preprint arXiv:1803.05392*, 2018. 4.1

[38] Jakub Černỳ, Branislav Boỳanskỳ, and Christopher Kiekintveld. Incremental strategy generation for stackelberg equilibria in extensive-form games. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 151–168. ACM, 2018. 7.8

[39] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 2011. 1.1, 3.11

[40] Antonin Chambolle and Thomas Pock. On the ergodic convergence rates of a first-order primal–dual algorithm. *Mathematical Programming*, 159(1-2):253–287, 2016. 1.1

[41] Katherine Chen and Michael Bowling. Tractable objectives for robust policy optimization. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2012. 1, 4.6.3, 4.9.6

[42] Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of computing two-player Nash equilibria. *Journal of the ACM*, 2009. 1, 6.2.1

[43] Xianfu Chen, Zhu Han, Honggang Zhang, Guoliang Xue, Yong Xiao, and Mehdi Bennis. Wireless resource scheduling in virtualized radio access networks using stochastic learning. *IEEE Transactions on Mobile Computing*, (1):1–1, 2018. 1, 4.9.6

[44] Yunmei Chen, Guanghui Lan, and Yuyuan Ouyang. Optimal primal-dual methods for a class of saddle point problems. *SIAM Journal on Optimization*, 24(4):1779–1814, 2014. 1.1

[45] Vincent Conitzer. Computing Slater rankings using similarities among candidates. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Boston, MA, 2006. Early version appeared as IBM RC 23748, 2005. 6, 7

[46] 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. 7, 7.1

[47] Antoine Augustin Cournot. *Recherches sur les principes mathématiques de la théorie des richesses (Researches into the Mathematical Principles of the Theory of Wealth)*. Hachette, Paris, 1838. 7

[48] Constantinos Daskalakis, Paul W Goldberg, and Christos H Papadimitriou. The complexity of computing a nash equilibrium. *SIAM Journal on Computing*, 39(1), 2009. 1, 3.1, 6.2.1

[49] Constantinos Daskalakis, Alan Deckelbaum, and Anthony Kim. Near-optimal no-regret algorithms for zero-sum games. *Games and Economic Behavior*, 92:327–348, 2015. 1, 3.1

[50] Giuseppe De Nittis, Alberto Marchesi, and Nicola Gatti. Computing the strategy to commit to in polymatrix games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018. 7

[51] 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. 1, 4.9.6

[52] Karel Durkota, Viliam Lisỳ, Branislav Bošanskỳ, and Christopher Kiekintveld. Optimal network security hardening using attack graph games. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2015. 8

[53] Karel Durkota, Viliam Lisỳ, Christopher Kiekintveld, Branislav Bošanskỳ, and Michal Pěchouček. Case studies of network defense with attack graph games. *IEEE Intelligent Systems*, 31(5):24–30, 2016. 8

[54] Kousha Etessami and Mihalis Yannakakis. On the complexity of Nash equilibria and other fixed points (extended abstract). In *Proceedings of the Annual Symposium on Foundations of Computer Science (FOCS)*, pages 113–123, 2007. 1

[55] Fei Fang, Peter Stone, and Milind Tambe. When security games go green: Designing

defender strategies to prevent poaching and illegal fishing. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2015. 8

[56] Fei Fang, Thanh H Nguyen, Robert Pickles, Wai Y Lam, Gopalasamy R Clements, Bo An, Amandeep Singh, Brian C Schwedock, Milind Tambe, and Andrew Lemieux. Paws–a deployed game-theoretic application to combat poaching. *AI Magazine*, 38(1):23–37, 2017. 8

[57] Gabriele Farina and Nicola Gatti. Extensive-form perfect equilibrium computation in two-player games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2017. 5

[58] Gabriele Farina, Christian Kroer, and Tuomas Sandholm. Regret minimization in behaviorally-constrained zero-sum games. In *International Conference on Machine Learning (ICML)*, 2017. 1.3, 4.2

[59] Gabriele Farina, Christian Kroer, and Tuomas Sandholm. Online convex optimization for sequential decision processes and extensive-form games. In *arXiv*, 2018. 1.1

[60] Gabriele Farina, Alberto Marchesi, Christian Kroer, Nicola Gatti, and Tuomas Sandholm. Trembling-hand perfection in extensive-form games with commitment. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 233–239, 2018. 7.8

[61] Tomás Feder and Daniel Greene. Optimal algorithms for approximate clustering. In *Proceedings of the Annual Symposium on Theory of Computing (STOC)*, 1988. 4.6.3

[62] 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. 4.5.2

[63] 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. 4.9.7, 4.9.8

[64] 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. 1.2, 4.1, 4.2, 4.5.2, 6.2.2, 6.5

[65] 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. 3

[66] 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. 1.2, 4, 4.1, 4.5, 4.5.2

[67] 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, 2007. 1.2, 4.1, 4.5.2, 4.6.3

[68] Andrew Gilpin and Tuomas Sandholm. Lossless abstraction of imperfect information games. *Journal of the ACM*, 54(5), 2007. 1, 3.1, 4, 4.1, 4.5.2, 4.5.3, 12, 4.5.3, 4.5.4, 5, 6.2.2, 7.8, 8

[69] 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. 1.2, 4.1, 4.5

[70] 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. 1.2, 4.1, 4.5, 4.5.2, 4.6.3

[71] 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. 4, 4.1, 4.5.2

[72] 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. 3.1, 3.6.1

[73] Teofilo F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38, 1985. 4.6.3, 4.6.3

[74] 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. 4.6.3

[75] John Hawkin, Robert Holte, and Duane Szafron. Automated action abstraction of imperfect information extensive-form games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2011. 4, 4.1

[76] 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. 4, 4.1

[77] John Hillas and Elon Kohlberg. Foundations of strategic equilibrium. *Handbook of Game Theory with Economic Applications*, 2002. 5

[78] Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Fundamentals of convex analysis*. 2001. 3.5

[79] Samid Hoda, Andrew Gilpin, Javier Peña, and Tuomas Sandholm. Smoothing techniques for computing Nash equilibria of sequential games. *Mathematics of Operations Research*, 35(2), 2010. 1.1, 3, 3.1, 3.4, 3.4, 3.5, 3.5, 3.6.1, 3.7, 3.8, 3.9, 4.2, 4.5, 4.9, 5.2, 6.2.2

[80] Sune K Jakobsen, Troels B Sørensen, and Vincent Conitzer. Timeability of extensive-form games. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, pages 191–199. ACM, 2016. 4.8

[81] 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. 1, 3.1

[82] 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. 3.9, 6.2.1

[83] 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. 3, 4.9

[84] 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. 4, 4.1, 4.2, 4.5.2

[85] 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, 2011. 3.2.2, 3.5

[86] 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, 2011. 3.2.2

[87] Anatoli Juditsky, Arkadi Nemirovski, and Claire Tauvel. Solving variational inequalities with stochastic mirror-prox algorithm. *Stochastic Systems*, 1(1):17–58, 2011. 1.1, 3.6.1, 3.10

[88] Debarun Kar, Benjamin Ford, Shahrzad Gholami, Fei Fang, Andrew Plumptre, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba, Mustapha Nsubaga, et al. Cloudy with a chance of poaching: adversary behavior modeling and forecasting with real-world poaching data. In *Autonomous Agents and Multi-Agent Systems*, pages 159–167. International Foundation for Autonomous Agents and Multiagent Systems, 2017. 8

[89] Christopher Kiekintveld, Milind Tambe, and Janusz Marecki. Robust Bayesian methods for Stackelberg security games (extended abstract). In *Autonomous Agents and Multi-Agent Systems*, 2010. Short paper. 7

[90] Christopher Kiekintveld, Janusz Marecki, and Milind Tambe. Approximation methods for infinite bayesian stackelberg games: modeling distributional payoff uncertainty. In *10^{th} International Conference on Autonomous Agents and Multiagent Systems AAMAS 2011*, 2011. 4.9.6

[91] Christopher Kiekintveld, Towhidul Islam, and Vladik Kreinovich. Security games with interval uncertainty. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 231–238. International Foundation for Autonomous Agents and Multiagent Systems, 2013. 1.5, 7

[92] 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. 2.1, 4.6

[93] Daphne Koller, Nimrod Megiddo, and Bernhard von Stengel. Efficient computation of

equilibria for extensive two-person games. *Games and Economic Behavior*, 14(2), 1996. 1, 3.1, 3.2.2

[94] Richard Korf. Real-time heuristic search. *Artificial Intelligence*, 42(2-3):189–211, March 1990. 6

[95] Christian Kroer and Tuomas Sandholm. Extensive-form game abstraction with bounds. In *Proceedings of the ACM Conference on Economics and Computation (EC)*, 2014. 1.2, 3.1, 4.4, 4.9, 4.9.4, 6.2.2, 7.8

[96] 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)*, 2015. 1.2

[97] Christian Kroer and Tuomas Sandholm. Limited lookahead in imperfect-information games. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2015. 1.4, 7, 7.2, 7.5, 7.6, 7.7

[98] Christian Kroer and Tuomas Sandholm. Imperfect-recall abstractions with bounds in games. In *Proceedings of the ACM Conference on Economics and Computation (EC)*, 2016. 1.2, 4.4, 4.9.4, 6.2.2, 7.8

[99] Christian Kroer and Tuomas Sandholm. Sequential planning for steering immune system adaptation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2016. 7, 8

[100] Christian Kroer and Tuomas Sandholm. A unified framework for extensive-form game abstraction with bounds. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2018. 1.2

[101] 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. (document), 1.1, 3, 3.1, 3.6.1, 3.8, 3.8, 3.3, 3.8, 4.2, 4.5

[102] Christian Kroer, Gabriele Farina, and Tuomas Sandholm. Smoothing method for approximate extensive-form perfect equilibrium. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2017. 1.3

[103] Christian Kroer, Kevin Waugh, Fatma Kılınç-Karzan, and Tuomas Sandholm. Theoretical and practical advances on smoothing for extensive-form games. In *Proceedings of the ACM Conference on Economics and Computation (EC)*, 2017. 1.1, 3.9, 4.2, 4.5, 5.2, 5.3

[104] Christian Kroer, Gabriele Farina, and Tuomas Sandholm. Robust stackelberg equilibria in extensive-form games and extension to limited lookahead. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018. 1.5

[105] Christian Kroer, Gabriele Farina, and Tuomas Sandholm. Solving large sequential games with the excessive gap technique. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*, 2018. 1.1

[106] 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. 6.4, 7.7

[107] Guanghui Lan. An optimal method for stochastic composite optimization. *Mathematical Programming*, 133(1-2):365–397, 2012. 3.10

[108] 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)*, 2009. 3, 3.1, 3.6.1, 3.9, 3.10, 4.2, 4.5, 4.9, 6.2.2

[109] 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. 1.2, 4, 4.1, 1, 4.4, 4.5.1, 4.6, 4.6.2, 4.6.3, 4.6.4, 4.6.4, 4.7, 7.8

[110] 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. 6, 6.2.2, 7

[111] Joshua Letchford, Vincent Conitzer, and Kamesh Munagala. Learning and approximating the optimal strategy to commit to. In *International Symposium on Algorithmic Game Theory*, pages 250–262. Springer, 2009. 7

[112] 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. 1, 3.1

[113] Viliam Lisý, Trevor Davis, and Michael Bowling. Counterfactual regret minimization in sequential security games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2016. 1

[114] 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. 1, 3.1

[115] Bo Liu, Ji Liu, Mohammad Ghavamzadeh, Sridhar Mahadevan, and Marek Petrik. Finite-sample analysis of proximal gradient td algorithms. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, pages 504–513. AUAI Press, 2015. 3.11

[116] Bo Liu, Ji Liu, Mohammad Ghavamzadeh, Sridhar Mahadevan, and Marek Petrik. Proximal gradient temporal difference learning algorithms. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4195–4199, 2016. 3.11

[117] Mitja Luštrek, Matjaž Gams, and Ivan Bratko. Is real-valued minimax pathological? *Artificial Intelligence*, 170(6):620–642, 2006. 6.4

[118] Sridhar Mahadevan, Bo Liu, Philip Thomas, Will Dabney, Steve Giguere, Nicholas Jacek, Ian Gemp, and Ji Liu. Proximal reinforcement learning: A new theory of sequential decision making in primal-dual spaces. *arXiv preprint arXiv:1405.6757*, 2014. 3.11

[119] Janusz Marecki, Gerald Tesauro, and Richard Segal. Playing repeated stackelberg games with unknown opponents. In *International Conference on Autonomous Agents and Multi-agent Systems, AAMAS 2012*, 2012. 4.9.6

[120] Peter Bro Miltersen and Troels Bjerre Sørensen. Fast algorithms for finding proper strategies in game trees. In *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2008. 1.3, 5, 5.1

[121] Peter Bro Miltersen and Troels Bjerre Sørensen. Computing a quasi-perfect equilibrium of a two-player game. *Economic Theory*, 42(1), 2010. 1.3, 5

[122] 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. 6

[123] Tyler Moore, Allan Friedman, and Ariel D Procaccia. Would a'cyber warrior'protect us: exploring trade-offs between attack and defense of information systems. In *Proceedings of the 2010 New Security Paradigms Workshop*, pages 85–94. ACM, 2010. 8

[124] Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337), May 2017. 3, 4.1, 8

[125] 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. 1

[126] John Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36:48–49, 1950. 4.1

[127] John Nash. Non-cooperative games. *Annals of Mathematics*, 54:289–295, 1951. 2.2

[128] Dana S Nau. Pathology on game trees revisited, and an alternative to minimaxing. *Artificial intelligence*, 21(1):221–244, 1983. 6, 6.4

[129] 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. 6, 6.4

[130] 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), 2004. 1.1, 3, 3.11

[131] Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609, 2009. 3.10

[132] Yurii Nesterov. Excessive gap technique in nonsmooth convex minimization. *SIAM Journal of Optimization*, 16(1), 2005. 1.1, 3, 3.3.1, 3.3.1, 3.3.1, 3.8, 3.9

[133] Yurii Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103, 2005. 1.1, 3.3.1, 3.3.1, 4.9.5

[134] Yurii Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical programming*, 120(1):221–259, 2009. 1.1, 3.1

[135] Thanh H Nguyen, Francesco M Delle Fave, Debarun Kar, Aravind S Lakshminarayanan, Amulya Yadav, Milind Tambe, Noa Agmon, Andrew J Plumptre, Margaret Driciru, Fred Wanyama, et al. Making the most of our regrets: Regret-based solutions to handle payoff uncertainty and elicitation in green security games. In *International Conference on Decision and Game Theory for Security*, pages 170–191. Springer, 2015. 7

[136] Thanh Hong Nguyen, Amulya Yadav, Bo An, Milind Tambe, and Craig Boutilier. Regret-based optimization and preference elicitation for Stackelberg security games with uncertainty. In *AAAI*, pages 756–762, 2014. 1.5, 7

[137] 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. 6.2.1

[138] Praveen Paruchuri, Jonathan P Pearce, Janusz Marecki, Milind Tambe, Fernando Ordonez, and Sarit Kraus. Playing games for security: An efficient exact algorithm for solving Bayesian Stackelberg games. In *Autonomous Agents and Multi-Agent Systems*, 2008. 1.5, 7, 7.1

[139] François Pays. An interior point approach to large games of incomplete information. In *AAAI Computer Poker Workshop*, 2014. 4.9

[140] Judea Pearl. Heuristic search theory: Survey of recent results. In *IJCAI*, volume 1, pages 554–562, 1981. 6, 6.4

[141] Judea Pearl. On the nature of pathology in game searching. *Artificial Intelligence*, 20(4): 427–453, 1983. 6

[142] Bezalel Peleg, Joachim Rosenmüller, and Peter Sudhölter. *The canonical extensive form of a game form: Symmetries*. Springer, 1999. 4.5.2

[143] Raghuram Ramanujan and Bart Selman. Trade-offs in sampling-based adversarial planning. In *ICAPS*, pages 202–209, 2011. 6

[144] Raghuram Ramanujan, Ashish Sabharwal, and Bart Selman. On adversarial search spaces and sampling-based planning. In *ICAPS*, volume 10, pages 242–245, 2010. 6

[145] R. Tyrell Rockafellar. *Convex Analysis*. Princeton University Press, 1970. 4.9.5

[146] I. Romanovskii. Reduction of a game with complete memory to a matrix game. *Soviet Mathematics*, 3, 1962. 3.2.2

[147] Ariel Rosenfeld and Sarit Kraus. When security games hit traffic: Optimal traffic enforcement under one sided uncertainty. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2017. 7

[148] Tuomas Sandholm. The state of solving large incomplete-information games, and application to poker. *AI Magazine*, 2010. Special issue on Algorithmic Game Theory. 1, 3.1, 3.11, 4.9.7, 6.5

[149] Tuomas Sandholm. Abstraction for solving large incomplete-information games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2015. Senior Member Track. 4.1

[150] Tuomas Sandholm. Steering evolution strategically: Computational game theory and op-

ponent exploitation for treatment planning, drug design, and synthetic biology. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2015. Senior Member Track. 1.4, 6, 6.3, 7

[151] 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. 8

[152] Tuomas Sandholm and Satinder Singh. Lossy stochastic game abstraction with bounds. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, 2012. 4, 4.1, 4.2, 4.5.1, 4.5.1, 4.5.2, 3, 6

[153] 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. 4

[154] Reinhard Selten. Reexamination of the perfectness concept for equilibrium points in extensive games. *International journal of game theory*, 1975. 5, 5.1.1

[155] 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, 2000. Springer-Verlag. 3.1, 4, 4.1

[156] Yoav Shoham and Kevin Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2009. 7.6

[157] 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)*, July 2005. 3, 3.8, 5.3

[158] Theodoros Spyridopoulos, G Karanikas, Theodore Tryfonas, and Georgios Oikonomou. A game theoretic defence framework against dos/ddos cyber attacks. *Computers & Security*, 38:39–50, 2013. 8

[159] 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. 4.5.2

[160] Milind Tambe. Security and game theory: algorithms, deployed systems, lessons learned, 2011. 1.5, 7, 7.1

[161] 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. 1.1, 3, 3.1, 3.9, 5.3

[162] Eric van Damme. A relation between perfect equilibria in extensive form games and proper equilibria in normal form games. *International Journal of Game Theory*, 1984. 5

[163] Heinrich von Stackelberg. *Marktform und Gleichgewicht*. Springer, Vienna, 1934. 7

[164] B. von Stengel, A. H. van den Elzen, and A. J. J. Talman. Computing normal form perfect equilibria for extensive two-person games. *Econometrica*, 70, 2002. 5

[165] Bernhard von Stengel. Efficient computation of behavior strategies. *Games and Economic*

Behavior, 14(2):220–246, 1996. 1, 1.1, 1.3, 3, 3.2.2, 3.4, 4.5, 4.9, 6.2.2, 6.3, 7.4, 7.6

[166] Bernhard von Stengel and Françoise Forges. Extensive-form correlated equilibrium: Definition and computational complexity. *Mathematics of Operations Research*, 33(4):1002–1022, 2008. 3.11

[167] Bernhard von Stengel and Shmuel Zamir. Leadership games with convex strategy sets. *Games and Economic Behavior*, 69(2):446–457, 2010. 7

[168] Kevin Waugh. Abstraction in large extensive games. Master's thesis, University of Alberta, 2009. 4, 4.1, 4.5.4

[169] 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. 1, 3.1

[170] 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)*, 2009. 4.6

[171] Kevin Waugh, Dustin Morrill, Drew Bagnell, and Michael Bowling. Solving games with functional regret estimation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2015. 4, 4.1

[172] Michael P Wellman and Achintya Prakash. Empirical game-theoretic analysis of an adaptive cyber-defense scenario (preliminary report). In *International Conference on Decision and Game Theory for Security*, pages 43–58. Springer, 2014. 8

[173] 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. 4.1

[174] Brandon Wilson, Inon Zuckerman, Austin Parker, and Dana S Nau. Improving local decisions in adversarial search. In *ECAI*, pages 840–845, 2012. 6.4

[175] Lin Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research*, 11(Oct):2543–2596, 2010. 3.10

[176] 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. In *Innovative Applications of Artificial Intelligence (IAAI) Conference*, 2012. 1.4, 6, 6.3, 7

[177] 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. 4.9.6

[178] Chao Zhang, Shahrzad Gholami, Debarun Kar, Arunesh Sinha, Manish Jain, Ripple Goyal, and Milind Tambe. Keeping pace with criminals: An extended study of designing patrol allocation against adaptive opportunistic criminals. *Games*, 7(3):15, 2016. 7

[179] 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. 1, 3, 3.1, 3.9, 4, 4.2, 4.2,

4.5, 4.5.2, 4.9, 6.2.2

Appendix

MIP for generating a perfect-recall abstraction by considering all levels at once

Motivated by the problems described in Section 4.5.2, 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 ex-ante variant of the bound given in Theorem 2. 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:

$$\max_{\delta_{i,j}, p_i, I_{i,j}} \sum_{i,j} \delta_{i,j} \tag{1}$$

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:

$$\min_{\delta_{i,j}, p_i, I_{i,j}} \sum_{z \in Z} \sum_{\hat{z} \in Z} \max_{i \in N} |V_i(z) - V_i(\hat{z})| \delta_{i,j} \tag{2}$$

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.

$$\sum_{j \in S_k} \delta_{i,j} \leq 1, \forall s_i \in S, \qquad p_i + \sum_{j \in S_k} \delta_{i,j} \geq 1, \forall s_i \in S, \qquad \delta_{j,i} - p_i \leq 0, \forall k \in \mathcal{H}, s_i, s_j \in S_k \tag{3}$$

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, satisfying the condition of Corollary 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.

$$\delta_{i,j} - \sum_{c_j \in C_j} s_{c_i,c_j} \leq 0, \forall s_i \in S \setminus Z, c_i \in C_i, \tag{4}$$

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 $\delta_{i,j} = 1$.

$$\delta_{i,j} - \delta_{p_i,p_j} \leq 0, \forall k \in \mathcal{H}, s_i, s_j \in S_k \tag{5}$$

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.

$$\delta_{i,j} - I_{i,j} \leq 0, \forall k \in \mathcal{H}, s_i, s_j \in S_k \tag{6}$$

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:

$$I_{i,j} + I_{i,l} - I_{j,l} \leq 1, \forall k \in \mathcal{H}, I_i, I_j, I_l \in \mathcal{I}_k \tag{7}$$

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.

$$p_l + I_{i,j} - \sum_{s_m \in I_j} \delta_{m,l} \leq 1, \forall I_i, I_j, s_l \in I_i \tag{8}$$

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.

$$\delta_{c_i,c_j} = 0, \forall s_i, s_j, c_i \in C_i, c_j \in C_j, \sigma(s_i, c_i) \neq \sigma(s_j, c_j) \tag{9}$$

$$\sum_{c_j \in C_j} \delta_{c_j,c_i} = \sum_{c_k \in C_i} \delta_{c_k,c_i} + 1, \forall s_i, s_j \in S \setminus Z, c_i \in C_i \tag{10}$$

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.

$$\sum_{z_i, z_j \in Z} \max_{i \in N} |V_i(z_i) - V_i(\hat{z_j})| \delta_{i,j} \leq \epsilon^R \tag{11}$$

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.

$$\max_{\delta_{i,j}, p_i, I_{i,j}} \sum_{i,j} \delta_{i,j} \geq |S| - K \tag{12}$$

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 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 $O(|Z|^2)$.

Games of ordered signals

Proof that the game in Figure 4.6 is a game of ordered signals

The game given in Figure 4.6 is only a signal tree. It can be coupled with any betting tree to form a simple poker game. We prove that it satisfies the conditions for any betting tree (by betting tree we mean any tree where the players take turns calling, checking, betting, raising, and folding, with each action leading to the usual outcomes of such actions in a poker game). We assume that a betting tree is played after the private cards are dealt, and another betting tree is played after the public card is dealt.

Proof. We go through the conditions for games of ordered signals as given by Gilpin and Sandholm [68].

1. The number of players is 2 which is finite.
2. The game gives only a signal tree that can be used to define winners, and thus works with any single-stage-game betting tree.
3. We only give a signal tree so this is not relevant.
4. The set of signals is {J1,J2,K1,K2}.
5. $\kappa = \{1\}, \gamma = \{1\}$.
6. The distribution over signals is the uniform distribution.
7. Any partial ordering with (K2,J2) ranked highest and (K2,J1) ranked lowest works.
8. Terminal nodes in the first stage game always map to *over* since we have only one stage game.
9. We need to check that the utility of players satisfy the ordering property. If either player has folded, the other player wins all money in the pot. If neither player folds, the signal tree is used to determine who wins the pot; 1 means that Player 1 wins, -1 means that Player 2 wins. First we check that utilities are ordered for Player 1. The definition only requires weak inequality, and Player 1 has the same payoff everywhere except when Player 1 has (K2,J1) and Player 2 has (K2,J2); since (K2,J1) is the lowest-ranked hand the ordering is satisfied for Player 1. Conversely, Player 2 loses everywhere except for (K2,J1,J2) where they win. Since (K2,J2) is ranked highest this satisfies the ordering.

□

Proof that merging K1 and K2 in Figure 4.6 is a an ordered game isomorphic abstraction transformation

Proof. First we note which nodes are not isomorphic. At the first level the J1 node is not isomorphic to any other subtree. At the second level the J1,J2 node is not isomorphic to any other subtree. At the third level the J1,J2,K2 leaf node is not isomorphic to any other leaf node.

We now consider the isomorphic nodes corresponding to private signals of K1 and K2 (bold and in red in Figure 4.6). Merging the private K1 and K2 signals into a single information set constitutes an ordered game isomorphic abstraction transformation: For each pair of K1 and K2 nodes at the two first levels, the subtrees are isomorphic. □