Politecnico di Torino

ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation
Doctoral Program in National Ph.D. in Artificial Intelligence (37^{th}cycle)

Efficient Learning in Team Games

A Coordination-Competition Dilemma

By

Luca Carminati

******

Supervisor(s):
Prof. Nicola Gatti, Supervisor
Prof. Marcello Restelli, Tutor

Doctoral Examination Committee:
Prof. Cesa-Bianchi Nicolò , Referee, University of Milan
Prof. Ferraioli Diodato, Referee, University of Salerno
Prof. Andrea Celli, Bocconi University
Prof. Stefano Moretti, Paris Dauphine University
Prof. Stefano Di Carlo, Politecnico di Torino

Politecnico di Torino
2025

Declaration

I hereby declare that, the contents and organization of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

Luca Carminati
2025

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

I would like to dedicate this dissertation to my loving family

Acknowledgements

My academic journey has been an adventure, one that many people have either willingly joined or been drawn into simply by being around me. I want to express my heartfelt appreciation to all of you.

First and foremost, I would like to thank the people most involved in my professional development, starting with my advisor, Nicola. Thanks for your insightful mentorship and support throughout those years and for the academic freedom you've always allowed me to have. I would also like to thank my close collaborators Brian, Federico, Gabriele, and Gianluca for our exchanges and good times together. A large share of what I've done would not have been possible without your contributions.

Secondly, I would like to thank the people who made the journey more enjoyable, both professionally and personally. While I lack the space to thank everyone individually, I am grateful to all of you. I start by thanking the *drappers* Alberto, Riccardo P., Riccardo Z., Gianluca, Alessio, Paolo, and all the other people from Airlab for the excellent time spent together inside and outside the lab  . I am deeply grateful to the *Mediterranean group* —Gabriele, Ioannis, Miguel, Margarida— and the many other friends I met during my time at CMU, and my old group of *survivors* from the M.Sc. Back in my hometown, I wish to thank my lifelong *ministri* friends Aspe, Facco, Teo, Paolo, Manu, and Mirko, as well as the *Cappuccinese* futsal team and climbing companions Gabriele and Andrei, for the joyful moments and camaraderie we've shared. Let's not forget all the people I bothered chatting with during those early morning commutes to Milan.

Finally, a very warm and special thanks goes to my mother, Rosanna, my father, Bruno, my sister, Laura, and my grandmother, Cia, for their unconditional support throughout those years ...and for their patience in hearing me when things were not going my way.

Abstract

A significant challenge in the field of Artificial Intelligence lies in addressing the sequential decision-making processes of multiple rational agents operating concurrently within a shared environment. *Algorithmic Game Theory* provides a compelling framework combining a rigorous game theoretical approach with a computational one; the main questions it addresses are to characterize equilibrium behavior and to compute optimal strategies efficiently. Significant results have been obtained in the well-understood two-player zero-sum setting with imperfect information. Experimentally effective techniques apply learning procedures to compute approximated optimal strategies, obtaining superhuman performance in large-scale games such as Poker or Diplomacy.

On the other hand, only a small amount of progress has been made in dealing with a larger number of players. This dissertation focuses on the *team* setting with imperfect information where two teams with multiple players have opposite utilities. This setting presents a coordination-competition dilemma. On the one hand, players desire to play simple strategies whose actions reveal their private information to team members; this would allow more effective cooperation. On the other hand, revealing private information increases the risk of exploitation by opponents. The tension between these opposite incentives is the common thread throughout the dissertation.

While previous approaches navigate the complexity with a progressive expansion of the correlated strategy space of each team, we reformulate the problem as an equivalent two-player zero-sum game between team coordinators. This formulation relies on novel efficient constructions. We can therefore adopt the scalable learning approaches developed for the two-player zero-sum setting, which can be adapted to better exploit the structure of our representations. We apply this approach to propose novel efficient representations for *adversarial team games*, where the teams are known before the start of the game, and, therefore, pre-play coordination allows team members to jointly sample their strategies. This representation also allows the definition of a *team-maxmargin* algorithm for solving subgames rooted at arbitrary points of the game. We also extend our analysis of team games to a novel setting called *hidden-role*

games. The particularity of this setting is that teams are sampled randomly at the start of the game, and players can usually communicate to coordinate and deceive the others, as seen in games like *Mafia* and *Avalon*. Experimental results show that our approaches enable faster and more scalable computation of equilibrium strategies.

Contents

List of Figures xi

List of Tables xii

1 Introduction 1

1.1 The Team Games Setting . . . 2

1.2 Original Contributions . . . 3

1.3 Dissertation Structure . . . 4

2 Preliminaries on Games and Learning 6

2.1 Games and Strategies Representations . . . 6

2.1.1 Extensive-Form Games . . . 6

2.1.2 Strategies . . . 9

2.1.3 Tree-Form Decision Problems . . . 12

2.2 Learning Equilibria in Games . . . 14

2.2.1 Nash Equilibrium and Max-Min Strategies . . . 14

2.2.2 Online Learning in Games . . . 15

2.3 Equivalence Across Games . . . 17

2.4 Adversarial Team Games . . . 17

3 Efficient Computation of Team and Imperfect-Recall Equilibria 22

3.1 Related Research . . . 24

3.2 Mixed Nash Computation Through Beliefs and Observations 26
3.2.1 Beliefs and Observations 26
3.2.2 Belief Game Construction 33
3.2.3 Strategic Equivalence 36
3.2.4 Worst-Case Dimension of the Belief Game 38
3.2.5 Regret Minimization on Team Games 40
3.3 DAG Decision Problems 41
3.4 DAG Decision Problems in Team Games 45
3.4.1 Size Analysis of the TB-DAG 47
3.4.2 Fixed-Parameter Hardness 48
3.4.3 Branching Factor Reduction 49
3.5 Complexity of Adversarial Team Games 50
3.5.1 Behavioral Max-Min Strategies 52
3.5.2 Mixed Nash Equilibria 54
3.6 Discussion 56
3.6.1 Public States vs Observations 56
3.6.2 Tree vs DAG Representation 61
3.6.3 Definition of Information Complexity and Comparison of Bounds . . 62
3.6.4 Connection With Tree Decomposition 62
3.6.5 Postprocessing Techniques That Can Be Used To Shrink the TB-DAG 64
3.7 Experiments 65
3.7.1 Experimental Setting 65
3.7.2 Discussion of the Results 68

4 Subgame Solving in Adversarial Team Games 70
4.1 Maxmargin 72
4.1.1 Value Computation in Games 72
4.1.2 Maxmargin Algorithm 73

4.2 Team-maxmargin 75
4.2.1 Linear Programming Formulation 75
4.2.2 Procedure for DAG Gadget 78
4.2.3 Safety of the Refinement 80
4.3 Column Generation for Sparser Solutions 80
4.4 Experimental Evaluation 82
4.4.1 Experimental Setting 82
4.4.2 Discussion of the Results 83

5 Hidden-Role Games: Equilibrium Concepts and Computation 89
5.1 Related Works 91
5.2 High-Level Contributions 92
5.2.1 Main Modeling Contributions 92
5.2.2 Main Computational Contributions 95
5.2.3 Experiments: *Avalon* 98
5.2.4 Examples 98
5.3 Equilibrium Concepts for Hidden-Role Games 100
5.3.1 Additional Notation 100
5.3.2 Hidden-Role Games 102
5.3.3 Models of Communication 103
5.3.4 Split Personalities 104
5.3.5 Equilibrium Notions 105
5.4 Computing Hidden-Role Equilibria 106
5.4.1 Computing Private-Communication Equilibria 106
5.4.2 Computing No/Public-Communication Equilibria 111
5.5 Worked Example 112
5.6 Properties of Hidden-role Equilibria 113
5.6.1 The Price of Hidden Roles 113

5.6.2 Order of Commitment and Duality Gap 115
5.7 Experimental Evaluation: *Avalon* 116

6 Conclusion and Future Research **118**

References **121**

Appendix A Additional Results from Chapter 4 **130**
A.1 Other Subgame Solving Techniques 130
A.1.1 Gifts 130
A.1.2 Resolving 131

Appendix B Additional Results from Chapter 5 **133**
B.1 Multi-Party Computation and Proof of Theorem 5.4.3 133
B.1.1 Secure MPC 133
B.1.2 Verifiable Secret Sharing 134
B.1.3 Simulating a Mediator 135
B.2 Connection to Communication Equilibria 136
B.3 Complexity Bounds and Proofs 138
B.4 The Game *Avalon* 146
B.4.1 Equivalence of Split-Personality Games 146
B.4.2 Abstractions 148
B.4.3 A Description of *Avalon* in Reduced Representation 153
B.4.4 Example of Optimal Play in *Avalon* 154

List of Figures

2.1 An example of an adversarial team game. 18

3.1 An example of team game and its corresponding connectivity graph for ▲. . . . 27
3.2 Another example of team game and its corresponding connectivity graph for ▲. 28
3.3 Belief game corresponding to Figure 3.1 35
3.4 Belief game corresponding to Figure 3.2 35
3.5 An example of imperfect-recall game derived from a team game whose rationale is described in the proof of Theorem 3.2.5. 38
3.6 An example of TB-DAG corresponding to Figure 3.1 46
3.7 Another example of TB-DAG corresponding to Figure 3.2 47
3.8 A game showing that public state-based approaches do not subsume inflation. 57
3.9 A pictorial representation of the proof of Proposition 3.6.2. 59
3.10 The counterexample for Proposition 3.6.3, for $C = 6$. 60

4.1 Value of the team's refined strategy as varying the refinement time limit in the K34, K36, K38, K312, K45, L3133 instances. 86
4.2 Value of the team's refined strategy as varying the refinement time limit in the L3143, L3151, L3153, L3223, L3523, D33 instances. 87
4.3 Value of the team's refined strategy as varying the refinement time limit in the D34, D62, T350, T3100 instances. 88

List of Tables

2.1 Translation table between terms commonly employed in the adversarial team games and two-player imperfect recall games. 20

3.1 Summary of most of the complexity results shown in Section 3.5. 50

3.2 Game sizes of the equivalent representations proposed in the paper on several standard parametric benchmark team games. 66

3.3 Runtime of our CFR-based algorithm using the team belief DAG form, compared to the prior state-of-the-art algorithms based on linear programming and column generation, on several standard parametric benchmark games. 67

4.1 Team's values against a best-responding opponent when playing the equilibrium strategy, blueprint, or refinement strategy with $\alpha = 5$, and corresponding algorithms running times. 84

4.2 Team's values against a best-responding opponent when playing the equilibrium strategy, blueprint, or refinement strategy with $\alpha = 1$, and corresponding algorithms running times. 84

5.1 Complexity results for computing hidden-role value with a constant number of players, for various assumptions about the adversary team and notions of communication. 112

5.2 Exact equilibrium values for 5- and 6-player *Avalon*. 116

B.1 Breakdown of equilibrium outcome probabilities for each variant of *Avalon*. . 154

Chapter 1

Introduction

Recent years have been characterized by multiple instances of *superhuman performances* of autonomous agents in decision-making problems: examples of this trend in tabletop or card games are [65, 9, 13, 81, 79, 73, 80], with interesting applied results obtained in [87, 6, 60]. These successes are characterized by a *learning* approach to the computation of optimal strategies within the framework of *Algorithmic Game Theory*, which provides a principled way of approaching learning against an adversary trying to exploit the agent. In this setting, two-player zero-sum games with imperfect information and the *Nash equilibrium* solution concept offer a clear framework to achieve robustness to exploiting behavior while being computable in polynomial time with respect to the size of the game instance. The learning approach consists in iteratively refining each player's strategy after repeated plays of the game. The main benefits consist of the updates being uncoupled between players and in the possibility of forgetting past plays because the strategy is sufficient. This allowed the development of scalable algorithms to optimize strategies, which have been employed in the settings mentioned earlier.

The dynamics change significantly whenever the two-player zero-sum imperfect-information setting is generalized to either include multiple players or allow for general-sum payoffs. In this case, theoretical results are often negative (PPAD-completeness of computing a Nash in the general case [23]), and practical positive results [11] depend on game-specific characteristics that make it similar to two-player zero-sum games.

1.1 The Team Games Setting

The main objective of this dissertation is to investigate *team games*, a specific setting of multiplayer games where *two teams* of players face each other in a zero-sum interaction with constrained communication. In other words, we lift the assumption of two players while maintaining a zero-sum interaction at a *two-team* level. In this case, coordination between team members is challenging due to imperfect information available to only some players. The team, therefore, cannot act as a single player; instead, it is crucial to find means of communicating this information to team members while satisfying the constraints on communication imposed by the game rules. Coordination can be achieved: i) by having team members reveal as much private information as possible through communication; or ii) by playing simple strategies whose observable actions leak this information. However, private communication is often banned during play and the presence of an exploiting adversary strongly punishes revealing information too openly. This establishes a *coordination-competition dilemma*.

The difference between the two-player and the two-team settings can be made explicit by considering an equivalent point of view where an *imperfect-recall* coordinator for each team plays in place of each team's members. The coordinator is imperfect-recall because it must pick actions while accessing only the information available to the active team member, in order not to break the constraints on communication. Under this view, team games are a generalization of two-player zero-sum games where the two players (*i.e.* the coordinators of each team) have imperfect recall. This connection is made formal in Sections 2.1 and 2.4.

Team games are an interesting benchmark that involves a mix of cooperation and competition among the agents in an environment. Any success in these mixed settings would bring the research closer to extend the superhuman performance seen in two-player zero-sum games to the complex real-world settings of negotiation, team cooperation and swarm techniques. Team games have therefore attracted the interest of different researchers sharing the objective of generalizing the results seen in the two-player zero-sum setting. Interesting applications include card games such as Bridge and Dou Dizhu [56], real-world settings such as patrolling [3], network security, or military applications.

The study of team games is not entirely novel. An important area of previous works focuses on *adversarial team games*, in which the teams are known and fixed in advance. They were introduced by von Stengel and Koller [90] in the normal-form setting and extended by Celli and Gatti [18] to the extensive-form setting. The available communication among team members influences the coordination capabilities and thus determines the solution concept. The most studied solution concept is the *team-maxmin with correlation* (TMECor), where team

members communicate and correlate their strategies before the start of the game. However, they have no means of communication during the game; they only play and observe information according to the game's rules. Traditional approaches focus on efficiently representing the correlated strategy space of the members. TMECor is a solution concept that is different from the *team-maxmin* (TME), which assumes no communication capabilities, and the *team-maxmin with communication* (TMECom), which instead assumes the availability of communication channels anytime. A more detailed review can be found in Section 3.1.

Other lines of research are related to the team setting but adopt different frameworks. This is the case with multi-agent reinforcement learning [102], mean-field games [55] and more classical algorithmic game theory [70]. This dissertation maintains the classical extensive-form games framework as the previous works on superhuman AI in games.

The main challenge addressed by previous approaches is that full strategy representation for the team is exponentially large with respect to the size of the games, which poses critical memory complexity challenges when implementing practical algorithms. Traditional approaches are based on custom optimization procedures that iteratively explore the space of correlated strategies. An unanswered challenge in the literature is whether a learning-based approach can be used to approximate optimal strategies in a team setting; this is a nontrivial question because they usually need a full representation of the strategy space. We answer this question positively: it is possible to efficiently represent team games as two-player zero-sum games in the case of both ex-ante correlation and hidden-role games. As we will show, doing so enables the use of state-of-the-art learning techniques to approximate optimal play. These contributions open a novel research direction, based on generalizing techniques from the two-player to the two-team setting. We believe these contributions can be pivotal in the development of superhuman AIs that play in team settings.

1.2 Original Contributions

We briefly summarize the main original contributions of this dissertation. Our focus is characterizing rational behaviors and learning-based methods to compute those behaviors in extensive-form games with imperfect information and a team structure. In particular, we develop:

- **A novel representation and algorithmic framework for the computation of TMECors** in adversarial team games and timeable imperfect-recall two-player zero-sum games.
- **Novel complexity results** on the problem of computing a TMECor in a team-vs-team setting.

- **An experimental evaluation** of learning techniques on the novel representation of the correlated team strategy space.
- **An efficient subgame solving procedure** tailored to adversarial team games and imperfect-recall games.
- **A novel model of agent rationality in hidden-team setting**, which are called *hidden-role* games.
- **A comprehensive study of the computational complexity** of computing equilibria in hidden-role games.
- **A large-scale, exact computation of equilibrium values** in Avalon with communication and **study** of the computed equilibrium.

We remark that in light of the equivalence of team games with timeable imperfect-recall two-player zero-sum games, our results immediately generalize to this setting as well.

1.3 Dissertation Structure

The main contents of the dissertation are structured as follows.

Chapter 2 defines the preliminary notions at the base of the work presented. The chapter focuses on the formal representations of games and strategies, on the definition of solution concepts for two-player and two-team settings and on the use of learning algorithms to approximate equilibria.

Chapter 3 introduces a novel representation of the strategy space of a coordinated team. This is achieved by explicitly reasoning on *beliefs* of the team and the construction of an equivalent representation for the decision process faced by the team. The proposed solution is theoretically analyzed and learning algorithms to approximate optimal strategies are developed. Empirical evaluation shows state-of-the-art performance on most of the game instances customarily employed.

Chapter 4 enhances the representation introduced in Chapter 3 with a *subgame solving* procedure, inspired by similar algorithms in the two-player zero-sum setting[64, 8]. Subgame solving is a learning-based, iterative, refinement procedure of a precomputed strategy limited to a specific part of the game that is reached during play. In team games, this algorithm can be adapted by carefully managing the reach probabilities of the teams to various parts of the game.

Experimental results show encouraging refinement capabilities, analogous to the performances in two-player zero-sum games.

Chapter 5 steps back from considering a known, fixed, coordinated team of agents and instead considers *hidden-role* games, where players are randomly assigned to teams at the start of the game. In this setting, communication plays a central strategic role since it allows players to discover and negotiate with possible teammates but also allows for deception and betrayal. We formally define this novel game setting, analyze the complexity of computing optimal behavior under different assumptions, and then apply our solution concept to large-scale instances of the board game *Avalon*. Exact equilibria are computed and analyzed.

Chapter 6 summarizes the main contributions and presents interesting future directions of the work presented in the dissertation.

Chapter 2

Preliminaries on Games and Learning

In this chapter, the main game-theoretical notions needed for the remainder of the work are presented. In particular, Section 2.1 introduces the concepts of *game*, *strategy*, *equilibrium* with a focus on classical results regarding strategy representation. Those concepts play an important role when extending the classical results from the setting of *perfect-recall* games to *imperfect-recall* and *adversarial team games*. Section 2.2 introduces *max-min* optimization problems and highlights their relation to the computation of optimal equilibria. It also presents how *online learning* techniques can be applied to approximate those equilibria whenever the strategy spaces of the players are convex polytopes. Section 2.3 rigorously defines what it means for two games to be *strategically equivalent*. Equivalence across games allows one to find an equilibrium by computing one in an equivalent game and then converting it. Section 2.4 defines *adversarial team games*, which are the class of games addressed by this work. Importantly, the equivalence between adversarial team games and two-player imperfect-recall games is established.

This chapter's content is adapted with minor modifications from a forthcoming first-author journal article currently under review at the *Journal of AI (JAI)*.

2.1 Games and Strategies Representations

2.1.1 Extensive-Form Games

A classical representation for sequential games is the one of *extensive-form games* (EFG). In the following, we introduce the notation adopted to describe their different components. A crucial

aspect is that players have imperfect information about the state of the game during gameplay. That is, they may not know the exact point of the game that has been reached; instead, they have to consider a set of possible candidates that fit the information they know. These sets are called *information sets* (in short *infosets*).

Definition 2.1.1 (Extensive-form game)**.** An *extensive-form game* G is a tuple $(\mathcal{N}, \mathcal{H}, \mathcal{Z}, \mathcal{A}, \mathcal{I}, \boldsymbol{p}, u)$ that describes a sequential game, where

- $\mathcal{N}$ is the set of *players* in the game. A special symbol $\mathsf{N} \in \mathcal{N}$ is used to indicate *nature* player (also called *chance*). Nature encodes all the stochasticity of the environment known a priori, and therefore it picks actions according to a fixed probability distribution. We commonly use a letter i to indicate any player $i \in \mathcal{N}$, while we will use $-i$ to indicate $\mathcal{N} \setminus i$.

- $\mathcal{H}$ is a rooted tree of *nodes* (also called *histories*) of the game. There is exactly one acting player picking an action at every node, apart from the terminal nodes $\mathcal{Z} \subset \mathcal{H}$, which represent possible game endings after which nobody plays anymore. $\mathcal{H} \setminus \mathcal{Z}$ is partitioned into sets $\{\mathcal{H}_i\}_{i \in \mathcal{N}}$ representing nodes in which player i is the player choosing an action. We use h to indicate a generic node $h \in \mathcal{H}$, z to indicate a generic terminal node $z \in \mathcal{Z}$, and $\varnothing$ to indicate the root node of the game tree. Moreover, we use the symbol $\mathcal{H}_{-i}$ to indicate the nodes at which player i is *inactive*, that is, nodes in which other players $j \in \mathcal{N} \setminus i$ are playing.

- $\mathcal{A}$ is the set of actions in the game. A non-empty set of actions $A_h \subseteq \mathcal{A}$ is available to i in any $h \in \mathcal{H}_i$ node. By assumption, each player knows the actions available at each node. Playing an action a at h leads to a node h' which we denote as $ha \equiv h'$.

- $\mathcal{I}$ is the set of all information sets. An information set $I \subseteq \mathcal{H}_i$ is a set of nodes that are indistinguishable by the active player i. It is assumed that players know the actions available to them whenever they are active, so it is required that for each $I \in \mathcal{I}$, all $h \in I$ have identical A_h. This common action set is denoted A_I. $\mathcal{I}$ is partitioned into $\{\mathcal{I}_i\}_{i \in \mathcal{N} \setminus \mathsf{N}}$ which are the sets of information sets of a player i. Exactly one information set $I_h \in \mathcal{I}_i$ contains every node $h \in \mathcal{H}_i$ where each player is active.

- $\boldsymbol{p} \in [0,1]^{\mathcal{Z}}$ is a vector encoding nature's fixed strategy. $\boldsymbol{p}[z]$ denotes the probability that nature plays all the actions leading to $z \in \mathcal{Z}$. The overloaded notation $\boldsymbol{p}[h] \coloneqq \sum_{z \succeq h} \boldsymbol{p}[z]$ indicates nature's reach probability at any node $h \in \mathcal{H}$, defined as the sum of the reach probability of each terminal having h as ancestor.

- $u_i : \mathcal{Z} \to \mathbb{R}$ is the *utility* function, associating each terminal node to the payoff player $i \in \mathcal{N} \setminus \mathsf{N}$ receives when the game reaches such terminal node.[1]

In the rest of this paper, we will use the following shorthand to indicate structural relationships on a game tree.

- The symbol $|\cdot|$ denotes the length of a node h, that is, the depth of the node from the root. The root node $\varnothing$ has length 0 and for each $h \in \mathcal{H} \setminus \mathcal{Z}$ and $a \in \mathcal{A}_h$, it holds that $|ha| = |h| + 1$. The *depth* of G is the length of the longest node in G.
- a set of nodes S (usually S will be an information set) precedes another set S', denoted $S \preceq S'$, if there are nodes $h \in S, h' \in S'$ such that h' is a descendant of h (or $h' = h$). If one of S and S' is a singleton, we omit the braces; for example, $h \preceq I$ means there is a path from h to some node $h' \in I$.

In the following, we analyze the information structure of extensive-form games. This is achieved through the concept of *sequences*. Sequences characterize the structure of the information revealed to a player at any node of the game.

Definition 2.1.2 (Sequence)**.** The *sequence* $\sigma_i(h)$ of player i at node h is the ordered list of pairs of infosets reached and actions played, by player i, on the $\varnothing \to h$ path, excluding the information set of h. A special sequence $\varnothing$ corresponding to an empty list is also considered for each player.

The set containing all sequences of all players and of a single player are respectively denoted Σ and Σ_i, with $\Sigma \equiv \bigcup_{i \in \mathcal{N} \setminus \mathsf{N}} \Sigma_i$. We use the notation $\sigma_i + (I, a)$ to indicate the sequence obtained by extending σ_i by adding an (infoset, action) pair (I, a).

Other important notions are those of *public state* and *subgame*. Intuitively, a public state is a set of nodes such that all players can detect whether a node in the set has been traversed or not. This is translated on a closure property on the information sets, which won't include nodes not traversing the public state from the public state on. A subgame is the part of the game rooted at a public state and including all the following nodes. This cut of the game does not separate nodes on different information sets.

Definition 2.1.3 (Public State)**.** A *public state* $P \subset \mathcal{H}$ is a set of same-depth nodes such that $P \preceq I$ implies $P \preceq h$ for all $h \in I, I \in \mathcal{I}$.

[1] Whenever the scale of the utilities matters for complexity results, we assume a utility function that is *normalized* so that $u_i : \mathcal{Z} \to [-1, 1]$. The normalization of the utility functions can be done WLOG in any game.

Definition 2.1.4 (Subgame)**.** A *subgame* S_P rooted at P of a game G is defined by reducing the node set of G to $\{h \in \mathcal{H} \text{ s.t. } h \succeq P\}$.

We remark that, in general, a subgame is not an extensive-form game. All nodes $h \in P$ have no parent in the subgame, and therefore, there may be no unique root of the game.

This paper focuses on extensive-form games that satisfy some regularity properties.

Definition 2.1.5 (Timeability)**.** An extensive-form game G is *timeable* if any path from the root to any node in the same infoset has the same length (equivalently, all nodes belonging to the same infoset have the same depth). Formally, G is timeable if for every infoset $I \in \mathcal{I}$ and every $h, h' \in I$, we have $|h| = |h'|$.

As an assumption, we will consider timeable games throughout the paper. Intuitively, this corresponds to assuming that all players know the number of actions played in the game anytime they play.

Definition 2.1.6 (Perfect recallness)**.** A player i has *perfect recall* if all nodes in the same infoset of that player have the same sequence. Formally, player i has perfect recall if for every infoset $I \in \mathcal{I}_i$ and every $h, h' \in I$, we have $\sigma_i(h) = \sigma_i(h')$. The common sequence is denoted σ_I.

When a player has perfect recall, its nonempty sequences are uniquely identified by the last infoset-action pair contained within them. Thus, when player i has perfect recall, we will use $Ia \in \Sigma_i$ to indicate the sequence of player i that ends with playing action a at infoset I. Intuitively, assuming that all the players have perfect recall corresponds to assuming that each of them never forgets information they previously acquired during the game's unfolding. A game is said to be perfect recall when all players in the game have perfect recall. We say the player/game is *imperfect recall* whenever the perfect recall condition does not hold.

We remark that the timeability assumption excludes *absentmindedness*. That is, players never have nodes on the same path belonging to an identical information set. Intuitively, knowing the number of actions played in the game allows the player to distinguish nodes on the same path.

2.1.2 Strategies

Extensive-form games represent the sequential interaction among players that are required to take an action given a specific information state. The description of a possibly stochastic

behavior of a player in a game is called *strategy*. The *strategy space* of a game is the set of all possible strategies that players can play. It is denoted by $\mathcal{X}$, while the strategy space of a player $i \in \mathcal{N}$ is denoted by $\mathcal{X}_i$.

Useful strategies in games are found by optimizing an objective function over the strategy space. Therefore, it is convenient to represent the strategy space in a way that allows efficient optimization procedures. Common optimization objectives employ *reach probabilities*, defined as the probability that a player plays to reach a terminal node z. *Realization-form* representation of strategies addresses this need by representing the probability of a player playing all actions leading to each terminal.

Definition 2.1.7 (Realization-form strategy)**.** A *realization-form strategy* is a $[0,1]^{\mathcal{Z}}$ vector, where each terminal z is associated with the probability that a player plays all the actions on the $\varnothing \to z$ path.

It is to be noted that, in general, a realization-form strategy is not a probability distribution over $\mathcal{Z}$.

Realization-form strategies provide a general representation of a strategy through its contributions to reaching terminal nodes. However, they characterize the players' behavior only for their effects on the terminal nodes, leaving unspecified the actual behavior creating such a contribution on the terminals. This behavior can be characterized according to different definitions, which can then be mapped to their compact, definition-agnostic realization form. The main notions of strategy that will be described in the following are *pure strategy*, *mixed strategy*, *behavioral strategy*, and *sequence-form strategy*.

A *pure strategy* is a deterministic selection of one action at each infoset. We consider only *reduced* pure strategies, where actions are undefined ($\perp$) in any infoset that cannot be reached considering previous actions taken by the player.

Definition 2.1.8 (Pure strategy)**.** A pure strategy $\boldsymbol{\pi}$ for player i is a vector indexed by information sets where $\boldsymbol{\pi}[I] \neq \perp$ if it exists $h \in I$ such that for all $(I', a') \in \sigma_i(h)$ we have $\boldsymbol{\pi}[I'] = a'$. The realization form of a pure strategy is the vector $\boldsymbol{x} \in \{0,1\}^{\mathcal{Z}}$ where $x[z] = 1$ if and only if the strategy prescribes all player actions on the path $\varnothing \to z$.

We use Π_i to indicate the set of pure strategies in realization form of player i. This set can be formally defined as:

$$\begin{aligned}
\boldsymbol{\pi} \in \mathcal{P}_i \subseteq \bigtimes_{I \in \mathcal{I}_i} \mathcal{A}_I \cup \{\bot\} & \qquad \text{Pure strategies domain} \\
\rho(\boldsymbol{\pi}, h) = \begin{cases} 1 & \text{if } \forall (I,a) \in \sigma_i(h)\ \boldsymbol{\pi}[I] = a \\ 0 & \text{otherwise} \end{cases} & \qquad \text{Reach at a history } h \\
\boldsymbol{\pi}[I] = \bot \iff \forall h \in I\ \rho(\boldsymbol{\pi}, h) = 0 & \qquad \text{Constraint on } \bot \\
\Pi_i \coloneqq \left\{ \boldsymbol{x} \in \{0,1\}^{\mathcal{Z}} : \exists \boldsymbol{\pi} \in \mathcal{P}_i \text{ s.t. } \boldsymbol{x}[z] = \rho(\boldsymbol{\pi}, z) \right\} & \qquad \text{Pure strategies set in realization form}
\end{aligned}$$

The definition of reduced pure strategies allows us to rigorously define the notions of *played* sequence and *reached* infoset.

Definition 2.1.9 (Played sequence)**.** A sequence $\sigma_i(h)$ is *played* by pure strategy $\boldsymbol{\pi}$ if for all $(I', a') \in \sigma_i(h)$ we have $\boldsymbol{\pi}[I'] = a'$.

Definition 2.1.10 (Reached node)**.** A node $h \in \mathcal{H}_i$ is *reached* or *played to* by $\boldsymbol{\pi}$ if $\sigma_i(h)$ is played. By extension, an infoset I is said reached if any $h \in I$ is reached.

The set of pure strategies $\Pi \equiv \{\Pi_i\}_{i \in \mathcal{N} \setminus \mathsf{N}}$ is finite, and so is the set of possible realization-form containing all possible realization form-vectors obtained by enumerating all pure strategies and considering their reach.

Definition 2.1.11 (Mixed strategy)**.** A *mixed strategy* is a distribution over pure strategies. The realization form of a mixed strategy is the vector $\boldsymbol{x} \in [0,1]^{\mathcal{Z}}$ obtained through the convex combination of pure strategies.
We use $\mathcal{X}_i$ to indicate the set of mixed strategies in realization form. Formally, we have:

$$\mathcal{X}_i \coloneqq \operatorname{co} \Pi_i$$

Definition 2.1.12 (Behavioral strategy)**.** A *behavioral strategy* describes a player's behavior as independent local distributions over the actions available at each information set. The realization form of a behavioral strategy is obtained by multiplying the probability of picking each player's action on the path $\varnothing \to z$.
We use $\hat{\mathcal{B}}_i$ to indicate the set of behavioral strategies, while $\hat{\mathcal{X}}_i$ in realization form. This set can be formally defined as:

$$\begin{aligned}
\hat{\mathcal{B}}_i \coloneqq \bigtimes_{I \in I_i} \Delta^{A_I} & \qquad \text{Behavioral strategies set} \\
\rho(\boldsymbol{b}, h) = \textstyle\prod_{(I,a) \in \sigma_i(h)} \boldsymbol{b}(I)[a] & \qquad \text{Reach at a history } h \\
\hat{\mathcal{X}}_i \coloneqq \left\{ \hat{\boldsymbol{x}} \in [0,1]^{\mathcal{Z}} : \exists \boldsymbol{b} \in \hat{\mathcal{B}}_i \text{ s.t. } \hat{\boldsymbol{x}}[z] = \rho(\boldsymbol{b}, z) \right\} & \qquad \text{Behavioral strategies set in realization form}
\end{aligned}$$

By definition, the set of mixed strategies in realization form is a convex, compact set, with pure strategies (in realization form) as vertices. In contrast, the set of behavioral strategies in realization form is, in general, non-convex since not all combinations of mixed strategies can be formulated as a combination of probabilities at each information set.

A (mixed) *strategy profile* is a tuple $\boldsymbol{x} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_n) \in \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$ consisting of one mixed strategy for each player. The *expected utility* of player i is $u_i(\boldsymbol{x}) := \mathbb{E}_{z \sim \boldsymbol{x}}\, u_i(z)$, where $z \sim \boldsymbol{x}$ denotes sampling a terminal node z by following the profile $\boldsymbol{x}$, that is, $u_i(\boldsymbol{x}) := \sum_{z \in \mathcal{Z}} u_i(z) \boldsymbol{p}[z] \prod_{j \in \mathcal{N} \setminus \mathsf{N}} \boldsymbol{x}_j[z]$. A crucial property of the realization form, which we will use repeatedly, is that the utility functions $u_i : \mathcal{X}_1 \times \dots \mathcal{X}_n \to \mathbb{R}$ are linear in each $\boldsymbol{x}_i$.

A classical question is to characterize the strategy spaces given their definitions and find the conditions guaranteeing the equivalence of the different notions. In particular, the local decomposability at each infoset offered by behavioral strategies offers efficiency advantages when describing strategies over an extensive-form game. Luckily, the only condition required for their equivalence is that the considered game is perfect-recall, as stated in the following classical theorem.

Theorem 2.1.1 (Kuhn's theorem, [53])**.** *In every extensive game with perfect recall, for each mixed strategy, there is a behavior strategy that has the same realization form. That is,* $\hat{\mathcal{X}}_i = \mathcal{X}_i$.

2.1.3 Tree-Form Decision Problems

This subsection describes *tree-form decision problems*, originally introduced by Farina et al. [27]. They are highly efficient representations of the strategy space of a specific player in the game. In games with perfect-recall, tree-form decision problems lead to the *sequence-form* representation of the strategy space, first noted by Romanovskii [76] and later rediscovered by von Stengel [88]. The tree-from decision problem will later be generalized to DAG structures arising from imperfect-recall games.

Definition 2.1.13 (Tree-form decision problem)**.** A *tree-form decision problem* is a rooted tree of nodes representing an interaction between a *player* and an *environment*. Each node s is one of the following types:

- *decision points* $s \in \mathcal{D}$, at which the player takes an action;
- *observation points* $s \in \mathcal{S}$, at which the environment selects the next state.

We assume WLOG that the root node ($\varnothing$) and all terminal nodes are observation points and that decision points and observation points alternate. At a decision point s, the set of legal

actions is denoted $\mathcal{A}_s$. Each action $a \in \mathcal{A}_s$ leads to a different observation node, denoted sa. The observation node parent of s is denoted p_s. For notational simplicity, when $\boldsymbol{x} \in \mathbb{R}^{\mathcal{S}}$ is any vector indexed by observation points and s is a decision point, we use $\boldsymbol{x}[s*] \in \mathbb{R}^{\mathcal{A}_s}$ to denote the subvector of $\boldsymbol{x}$ indexed only by the children of s.

Similar definitions of strategy notions as those from extensive-form games hold in tree-form decision problems. A *pure strategy* is an assignment of one action to each decision node. Pure strategies are associated with vectors in $\boldsymbol{x} \in \{0, 1\}^{\mathcal{S}}$ in the following way: for a node $s \in \mathcal{S}$, $\boldsymbol{x}[s] = 1$ if and only if the player plays every action on the path from the root to s. We will call such vectors *tree-form (pure) strategies*. A *tree-form mixed strategy* $\boldsymbol{x} \in \mathcal{Q}$ is a convex combination of tree-form pure strategies.

The set of tree-form mixed strategies has a natural description as a set of linear inequalities. Namely:

Proposition 2.1.2. *$\mathcal{Q}$ is the set of nonnegative vectors $\boldsymbol{q} \in \mathbb{R}^{\mathcal{S}}$ satisfying the following constraints:*

$$\begin{cases} \boldsymbol{q}[\varnothing] = 1 & \text{(2.1a)} \\ \boldsymbol{q}[p_s] = \sum_{a \in \mathcal{A}_s} \boldsymbol{q}[sa] \quad \textit{for all } s \in \mathcal{D}. & \text{(2.1b)} \end{cases}$$

The tree-form constraints are *probability flow* constraints: the probability of reaching a node starts at 1 at the root (2.1a), and is split at each decision point on the available actions (2.1b).

There is an important connection with extensive-form games whenever a player i has perfect recall. In this case, a tree-form decision problem can be extracted from the game by considering the point of view of player i. We have that information states are the decision points, and every contribution to the game the player observes is associated with observation points. More formally, this corresponds to having one observation point per sequence, where the infosets reached when playing that sequence are the possible observations. We obtain a tree-form decision problem where $\mathcal{D} = \mathcal{I}_i$ and $\mathcal{S} = \Sigma_i$. In this case, tree-form strategies correspond to the so-called *sequence-form* strategies, and there is a natural correspondence between the tree-form strategies and the realization-form strategies.

Definition 2.1.14 (Sequence-form strategies)**.** In a game with perfect recall, consider a player i and let $\mathcal{Q}$ defined according to Proposition 2.1.2, with $\mathcal{D} = \mathcal{I}_i$ and $\mathcal{S} = \Sigma_i$. A *sequence-form strategy* $\boldsymbol{q} \in \mathcal{Q}$ of player i associates to each $\sigma \in \Sigma_i$ the probability that the player plays all actions in σ. The realization-form strategy $\boldsymbol{x}_i \in \mathcal{X}_i$ of a sequence-form strategy $\boldsymbol{q}$ is

$\boldsymbol{x}_i[z] = \boldsymbol{q}[\sigma_i(z)]$ since that is the probability that the player plays all actions on the path to a terminal node $z \in \mathcal{Z}$.

This is a much more compact representation than the naive one offered by the definition of mixed strategy. Mixed strategies maintain a probability distribution over an exponential number of pure strategies. On the other hand, sequence-form strategies have one element per sequence, which corresponds to an infoset-action pair, similar to behavioral strategies.

2.2 Learning Equilibria in Games

2.2.1 Nash Equilibrium and Max-Min Strategies

After deciding the most suitable representation for strategies in a given problem, an objective to be optimized to find the optimal strategy is required. This is done through *solution concepts*, which describe the characteristics that make a strategy optimal in a context. One solution concept that is relevant for the paper is the *Nash equilibrium*:

Definition 2.2.1 (Nash equilibrium)**.** A *Nash equilibrium* is a mixed strategy profile x such that no player has a profitable deviation. That is, for every player i and every $\boldsymbol{x}'_i \in \mathcal{X}_i$, we have $u_i(\boldsymbol{x}'_i, \boldsymbol{x}_{-i}) \leq u_i(\boldsymbol{x})$.

The most studied type of extensive-form games are *two-player zero-sum games*. In this class of games, two players face each other in a setting where a payoff for anyone is a cost for the other. In this section, we define two-player zero-sum games and explore some basic solution concepts for them.

Definition 2.2.2 (Two-player zero-sum game)**.** An extensive-form game is two-player zero-sum if

- the player set contains nature and two players, called ▲ and ▼, namely $\mathcal{N} = \{\mathsf{N}, ▲, ▼\}$, and
- the utilities of the two players are opposite, namely $u_▲ = -u_▼$.

In this case, we will use a single utility function $u = u_▲ = -u_▼$ and use $\mathcal{X}$ and $\mathcal{Y}$ to represent the mixed strategy sets of the two players. In two-player zero-sum games, Nash equilibria $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{X} \times \mathcal{Y}$ are the saddle-point solutions to the bilinear saddle-point problem

$$\max_{\boldsymbol{x}\in\mathcal{X}} \min_{\boldsymbol{y}\in\mathcal{Y}} \boldsymbol{x}^\top \boldsymbol{U} \boldsymbol{y} = \min_{\boldsymbol{y}\in\mathcal{Y}} \max_{\boldsymbol{x}\in\mathcal{X}} \boldsymbol{x}^\top \boldsymbol{U} \boldsymbol{y} \tag{2.2}$$

Algorithm CFR Counterfactual regret minimization on tree-form decision problems $\mathcal{Q}$. For each decision point s, $\mathcal{R}_s$ is a regret minimizer on $\Delta(\mathcal{A}_s)$.

procedure NEXTSTRATEGY
  $\boldsymbol{x}^t[\varnothing] \leftarrow 1$
  for each decision point s, in top-down order **do**
    $\boldsymbol{r}_s^t \leftarrow \mathcal{R}_s$.NEXTSTRATEGY()
    $\boldsymbol{x}^t[s*] \leftarrow \boldsymbol{x}^t[p_s]\boldsymbol{r}_s^t$
  return $\boldsymbol{x}^t$
procedure OBSERVEUTILITY($\boldsymbol{u}^t$)
  $\boldsymbol{v}^t \leftarrow \boldsymbol{u}^t$
  for each decision point s, in bottom-up order **do**
    $\mathcal{R}_s$.OBSERVEUTILITY($\boldsymbol{v}^t[s*]$)
    $\boldsymbol{v}^t[p_s] \leftarrow \boldsymbol{v}^t[p_s] + \langle \boldsymbol{r}_s^t, \boldsymbol{v}^t[s*] \rangle$
  $t \leftarrow t + 1$

where $\boldsymbol{U} = \text{diag}(\{u(z)\boldsymbol{p}[z]\}_{z \in \mathcal{Z}})$ so that $\boldsymbol{x}^\top \boldsymbol{U} \boldsymbol{y} = \mathbb{E}_{z \sim (\boldsymbol{x}, \boldsymbol{y})}\, u(z)$. The *Nash value* u^* of a two-player zero-sum game is the value for ▲ in any equilibrium. As discussed in the previous subsection, in games with *perfect recall*, mixed and behavioral strategies are equivalent by Theorem 2.1.1. In games without perfect recall, the above definition of Nash equilibrium is still valid; however, it will also be useful to define the behavioral max-min strategy:

Definition 2.2.3 (Behavioral max-min strategy)**.** In a two-player zero-sum game, a *behavioral max-min strategy* $\boldsymbol{x} \in \hat{\mathcal{X}}$ is a solution to the optimization problem

$$\max_{\boldsymbol{x} \in \hat{\mathcal{X}}} \min_{\boldsymbol{y} \in \hat{\mathcal{Y}}} \boldsymbol{x}^\top \boldsymbol{U} \boldsymbol{y}.$$

The *behavioral max-min value* is the optimal value of the above problem. Since $\hat{\mathcal{X}}$ and $\hat{\mathcal{Y}}$ are not necessarily convex sets, the minimax theorem does not apply, so the maximization and minimization can not necessarily be swapped. Therefore, the behavioral max-min strategy is not an *equilibrium*. Further, in games with imperfect recall, the tree-form decision problem is not a valid representation of the set of realization-form strategies. Therefore, we will need different techniques to tackle such games.

2.2.2 Online Learning in Games

In this work, we will leverage the *online convex optimization* [107] (OCO) framework for modeling repeated interactions between a player and an arbitrary environment. According to this model, the player iteratively interacts with the environment by selecting at each timestep

t a strategy $\boldsymbol{x}^t$ from a convex, compact set $\mathcal{Q} \subseteq \mathbb{R}^m$ and observes a linear utility function $u^t : \mathcal{Q} \to [-1, +1]$ chosen (possibly adversarially) by the environment. Since utility functions are linear, we will represent them as vectors $\boldsymbol{u}^t \in \mathbb{R}^m$. Considering an interaction repeated over T steps, the objective of the player is to minimize the *cumulative regret*, which is defined as

$$R_{\mathcal{Q}}^T := \max_{\boldsymbol{x} \in \mathcal{Q}} \sum_{t=1}^{T} \left\langle \boldsymbol{u}^t, \boldsymbol{x} - \boldsymbol{x}^t \right\rangle$$

We say that an algorithm is a *regret minimizer* if it guarantees that the cumulative regret grows sublinearly with T, that is, $R^T = o(T)$.

Regret minimization and equilibrium computation in two-player zero-sum games are tightly connected. In particular, consider the scenario of a repeated two-player zero-sum game in which ▲ and ▼ sequentially interact for T rounds. Both players use regret minimizers over their strategy sets $\mathcal{X}$ and $\mathcal{Y}$, respectively. Let, at each timestep t, the utility vectors observed by the two players be $\boldsymbol{u}_{▲}^t = \boldsymbol{U}\boldsymbol{y}^t$ for ▲ and $\boldsymbol{u}_{▼}^t = -\boldsymbol{U}\boldsymbol{x}^t$ for ▼. Then, it is known that the equilibrium gap computed w.r.t. the average strategies $\bar{\boldsymbol{x}} := \sum_{t=1}^T \boldsymbol{x}^t/T$, $\bar{\boldsymbol{y}} := \sum_{t=1}^T \boldsymbol{y}^t/T$ is bounded by the cumulative regrets $R_{▲}^T$, $R_{▼}^T$ experienced by the players ▲ and ▼, respectively:

$$\max_{\boldsymbol{x} \in \mathcal{X}} \boldsymbol{x}^\top \boldsymbol{U} \bar{\boldsymbol{y}} - \min_{\boldsymbol{y} \in \mathcal{Y}} \bar{\boldsymbol{x}}^\top \boldsymbol{U} \boldsymbol{y} \leq \frac{R_{▲}^T + R_{▼}^T}{T}.$$

Thus, if the two players pick their strategies according to a no-regret algorithm, this property guarantees the convergence of the average strategies $\bar{\boldsymbol{x}}$ and $\bar{\boldsymbol{y}}$ to a Nash Equilibrium as $T \to \infty$.

We now show how to construct regret minimizers for arbitrary tree-form decision problems (and thus, in particular, also for zero-sum games of perfect recall). For this problem, we use the *counterfactual regret minimization* (CFR) framework [108, 27]. Intuitively, CFR allows one to *build* a regret minimizer on a tree-form strategy set $\mathcal{Q}$ by running *local* regret minimizers at each decision point and combining them in a clever way. The guarantee given by CFR can be expressed as follows. Call a subset $P \subseteq \mathcal{D}$ *playable* if there is a pure strategy that reaches every decision point in P, that is, there is a pure strategy $\boldsymbol{x} \in \mathcal{Q}$ such that $\boldsymbol{x}[p_s] = 1$ for every $s \in P$.

Theorem 2.2.1 ([108, 27])**.** *After the execution of* CFR *for T rounds, with the set of mixed strategies $\mathcal{Q}$ as decision set, the cumulative regret is at most*

$$\max_{P} \sum_{s \in \mathcal{P}} R_s^T,$$

where R_s^T is the regret of the local regret minimizer $\mathcal{R}_s$ and the maximum is taken over all playable sets P.

Local regret minimizers on simplices $\Delta(m)$ whose regret scales as $O(\sqrt{mT})$ or even $O(\sqrt{T \log m})$ are well known; two examples are *regret matching* [37], which has the former guarantee, and *multiplicative weights*, which has the latter guarantee. Picking either of these algorithms (or, indeed, any of many regret minimizers with similar guarantees), we have:

Corollary 2.2.2. *If both players in an extensive-form zero-sum game of perfect recall use* CFR *with any efficient local regret minimizer, then after T rounds, the average strategy profile $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}})$ will be an $O(|\Sigma_{\blacktriangle}| + |\Sigma_{\blacktriangledown}|)/\sqrt{T}$-approximate Nash equilibrium.*

We refer the interested reader to [10, 29] for further reading on practical state-of-the-art no-regret-based solvers for two-player zero-sum games.

2.3 Equivalence Across Games

The contributions presented in this work will rely on auxiliary games to represent strategy optimization problems. In order for the results obtained in the auxiliary game to map to the original one we want to solve, we need to define what it means for two games to be equivalent. Let $G = (\mathcal{N}, \mathcal{H}, \mathcal{Z}, \mathcal{A}, \mathcal{I}, \boldsymbol{p}, u)$ and $G' = (\mathcal{N}, \mathcal{H}', \mathcal{Z}', \mathcal{A}', \mathcal{I}', \boldsymbol{p}', u')$ be extensive-form games with the same set of players. Let Π_i and Π'_i be player i's pure strategy set in G and G', respectively, and similarly let u_i and u'_i be player i's utility function in G and G' respectively.

Definition 2.3.1 (Strategic Equivalence)**.** Two games G and G' are *strategically equivalent* if there are bijective *strategy maps* $\rho_i : \Pi_i \to \Pi'_i$ for each player i such that, for every profile $\boldsymbol{\pi} \in \Pi$ and every player i we have $u_i(\boldsymbol{\pi}) = u'_i(\rho(\boldsymbol{\pi}))$ where $\rho(\boldsymbol{\pi}) := (\rho_i(\boldsymbol{\pi}_i))_{i \in \mathcal{N} \setminus 0}$.

This definition is a very strong notion of equivalence: if two extensive-form games are equivalent in the above sense, then every strategy $\boldsymbol{\pi}_i$ in one of the games is equivalent to some strategy $\rho(\boldsymbol{\pi}_i)$ in the other game. Thus, in particular, a solution to one game will give a solution to the other game.

2.4 Adversarial Team Games

The general framework of adversarial team games has first been studied by von Stengel and Koller [90] in the context of normal form games, while Celli and Gatti [18] first addressed them in an extensive-form setting. Adversarial team games describe situations where multiple agents are organized in two teams, receiving zero-sum payoffs. Like the above papers, ours focuses

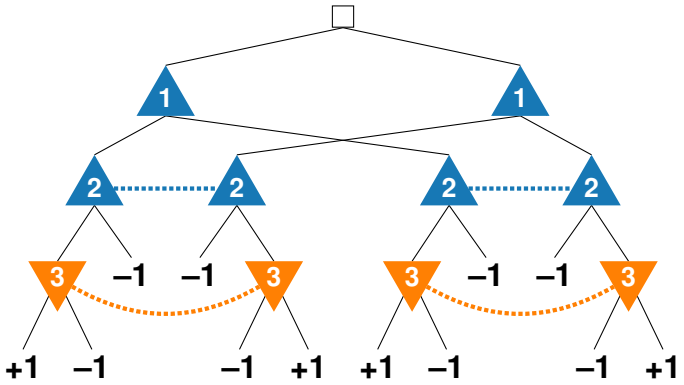


Figure 2.1: An example of an adversarial team game. There are three players: P1 and P2 are on team $\triangle$, and P3 is on team $\triangledown$. Dotted lines connect nodes in the same information set. The (total) utility of $\triangle$ is listed on each terminal node. The root node is a nature node, at which nature selects uniformly at random.

on the setting in which no extra communication channel is available to the players during the game, but they are allowed to communicate freely before the start of the game. This means that the only form of coordination across players' strategies available is *preplay coordination*; that is, any coordination has to be prepared before the start of the game.

Adversarial team games can be modeled as extensive-form games as follows.

Definition 2.4.1 (Adversarial team game)**.** An extensive-form, perfect-recall game $(\mathcal{N}, \mathcal{H}, \mathcal{Z}, \mathcal{A}, \mathcal{I}, \boldsymbol{p}, u)$ is said to be an *adversarial team game* (ATG), or *two-team zero-sum game* if

- the player set is partitioned in two sets called *teams*, symbolized by $\triangle$ and $\triangledown$. Formally, $\mathcal{N} = \triangle \cup \triangledown \cup \{\mathsf{N}\}$, and
- the utilities of the players belonging to the same team are identical, and the total utilities of the two team are opposites. Formally,

$$\begin{aligned} u_i &= u_j \quad \text{for all } i, j \in \triangle \\ u_i &= u_j \quad \text{for all } i, j \in \triangledown \\ \sum_{i \in \triangle} u_i &= -\sum_{j \in \triangledown} u_j. \end{aligned}$$

In adversarial team games, the Nash equilibrium fails to consider that teams can coordinate among themselves. Indeed, it is possible for there to be a Nash equilibrium in which two teammates could profit by *jointly* switching strategies, but no *individual* player can profit from a unilateral deviation. To consider these joint deviations, it is most natural to reformulate an adversarial team game as a two-player zero-sum game of imperfect recall, in which a *team coordinator* plays on behalf of all team members. In this manner, deviations of the

team coordinator correspond to *simultaneous, joint* deviations of all team members. We now formalize this conversion.

Definition 2.4.2 (Coordinator game)**.** Let $G = (\mathcal{N}, \mathcal{H}, \mathcal{Z}, \mathcal{A}, \mathcal{I}, \boldsymbol{p}, u)$ be an adversarial team game. The *coordinator game* G' corresponding to G is the two-player zero-sum imperfect-recall game defined $G' = (\{\mathsf{N}, \blacktriangle, \blacktriangledown\}, \mathcal{H}, \mathcal{Z}, \mathcal{A}, \mathcal{I}', \boldsymbol{p}, u')$, where

$$\mathcal{I}'_{\blacktriangle} = \bigcup_{i \in \triangle} \mathcal{I}_i, \qquad \mathcal{I}'_{\blacktriangledown} = \bigcup_{i \in \triangledown} \mathcal{I}_i, \qquad u'_{\blacktriangle} = \sum_{i \in \triangle} u_i, \quad \text{and} \quad u'_{\blacktriangledown} = \sum_{i \in \triangledown} u_i.$$

The coordinator game merges all members of a team ($\triangle$ or $\triangledown$) into a coordinator ($\blacktriangle$ or $\blacktriangledown$). Therefore:

- Pure strategies of a coordinator correspond to *pure profiles* of the team.
- Behavioral strategies of a coordinator correspond to *behavioral profiles* of the members of the team. Since behavioral strategies enforce actions at different infosets to be independently sampled, this means that team members can *privately* sample randomness for their own personal use but cannot share that randomness with teammates.
- Mixed strategies of a coordinator correspond to *correlated* strategy profiles of the members of the team. In a correlated profile, team members may *jointly* sample randomness that they use to correlate their actions.

We remark on the role that preplay coordination has in allowing the coordination capabilities modeled by the coordinator game. In fact, before starting the game, players are allowed to jointly sample a pure plan from their coordinator's mixed strategy and then individually play the specified actions at the infoset in which they play. This allows the team to play any randomized strategy of the coordinator effectively.

The coordinator game allows us to define notions of equilibrium specialized for team games:

Definition 2.4.3. A *team max-min equilibrium with correlation* (TMECor) of an ATG G is a mixed-strategy Nash equilibrium of G'.

Definition 2.4.4. A *team max-min equilibrium* (TME) of an ATG G is a behavioral max-min strategy of G'.

The *TMECor value* and *TME value* are defined analogous to the Nash value and behavioral max-min value. As discussed before, behavioral max-min strategies in G' are not equilibria in

Adversarial Team Games	Imperfect-Recall Games
Team $\triangle\triangledown$	Player $\blacktriangle\blacktriangledown$
Correlated team strategy	Mixed strategy
Uncorrelated team strategy	Behavioral strategy
TMECor	Mixed-strategy Nash Equilibrium
TME	Behavioral max-min strategy

Table 2.1: Translation table between terms commonly employed in the adversarial team games and two-player imperfect recall games. The translation happens through the introduction of coordinator games (Definition 2.4.2).

G', so one may wonder about the name "team max-min equilibrium". However, there *is* a sense in which TMEs are equilibria: von Stengel and Koller [90] showed that, at least in the case where $|\triangledown| = 1$, the TMEs are precisely the Nash equilibria of the team in G that maximize the utility of team $\triangle$.

An example adversarial team game in which the difference between TME and TMECor is relevant can be found in Figure 2.1. The coordinator game is constructed simply by erasing the player labels, creating a two-player zero-sum game. This game is a simple signaling game: nature selects a bit, which is privately revealed to P1. P1 then communicates a single bit, which is publicly revealed. Then P2 and P3 both attempt to guess nature's selected bit, and $\triangle$ wins if and only if P2's guess is correct. Therefore, the goal of P1 and P2 is for P1 to "securely" communicate the bit to P2 without also revealing it to P3. With a behavioral profile, this is impossible since P1 and P2 cannot correlate their strategies; therefore, the TME value is $-1/2$. However, if P1 and P2 are allowed to correlate their strategies, they can do the following: jointly flip a coin. If that coin landed heads, P1 communicates the true bit, and P2 plays what P1 communicates. If that coin landed tails, P1 communicates the opposite of the true bit, and P2 plays the opposite of what P1 communicates. In this way, P2 will always play the true bit, but P3 (who does not know the outcome of the correlating coin flip) does not learn any information. Therefore, the value of this strategy for $\triangle$ is 0 (since P2 wins half the time by randomly guessing the bit).

We have defined equilibrium concepts for team games using an "equivalent" coordinator game that is a two-player zero-sum imperfect recall. It turns out that, in fact, every two-player zero-sum imperfect-recall game G' has an ATG whose coordinator game is G': indeed, given such a G', consider the ATG G in which every information set is assigned to a different player. Therefore, team games and imperfect-recall games are equivalent in a very strong sense. All of the results of this work, unless otherwise stated, therefore apply equally to team games and to two-player zero-sum imperfect-recall games.

In the remainder of the paper, we consider the point of view of two-player zero-sum games with imperfect recall, that is the coordinator game from Definition 2.4.2. A summary of the different equivalent terms that are used in the two settings can be found in Table 2.1.

Chapter 3

Efficient Computation of Team and Imperfect-Recall Equilibria

This chapter's main objective is to present a novel contribution to the efficient computation of mixed-strategy equilibria in two-player zero-sum imperfect-recall games (or, equivalently, TMECors in adversarial team games). It consists of a novel algorithm transforming any game in these classes into a *strategically equivalent* two-player zero-sum perfect-recall decision problem called *Team Belief DAG (TB-DAG)*. This structure allows one to find an equilibrium using already existing efficient techniques. Other novel contributions are the proof of Σ_2^P-completeness and Δ_2^P-completeness of the problem of finding TMECors and TMEs. The theoretical results are accompanied by some experiments that show how the presented approach achieves state-of-the-art performance on a broad set of benchmark games.

The most interesting aspect of the proposed approach is that coordination among team members is represented in the converted game structure rather than as constraints on the joint strategy space or in the algorithm's logic. This approach is both interpretable and efficient, leading to state-of-the-art equilibrium computation in team games.

We remark that thanks to the equivalence between adversarial team games and two-player zero-sum imperfect-recall games presented in Section 2.4 (summarized in Table 2.1), it is equivalent to pick both the team and imperfect-recall perspectives. We, therefore, opt for the imperfect-recall framework to present our results, as this leads to a cleaner formalism while falling back to the team setting for some examples whenever those provide a better intuition.

The structure of the chapter is as follows. Section 3.1 presents a complete analysis of the related works from both the imperfect-recall games and adversarial team game literature. Sections 3.2.1 and 3.2.2 present a preliminary, interpretable algorithm that converts any two-

player zero-sum imperfect-recall game into a strategically-equivalent *perfect-recall game* which we call the *belief game*. In Section 3.2.3, we formally prove that strategic equivalence is preserved during the conversion, and Section 3.2.4 presents worst-case bounds on the size of the belief game in terms of the number of nodes of the original game. In particular, the worst case the number of histories of the belief game is $O(b^{dk})$, where b is the maximum branching factor of the original game, d is its depth, and k is a parameter we introduce called the *information complexity*, which intuitively measures the amount of information that the player can forget —or, in the case of team games, the amount of *information asymmetry* between players on the team. Section 3.3 introduces a notion of DAG-form decision-making that we use to generalize counterfactual regret minimization (CFR) beyond tree-form games. In Section 3.4, we use DAG-form decision problems to efficiently represent each player's strategy space in the belief game through a construction we call the *team-belief DAG* (TB-DAG). We show that the TB-DAG representation of a game with imperfect recall can be exponentially smaller than the size of the belief game and that it can be constructed directly from the original game without first constructing the belief game, thus leading to exponentially faster algorithms in the worst case. This construction improves the worst-case efficiency[1] of our technique to $O(|\mathcal{H}|(b+1)^{k+1})$, where $|\mathcal{H}|$ is the number of nodes in the original game. We also show that this bound is essentially optimal: under reasonable computational assumptions (namely, the exponential time hypothesis), we show that there cannot exist an algorithm for solving even single-player games of imperfect recall whose runtime is of the form $f(k)\mathsf{poly}(|\mathcal{H}|)$, for *any* function f. Section 3.5 investigated the computational complexity of computing mixed Nash equilibria with imperfect recall. We prove that computing a max-min strategy in mixed or behavioral strategies in games where both players have imperfect recall is Δ_2^{P}-complete and Σ_2^{P}-complete respectively. Section 3.6 presents further discussions comparing different notions presented in the paper, providing additional insights on the technical decisions made. In Section 3.7, we evaluate our methods empirically by benchmarking our construction on a standard testbed of imperfect-information games, compared to state-of-the-art baselines. Our technique allows much faster equilibrium computation when the information complexity k of the game is low.

This chapter's content is derived almost verbatim from a forthcoming first-author journal article currently under review at the *Journal of AI (JAI)*. Such a journal publication combines, synthesizes, and improves upon the results from the following three conference papers: Carminati et al. [15], Zhang and Sandholm [96], Carminati et al. [16], Zhang et al. [99].

[1] By *efficiency* here we mean the size of the representation of the players' strategy spaces. Algorithms such as CFR have per-iteration complexity that scales linearly in this size.

3.1 Related Research

This section describes the most important related works on computing Nash equilibria in imperfect-recall and team games. We start by reviewing the results in the literature on imperfect-recall games. To the best of our knowledge, there are no previous works concerned specifically with efficient algorithms for computing mixed-strategy Nash equilibria in imperfect-recall and timeable games, which, instead, is our primary objective. A similar approach to ours is taken by Tewolde et al. [84], investigating imperfect-recall games without the timeability assumption; that is, they include the absent-minded case; however, their solution concept is weaker than ours, and their setting is more difficult.

A stream of works studies the computational complexity of finding Nash equilibria in imperfect-recall games. Koller and Megiddo [45] study the NP-hardness of computing mixed Nash equilibria in imperfect-recall games. In particular, they proved that computing Nash equilibria in imperfect-recall games is NP-hard and that computing a max-min equilibrium in pure strategies is Σ_2^{P}-complete. Our results from Section 3.5 refine those results proving Δ_2^{P}-completeness and Σ_2^{P}-completeness of computing Nash equilibria in mixed or behavioral strategies, respectively.

Gimbert et al. [34] study imperfect-recall games with or without absentmindedness and one or two players. In particular, they connect the problem of computing optimal values in behavioral strategies with polynomial optimization. They also use their results on imperfect-recall games to characterize the complexity of the bidding phase of the *team* game of Bridge, acknowledging the equivalence of team games and imperfect-recall games. Our complexity results from Section 3.5 focus on the computation of mixed-strategy Nash equilibria in two-player imperfect recall with no absentmindedness and can be thought of as a refined (more powerful) version of the results of Gimbert et al. [34].

Similarly, previous uses of imperfect recall for abstractions in two-player zero-sum games make use of behavioral strategies to compute an approximate Nash equilibrium in the original perfect-recall game [92, 54, 50, 19]. In this case, the imperfect-recall game is used as a compressed representation to compute behavioral strategies that are then mapped to a hopefully good strategy in the original game. Our work is distinguished from theirs as our approach targets imperfect-recall games per se, and we are not interested in mapping optimal strategies to a perfect-recall game from which the game is generated. Some works, in particular, are interested in A-loss imperfect-recall games. The A-loss condition was originally introduced by Kaneko and Kline [43] and is related to imperfect recall due to a player forgetting actions played in the past but not the information revealed to her. Our results are generalizing theirs in the

sense that, since A-loss games are those that inflate to games of perfect recall, Proposition 3.6.2 guarantees that our algorithm will run in polynomial time for games with A-loss recall.

On the other hand, the adversarial team game literature presents many works interested in finding TMECors (Definition 2.4.3), which corresponds to finding mixed-strategy Nash equilibria in timeable imperfect-recall games. Past works typically overcome the issue of coordination among team members by directly optimizing over the space of coordinated strategies, that is, the space of mixed strategies of the coordinator game. The algorithm commonly adopted is that of *column generation (CG)*, which for games essentially amounts to the *double oracle* [62] algorithm. CG is an iterative approach that considers only a subset of pure strategies for the team at a time and progressively expands it to approximate the support of the TMECor in the best way. Each CG iteration comprises two steps: firstly, a TMECor restricted to the current set of pure strategies is computed; secondly, a joint best response to the current equilibrium is computed for each team, and these strategies are added to the sets of available strategies. Different variants of this algorithm have been developed in the literature. Celli and Gatti [18] first defined TMECors in the extensive-form games setting and proposed a CG algorithm based on integer linear programming for the best response computation and linear programming for the equilibrium computation. Successive works [25, 104, 29, 97] further refined the results regarding execution speed and scalability while still operating in the CG framework. Similarly, Zhang and An [103], Zhang et al. [104] focused on the computation of TMEs (Definition 2.4.4), which are behavioral-strategy equilibria for the settings in which no coordination signals are shared among the team.

Contrary to the direction followed by previous works, the purpose of this paper is to enable a sequence-form-like description of each team's strategy space. Thus, it bridges the gap in the literature between two-player zero-sum games and adversarial team games and empirically evaluates the tradeoff imposed by the solution proposed.

A technique based on a belief-prescription scheme was originally proposed by Nayyar et al. [69] in the field of decentralized stochastic control in a cooperative setting. The term *prescriptions* was later proposed by [30] in the setting of cooperative reinforcement learning to indicate the meta-actions played when coordinating multiple agents. Our construction of the auxiliary game in Section 3.2.2 can be seen as a generalization of those approaches to a two-team adversarial setting. In addition, the adoption of observations (defined in Section 3.2.1) in place of public states and the DAG approach from Section 3.4 are novel contributions that refine the efficiency of our methods compared to theirs. A more detailed comparison between observations and public states is given in Section 3.6.1.

Lastly, the idea of DAG decision problems in which identical paths of play are joined (instead of being repeated as it would happen in a tree structure) shares the same intent of the notion of *well-formed games* from Lanctot et al. [54]. The main difference with this approach is that in our case, the common paths of play are merged at a game-representation level, while the main focus in [54] was to describe a possible optimization to the CFR algorithm.

3.2 Mixed Nash Computation Through Beliefs and Observations

We introduce the paper's fundamental contribution: a novel technique for computing a mixed Nash equilibrium in two-player zero-sum imperfect-recall games (or equivalently for computing a TMECor in adversarial team games) based on the construction of an equivalent two-player zero-sum game with perfect recall.

Our technique attains the perfect recall condition by suitably changing the information available to the players and their action sets. The main intuition behind the belief game is to consider the point of view of a perfect recall player instead of the imperfect recall one. Unlike the imperfect-recall player, this player reasons only using information the player would never forget due to imperfect recall and chooses an action for every possible information set the imperfect-recall player may be in. The game then transitions by applying the action corresponding to the information set of the current node. Crucially, the perfect-recall player can strategically refine the set of reached nodes over time by carefully considering reachable nodes given the played strategy and the perfect-recall results of her actions.

After introducing the main concepts and the construction algorithm, we prove that the original and the belief games are strategically equivalent. This means that the perfect-recall player we introduce is an equivalent representation of both the imperfect-recall player and the corresponding preplay coordinated team (thanks to the considerations from Section 2.4).

3.2.1 Beliefs and Observations

The main purpose of this section is to formally define *beliefs*, which are sets of nodes $B \subseteq \mathcal{H}$ derived from information sets $\mathcal{I}_i$ of player $i \in \{\blacktriangle, \blacktriangledown\}$. Informally, beliefs are the "information sets" that i would have if she could not distinguish nodes that cannot be distinguished using information from a later stage. This notion is formalized by putting in the same belief any two nodes with descendent nodes in the same information set (even if they belong to different

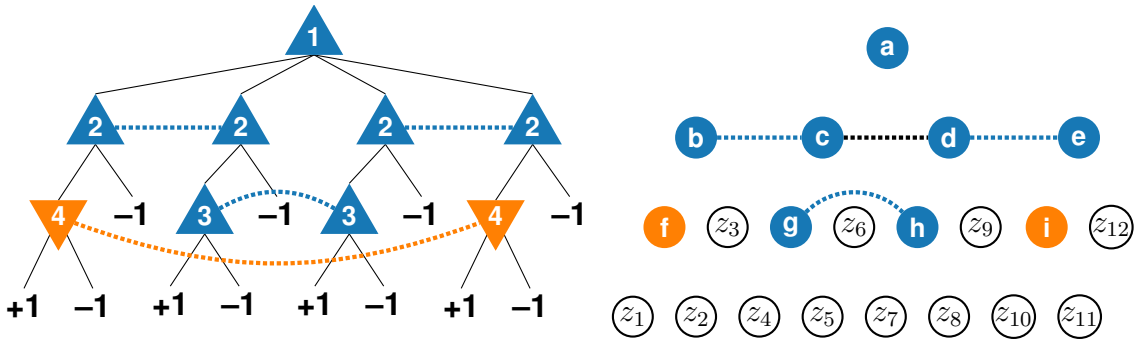


Figure 3.1: An example of team game on the left (using the same notation as Figure 2.1) and its corresponding connectivity graph for player ▲ on the right. Nodes in the two figures correspond in position.

information sets).[2] Similarly to information sets: i) nodes in beliefs would be indistinguishable to i ii) one action is chosen at each belief, and then this action is followed in all nodes in the belief iii) if the player knows that the current node of the game h lies in a set H of candidates, and the player observes that her current belief is B, then the set of candidates can be refined to $B \cap H$ (similarly to information sets, beliefs correspond to observations over the state of the game). Crucially, beliefs can be organized in the tree-like structure needed by algorithms finding Nash equilibria in two-player zero-sum games, as we will see in Sections 3.2.2 and 3.2.5. This is thanks to the guarantee that once a group of nodes is split among two different distinguishable beliefs, then any group of descendent nodes from one belief will be distinguishable from any group of descendants from the other.

In the following, we formalize the notion of beliefs and observations. We consider a two-player zero-sum game with imperfect recall G.

Connectivity graph We say that two nodes h and h' are *unforgettably distinguished* by i if they do not belong to the same infoset and no pair of children of those two nodes belong to the same infoset, that is, i will never be in an information set where these two ancestors are both possible. This condition guarantees that if the set of candidates is $H = \{h, h'\}$, then the player can discern h from h' and will never forget which of the two nodes has been reached in the next steps of the game.[3]

[2] We remark that this is a condition of imperfect recall equivalent to Definition 2.1.6, because information sets I, I' at the same level of the game having descendent nodes in the same information set J implies that J has nodes with different sequences.

[3] h, h' being distinguishable implies that in the corresponding team game, any team member can recall whether h or h' was reached upon reaching h or h'.

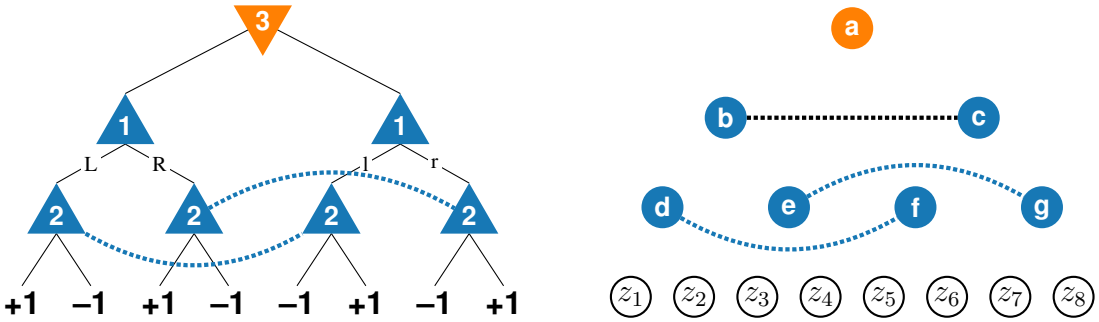


Figure 3.2: Another example of team game on the left (using the same notation as Figure 2.1) and its corresponding connectivity graph for player ▲ on the right. Nodes in the two figures correspond in position. Actions at nodes b and c are labeled for easier reference.

For the purpose of our definitions, we are concerned with pairs of nodes that are *not* distinguishable. This can be represented through a *connectivity graph* over $\mathcal{H}$ as follows.

Definition 3.2.1 (Connectivity graph)**.** The *connectivity graph* $\mathcal{G}_i = (\mathcal{H}, \mathcal{E}_i)$ for player $i \in \{▲, ▼\}$ is the graph with nodes $\mathcal{H}$ and edges $\mathcal{E}_i$, where $(h, h') \in \mathcal{E}_i$ if h and h' are at the same depth in G and there exists $I \in \mathcal{I}_i$ such that $h \preceq I$ and $h' \preceq I$.

Consider Figures 3.1 and 3.2 as examples of connectivity graphs for a game. Note in Figure 3.1 the blue edges, which correspond to connections due to infosets, and the black edge $c - d$ due to g, h belonging to the same infoset. Similarly for the black edge $b - c$ is due to d, f and e, g in Figure 3.2.

Beliefs Consider now a set H of nodes such that the induced subgraph $\mathcal{G}_i[H]$ is connected. Player i cannot distinguish any subset of H from the others because any node cannot be distinguished from its neighbors. *Beliefs* are defined as these sets of indistinguishable nodes.[4]

Definition 3.2.2 (Belief)**.** A set of nodes $B \subseteq \mathcal{H}$ is a *belief* for player i if the induced subgraph $\mathcal{G}_i[B]$ is connected.

We remark that the timeablility property assumed on G implies that any node belonging to the same belief has the same depth. Notice that a direct consequence of the definition of beliefs is that $\{\varnothing\}$ and $\{z\}$ for $z \in \mathcal{Z}$ are singleton beliefs for both teams.

[4]From a team game perspective, beliefs are sets of nodes with the guarantee that once reached, all team members know that any node $\mathcal{H} \setminus B$ is not reached; that is, it is team-common knowledge that the game reached a node in B.

Observations Consider instead a set H of nodes such that the induced subgraph $\mathcal{G}_i[B]$ has different connected components. In this case, player i can distinguish those components one from the other, thus partitioning H into multiple beliefs. Intuitively, the unforgettable information is enough to distinguish every node in a component from any node in other components. The player can, therefore, exclude nodes from components that are distinguishable from the current reached node. We say that upon reaching a node h among possible candidates H, player i *observes belief* $B \subseteq H$, meaning that player i uses the newly acquired unforgettable information acquired in h to refine its imperfect information from H to B. We formalize this notion of observation through the function $\textsc{SplitBelief}_i$:[5]

Definition 3.2.3 (Observation)**.** The *observation* for player $i \in \{\blacktriangle, \blacktriangledown\}$ when reaching node h among a set H of candidate nodes is:

$$\textsc{SplitBelief}_i(H, h) \coloneqq \text{the connected component of } \mathcal{G}_i[H] \text{ containing } h.$$

The set of all possible observations given a set of candidates is denoted[6] by

$$\mathcal{B}_H \coloneqq \{\textsc{SplitBelief}_i(H, h) : h \in H\}.$$

An example of observation can be given by considering the team game depicted in Figures 3.1 and 3.2 and a candidate set $H = \{b, c, e\}$. This candidate set is possible when player $\blacktriangle$ plays a strategy where player 1 plays a mixed strategy excluding d from its support. This, in turn, implies that player 2, at the next step, knows that the reached node h in the game is in H. Moreover, player 2 observes her current information set $I = b, c$ if $h \in \{b, c\}$ of $I = \{d, e\}$ if $h \in \{d, e\}$. I can be used to further refine H *as long as the information used will be known at player 1 next*. This is formalized in $\textsc{SplitBelief}_i(H, b) = \textsc{SplitBelief}_i(H, c) = \{b, c\}$ and $\textsc{SplitBelief}_i(H, e) = \{e\}$, which intuitively correspond to the fact that given those candidates, $\blacktriangle$ unforgettably distinguishes b and c from e. From the equivalent team game perspective: player 2 is active and can check her current infoset to distinguish the two beliefs; player 1 has stopped playing and therefore it is not relevant in terms of team knowledge; player

[5] In team games, the belief returned by $\textsc{SplitBelief}_i$ is the team-common knowledge update happening when reaching h among a set of candidates H.

[6] We remark that the belief-based constructions employed by the paper would also work when allowing $\textsc{SplitBelief}_i$ to return any superset of connected components. For example, in the framework of *factored-observation games* [48], it is valid to define $\textsc{SplitBelief}_i$ using the explicitly-given public observations. However, since the efficiency of the proposed algorithms depends on the size of the beliefs employed, we opt not to allow, by definition, the use of beliefs larger than needed. As we show in Section 3.2.4, any reduction in the size of the beliefs in a game brings exponential benefits in the size of the belief game obtained.

3 either will not play or will know that the current node was c once the game reached g, so she can safely assume that the game is in c. This means that every player distinguishes e from b, c.

Team public states We compare our notion of beliefs with *public states* (Definition 2.1.3), an alternative customarily used in the related literature. A public state P for player i can also be defined as a connected component of the connectivity graph $\mathcal{G}_i$. The set of all public states of i is denoted as $\mathcal{P}_i$.

Public states identify sets of nodes that are distinguishable to a player without considering a possibly pruned subgraph of $\mathcal{G}_i$ as instead done for team observations. Therefore, every belief is contained in a public state. In Figures 3.1 and 3.2 we have that $\mathcal{P}_{\blacktriangle} = \{\{a\}, \{b, c, d, e\}, \{g, h\}, \{f\}, \{i\}\} \cup \{\{z\} : z \in \mathcal{Z}\}$.

Public states are the customarily adopted alternative to observations when partitioning a set H of candidates in beliefs by splitting H in $\{H \cap P : P \in \mathcal{P}_i\}$. However, public states may return a coarser partition than the one returned by observations, as the absence of specific nodes from H may disconnect components in $\mathcal{G}$. We will, therefore, use observations in place of public states whenever possible. An example illustrating the difference between the two definitions is available in Section 3.6.1.

Prescriptions Restricting the information available to player i to her beliefs also affects the set of actions available. In fact, multiple infosets may intersect a given belief, and the player does not know in which infoset she finds herself. Therefore, she does not know what actions are available to her.

We overcome this issue by associating to each belief B a set of meta-actions A^i_B such that an action is specified for each possible infoset that intersects the belief. We call such structured meta-actions *prescriptions* and use a symbol $\boldsymbol{a}$ to indicate them. The concept is formally defined as follows.

Definition 3.2.4 (Prescription)**.** Consider a belief B of a player $i \in \{\blacktriangle, \blacktriangledown\}$. A *prescription* $\boldsymbol{a}$ is a selection of one action at each infoset having a nonempty intersection with B:

$$\boldsymbol{a} \in \bigtimes_{I \in \mathcal{I}_i[B]} A_I \quad \text{where} \quad \mathcal{I}_i[B] = \{I \in \mathcal{I}_i : I \cap B \neq \varnothing\}.$$

Given a prescription $\boldsymbol{a}$ for a belief B and an infoset I such that $I \cap B \neq \varnothing$, we denote as $\boldsymbol{a}[I]$ the action relative to infoset I which is specified by prescription $\boldsymbol{a}$. Note that we have *empty prescriptions* at beliefs containing no active nodes for a player.

As we will see in the next section, our equivalent belief game introduces one perfect-recall player per team, with information sets associated with beliefs corresponding to the perfect-recall part of the information available to this unique player. Prescriptions will allow this player to have an identical expressive power in terms of actions without accessing the exact information set of the player, which is her imperfect-recall information. Moreover, specifying a prescription at each reached belief for i incrementally defines a pure strategy of player i. This allows us to consider a reduced set of candidate nodes H for the reached node h from which the belief is observed, as a non-played action implies that all the descendant nodes are not reached and, therefore, excluded from the candidates.

For example, consider a 3-player poker instance where two players collude to form a team. At any time of the game, we can consider the point of view of a team coordinator, who acts as the single imperfect recall player. We can imagine this coordinator as sitting at the same table as the players, and therefore, she cannot access the private cards given to the players but can access the same public information as the players, that is, the bet, fold, and check actions of the players. Her belief at any point regards the private cards that each team member has. At the start of the game, this belief is uniform over all pairs of cards, as no information regarding these cards is available from an external point of view. The coordinator emits prescriptions for the players to follow as the game progresses. Since the coordinator does not know the card held by a player, she has to prescribe an action for each possible card the current player may hold. The player receives this prescription and follows the part that matches the private card. By observing the action played by the player, the coordinator can exclude from her belief the cards for which she prescribed different actions. While there are no means of communicating prescriptions during play at the poker table, this mechanism can be implemented *ex-ante*; that is, each team of players jointly samples a pure strategy of this coordinator before the start of the game and each member simulates the coordinator locally.

Information complexity We quantify the number of information sets reaching a belief through the notion of *information complexity* k. This quantity will allow us to bound the size of the belief games in Sections 3.2.4 and 3.4.1.

We first characterize the notion of *remembered* information sets and the set of *last-infosets* at a node. Intuitively, an infoset J remembers another infoset I if reaching a node in J implies traversing a node in I and picking a specific action. Therefore, knowing to be at a node in J allows the player to recall having traversed I and have played action a there. The last-infosets of player i at h are the information sets traversed by h and not remembered by any following

information set of the player up to h.[7] This set quantifies the knowledge lost by the player at a node due to imperfect recall.

Definition 3.2.5. An infoset J *remembers* another infoset I if there exists an action $a \in A_I$ such that, for every $h \in J$, we have $h'a \preceq h$ for some $h' \in I$.

Definition 3.2.6. The set of *last-infosets* at node h for player i is the set of infosets $I \in \mathcal{I}_i$ such that $I \preceq h$ and there is no other infoset $J \in \mathcal{I}_i$ such that $J \preceq h$ and J remembers I.

We will use $\mathrm{LI}_i(h)$ to denote the set of last-infosets at h for player i. Note that if $h \in \mathcal{H}_i$ then $I_h \in \mathrm{LI}_i(h)$.

Now define the *information complexity* k of a two-player game G as follows.

$$k = \max_{\substack{i \in \{\blacktriangle, \blacktriangledown\}, \\ P \in \mathcal{P}_i}} \left| \bigcup_{h \in P} \mathrm{LI}_i(h) \right|$$

Intuitively, k represents *how much information can be worst-case forgotten* by player i. In the team game interpretation, k represents how *asymmetric* the information is among team members. Note that $k = 1$ if and only if both players have perfect recall.

The information complexity characterizes both the number of beliefs in a public state P, and the number of prescriptions that are available at such beliefs. In fact, the actions played at information sets in $\bigcup_{h \in P} \mathrm{LI}_i(h)$ determine which nodes in P are reached (that is, a belief $B \subseteq P$).

As an example, consider the game from Figures 3.1 and 3.2 and the public state $P = \{g, h\}$. We have that the strategy played at

$$\bigcup_{h \in \{g,h\}} \mathrm{LI}_i(h) = \{I_a, I_c.I_g\} \cup \{I_a, I_d, I_h\} = \{I_a, I_c, I_d, I_g\}$$

is enough to characterize a belief $B \in P$ and a prescription at that belief. In fact, the action at I_a decides whether c and d are reached, the actions at I_c (respectively I_d) decide whether g (respectively h) is reached, and the action at $I_g = I_h$ is the prescription.

It is instructive to understand how k behaves in a simple game. Suppose that G is a team game such that there are n players on each team, each player is assigned one of t "private types" (in poker, these are the private hands), and all other information in the game is common

[7] From the perspective of an adversarial team game, the last-infosets at a node for team $t \in \{\blacktriangle, \blacktriangledown\}$ are the most recent infosets of each player in t, minus the infosets of players that are implied by other players' infosets.

Algorithm `MakeBeliefGame` Belief game construction

procedure MAKENODE$_{\blacktriangle}(h, B_{\blacktriangle}, B_{\blacktriangledown}, \tilde{\sigma}_{\blacktriangle}, \tilde{\sigma}_{\blacktriangledown})$
  create node $\tilde{h} \in \tilde{\mathcal{H}}_{\blacktriangle}$
  add $\tilde{h}$ to infoset labeled $(\tilde{\sigma}_{\blacktriangle}, B_{\blacktriangle})$
  for each prescription $\boldsymbol{a}_{\blacktriangle} \in \mathcal{A}^{\blacktriangle}_{B_{\blacktriangle}}$ **do**
    $\tilde{h}\boldsymbol{a}_{\blacktriangle} \leftarrow$ MAKENODE$_{\blacktriangledown}(h, B_{\blacktriangle}, B_{\blacktriangledown}, \tilde{\sigma}_{\blacktriangle}, \tilde{\sigma}_{\blacktriangledown}, a_{\blacktriangle})$
  return $\tilde{h}$
procedure MAKENODE$_{\blacktriangledown}(h, B_{\blacktriangle}, B_{\blacktriangledown}, \tilde{\sigma}_{\blacktriangle}, \tilde{\sigma}_{\blacktriangledown}, \boldsymbol{a}_{\blacktriangle})$
  create node $\tilde{h}\boldsymbol{a}_{\blacktriangle} \in \tilde{\mathcal{H}}_{\blacktriangledown}$
  add $\tilde{h}\boldsymbol{a}_{\blacktriangle}$ to infoset labeled $(\tilde{\sigma}_{\blacktriangledown}, B_{\blacktriangledown})$
  for each prescription $\boldsymbol{a}_{\blacktriangledown} \in \mathcal{A}^{\blacktriangledown}_{B_{\blacktriangledown}}$ **do**
    $\tilde{h}\boldsymbol{a}_{\blacktriangle}\boldsymbol{a}_{\blacktriangledown} \leftarrow$ MAKENODE$_{\mathsf{N}}(h, B_{\blacktriangle}, B_{\blacktriangledown}, \tilde{\sigma}_{\blacktriangle}, \tilde{\sigma}_{\blacktriangledown}, \boldsymbol{a}_{\blacktriangle}, \boldsymbol{a}_{\blacktriangledown})$
  return $\tilde{h}\boldsymbol{a}_{\blacktriangle}$
procedure MAKENODE$_{\mathsf{N}}(h, B_{\blacktriangle}, B_{\blacktriangledown}, \tilde{\sigma}_{\blacktriangle}, \tilde{\sigma}_{\blacktriangledown}, \boldsymbol{a}_{\blacktriangle}, \boldsymbol{a}_{\blacktriangledown})$
  if h is terminal node **then**
    create new terminal node $\tilde{h}\boldsymbol{a}_{\blacktriangle}\boldsymbol{a}_{\blacktriangledown} \in \tilde{\mathcal{Z}}$
    $u_i(\tilde{h}\boldsymbol{a}_{\blacktriangle}\boldsymbol{a}_{\blacktriangledown}) \leftarrow u_i(h)$ for each player i
    $\boldsymbol{p}[\tilde{h}\boldsymbol{a}_{\blacktriangle}\boldsymbol{a}_{\blacktriangledown}] \leftarrow \boldsymbol{p}[h]$
    return $\tilde{h}\boldsymbol{a}_{\blacktriangle}\boldsymbol{a}_{\blacktriangledown}$
  create new chance node $\tilde{h}\boldsymbol{a}_{\blacktriangle}\boldsymbol{a}_{\blacktriangledown} \in \mathcal{H}_{\mathsf{N}}$
  if h is a chance node **then** $S \leftarrow \{ha : a \in A_h\}$
  else $S \leftarrow \{h\boldsymbol{a}_i[I_h]\}$ where $h \in \mathcal{H}_i$
  for each node $ha \in S$ **do**
    $B'_i \leftarrow$ SPLITBELIEF$_i(B_i\boldsymbol{a}_i, ha)$ for each player i
    $\tilde{h}\boldsymbol{a}_{\blacktriangle}\boldsymbol{a}_{\blacktriangledown}a \leftarrow$ MAKENODE$_{\blacktriangle}(ha, B_{\blacktriangle}', B_{\blacktriangledown}', \tilde{\sigma}_{\blacktriangle} + (\tilde{\sigma}_{\blacktriangle}, a_{\blacktriangle}), \tilde{\sigma}_{\blacktriangledown} + (\tilde{\sigma}_{\blacktriangledown}, a_{\blacktriangledown}))$
  return $\tilde{h}\boldsymbol{a}_{\blacktriangle}\boldsymbol{a}_{\blacktriangledown}$

knowledge. Then, at each public state $P \in \mathcal{P}_i$, there are at most t last-infosets per player, so $k = nt$.

3.2.2 Belief Game Construction

We now introduce an algorithm that explicitly constructs a belief game given any two-player game. We will use $\tilde{G}$ to denote the belief game and distinguish components of the original game G from components of the belief game by writing tildes: for example, a generic node is $\tilde{h} \in \tilde{\mathcal{H}}$, a generic information set is $\tilde{I}_i \in \tilde{\mathcal{I}}_i$, and so on.

Here, for cleanliness, we will describe the evolution of the belief game as a game of *simultaneous moves*. Algorithm `MakeBeliefGame` describes the procedure that constructs

an extensive-form game (without simultaneous moves) that is equivalent to it.[8] In particular, $\textsc{MakeNode}_{\blacktriangle}(\varnothing, \{\varnothing\}, \{\varnothing\}, \varnothing, \varnothing)$ constructs the whole belief game.

A node $\tilde{h} \in \tilde{\mathcal{H}}$ in the belief game is identified by a tuple $(h, B_{\blacktriangle}, B_{\blacktriangledown})$ such that $h \in B_{\blacktriangle} \cap B_{\blacktriangledown}$, where $h \in \mathcal{H}$ is the corresponding node in the original game describing the underlying state of the game, $B_{\blacktriangle}, B_{\blacktriangledown}$ are the current beliefs of $\blacktriangle$ and $\blacktriangledown$ respectively. At $\tilde{h}$, each player $i \in \{\blacktriangle, \blacktriangledown\}$ has a (possibly empty) collection of infosets, $\mathcal{I}[B_i]$, at which it needs to prescribe an action. The two players simultaneously submit actions $\boldsymbol{a}_i \in A^i_{B_i}$. The next belief game node is $(ha, B'_{\blacktriangle}, B'_{\blacktriangledown})$, where the action a and the next beliefs $(B'_{\blacktriangle}, B'_{\blacktriangledown})$ are defined as follows.

(i) The action a is the one taken by the player at h: if h is chance node, then a is sampled from chance's action distribution at h; otherwise, $a = \boldsymbol{a}_i[I_h]$.

(ii) The beliefs evolve as follows. For each player i, the set of candidate next nodes in the original game compatible with i's current belief B_i and its prescription $\boldsymbol{a}_i$ is given by:

$$B_i\boldsymbol{a}_i := \underbrace{\{ha : h \in B_{\blacktriangle} \cap \mathcal{H}_i, a = \boldsymbol{a}[I_h]\}}_{\substack{\text{when player } i \text{ acts,}\\ \text{it must be according to the prescription}}} \cup \underbrace{\{ha : h \in B_{\blacktriangle} \setminus \mathcal{H}_i, a \in A_h\}}_{\substack{\text{when player } i \text{ does not act,}\\ i \text{ does not know what action is taken}}},$$

Next, player i observes the information revealed by the next node ha, thus arriving at belief:

$$\tilde{B}'_i := \textsc{SplitBelief}_i(B_i\boldsymbol{a}_i, ha).$$

The examples of application of the `MakeBeliefGame` algorithm to the adversarial team games from Figures 3.1 and 3.2 are shown in Figures 3.3 and 3.4 respectively.

We remark some characteristics of $\tilde{G} := \texttt{MakeBeliefGame}(G)$.

- Multiple different tree nodes $\tilde{h}$ can correspond to the same $(h, B_{\blacktriangle}, B_{\blacktriangledown})$ tuple. In particular, for each terminal node $z \in \mathcal{Z}$ there is only one state $(z, \{z\}, \{z\})$.
- Information sets in $\tilde{G}$ are associated with sequences of beliefs and prescriptions. In particular, such infosets can be described by tuples of the form $(B^1_i = \{\varnothing\}, \boldsymbol{a}^1_i, B^2_i, \boldsymbol{a}^2_i, \ldots, B^L_i)$, where $\boldsymbol{a}^\ell_i \in A_{B^\ell_i}$ and $B^{\ell+1}_i = \textsc{SplitBelief}_i(B^\ell_i\boldsymbol{a}^\ell_i, h)$ for some $h \in B^\ell_i\boldsymbol{a}^\ell_i$.

[8] We implement simultaneous actions by representing each step in the game as a sequence of one node per player $\blacktriangle$, $\blacktriangledown$, N where everyone acts; the effects of the actions taken are applied at the end.

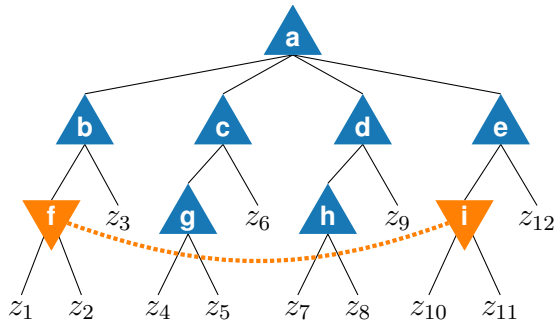


Figure 3.3: Belief game constructed from the adversarial team game in Figure 3.1. Trivial nodes with a single actions have been removed to clarify the resulting game. Note how ▲ never has beliefs containing more than one node, so there is only a constant factor increase in size with respect the original game.[10]

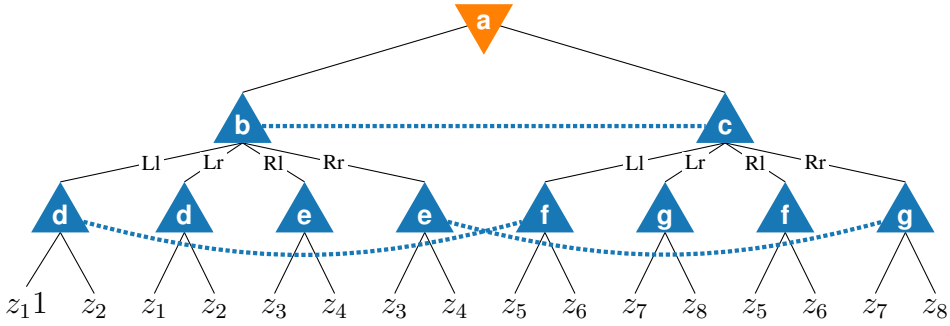


Figure 3.4: Belief game constructed from the adversarial team game in Figure 3.2. Trivial nodes with a single actions have been removed to clarify the resulting game. Note how ▲ in this case solves the imperfect recall of the team through presciptions. Non trivial presciptions at nodes b and c (where ▲ has belief {b,c}) have been marked with the corresponding actions at nodes.

- By construction of `MakeBeliefGame` we have that $\tilde{G}$ is a perfect-recall game. In fact, nodes with different sequences are associated with different information sets thanks to including sequences in each information set's label;
- When h is terminal, the belief game does not stop until both players have observed the trivial belief $\{h\}$ at h and then submitted their empty prescriptions at that belief. This is for notational convenience: it ensures that terminal sequences for a player i will always end with singleton beliefs, which will make the later analysis cleaner.
- Modulo trivial reformulations (namely, the insertion of nodes with a single child), if G is perfect recall, then $\tilde{G}$ is identical to G.

Given a pure strategy $\tilde{\boldsymbol{\pi}}_i \in \tilde{\Pi}_i$, we say that $\tilde{\boldsymbol{\pi}}_i$ plays to a belief B_i of player i if $\tilde{\boldsymbol{\pi}}_i$ plays to some node corresponding to (h, B_i, B_{-i}).

3.2.3 Strategic Equivalence

The main result of this section is the following.

Theorem 3.2.1. *Let G be any two-player imperfect-recall extensive-form game, and $\tilde{G}$ be the belief game constructed by* `MakeBeliefGame`*. G and $\tilde{G}$ are strategically equivalent.*

We use the remainder of the subsection to prove this result. The requirements for strategic equivalence as per Definition 2.3.1 that we will show are: i) it exists a function ρ mapping pure strategies in the two games ii) ρ is a bijective function iii) ρ is value-preserving, that is, the mapped strategies have the same expected utilities.

We first construct the strategy maps $\rho_i : \Pi_i \to \tilde{\Pi}_i$. A pure strategy $\boldsymbol{\pi}_i \in \Pi_i$ in G assigns one action to each information set.[11] From such a strategy we construct a strategy $\tilde{\boldsymbol{\pi}}_i = \rho_i(\boldsymbol{\pi}_i)$ which plays prescriptions consistent with $\boldsymbol{\pi}_i$. At belief B_i, $\tilde{\boldsymbol{\pi}}_i$ plays the prescription $\boldsymbol{a}_i$ given by $\boldsymbol{a}_i[I] = \boldsymbol{\pi}_i[I]$ for each $I \in \mathcal{I}_{B_i}$ reached by $\tilde{\boldsymbol{\pi}}$.

We now show that ρ_i is injective. This will follow from the following lemma.

Lemma 3.2.2. *Let $\tilde{\boldsymbol{\pi}}_i = \rho_i(\boldsymbol{\pi}_i) \in \tilde{\Pi}_i$. Then for every $z \in \mathcal{Z}$, $\tilde{\boldsymbol{\pi}}_i$ plays to belief $\{z\}$ if and only if $\boldsymbol{\pi}_i$ plays to z.*

Proof. First suppose $\tilde{\boldsymbol{\pi}}_i = \rho_i(\boldsymbol{\pi}_i) \in \tilde{\Pi}_i$ plays to $\{z\}$. Thus $\tilde{\boldsymbol{\pi}}_i$ plays some sequence of beliefs and prescriptions $(B_i^1 = \{\varnothing\}, \boldsymbol{a}_i^1, \ldots, B_i^2, \boldsymbol{a}_i^2, \ldots, B_i^L = \{z\})$. But then, by construction, every ancestor $h \prec z$ is included in one of the B_i^ℓs, and for $ha \preceq z$ to appear in $B_i^\ell \boldsymbol{a}_i^\ell$, if $h \in \mathcal{H}_i$ it must be the case that $\boldsymbol{\pi}_i$ plays a. Thus $\boldsymbol{\pi}_i$ plays to z.

Conversely suppose $\boldsymbol{\pi}_i$ plays to z. Then, construct a sequence of beliefs and prescriptions as follows. Let $\boldsymbol{a}_i^\ell$ be the prescrption played by $\tilde{\boldsymbol{\pi}}_i$ at belief B_i^ℓ, and $B_i^{\ell+1} = \textsc{SplitBelief}_i(B_i^\ell \boldsymbol{a}_i^\ell, h)$ where $h \in B_i^\ell \boldsymbol{a}_i^\ell$ and $h \preceq z$. (The fact that $\boldsymbol{\pi}_i$ plays to z ensures that such h must exist). Then, by induction, $\tilde{\boldsymbol{\pi}}_i$ plays to this sequence, and the sequence must eventually terminate at $\{z\}$ because it always contains at least one predecessor of z. Thus $\tilde{\boldsymbol{\pi}}_i$ plays to $\{z\}$. □

Since different pure strategies (by definition) play to different sets of terminal nodes, this immediately shows that ρ is injective. We now show that ρ_i is a surjection (and hence a

[11] At infosets not reached by $\boldsymbol{\pi}_i$, actions can be selected arbitrarily.

bijection), that is, any $\tilde{\boldsymbol{\pi}}_i \in \tilde{\Pi}_i$ is the image of some $\boldsymbol{\pi}_i \in \Pi_i$. We remark that a pure strategy $\tilde{\boldsymbol{\pi}}_i \in \tilde{\Pi}_i$ assigns one prescription to every reached infoset of player i in $\tilde{G}$.

We will require the following lemma. Informally, it states that no player can play to two nodes with intersecting beliefs.

Lemma 3.2.3. *Let $\tilde{\boldsymbol{\pi}}_i \in \tilde{\Pi}_i$ be any pure strategy. Let $\tilde{I}, \tilde{I}' \in \tilde{\mathcal{I}}_i$ be two distinct infosets of player i that are simultaneously reached by $\tilde{\boldsymbol{\pi}}_i$. Let B_i and B_i' be the beliefs for player i at $\tilde{I}$ and $\tilde{I}'$, respectively. Then, B_i and B_i' are not connected and do not intersect. That is, there do not exist nodes $h \in B_i, h' \in B_i'$ with $(h, h') \in \mathcal{E}_i$ or $h = h'$.*

Proof. Consider the tree-form decision problem $\tilde{\mathcal{T}}$ of player i in $\tilde{G}$. Since $\tilde{G}$ is perfect-recall, $\tilde{\mathcal{T}}$ is indeed a valid representation of player i's strategy set in $\tilde{G}$. Let s be the lowest common ancestor of $\tilde{I}$ and $\tilde{I}'$ in $\tilde{\mathcal{T}}$.

Since $\tilde{\boldsymbol{\pi}}_i$ plays to both $\tilde{I}$ and $\tilde{I}'$, the node s must be an observation point, as a pure strategy plays a single action at each decision point. At observation points, the next observation made by player i is the next belief. Let C_i and C_i' be the different observed beliefs at node s that lead to $\tilde{I}$ and $\tilde{I}'$, respectively. Then, by construction of SPLITBELIEF$_i$, C_i and C_i' are disconnected and disjoint. Since every node in B_i is a descendant of some node in C_i (and the same for C_i'), it follows that B_i and B_i' are also disconnected and disjoint. □

Thus, in particular, for any infoset $I \in \mathcal{I}_i$ in the original game, $\tilde{\boldsymbol{\pi}}_i$ can only play to one infoset $\tilde{I} \in \tilde{\mathcal{I}}_i$ whose belief B_i overlaps I. At B_i, the prescription chosen by $\tilde{\boldsymbol{\pi}}_i$ includes an action $\boldsymbol{a}[I]$ at I (by construction of prescriptions at a node). Thus, consider the strategy $\boldsymbol{\pi}_i$ defined such that $\boldsymbol{\pi}_i[I] = \boldsymbol{a}[I]$ for every infoset I such that $\boldsymbol{\pi}_i$ plays to a belief B_i overlapping I.

Lemma 3.2.4. *$\boldsymbol{\pi}_i$ is a well-defined strategy. That is, if $\boldsymbol{\pi}_i$ plays to a node $h \in \mathcal{H}_i$, then $\tilde{\boldsymbol{\pi}}_i$ plays to a belief $B_i \ni h$, and hence, if $h \in I \in \mathcal{I}_i$ then $\boldsymbol{\pi}_i[I]$ is defined.*

Proof. By induction on the length of the node h. For $h = \varnothing$ this is trivial. Now let $h = h'a'$ be a non-root node, and suppose $\boldsymbol{\pi}_i$ plays to h. Then by inductive hypothesis, $\tilde{\boldsymbol{\pi}}_i$ plays to a belief $B_i \ni h'$. Let $\boldsymbol{a}$ be the prescription played by $\tilde{\boldsymbol{\pi}}_i$ at B_i.

- If $h' \in I \in \mathcal{I}_i$, then by construction of $\boldsymbol{\pi}_i$, it must be the case that $\tilde{\boldsymbol{\pi}}_i$'s prescription $\boldsymbol{a}$ at B_i satisfies $\boldsymbol{a}[I] = a'$, so $B_i\boldsymbol{a} \ni h$.
- If $h' \notin \mathcal{H}_i$, then for *any* prescription $\boldsymbol{a}$ at B_i we have that $h \in B_i\boldsymbol{a}$.

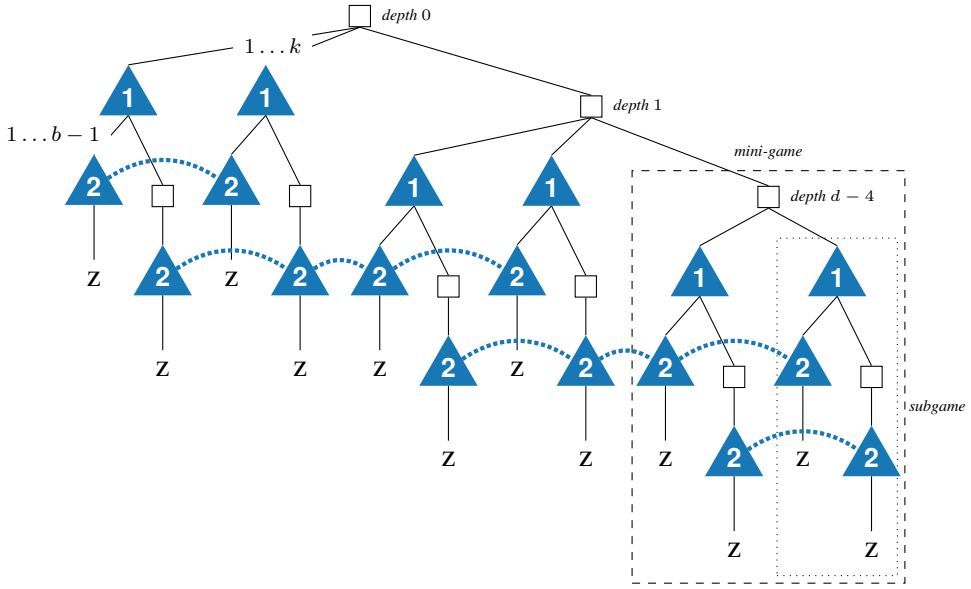


Figure 3.5: An example of imperfect-recall game derived from a team game whose rationale is described in the proof of Theorem 3.2.5. Given that the terminal values of this game are not relevant, we use z to indicate a generic payoff. The boxes indicated by *mini-game* and *subgame* correspond to the terms used in the description.

In either case, we have $B_i\boldsymbol{a} \ni h$. Thus, $\tilde{\boldsymbol{\pi}}_i$ must also play to the belief $\textsc{SplitBelief}_i(B_i\boldsymbol{a}, h) \ni h$. □

Further, from the definition of ρ_i it follows immediately that $\rho_i(\boldsymbol{\pi}_i) = \tilde{\boldsymbol{\pi}}_i$. It only remains to show that ρ_i is value-preserving. But this is easy: $\rho_i(\boldsymbol{\pi}_i)$ prescribes the same actions as $\boldsymbol{\pi}_i$. Thus, for any pure strategy profile $\boldsymbol{\pi} \in \Pi$, following profile $\boldsymbol{\pi}$ through G will yield exactly the same trajectory as following profile $\rho(\boldsymbol{\pi})$ through $\tilde{G}$. Thus, their expected utilities will also coincide, and the proof is complete.

3.2.4 Worst-Case Dimension of the Belief Game

The per-iteration time complexity of `CFR` depends linearly on the size of the game on which the algorithm is applied. Thus, it is critical for complexity analysis to bound the size of the belief game produced by `MakeBeliefGame`.

Lower Bound We first present a lower bound of the worst-case size of the belief game by describing a worst-case game whose belief game has a large number of nodes.

Theorem 3.2.5. *There exists a game G with depth d, information complexity k, and maximum branching factor at a node b such that the number of nodes in the belief game $\tilde{G}$ is $|\tilde{\mathcal{H}}| \geq b^{2k(d-4)}$.*

Proof. We construct a parametric game for depth $d \geq 4$, information complexity k, and branching factor $b \geq k+1$.

Consider a game that consists of $d-3$ repetitions of the following *mini-game*. There is a *"root"* nature node with k nodes of ▲, k nodes of ▼ and the root of the next repetition of the mini-game as children. Each of ▲'s and ▼'s nodes belong to different information sets. Each of those is the root of *subgames* with identical structures. Their children are $b-1$ player nodes followed by a single terminal node, and there is a chance node followed by a player node with a single terminal node. These player nodes belong to the same player as the root of the subtree, namely ▲ for the first k children and ▼ for the second k children of the *"root"* of the mini-game. All nodes of ▲ and ▼ belong to the same level of the game apart from the $2k$ nodes that belong to the same information set.

A representation of such a game for $k=2, b=2, d=6$ is given in Figure 3.5, where nodes of ▼ have been omitted to improve clarity. Those have the same structure as the nodes of ▲. The main point of this game is that at any level $l = 1, \dots, d-3$, both ▲ and ▼ have a public state containing the k infosets corresponding to the nodes after N's *"root"* of the *mini-game* at depth $l-1$. At each of those public states, both ▲ and ▼ have a single belief containing k information sets and the chance node that leads to the next *mini-game*. Therefore, there are b^k prescriptions at this belief, and each prescription played leads, among the others, to the belief containing k information sets of the next *mini-game*.

Multiplying this factor for each step of the level gives $|\mathcal{H}'| \geq \left(b^{2k}\right)^{d-4}$. □

Upper Bound We now present an upper bound on the number of nodes of the belief game.

Theorem 3.2.6. *Let G be a game with depth d, information complexity k, and maximum branching factor at a node b. The number of nodes in the belief game $\tilde{G}$ is $|\tilde{\mathcal{H}}| \leq b^{2kd+d}$.*

Proof. Consider the algorithm `MakeBeliefGame`. Grouping levels of the belief game $\tilde{G}$ three by three, we have that ▲, ▼, and N each play at a different level in each group, and we have d groups. Moreover, no more than b actions for the chance player and b^k prescriptions are available to each player. We, therefore, have that the number of nodes in-game tree $\tilde{G}$ is $|\tilde{\mathcal{H}}| \leq b^{d(2k+1)}$. □

Discussion The bounds presented in this section highlight the main computational limitation of Algorithm `MakeBeliefGame`, the explicit dependence on depth introduced by explicitly using sequences to distinguish information sets in the belief game.

We remark that we can replace k here with the maximum number of infosets (not the last-infosets) in any public state. We opted not to introduce two different notions of information complexity to have bounds comparable with the TB-DAG ones in Section 3.4.1. We will explore the effects of introducing the different definitions of k in Section 3.6.3.

3.2.5 Regret Minimization on Team Games

This section shows how to find a mixed Nash equilibrium in a generic two-player zero-sum game with imperfect recall G by applying `CFR` on the belief game $\tilde{G}$ obtained by running Algorithm `MakeBeliefGame` on G.

Let $\tilde{\mathcal{X}}$ and $\tilde{\mathcal{Y}}$ be the realization-form strategy spaces for ▲ and ▼ in $\tilde{G}$ derived from the sequence-form representation as in Section 2.1.3. Specifically, vectors $\tilde{\boldsymbol{x}} \in \tilde{\mathcal{X}}$ are indexed by terminal sequences for ▲ in $\tilde{G}$ (similarly for ▼). Such a sequence σ can be identified by a list of beliefs and prescriptions, ending in a singleton belief $\{z\}$ for terminal node $z \in \mathcal{Z}$. For any terminal node z, let $\Sigma^z_{\blacktriangle}$ be the set of terminal sequences for ▲ that end at belief $\{z\}$. Then computing a Nash equilibrium in $\tilde{G}$ (and hence a mixed Nash in G) can be done by solving the max-min problem

$$\max_{\tilde{\boldsymbol{x}} \in \tilde{\mathcal{X}}} \min_{\tilde{\boldsymbol{y}} \in \tilde{\mathcal{Y}}} \sum_{z \in \mathcal{Z}} u(z) \sum_{\tilde{\sigma}_{\blacktriangle} \in \Sigma^z_{\blacktriangle}} \tilde{\boldsymbol{x}}[\tilde{\sigma}_{\blacktriangle}] \sum_{\tilde{\sigma}_{\blacktriangledown} \in \Sigma^z_{\blacktriangledown}} \tilde{\boldsymbol{y}}[\tilde{\sigma}_{\blacktriangledown}]. \tag{3.1}$$

This is equivalent to the max-min problem for the coordinator game (Eq. (2.2)) by setting $\boldsymbol{x}[z] := \sum_{\tilde{\sigma}_{\blacktriangle} \in \Sigma^z_{\blacktriangle}} \tilde{\boldsymbol{x}}[\tilde{\sigma}_{\blacktriangle}]$ (and similar for $\boldsymbol{y}$). That is, from an optimization perspective, what has happened is that we have constructed sets $\tilde{\mathcal{X}}$ and $\tilde{\mathcal{Y}}$ that are described by linear constraints, just like the sequence form, and *project* onto $\mathcal{X}$ and $\mathcal{Y}$ respectively, allowing the reformulation and equivalence of problems.

We now analyze the time complexity and regret of running CFR on $\tilde{G}$. Fix a player, say, ▲. (The same analysis will apply to ▼.) First, recall from Section 2.1.3 that, in a decision problem, a set P of ▲-decision points is called *playable* if there exists a pure strategy of ▲ that plays to all the decision points in P. But by Lemma 3.2.3, the size of any playable set P of ▲ is at most $|\mathcal{H}|$. Further, the branching factor of $\tilde{G}$ is at most b^k, where b is the branching factor of G and k is the information complexity (see Section 3.2.4). Thus, applying multiplicative weights (`MWU`) as the local regret minimizer at each decision point and using Theorem 2.2.1, we have:

Algorithm `DAG-Generic` Generic construction of a regret minimizer $\mathcal{R}$ on $\mathcal{Q}$ from a regret minimizer $\hat{\mathcal{R}}$ on its tree form $\hat{\mathcal{Q}}$.

procedure NEXTSTRATEGY
  $\hat{\boldsymbol{x}}^t \leftarrow \hat{\mathcal{R}}$.NEXTSTRATEGY()
  return $\boldsymbol{D}\hat{\boldsymbol{x}}^t$
procedure OBSERVEUTILITY($\boldsymbol{u}^t$)
  $\hat{\mathcal{R}}$.OBSERVEUTILITY($\boldsymbol{D}^\top \boldsymbol{u}^t$)

Algorithm `DAG-CFR` Counterfactual regret minimization on DAG-form decision problems $\mathcal{Q}$. For each decision point s, $\mathcal{R}_s$ is a regret minimizer on $\Delta(\mathcal{A}_s)$.

procedure NEXTSTRATEGY
  $\boldsymbol{x}^t[\varnothing] \leftarrow 1$
  for each decision point s, in top-down order **do**
    $\boldsymbol{r}_s^t \leftarrow \mathcal{R}_s$.NEXTSTRATEGY()
    $\boldsymbol{x}^t[s*] \leftarrow \sum_{p \in P_s} \boldsymbol{x}^t[p] \boldsymbol{r}_s^t$
  return $\boldsymbol{x}^t$
procedure OBSERVEUTILITY($\boldsymbol{u}^t$)
  $\boldsymbol{v}^t \leftarrow \boldsymbol{u}^t$
  for each decision point s, in bottom-up order **do**
    $\mathcal{R}_s$.OBSERVEUTILITY($\boldsymbol{v}^t[s*]$)
    for $p \in P_s$ **do** $\boldsymbol{v}^t[p] \leftarrow \boldsymbol{v}^t[p] + \langle \boldsymbol{r}_s^t, \boldsymbol{v}^t[s*] \rangle$
  $t \leftarrow t + 1$

Theorem 3.2.7. *After T iterations of* `CFR` *on $\tilde{G}$ with* `MWU` *as the local regret minimizer, the average strategy profile $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}})$ is an $O(\varepsilon)$-Nash equilibrium of G, where*

$$\varepsilon = |\mathcal{H}| \sqrt{\frac{k \log b}{T}}.$$

The per-iteration complexity is linear in the size of $\tilde{G}$.

While the regret above is polynomial in $\mathcal{H}$, the per-iteration complexity depends on the size of $\tilde{G}$, which is worst-case exponentially larger than G, as shown in Section 3.2.4.

3.3 DAG Decision Problems

In this section, we will develop a general theory of DAG-form decision problems and regret minimization on them, analogous to the tree-form theory in Section 2.1.3. Although our main interest in DAG-form decision-making is its application to two-player imperfect-recall games

(which we will develop in Section 3.4), the observations made in this section also have general applicability beyond this setting. For example, since the publications of earlier versions of the present paper, DAG-form decision-making has been applied toward the efficient computation of many other solution concepts, including *linear correlated equilibria* and *optimal extensive-form correlated equilibria* [95, 97, 98, 100, 101].

As one may expect, DAG-form decision problems are identical to tree-form decision problems except that the graph of nodes is allowed to be a DAG, albeit with some restrictions.

Definition 3.3.1. A *DAG-form decision problem* is a DAG with a unique source (root node) $\varnothing$, wherein each node s is either a decision point ($s \in \mathcal{D}$) or an observation point ($s \in \mathcal{S}$), with the following properties.

1. Observation points other than the root have exactly one incoming edge.
2. For any two paths p_1 and p_2 from the root that end at the same node, the last node in common between p_1 and p_2 is a decision point.

As with tree-form decision problems, we will also assume (WLOG) that decision and observation points alternate along every path and that both the root node and all terminal nodes are observation points. A *pure strategy* is once again an assignment of one action to each decision point. The *DAG form* of a pure strategy is the vector $\boldsymbol{x} \in \{0,1\}^{\mathcal{S}}$, where $\boldsymbol{x}[s] = 1$ if there is *some* $\varnothing \to s$ path along which the player plays all actions. A mixed strategy $\boldsymbol{x} \in \mathcal{Q}$ is a convex combination of pure strategies. Since decision points can now have multiple parents, we will use P_s to denote the set of (observation points) parents of a decision point s.

Like tree-form decision problems, the mixed strategy set in a DAG-form decision problem has a convenient representation using linear constraints, namely:

$$\begin{cases} \boldsymbol{x}[\varnothing] = 1 \\ \sum_{p \in P_s} \boldsymbol{x}[p] = \sum_{a \in \mathcal{A}_s} \boldsymbol{x}[sa] & \text{for all} \quad s \in \mathcal{D}. \end{cases} \tag{3.2}$$

DAG-form decision problems and tree-form decision problems are closely related. Of course, all tree-form decision problems are DAG-form decision problems. Conversely, any DAG-form decision problem can be thought of as a "compressed" representation of the tree-form decision problem created by separating out all the different paths through the DAG. While this tree will generally be exponentially larger than the DAG, we will find it useful to compare the DAG and tree representations.

We now formulate a general theory of regret minimization for DAG-form decision problems. We will use hats $(\hat{\mathcal{Q}}, \hat{\mathcal{D}}, \hat{\mathcal{S}}, \hat{\boldsymbol{x}})$ to denote components of the tree form of a generic DAG-form regret minimizer. For each tree-form observation point, observation point $s \in \hat{\mathcal{S}}$ let $\delta(s) \in \mathcal{S}$ be the corresponding observation point in $\mathcal{S}$. Note that, by construction, δ is surjective but not injective unless the DAG happens to be a tree.

We now show how tree-form strategies and utilities correspond to DAG-form strategies and utilities. Concretely, we define a matrix $\boldsymbol{D} \in \mathbb{R}^{\mathcal{S} \times \hat{\mathcal{S}}}$ by $\boldsymbol{D}\hat{\boldsymbol{x}}[s] = \sum_{\hat{s}:\delta(\hat{s})=s} \hat{\boldsymbol{x}}[\hat{s}]$ for all $\hat{\boldsymbol{x}} \in \mathbb{R}^{\hat{\mathcal{S}}}$. This is the matrix of the linear map that transforms tree-form strategies to their corresponding DAG-form strategies. That is, $\boldsymbol{D} : \hat{\mathcal{X}} \to \mathcal{X}$ is a bijection.

Dually, for DAG-form utility vectors $\boldsymbol{u} \in \mathbb{R}^{\mathcal{S}}$, the vector $\boldsymbol{D}^\top \boldsymbol{u} \in \mathbb{R}^{\hat{\mathcal{S}}}$ is a utility vector on the tree form, with the property that $\left\langle \boldsymbol{D}^\top \boldsymbol{u}, \hat{\boldsymbol{x}} \right\rangle = \langle \boldsymbol{u}, \boldsymbol{D}\hat{\boldsymbol{x}} \rangle$ by definition of the inner product. That is, the DAG-form strategy $\boldsymbol{D}\hat{\boldsymbol{x}}$ achieves the same utility against DAG-form utility vector $\boldsymbol{u}$ as the tree-form strategy $\hat{\boldsymbol{x}}$ achieves against the utility $\boldsymbol{D}^\top \hat{\boldsymbol{u}}$.

The relationship between trees and DAGs allows us to use *any* regret minimizer on $\hat{\mathcal{Q}}$ to construct a regret minimizer with the same guarantee on $\mathcal{Q}$. We do this in Algorithm DAG-Generic.

Proposition 3.3.1. *Let $\mathcal{R}$ and $\hat{\mathcal{R}}$ be as in Algorithm* DAG-Generic*. Then the regret of $\mathcal{R}$ with utility sequence $\boldsymbol{u}^1, \dots, \boldsymbol{u}^T$ is equal to the regret of $\hat{\mathcal{R}}$ with utility sequence $\boldsymbol{D}^\top \boldsymbol{u}^1, \dots, \boldsymbol{D}^\top \boldsymbol{u}^T$.*

Proof. Using the fact that $\boldsymbol{D}$ is a bijection, we have

$$\begin{aligned} R_{\mathcal{Q}}^T &= \max_{\boldsymbol{x} \in \mathcal{Q}} \sum_{t=1}^{T} \left\langle \boldsymbol{u}^t, \boldsymbol{x} - \boldsymbol{x}^t \right\rangle \\ &= \max_{\hat{\boldsymbol{x}} \in \hat{\mathcal{Q}}} \sum_{t=1}^{T} \left\langle \boldsymbol{u}^t, \boldsymbol{D}\boldsymbol{x} - \boldsymbol{D}\hat{\boldsymbol{x}}^t \right\rangle \\ &= \max_{\hat{\boldsymbol{x}} \in \hat{\mathcal{Q}}} \sum_{t=1}^{T} \left\langle \boldsymbol{D}^\top \boldsymbol{u}^t, \boldsymbol{x} - \boldsymbol{x}^t \right\rangle = R_{\mathcal{Q}}^T. \end{aligned}$$

□

Applying the transformation DAG-Generic with CFR as the tree-form regret minimizer $\hat{\mathcal{R}}$, we arrive at a DAG form of CFR, which can be simulated efficiently: Algorithm DAG-CFR. One can think of DAG-CFR as a *more efficient implementation* of CFR when the decision tree happens to have a DAG structure. Of course, the regret bound $O(|\mathcal{S}|\sqrt{T})$ is only a worst-case bound; in special cases (such as Theorem 3.2.7), CFR does much better than its worst case, and therefore so will DAG-CFR.

Call a utility vector $\hat{\boldsymbol{u}}$ *consistent* if it is in the image of $\boldsymbol{D}^\top$. That is (expanding the definition of $\boldsymbol{D}^\top$), $\hat{\boldsymbol{u}} \in \mathbb{R}^{\hat{O}}$ is consistent if $\hat{\boldsymbol{u}}[\hat{s}] = \hat{\boldsymbol{u}}[\hat{s}']$ if $\delta(\hat{s}) = \delta(\hat{s}')$. In essence, a DAG-form regret minimizer can "simulate" a tree-form regret minimizer so long as the tree-form regret minimizer's utilities are always consistent. We now formalize this idea.

Theorem 3.3.2 (DAG regret minimization via CFR)**.** `DAG-CFR` *produces the same iterates as* `DAG-Generic` *with* `CFR` *as* $\hat{\mathcal{R}}$*. Therefore, in particular, the regret of* `DAG-CFR` *with utility sequence* $\boldsymbol{u}^1, \dots, \boldsymbol{u}^T$ *is the same as that of* `CFR` *on the tree form with utility sequence* $\hat{\boldsymbol{u}}^1 := \boldsymbol{D}^\top \boldsymbol{u}^1, \dots, \hat{\boldsymbol{u}}^t := \boldsymbol{D}^\top \boldsymbol{u}^T$*. Moreover, the per-iteration runtime of* `DAG-CFR` *is linear in the number of edges in the DAG. In particular, taking any reasonably efficient regret minimizer over simplices, the regret of* `DAG-CFR` *after* T *iterations is at most* $O(|\mathcal{S}|\sqrt{T})$.

Proof. Let $s \in \mathcal{D}$ be any decision point of $\mathcal{Q}$, and let $\hat{s} \in \hat{\mathcal{D}}$ be any decision point in $\hat{\mathcal{Q}}$ with $\delta(\hat{s}) = s$. It is enough to show that the sequence of utility vectors observed by $\mathcal{R}_{\hat{s}}$ when running `DAG-Generic` with `CFR` as $\hat{\mathcal{R}}$ is the same as the sequence of utility vectors observed by $\mathcal{R}_s$ in `DAG-CFR`. We show this by induction on the decision points $\hat{s}$, leaves first.

First, if $\hat{s}$ has no decision point descendants, then the claim is trivial because, by construction of $\boldsymbol{D}^\top$, we have $\hat{\boldsymbol{u}}^t[\hat{s}a] = \boldsymbol{u}^t[sa]$ for every $a \in A_s$. Now let $\hat{s} \in \hat{\mathcal{D}}$ be any internal node and $s = \delta(\hat{s})$. By inductive hypothesis, for every decision point descendant $\hat{s}'$ of $\hat{s}$, at every timestep t, $\mathcal{R}_{\hat{s}'}$ receives the same utility vector as $\mathcal{R}_{s'}$ where $s' = \delta(\hat{s}')$, and thus produces the same behavioral strategy $\boldsymbol{r}^t_{s'} = \hat{\boldsymbol{r}}^t_{\hat{s}'}$. Thus, at any timestep t the utility vector $\hat{\boldsymbol{v}}^t[\hat{s}*]$ that is passed to $\mathcal{R}_{\hat{s}}$ is given by

$$\begin{aligned}
\hat{\boldsymbol{v}}^t[\hat{s}a] &= \hat{\boldsymbol{u}}^t[\hat{s}a] + \sum_{\hat{s}' : p_{\hat{s}'} = \hat{s}a} \left\langle \hat{\boldsymbol{r}}^t_{\hat{s}'}, \hat{\boldsymbol{v}}^t[\hat{s}'*] \right\rangle \\
&= \boldsymbol{u}^t[sa] + \sum_{\hat{s}' : p_{\hat{s}'} = \hat{s}a} \left\langle \boldsymbol{r}^t_{s'}, \boldsymbol{v}^t[s'*] \right\rangle \\
&= \boldsymbol{u}^t[sa] + \sum_{s' : sa \in P_s} \left\langle \boldsymbol{r}^t_{s'}, \boldsymbol{v}^t[s'*] \right\rangle \\
&= \boldsymbol{v}^t[sa]
\end{aligned}$$

where once again we use the notation $s' := \delta(\hat{s}')$, and the inductive hypothesis is used in the second equality on every term in the sum. □

Algorithm `ConstructTB-DAG` Constructing the TB-DAG. Inputs: imperfect-recall game G, player i

procedure MAKEDECISIONPOINT(B) ▷ $B \subseteq \mathcal{H}$ *is a belief*
  if a decision point s with belief B already exists **then return** s
  if $B = \{z\}$ for $z \in \mathcal{Z}$ **then return** new terminal node with belief $\{z\}$
  $s \leftarrow$ new decision point with belief B
  for each prescription $\boldsymbol{a} \in A^i_B$ **do**
    add edge $s \to$ MAKEOBSERVATIONPOINT($B\boldsymbol{a}$)
  return s
procedure MAKEOBSERVATIONPOINT(H)
  $s \leftarrow$ new observation point
  for each $B \in$ SPLITBELIEF$_i(H)$ **do**
    add edge $s \to$ MAKEDECISIONPOINT(B)
  return s

3.4 DAG Decision Problems in Team Games

In Section 3.2.5, it emerged that applying the `CFR` procedure to the belief game produced by `MakeBeliefGame` suffers from the size of the game to solve, which may grow exponentially fast as shown in Section 3.2.4. In this section, we show how DAG decision problems can greatly reduce the inefficiencies caused by the previous construction.

The main observation is that `MakeBeliefGame` enforces perfect recallness of the belief game by including the players' sequences in the infoset definition. On the other hand, the strategic aspect of the game is governed solely by the nodes contained in beliefs. Once the set of possible nodes is fixed, the exact sequence of prescriptions and observations is irrelevant, as the game will evolve identically from that point onwards. This observation leads to considering a DAG structure for the decision problems, where beliefs identify decision nodes.

The Nash equilibrium problem in $\tilde{G}$, namely (3.1), indeed guarantees that both players' utility vectors will be consistent with respect to these DAG-form decision problems. We will call the resulting DAG decision problems the *team belief DAGs* (TB-DAGs)[12]. Therefore, using `DAG-CFR` as the regret minimizer for both players, we recover the regret guarantee of Theorem 3.2.7 with *per-iteration complexity* proportional to the total size of both DAGs.

However, this proposed algorithm still depends on the size of $\tilde{G}$, because, naively, to construct the DAG representations, one first constructs the augmented game $\tilde{G}$, and only then does the merging of decision points to create the DAGs. We, therefore, describe an

[12]We keep the name *team belief DAG* for continuity with previous versions of the work, even though it applies equally well in the team and imperfect recall settings.

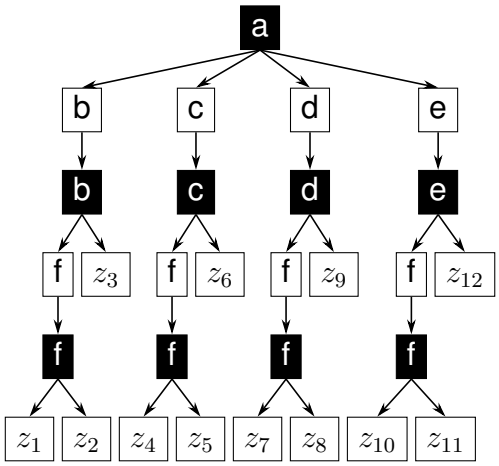


Figure 3.6: An example TB-DAG corresponding to the team game from Figure 3.1 and belief game from Figure 3.3. Beliefs and observation points are represented in black and white respectively. Since the imperfect-recall of the team in the original game is only due to forgetting actions played, the DAG in this case is a tree and there is only a constant factor increase in size with respect the original game.

algorithm `ConstructTB-DAG` that recursively constructs the team belief DAGs *directly from the original game* G, thus bypassing the construction of $\tilde{G}$. Therefore, we have the following result. For each player $i \in \{\blacktriangle, \blacktriangledown\}$, let E_i be the number of edges in the TB-DAG of player i.

Theorem 3.4.1 (TB-DAG and CFR)**.** *Suppose that both players run the algorithm* `Construct TB-DAG` *to construct their strategy spaces* $\tilde{\mathcal{X}}, \tilde{\mathcal{Y}}$*, and then run* `DAG-CFR`*. Then their average strategy profile converges at the rate shown in Theorem 3.2.7, and the per-iteration runtime complexity is* $O(E_{\blacktriangle} + E_{\blacktriangledown})$.

Proof. `ConstructTB-DAG` is designed to construct a DAG-form decision problem whose tree form corresponds precisely to the decision problem faced by player i in the belief game. Thus, it only remains to show that Theorem 3.3.2 applies. That is, we need to show that, in the belief game $\tilde{G}$, the utility vectors $\hat{\boldsymbol{u}}$ that would be observed by `CFR` for $\blacktriangle$ (the same proof would hold for $\blacktriangledown$) are such that $\hat{\boldsymbol{u}}[\tilde{z}] = \hat{\boldsymbol{u}}[\tilde{z}']$ whenever nodes $\tilde{z}, \tilde{z}' \in \tilde{\mathcal{Z}}$ of the belief game represent the same node $z \in \mathcal{Z}$ of the original game. But indeed, for any pure strategy $\tilde{\boldsymbol{y}} \in \tilde{\mathcal{Y}}$, by Theorem 3.2.1 there is a pure strategy $\boldsymbol{y} \in \tilde{\mathcal{Y}}$ such that $\tilde{\boldsymbol{y}}[\tilde{z}] = \tilde{\boldsymbol{y}}[\tilde{z}'] = \boldsymbol{y}[z]$. Thus, if $\blacktriangledown$ plays $\tilde{\boldsymbol{y}}$, the utility observed by $\blacktriangle$ will be given by

$$\hat{\boldsymbol{u}}[\tilde{z}] = \tilde{\boldsymbol{p}}[\tilde{z}]\tilde{\boldsymbol{y}}[\tilde{z}] = \boldsymbol{p}[z]\boldsymbol{y}[z] = \tilde{\boldsymbol{p}}[\tilde{z}']\tilde{\boldsymbol{y}}[\tilde{z}'] = \hat{\boldsymbol{u}}[\tilde{z}']$$

which is indeed consistent in the required sense. □

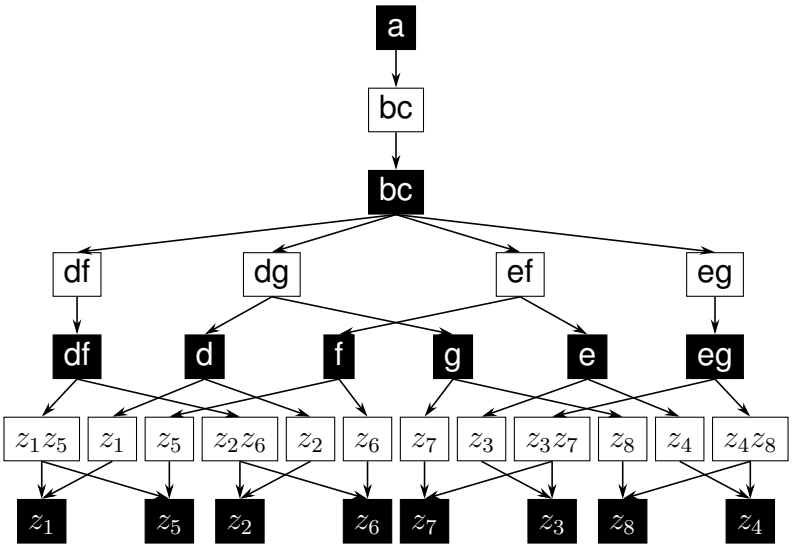


Figure 3.7: Another example TB-DAG corresponding to the team game from Figure 3.2 and belief game from Figure 3.4. Beliefs and observation points are represented in black and white respectively. Note that endgames that are repeated by the belief game are now merged.

Figures 3.1 and 3.2 are shown in Figures 3.3 and 3.4 respectively.

As an example, the TB-DAGs corresponding to the team game from Figures 3.1 and 3.2 are reported in Figures 3.6 and 3.7 respectively.

3.4.1 Size Analysis of the TB-DAG

The per-iteration runtime above depends on the number of edges in the TB-DAGs, so it is important to bound this number. We will do so now. Here, we use the same notation as in Section 3.2.4.

Theorem 3.4.2. *For each player i, we have $E_i \leq |\mathcal{H}|(b+1)^{k+1}$.*

Proof. At any given public state P of player i, let $\mathcal{I}_i(P) = \bigcup_{h \in P} \mathcal{I}_i(h)$ be the set of last-infosets. Consider a pure strategy $\boldsymbol{\pi} \in \Pi_i$. For each last-infoset $I \in \mathcal{I}_i(P)$, let $\boldsymbol{\pi}_P$ be the partial strategy defined only on infosets $I \in \mathcal{I}_i(P)$, by $\pi_P[I] = \boldsymbol{\pi}[I] \in A_I$ if $\boldsymbol{\pi}$ plays to at least one node in I, and $\pi_P[I] = \perp$ if it does not. There are thus at most $(b+1)^k$ such possible partial strategies, since $|\mathcal{I}_i(h)| \leq k$ by definition. By construction, each partial strategy $\boldsymbol{\pi}_P$ completely determines which nodes in P are reached by $\boldsymbol{\pi}$, as well as the actions played at any such nodes. Thus, $\boldsymbol{\pi}_P$ induces a disjoint collection of observation points $S \subseteq \mathcal{S}$ such that we have $B\boldsymbol{a} \in S$ for each observation point $B \subseteq P$. Now, each observation point $B\boldsymbol{a}$ has at most

one incoming edge and at most $|B\boldsymbol{a}|$ outcoming edges. Since the observation points $B\boldsymbol{a}$ are disjoint and have a total size at most $|P|b$, the total number of edges at public state P is at most $|P|(b+1)^{k+1}$. The proof finishes by summing over public states, noting that every node is in exactly one public state. □

Thus, from Theorem 3.4.1 and Theorem 3.4.2, it follows that:

Theorem 3.4.3 (Main theorem)**.** *Any given imperfect-recall game G can be solved by constructing the TB-DAGs using* `ConstructTB-DAG` *and running* `DAG-CFR`*. After T iterations, the average strategy profile will be an $O(\varepsilon)$-Nash equilibrium where ε is as in Theorem 3.2.7. The per-iteration complexity is $O(|\mathcal{H}|(b+1)^{k+1})$.*

Before proceeding, it is instructive to briefly compare Theorem 3.4.3 to Theorem 3.2.6. The latter result gives a per-iteration complexity that is $O(b^{d(2k+1)})$. Thus, Theorem 3.4.3 is strictly superior: for Theorem 3.2.6 to be superior, we would need to have $b^{d(2k+1)} < |\mathcal{H}|(b+1)^{k+1}$, which is impossible for $d \geq 1, k \geq 1, b \geq 2$. We give a more detailed comparison between the two bounds in Section 3.6.3.

3.4.2 Fixed-Parameter Hardness

Given the above result, one may ask whether the b can be removed more generally. It turns out that it cannot. Before proceeding, we must introduce some fundamental concepts surrounding *fixed-parameter tractability*.

Definition 3.4.1. A problem is *fixed-parameter tractable* with respect to a parameter k if it admits an algorithm whose runtime on inputs of length N is $f(k)\mathsf{poly}(N)$, for some arbitrary function f.

The k-*CLIQUE* problem is to, given a graph G and an integer k, decide where G has a k-clique. The computational assumption $\mathsf{FPT} \neq \mathsf{W}[1]$ states that k-CLIQUE is not fixed-parameter tractable. It is implied by the exponential time hypothesis [20].

Theorem 3.4.4. *Assuming $\mathsf{FPT} \neq \mathsf{W}[1]$, there is no algorithm for computing the mixed Nash value of a* one-player *game of imperfect recall whose runtime has the form $f(k)\mathsf{poly}(|\mathcal{H}|)$ where f is an arbitrary function.*

Proof. We reduce from k-CLIQUE. Given a graph $G = (V, E)$, we construct the following game with a single team with two members. Nature selects two vertices $v_1, v_2 \in V$ independently and uniformly at random. Then, both team members privately observe the vertices that have been assigned to them, and select bits $b_1, b_2 \in \{0, 1\}$.

Utilities are defined as the sum of the following terms.

- If $v_1 = v_2$ and $b_1 \neq b_2$ then the team scores $-|V|$.
- If $b_1 = b_2 = 1$ and $(v_1, v_2) \neq E$ and $v_1 \neq v_2$, then the team scores $-|V|$.
- If $b_1 = b_2 = 1$ then the team scores 1.

We claim that the value of this game is $\geq k$ if and only if G has a k-clique. Clearly if G has a k-clique then the value of the game is at least k: if both members play bits according to the k-clique (*i.e.*, $b_i = 1$ if v_i is in the clique) then they will score k.

Conversely, suppose there is no k-clique. Note first that we can assume WLOG that the two members play the same strategy. (If they do not, they get utility at most 0, but playing the all-zeros vector also gets utility 0.) A pure strategy is identified by the subset $S \subseteq V$ of vertices at which the player plays 1. Since both members are playing the same strategy, the utility function reduces to $|S| - |V| \cdot m$, where $m = |\{(i,j) : i \neq j; i, j \in S; (i,j) \notin E\}|$ is the number of times the clique constraint is violated. Once again, this number is ≤ 0 unless S is a clique, and if S is a clique then $|S| < k$ since there is no k-clique. □

Thus, it is impossible to replace the b in Theorem 3.4.3 with any absolute constant.

3.4.3 Branching Factor Reduction

Despite the worst-case hardness of removing the b in Theorem 3.4.3, it turns out that, for a natural class of games, we *can* remove b. In this subsection we will discuss games with *action recall*, and prove that in such games, it is without loss of generality to assume that the branching factor is 2. Intuitively, a player i has action recall if it remembers the entire sequence of actions she has taken in the past (including the timesteps at which such actions were taken). More formally:

Definition 3.4.2. At a node $h \in \mathcal{H}$, let $(a_1, \ldots, a_L) \in \mathcal{A}^L$ be the list of actions taken on the $\varnothing \to h$ path. Define the *action sequence* of player i as the sequence $(a'_1, \ldots, a'_L) \in (\mathcal{A} \sqcup \{\perp\})^L$ where $a'_\ell = a_\ell$ if action a_ℓ was taken by player i, and $a'_\ell = \perp$ otherwise. We say that player i has *action recall* if, for every infoset I of player i, every node in I shares the same action sequence.

Theorem 3.4.5. *Given a two-player imperfect-recall game G where both players have perfect action recall, there exists another strategically-equivalent game G' such that the branching*

	Team vs Player	Team vs Team
TMECor	NP-complete [45]	Δ_2^{P}-complete (This paper, Theorems 3.5.5 and 3.5.6)
TME	NP-complete [45]	Σ_2^{P}-complete (This paper, Theorems 3.5.2 and 3.5.3)

Table 3.1: Summary of most of the complexity results shown in Section 3.5.

factor of G' is at most 2 at each $h \in \mathcal{H}_{\blacktriangle} \cup \mathcal{H}_{\blacktriangledown}$, the parameter k in G' is the same as it in G, and the size of the game has increased by a factor of $O(\log |\mathcal{A}|)$.

Proof. To every action $a \in \mathcal{A}$, we associate a unique bitstring of length at most $\ell = O(\log |\mathcal{A}|)$. Assume without loss of generality, for simplicity of notation, that all such bitstrings end with a 0. We will call bitstrings $\tilde{a} \in \{0,1\}^{<\ell}$ "partial actions".

We replace every internal node $h \in \mathcal{H} \setminus \mathcal{Z}$ with a binary tree of depth ℓ, where bitstrings that are not prefixes of any action $a \in A_h$ are pruned. If h and h' are in the same infoset I, then for every partial action $\tilde{a} \in \{0,1\}^{<\ell}$ we connect $h\tilde{a}$ and $h'\tilde{a}$ in an infoset, which we will call $I\tilde{a}$. This creates a new game G', whose parameters we must now analyze.

For each node $h \in \mathcal{H} \setminus \mathcal{Z}$ and action a, G' has createad at most $O(\log |\mathcal{A}|)$ additional nodes (namely, the nodes $h\tilde{a}$ where $\tilde{a}$ is a prefix of a). Thus, the size of G' is at most $O(|\mathcal{H}| \log |\mathcal{A}|)$.

It thus remains only to bound the information complexity of G'. Let P be a public state of player i in G'. By construction of action sequences, P contains either only nodes in $\mathcal{H}_i$, or only nodes not in $\mathcal{H}_i$. In the latter case, there is nothing to check. In the former case, we have $P \subseteq \{h\tilde{a} : h \in P'\}$ for some public state P' of G, and partial action $\tilde{a} \in \{0,1\}^{<\ell}$. Now let I be a last-infoset of P' in G. Then I induces at most one last-infoset in P: namely, if I overlaps P, then this infoset is simply $I\tilde{a}$; otherwise, it is the infoset $I\tilde{a}'$ where $\tilde{a}' \in \{0,1\}^{\ell-1}$ is the partial action such that action $\tilde{a}'0$ leads to P (which must be uniquely defined by definition of action recall). Thus, P has as many last-infosets as P', so the information complexity of G' is, at most, the information complexity of G. □

Corollary 3.4.6. *In games where both players have action recall, Theorem 3.4.3 applies with the per-iteration runtime replaced with $O(3^k |\mathcal{H}| \log |\mathcal{A}|)$.*

3.5 Complexity of Adversarial Team Games

Here, we state and prove several results about the *complexity* of finding various equilibria in timeable two-player zero-sum games of imperfect recall.

In all cases, the goal is to solve the following promise problem: given game G, threshold value v, and error $\varepsilon > 0$ (where all the numbers are rational), determine whether the (mixed or behavioral) value of the game is $\geq v$, or $< v - \varepsilon$. The allowance of an exponentially-small error is to circumvent issues of bit complexity that arise since exact behavioral max-min strategies may not have rational coefficients [45]. Throughout this section, it will often be convenient to formulate the hardness gadgets in terms of adversarial team games. We will thus freely utilize the analogy between adversarial team games and coordinator games. For mixed Nash and behavioral Nash respectively, we will refer to the problems as MIXED and BEHAVIORAL.

Although we do not explicitly state it in the theorem statements, all the hardness results are proven by constructing adversarial team games in which both teams have a constant number of players.

Theorem 3.5.1 (45, 22, 89)**.** *Finding the optimal strategy in a one-player, timeable game of imperfect recall is* NP*-hard.*

Proof. We essentially follow the proof of Chu and Halpern [22], but we are more explicit about ordering the turns so that we can later bound the information complexity of our game. Given a 3-CNF formula ϕ with m clauses and n variables, construct the following game with two team members on the maximizing team and no opponent. Nature picks a random clause $j \in [m]$ and a random variable x_i that appears in clause j. P1 privately observes the variable and is asked to assign a value (either true or false) to it. P2 then privately observes the clause j (but not the variable i, nor P1's assignment), and must pick one of the three variables in the clause. The team wins if P2 picks variable i and P1 assigns the value to i that makes the clause true.

Note that the pure strategies of P1 are precisely the assignments $x \in \{0,1\}^n$. We claim that if P1 plays assignment x, the best value for the team is precisely $1/3$ of the maximum number of clauses satisfied by x. To see this, note that the team will lose with probability $2/3$ no matter what since P2 has only a $1/3$ chance of picking the same variable that was selected by nature. Conversely, if P2 happens to pick the right variable, the team wins if and only if P1's assignment to that variable satisfies the clause j. Thus, P2's best strategy, given P1 strategy x, is always to pick the satisfying variable in the clause, if there is one. By construction, this achieves the desired value. □

The above proof also shows, by the PCP theorem [38], that there exists an absolute constant ε such that computing the optimal value in a team game with no adversary to accuracy ε is NP-hard. Finally, the information complexity of the game used in the above construction is[13]

[13]Here we use the ordering of the players: namely, we have $k = n$ only because P1 plays before P2. If the order of the players were flipped, we would instead have $k = m$.

$k = n$, and the branching factor can be made an absolute constant by splitting the root chance node into $\Theta(\log m)$ layers. Finally, the size of the game is $O(mn)$. Thus, Theorem 3.4.3 implies a SAT-solving algorithm whose runtime is $2^{O(n)}$. Thus, in particular, the appearance of k in the exponent in Theorem 3.4.3 is unavoidable: if the k were replaced by any $o(k)$ term, then SAT would have a $2^{o(n)}$-time algorithm, violating the commonly-believed exponential time hypothesis.

3.5.1 Behavioral Max-Min Strategies

We first show results for BEHAVIORALMAXMIN. In particular, we will show that it is Σ_2^{P}-complete, first by showing inclusion and then constructing a gadget game to show completeness. (Recall that the inclusion will require an ε-approximation because exact behavioral max-min strategies may contain irrational values.)

Theorem 3.5.2. BEHAVIORALMAXMIN *is in* Σ_2^{P}. *If* ▼ *has perfect recall, it is in* NP.

Proof. Consider a behavioral max-min strategy represented by a distribution over the actions at each information set I. Let $\delta > 0$, and consider rounding each entry of the behavioral-form strategy by at most an additive δ so that the resulting strategy is rational. Let $\boldsymbol{x}'$ be the correlation plan of the resulting strategy. Thus, for any given terminal node s, the resulting reach probability $x'[s]$ is perturbed by at most an additive $O(N\delta)$ where N is the number of nodes in the game. Thus, $\|\boldsymbol{x}' - \boldsymbol{x}\|_1 \leq O(N^2\delta)$. Thus, for any realization-form strategy $\boldsymbol{y}$ for the opponent, we have $|\langle \boldsymbol{x}' - \boldsymbol{x}, \boldsymbol{A}\boldsymbol{y}\rangle| \leq \|\boldsymbol{x}' - \boldsymbol{x}\|_1 \|\boldsymbol{A}\boldsymbol{y}\|_\infty \leq O(N^2\delta)$, so x' is $O(N^2\delta)$-close to the optimal solution. Taking $\delta < O(\varepsilon/N^2)$ thus concludes the proof. □

Theorem 3.5.3. BEHAVIORALMAXMIN *is* Σ_2^{P}*-hard, even when the number of players is constant and there is no chance.*

Proof. We first give a reduction involving chance, then show how to relax this condition. We reduce from ∃∀3-SAT, which is known to be Σ_2^{P}-complete [77]. The ∃∀3-SAT problem is to, given a 3-DNF formula $\phi(X, Y)$, determine whether $\exists X\ \forall Y\ \phi(X, Y)$ holds.

Given a 3-DNF formula ϕ with m clauses, n_1 variables in X, and n_2 variables in Y, construct the following game between △ with 3 players and ▽ with 3 players. Nature chooses three variables x_1, x_2, x_3 from X and three variables y_1, y_2, y_3 from Y. For each variable x_i (respectively y_i), Player i of △ (respectively ▽) is asked for an assignment to the variable.

If any two players of Δ (respectively ∇) have the same variable but differ in their assignment, Δ gets value $-M$ (respectively M) where M is a large value. In addition, Δ gets value 1 if at least one term in the 3-DNF ϕ is satisfied by the assignments of Δ and ∇.

Let $n = \max(n_1, n_2)$. We complete the proof by showing that Δ gets at least $1/n^3$ if and only if $\exists x\ \forall y\ \phi(x, y)$ holds; otherwise, their value is at most 0. We first show that for large enough M, since players of Δ cannot correlate, Δ's pure strategies are dominant over non-pure ones.

Lemma 3.5.4. *Let $x \in X$ be a variable and $p \leq 1/2$ be the probability that Player i plays their less-likely action for x in a behavioral strategy. If $p > 0$ and $M \geq n_1$, then this strategy is strictly dominated by the strategy under which Player i only plays their more-likely action.*

Proof. Whenever variable x is picked for Player i and one of their teammate (probability strictly larger than $1/n_1^2$), the penalty incurred by the two players is strictly more than $(M/n_1^2)(p(1-q) + q(1-p)) \geq (M/n_1^2)(p + q(1/2 - p)) \geq (M/n_1^2)p$. On the other hand, Player i gains no more than 1 by playing their less-likely action (probability p/n_1). Hence, if $M \geq n_1$, any strategy with $p > 0$ is dominated by a pure strategy. □

The pure strategies of a player of Δ (respectively ∇) are precisely the assignments in $\{0,1\}^X$ (respectively in $\{0,1\}^Y$). By a similar argument, it is straightforward to show that it is a dominant strategy for Δ (respectively for ∇) to let all players pick the same assignment to avoid a large penalty.

The hardness then follows from the following observation. If $\exists X\ \forall Y\ \phi(X, Y)$ holds, then Δ can play the corresponding assignment to force a value of at least $1/n^3$: no matter what assignment ∇ picks, at least one term in ϕ is true, which is discovered with a probability of at least $1/n^3$ (whenever all the variables in such a term are picked by Nature). On the other hand, if $\exists X\ \forall Y\ \phi(X, Y)$ does not hold, then no matter what assignment Δ picks, there is an assignment that ∇ can pick such that none of the terms is satisfied, which forces a value of 0 for Δ.

To show that the same hardness holds even when there is no chance, we use the following gadget to eliminate the need for Nature. Let us introduce a new Δ-player called Δ-Nature and a new ∇-player called ∇-Spoiler. The gadget will be such that Δ-Nature can incur a large penalty whenever they do not mimic perfectly Nature's behavior. More concretely, as Nature in the construction above, Δ-Nature picks $s = (x_1, x_2, x_3, y_1, y_2, y_3) \in X^3 \times Y^3$. ∇-Spoiler then guesses Δ-Nature's choice by picking $s' \in X^3 \times Y^3$. Δ receives $-N(n_1^3 n_2^3 - 1)$ If $s = s'$, otherwise N, where N is a large number. The game then continues as in the construction above.

By a similar argument to the one used in the proof of the lemma above, Δ-Nature's dominant strategy is to pick s uniformly at random. Since Δ cannot correlate, the game plays exactly like the construction above; Δ can force a value of $1/n^3$ if and only if $\exists X\ \forall Y\ \phi(X, Y)$ holds. □

3.5.2 Mixed Nash Equilibria

We now show results for MIXEDNASH, namely, we will show that MIXEDNASH is Δ_2^{P}-complete, again by showing inclusion first and then completeness. Unlike for BEHAVIORAL-MAXMIN (Theorem 3.5.2), here we will directly construct a separation oracle, and thus be able to recover algorithms for *exact* computation.

Theorem 3.5.5. MIXEDNASH *is in* Δ_2^{P}*, even for exact computation* ($\varepsilon = 0$).

Proof. Let $\mathcal{X} \subset \mathbb{R}^m, \mathcal{Y} \subset \mathbb{R}^n$ be the space of realization-form pure strategies of both players, and $\mathcal{A}$ be the payoff matrix. Then our goal is to decide whether the polytope

$$\mathcal{X}^* := \left\{ \boldsymbol{x} \in \mathbb{R}^m : \begin{array}{cc} ① & \boldsymbol{x} \in \operatorname{co} \mathcal{X}, \\ ② & \boldsymbol{y}^\top \boldsymbol{A}\boldsymbol{x} \le v\ \forall \boldsymbol{y} \in \mathcal{Y} \end{array} \right\}$$

is empty. We will show how to separate over $\mathcal{X}^*$ with a mixed-integer linear programming oracle, which suffices to complete the proof because such a separating oracle can be used to run the ellipsoid algorithm.

Given a candidate solution $\boldsymbol{x}$, we check both constraints. If ② is violated for some $\boldsymbol{y}^* \in \mathcal{Y}$, then $\boldsymbol{A}\boldsymbol{y}^*$ is a separating direction; such $\boldsymbol{y}^*$ can be found by an integer programming oracle. If ① is violated, then a separating direction can be found because (strong) separation and optimization are equivalent for well-described polytopes [35], and optimization over $\operatorname{co} \mathcal{X}$ is an integer program. □

Theorem 3.5.6. MIXEDNASH *is* Δ_2^{P}*-hard, even when the number of players is constant and there is no chance.*

Proof. We reduce from Last-SAT, which is known to be Δ_2^{P}-complete [49]. The Last-SAT problem is to, given a 3-CNF formula $\phi(x)$, decide whether the lexicographically last satisfying assignment of ϕ has a 1 in the least-significant bit.

Given a 3-CNF formula ϕ with m clauses over a set of n variables $X = \{x_1, \dots x_n\}$, we construct the following zero-sum game with 3 players on each team. First, nature chooses 6 variables $x_1^{\Delta}, x_2^{\Delta}, x_3^{\Delta}, x_1^{\nabla}, x_2^{\nabla}, x_3^{\nabla} \in X$ independently and uniformly at random. Player i of Δ

(respectively of ∇) is asked concurrently and independently to assign either true or false to the variable x_i^{Δ} (respectively x_i^{∇}). The payoff for Δ is a sum of terms, determined by the following conditions.

- If any two players of Δ (respectively of ∇) assign different values to the same variable, Δ receives $-N^2$ (respectively $+N^2$), where N is a large number.
- If any clause in ϕ is rendered false by the assignment of Δ (respectively ∇), Δ receives $-(n^2 + N)$ (respectively $+N$).
- If the variable shown to player 1 of Δ (resp. of ∇) is $x_k \in X$ and they assign true to this variable, then Δ receives $+2^{n-k}$ (respectively -2^{n-k}).
- If the variable shown to player 1 of Δ is the last variable x_n and they assign false to this variable, then Δ receives an additional penalty of -1.

It is straightforward to verify if N is large enough (e.g. $N = n^2 2^n$), for both Δ and ∇, the dominant pure strategy is to let all the players in the same team pick the lexicographically last maximum-satisfying assignment $X' \subseteq X$. In particular, this strategy is also the dominant mixed strategy. Now consider Δ's payoff when both teams play this pure strategy.

- If ϕ is not satisfiable, then every clause that is false under the assignment X' is detected with a probability of at least $1/n^3$ (whenever all the variables in the clause are picked by Nature for the same team). This means the second term yields to Δ an expected payoff at most $-1/n$. Other terms yields a non-positive expected payoff.
- If ϕ is satisfiable, then the only non-zero expected payoff for Δ comes from the last term, which is 0 if $x_n \in X'$, otherwise $-1/n$.

Therefore, the game's value is at least 0 if and only if the Last-SAT instance is true; otherwise, it is at most $-1/n$.

To eliminate nature from the construction, we use a similar gadget to the one in the proof of 3.5.2. Let us introduce a new Δ-player called Δ-Nature and two new ∇-player called ∇-Spoiler and ∇-Anticorrelator. The gadget will be such that ∇-Spoiler (respectively ∇-Anticorrelator) can impose a strictly negative expected payoff whenever Δ-Nature does not mimic perfectly Nature's behavior (respectively whenever other players of Δ correlate with Δ-Nature).

More concretely, the game proceeds as follows: First, Δ-Nature picks a sextuple from X^6. ∇-Spoiler then decides whether to guess the choice of Δ-Nature without observing it. If they

do, Δ receives $+1$, and an additional $-n^6$ if ∇-Spoiler guesses correctly. If ∇-Spoiler decides not to guess, then the game continues as before: each player is shown their variable and nothing else, to which they assign either true or false. Then, ∇-Anticorrelator can choose to do nothing or to pick an $i \in 1, 2, 3$. In the former case, the payoff of Δ is computed as in the construction above. In the latter case, ∇-Anticorrelator observes the variable x_i^Δ shown to player i of Δ and the truth value that player i assigns to this variable. ∇-Anticorrelator then guesses the other 5 variables picked by Δ-Nature; Δ receives $+1$, and an additional $-n^5$ if ∇-Anticorrelator guesses correctly.

To see that this construction works, notice that if Δ-Nature does not pick the sextuple uniformly at random, then ∇-Spoiler can guess correctly the sextuple with a probability strictly larger than $1/n^6$, thus yielding a strictly negative reward to Δ. Similarly, if player i's assignment to x_i^Δ depends on the other 5 variables (which they cannot observe in the construction with chance above), then ∇-Anticorrelator can guess correctly with a probability strictly larger than $1/n^5$. Therefore, Δ-Nature's dominant strategy is to pick the sextuple uniformly at random; it is also dominant for the other 3 players of Δ not to correlate to Δ-Nature. It is then straightforward to see that this game is equivalent to the game with chance above, and the value of the game is 0 if and only if the Last-SAT instance is true. □

3.6 Discussion

In this section, we discuss important details that may help the interested reader in clarifying some technical aspects of our contributions.

3.6.1 Public States vs Observations

In this section, we discuss in depth the difference between public *states* and public *observations*. Intuitively, the difference is that observations are *localized* to a particular node in the TB-DAG: if a fact is public to the team *conditional on the part of the team strategy that has been played to reach this point*, then it is an observation. On the other hand, public *states* only encode *unconditionally* public information. As we will see, using observations is strictly preferable to public states from both conceptual and theoretical perspectives.

Comparision to using public states We envision an alternative construction of the TB-DAG in which the team coordinator observes only the *public state* containing the current node. That

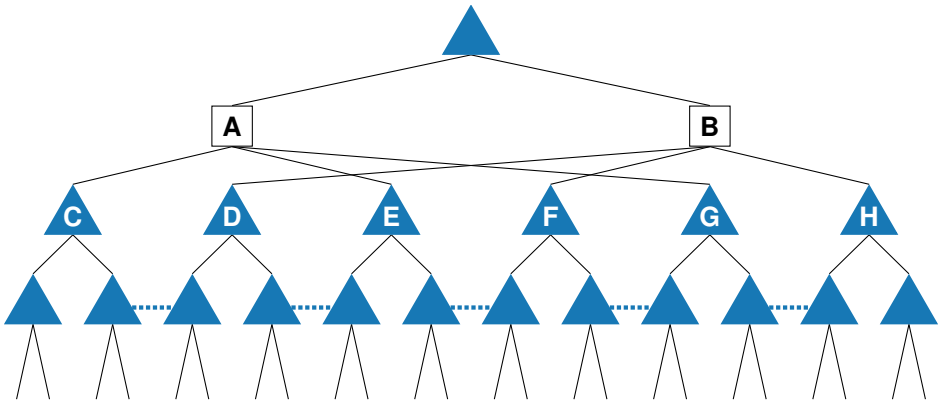


Figure 3.8: A game showing that public state-based approaches do not subsume inflation.

is, the definition of SPLITBELIEF is replaced by:

$$\text{SPLITBELIEF}_i^{\text{pub}}(H, h) \coloneqq H \cap P \text{ where } h \in P \in \mathsf{P}_i.$$

and $\text{SPLITBELIEF}_i^{\text{pub}}(H)$ defined analogously. Then, in `ConstructTB-DAG`, we replace $\text{SPLITBELIEF}_i(H)$ with $\text{SPLITBELIEF}_i^{\text{pub}}(H)$. We will call this new construction the *public-state TB-DAG* and spend the rest of this subsection contrasting it with the (observation) TB-DAG constructed by `ConstructTB-DAG`.

Our first result is that the TB-DAG can never be too much larger than the public state TB-DAG:

Proposition 3.6.1. *Let N and N' be the number of nodes in the TB-DAG and public state TB-DAG, respectively. Then $N \leq 2pN'$, where p is the largest size (in number of nodes) of any belief in the public state TB-DAG.*

Proof. Let B be any belief in the public state TB-DAG. In the (non-public-state) TB-DAG, B splits into disjoint beliefs $B_1, \ldots, B_m$. Let $A_1, \ldots, A_m$ be the sizes of the prescription spaces at $B_1, \ldots, B_m$, respectively. Then B has $A_1 A_2 \ldots A_m$ children, so B induces $1 + A_1 A_2 \ldots A_m$ nodes in the public state TB-DAG. On the other hand, the beliefs $B_1, \ldots, B_m$ in the TB-DAG will have $A_1, \ldots, A_m$ children respectively, accounting for a total of $m + A_1 + \cdots + A_m \leq 2mA_1 \ldots A_m$ nodes. Now, observing simply that $m \leq p$ completes the proof. □

Thus, using observations is never much worse than using public states.

Comparision to using inflated public states Previous works [96, 15] used public states and required to *inflate* the information partition of the team before the new representation can be constructed. *Complete inflation* [43], which we simply call *inflation* for short, is an algorithm that splits an infoset I into two infosets $I = I_1 \sqcup I_2$ if no pure strategy of the team can simultaneously play to a node in I_1 and a node in I_2, and repeats this process until no more such splits are possible. This preserves strategic equivalence. However, inflation can lead to the break-up of public states, reducing the size of public state TB-DAG.

Indeed, consider the game in Figure 3.8. Due to the information sets marked in the last layer of the game tree, the connectivity graph contains a path C—D—E—...—H. Therefore, {C, D, ..., H} form a public state. Also, the combinations CEG and DFH can be reached (if the player at the root plays left or right, respectively). Therefore, CEG and DFH are beliefs in the public-state TB-DAG. In the observation TB-DAG, consider, for example, what happens if the left action is played at the root so that C, E, and G are all reached. Note that there are no edges connecting C, E, and G—the path connecting C to E in the connectivity graph passes through D, which is not reached; therefore, C, E, and G are three different observations and hence three different beliefs, resulting in an exponentially-smaller TB-DAG. Inflation would remove the nontrivial information sets in the second black layer, ultimately having the same effect in this example as using observations.

The number 3 is not special in this construction; it can be increased arbitrarily by simply increasing the number of children of A and B. Therefore, in particular, one can construct a family of games in which the public state TB-DAG (without inflation) has exponential size, while the (observation) TB-DAG has polynomial size. This is why Zhang and Sandholm [96] and Carminati et al. [15] insist that inflation be done as a preprocessing step before beginning their constructions.

The use of observations, however, removes the need for this step:

Proposition 3.6.2. *Given any team decision problem $\mathcal{T}$, the TB-DAG of $\mathcal{T}$ is the same no matter whether inflation is applied to $\mathcal{T}$ before the construction.*

Proof. Inflation operations affect the connectivity graph $\mathcal{G}_i$, thus changing the results of SPLITBELIEF$_i$ operations at a terminal node. Consider, therefore, any observation node O in the TB-DAG, and let $h, h' \in O$, such that $I = I_1 \sqcup I_2$ is an inflatable infoset and $h \preceq u \in I_1$ and $h' \preceq u' \in I_2$. We must show that inflating I into I_1 and I_2 cannot remove the (h, h') edge in $\mathcal{G}_i[O]$.

Assume for contradiction that inflating would remove the (h, h') edge and that therefore SPLITBELIEF$_i$ would split h, h' into two different beliefs. We have that O is a valid observation

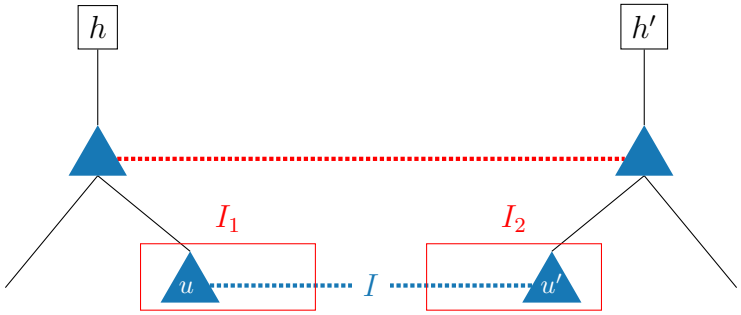


Figure 3.9: A pictorial representation of the proof of Proposition 3.6.2. Since h and h' can be played simultaneously but u and u' cannot, there must be an infoset like the red dotted one connecting a child of h to a child of h'. Therefore, inflation cannot break existing edges between played nodes.

node, so the player can play to both nodes h and h' simultaneously. But then there must be an infoset I' connecting some node on the $h \to u$ path to some node on the $h' \to u'$ path—otherwise, it would be possible for the player to play to both u and u' simultaneously, which violates inflatability of I. But then there is still an (h, h') edge in $\mathcal{G}_i[O]$, which is a contradiction. □

Although inflation *can* be performed efficiently, not requiring it as a preprocessing step simplifies the code and makes for a conceptually cleaner construction. However, the benefits of observations go beyond making inflation unnecessary. In fact, even with inflation, there are still cases where using observations instead represents an exponential improvement.

Proposition 3.6.3. *There exists a family of team decision problems in which the TB-DAG has a polynomial size, but the public state TB-DAG has exponential size, even if inflation is applied as a preprocessing step before building the latter.*

Proof. The counterexample in Figure 3.8 would work if it were not for the fact that all of the infosets in the last layer inflate. Therefore, we use a similar gadget at the bottom of the game to prove this result but ensure that inflation does nothing.

Consider the following family of games, parameterized by an integer $C > 1$. First, nature picks an integer $c \in \{1, \ldots, C\}$. Over the next $C-2$ layers $t = 1, 2, \ldots, C-2$, if $c \in \{t, t+2\}$, a player who cannot distinguish the two cases chooses an action $a \in \{0, 2\}$. If $c = t + a$, the game continues; otherwise, the game ends.

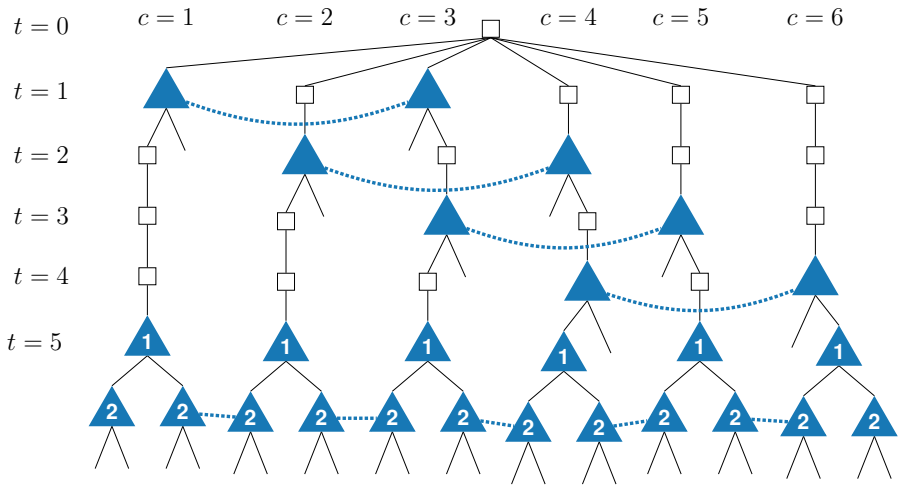


Figure 3.10: The counterexample for Proposition 3.6.3, for $C = 6$.

Finally, P1, who has perfect information about c, chooses between two actions numbered either c or $c + 1$. Then, player P2, observing P1's action number but *not* the value c, picks one of two options.

The resulting game is visualized in Figure 3.10. We observe the following things about it.

1. No infoset inflates: all nontrivial infosets have size 2, and it is always possible to play to both nodes in them. This starkly contrasts the earlier counterexample, in which inflation was enough to achieve a small representation.

2. Every P2-node in layer $C - 1$ is in the same public state, and playing to at least $C/2$ of them is always possible. Therefore, if using public-state-based beliefs, there will be a belief with $2^{C/2}$ prescriptions. Thus, the public-state-based team belief DAG will have a size of at least $2^{C/2}$.

We claim that layer $t \leq C - 1$ does not have too many beliefs. Let $B \subseteq \mathcal{H}_t$ be a belief, and in the below discussion, let $[a..b]$ denote the set of integers $\{a, \dots, b\}$.

1. $B \subseteq [t + 2..C]$. Since all nodes $j \geq t + 2$ must be played to, we have $B = [t + 2..C]$.

2. $B \not\subseteq [t + 2..C]$. Then let $j = \max(B \setminus [t + 2..C])$. Since $j \leq t + 1$, we have $j - 2 \notin B$, since j and $j-2$ are descendants of different actions taken at the infoset on layer $j-2 < t$. Thus B does not contain any node $j' < j - 2$ either, since in $\mathcal{G}[\mathcal{H}_t]$ such nodes j' are only connected to j through $j - 2$.

Thus, $B \cap [1..t+1]$ is either $\{j\}$ or $\{j, j-1\}$. Further, since all nodes $j \geq t+2$ are played to and connected in $\mathcal{G}[\mathcal{H}_t]$, it follows that $B \cap [t+2..C]$ is either $[t+2..C]$ or empty. Thus, for each possible choice of $j \leq t+1$, there are at most 4 valid beliefs.

Thus, the number of beliefs in layer t is at most $4(t+1)+1+2 = 4t+7$, where the $+2$ comes from counting the terminal beliefs, of which there are at most two.

Further, at each belief B, we claim that the number of active information sets is, at most, a constant. For $t < C-1$ this is obvious since $\mathcal{H}_t$ contains only one information set (namely $\{t, t+2\}$). For $t = C+1$, by the above argument, we have $|B| \leq 2$, so B overlaps at most two information sets.

Overall, there are $O(C)$ beliefs at each of the C layers of the game, and such beliefs never touch more than $O(1)$ different infosets. This is also true at the final layer because each infoset contains only two nodes at most. It is therefore proven that the team belief DAG has a size of at most $O(C^2)$. □

A practical experiment backs up these results. When $C = 16$, using observations generates a DAG with around 1000 edges; using public states generates a DAG with 30 million edges.

3.6.2 Tree vs DAG Representation

Here, we give an explicit example in which the TB-DAGs will be exponentially smaller than the game tree generated by `MakeBeliefGame`. This construction would work for most nontrivial adversarial team games, but for concreteness, consider the game G depicted in Figure 2.1. Call the leftmost terminal node in that diagram z. Consider adding another copy of G rooted in node z, and then repeating this process until ℓ copies of the game tree have been created, thus forming a game G^ℓ. That is, G^ℓ is the game in which G is played repeatedly until ℓ repetitions have been reached, or the terminal node reached is not z.

Note that when running `MakeBeliefGame` on G, multiple copies of node z will appear. Thus, the number of nodes in the auxiliary game will be exponential in ℓ. However, in the TB-DAG, after the ith repetition of the game finishes, the belief will always be $\{z_i\}$ (where z_i is the copy of z in the ith repetition of the game). Thus, the size of the TB-DAG will scale linearly with ℓ. Thus, as ℓ grows, the TB-DAG will be exponentially smaller than the auxiliary game, and in particular, the TB-DAG will have polynomial size. In contrast, the auxiliary game will have exponential size.

3.6.3 Definition of Information Complexity and Comparison of Bounds

We discuss the comparison between the bounds from Theorem 3.2.6 and Theorem 3.4.2 in more detail.

In Section 3.2.1, we defined the information complexity as the maximum number of *last-infosets* in any public state. This definition was made with Theorem 3.4.3 in mind, because it is the correct parameterization for that result. However, we could have used a tighter parameterization for Theorem 3.2.6. In particular, we could have defined a parameter κ as the number of infosets (not last-infosets) in any public state. Then $O(b^{2\kappa d+d})$ would be a valid upper bound in Theorem 3.2.6. One might ask how this new upper bound compares to that of Theorem 3.4.3. To this end, we now compare the two bounds.

Lemma 3.6.4. $k \leq \kappa d$.

Proof. Every last-infoset at a public state P will be an infoset intersecting some public state ancestor of P. Thus, there can be at most κd of these. □

Thus, the bound in Theorem 3.4.3 is at most

$$|\mathcal{H}|(b+1)^{k+1} \leq |\mathcal{H}|(b+1)^{\kappa d} < |\mathcal{H}|b^{2\kappa d} \leq b^{2\kappa d+d}$$

where we use the bounds $b \geq 2$ (which holds for every nontrivial game) and $|\mathcal{H}| \leq b^d$. Thus, we conclude that the bound in Theorem 3.4.3 is always strictly tighter than the bound in Theorem 3.2.6.

We also remark that in any case, $\kappa \leq |\mathcal{H}|$ is a loose bound that still ensures that the overall bound in Theorem 3.2.7 is polynomial in $|\mathcal{H}|$.

3.6.4 Connection With Tree Decomposition

The public *state* TB-DAG can be viewed from the perspective of graphical models, specifically, using *tree decompositions*. Here, we review tree decompositions and show the tree decomposition-based perspective of the public-state TB-DAG.

Definition 3.6.1. Given a (simple) graph $\mathcal{G} = (V, E)$, a *tree decomposition*[14] is a tree $\mathcal{J}$, with the following properties:

[14] also known as a *clique tree* or *junction tree*

1. the nodes of $\mathcal{J}$ are subsets of V, called *bags*;
2. for each edge $(u, v) \in E$, there is a bag containing both u and v; and
3. for each vertex $u \in V$, the subset of nodes of $\mathcal{J}$ whose bags contain u is connected.

Consider an arbitrary set of the form

$$\Pi = \{\boldsymbol{x} \in \{0,1\}^n : g_k(\boldsymbol{x}) = 0 \ \ \forall k \in [m]\}$$

where the g_ks are arbitrary constraints, and as before let $\mathcal{X} = \operatorname{co}\Pi$. Each constraint g_k has a *scope* $S_k \subseteq [n]$ of variables it depends on. The *dependency graph* of Π is the graph $\mathcal{G}_\Pi$ whose nodes are the integers $1, \ldots, n$, and where there is an edge (i, j) if there is a constraint whose scope S_k contains both i and j. For a subset $U \subseteq [n]$, a vector $\tilde{\boldsymbol{x}} \in \{0,1\}^U$ is *locally feasible* if $\tilde{\boldsymbol{x}} = \boldsymbol{x}_H$ for some $\boldsymbol{x} \in \Pi$. We will use Π_U to denote the set of all locally feasible vectors on U. Of course, $\Pi_{[n]} = \Pi$.

The main result of interest to us is a corollary of the junction tree theorem (see, for example, [91]), which allows an arbitrary set $\mathcal{X} = \operatorname{co}\Pi$ to be described with a constraint system whose size is related to the sizes of tree decompositions of $\mathcal{G}_\Pi$.

Theorem 3.6.5 (91)**.** *Let $\mathcal{J}$ be a tree decomposition of $\mathcal{G}_\Pi$. Then $\boldsymbol{x} \in \mathcal{X}$ if and only if there are vectors $\boldsymbol{\lambda}_U \in \Delta(\Pi_H)$ for each bag U of $\mathcal{J}$, such that:*

$$\boldsymbol{x}_U = \sum_{\tilde{\boldsymbol{x}} \in \Pi_U} \boldsymbol{\lambda}_U[\tilde{\boldsymbol{x}}] \cdot \tilde{\boldsymbol{x}} \qquad \forall \text{ bags } U \text{ in } \mathcal{J}$$

$$\sum_{\substack{\tilde{\boldsymbol{x}} \in \Pi_U \\ \tilde{\boldsymbol{x}}_{U\cap V} = \tilde{\boldsymbol{x}}^*}} \boldsymbol{\lambda}_U[\tilde{\boldsymbol{x}}] = \sum_{\substack{\tilde{\boldsymbol{x}} \in \Pi_V \\ \tilde{\boldsymbol{x}}_{U\cap V} = \tilde{\boldsymbol{x}}^*}} \boldsymbol{\lambda}_V[\tilde{\boldsymbol{x}}] \qquad \forall \text{ edges } (U, V) \text{ of } \mathcal{J} \text{ and } \tilde{\boldsymbol{x}}^* \in \Pi_{U\cap V}$$

Intuitively, the first constraint says that every $\boldsymbol{x}_U$ must be a convex combination of locally feasible $\tilde{\boldsymbol{x}} \in \Pi_U$. This is, of course, a necessary condition. The second constraint says that marginal probabilities on edges (U, V) must be consistent. This is also clearly a necessary condition, so the difficulty of proving the above result lies in showing that these two constraints are *sufficient*. We will not prove the result here, but we will use it as a black box.

In this section, we will work with a representation slightly different from the realization form. For a player i in a coordinator game G and a pure strategy of that player, the *history form* of the strategy as the vector $\boldsymbol{x} \in \{0,1\}^{\mathcal{H}}$ where $\boldsymbol{x}[h] = 1$ if and only if the team plays all actions on the $\varnothing \to h$ path. (Of course, the realization form is just the subvector of $\boldsymbol{x}$ indexed by $\mathcal{Z}$.) As usual, we will use Π for the set of pure strategies in history form, and $\boldsymbol{x} = \operatorname{co}\Pi$.

The history form is the set of vectors $\boldsymbol{x} \in \{0,1\}^{\mathcal{H}}$ satisfying the following constraint system.

$$\begin{aligned} \boldsymbol{x}[\varnothing] &= 1 \\ \boldsymbol{x}[ha] &= \boldsymbol{x}[h] && \text{if} \quad h \notin \mathcal{H}_i \\ \boldsymbol{x}[h] &= \sum_{a \in A_h} \boldsymbol{x}[ha] && \text{if} \quad h \in \mathcal{H}_i \\ \boldsymbol{x}[ha]\boldsymbol{x}[h'] &= \boldsymbol{x}[h'a]\boldsymbol{x}[h] && \text{if} \quad h, h' \in I \in \mathcal{I}_i; a \in A_h \end{aligned}$$

This constraint system defines a dependency graph $\mathcal{G}_\Pi$, whose nodes are nodes of the tree, and in which there is an edge (h, h') if either h' is a child of h, or h and h' are in the same infoset of player i.

Now consider the following tree decomposition of $\mathcal{J}$ of $\mathcal{G}_\Pi$. For each public state P, the tree decomposition $\mathcal{J}$ has a bag U_P that contains all nodes in P and all children of nodes in P. The edges of $\mathcal{J}$ are the obvious edges, connecting each U_P to $U_{P'}$ if $U_P \cap U_{P'} \neq \varnothing$.

One can check that, up to trivial reformulations (that is, removal of redundant variables and constraints), the constraint system from Theorem 3.6.5 associated with $\mathcal{J}$ is identical to the constraint system associated with the public state TB-DAG (via (3.2)). Thus, it is possible to interpret the public state TB-DAG entirely from the point of view of tree decompositions. We do not take this perspective in the rest of the paper because using beliefs is more interpretable and understandable from a game-theoretic perspective.

3.6.5 Postprocessing Techniques That Can Be Used To Shrink the TB-DAG

In practice, `ConstructTB-DAG` is suboptimal in several ways. Here, we state some straightforward postprocessing techniques that can be used to shrink the size of the TB-DAG. These do not affect the theoretical statements as the primary focus of those is isolating the dependency on our parameters of interest. Still, they can significantly affect the practical performance, so we apply them in the experiments.

1. If two terminal nodes z, z' have the same sequence, we remove one of them (say, z') from our DAG because it is redundant, and alias $\boldsymbol{x}[\{z'\}]$ to $\boldsymbol{x}[\{z\}]$. If this removal causes a section of the DAG to no longer contain any terminal descendants, we also remove that section.

2. If a decision point in the TB-DAG has (at most) one parent and (at most) one child, we remove the decision point and directly connect the parent observation node to the grandchild decision points.

In particular, if the team has perfect recall, the above two optimizations are sufficient for the TB-DAG to coincide with the sequence form.

3.7 Experiments

This section investigates the empirical benefits of applying the TB-DAG when computing mixed-Nash equilibria. As highlighted in Section 3.1, the literature on team games has been the one most concerned with the efficient computation of mixed Nash, with different works establishing benchmarks and proposing algorithms. We will, therefore, focus on comparing our approach against those previous related works. Our main results are reported in Table 3.2, which reports the size of the original games and our derived representations, and in Table 3.3, which reports the time required to solve those instances up to an approximation factor.

3.7.1 Experimental Setting

First, we give a complete description of the experimental setting in which the different algorithms are tested.

Game instances We run experiments on commonly adopted parametric benchmarks in the team games literature. The following is the naming convention adopted for the instances considered:

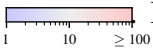
- ${}^{n}\mathbf{K}r$: n-player Kuhn poker with r ranks [52].
- ${}^{n}\mathbf{L}brs$: n-player Leduc poker with a b-bet maximum in each betting round, r ranks, and s suits [83].
- ${}^{n}\mathbf{D}d$: n-player Liar's Dice with one d-sided die for each player [58].

The full description of these games can be found in Farina et al. [28]. For each game, the players belonging to team ∇ are represented along with the name. For example, ^{4}L133 {3,4} indicates a 4-player Leduc poker game with 1 bet each round, 3 ranks, 3 suits, where players 3 and 4 belong to team ∇ and are therefore coordinated by player $\blacktriangledown$.

	Original game G							Belief Game $\tilde{G}$					Team ▲'s DAG		Team ▼'s DAG	
	Nodes	Infosets		Sequences		Information		Nodes	Infosets		Sequences		Vertices	Edges	Vertices	Edges
G	$\lvert\mathcal{H}\rvert$	$\lvert\mathcal{I}_{▲}\rvert$	$\lvert\mathcal{I}_{▼}\rvert$	$\lvert\Sigma_{▲}\rvert$	$\lvert\Sigma_{▼}\rvert$	$\max_{\mathcal{P}}\lvert P\rvert$	$\boldsymbol{k}$	$\lvert\tilde{\mathcal{H}}\rvert$	$\lvert\tilde{\mathcal{I}}_{▲}\rvert$	$\lvert\tilde{\mathcal{I}}_{▼}\rvert$	$\lvert\tilde{\Sigma}_{▲}\rvert$	$\lvert\tilde{\Sigma}_{▼}\rvert$	$\lvert\mathcal{D}_{▲}\cup\mathcal{S}_{▲}\rvert$	$\lvert E_{▲}\rvert$	$\lvert\mathcal{D}_{▼}\cup\mathcal{S}_{▼}\rvert$	$\lvert E_{▼}\rvert$
^{3}K3 {3}	151	24	12	48	24	6	6	2,119	486	12	1,062	24	487	918	37	36
^{3}K4 {3}	601	32	16	64	32	12	8	45,049	4,487	16	9,800	32	2,100	6,711	49	48
^{3}K6 {3}	3,001	48	24	96	48	30	12	6,768,601	267,184	24	574,588	48	54,255	336,944	73	72
^{3}K8 {3}	8,401	64	32	128	64	56	16	617,929,873	13,194,749	32	27,978,704	64	1,783,926	15,564,765	97	96
^{3}K12 {3}	33,001	96	48	192	96	132	24	—	—	—	—	—	—	—	—	—
^{4}K5 {3,4}	7,801	80	80	160	160	20	10	577,764,601	102,725	10,385	221,810	21,740	26,566	124,875	4,621	15,415
^{4}K5 {4}	7,801	120	40	240	80	60	15	174,273,721	11,739,640	40	25,581,730	80	998,471	4,658,070	121	120
^{3}L133 {3}	12,688	456	228	912	456	9	6	1,293,658	96,115	228	208,136	456	23,983	49,005	685	684
^{3}L143 {3}	40,409	800	400	1,600	800	16	8	52,745,745	2,625,209	400	5,736,592	800	139,964	417,027	1,201	1,200
^{3}L151 {3}	19,981	1,000	500	2,000	1,000	20	10	152,692,141	16,564,617	500	36,016,124	1,000	150,707	496,196	1,501	1,500
^{3}L153 {3}	98,606	1,240	620	2,480	1,240	25	10	1,833,113,016	67,400,747	500	147,671,104	1,240	855,397	3,486,091	1,861	1,860
^{3}L223 {3}	15,659	1,260	630	2,884	1,442	4	4	521,285	47,579	812	100,420	1,624	32,750	45,913	2,437	2,436
^{3}L523 {3}	1,299,005	99,168	49,584	246,304	123,152	4	4	178,141,285	19,499,329	73,568	40,224,140	147,136	2,911,352	4,183,685	220,705	220,704
^{4}L133 {3,4}	159,001	1,632	1,632	3,264	3,264	9	6	985,916,371	475,081	135,322	1,011,500	292,400	79,351	158,058	75,157	155,475
^{3}D3 {3}	27,622	1,023	513	2,046	1,020	9	6	70,704,118	3,235,954	765	5,501,789	1,272	91,858	215,967	1,522	1,521
^{3}D4 {3}	524,225	10,924	5,460	21,840	10,920	16	8	—	—	—	—	—	4,043,377	13,749,608	16,381	16,380
^{4}D3 {2,4}	663,472	6,144	6,144	12,285	12,285	9	6	—	—	—	—	—	514,120	1,217,310	486,442	1,155,144
^{6}D2 {2,4,6}	524,225	4,096	4,096	8,190	8,190	8	6	5,879,066,753	1,094,865	701,001	1,869,170	1,202,948	254,758	457,795	218,570	389,995
^{6}D2 {4,6}	524,225	5,704	2,488	10,920	5,460	16	8	4,992,649,921	15,032,900	33,905	25,363,692	57,194	991,861	2,029,546	46,236	60,717
^{6}D2 {6}	524,225	6,584	1,608	12,922	3,458	32	10	2,126,796,737	126,748,497	2,532	208,964,598	4,382	3,158,364	7,395,885	5,551	5,550

Table 3.2: Game sizes of the equivalent representations proposed in the paper (that is, belief game and TB-DAG) on several standard parametric benchmark team games. See Section 3.7 for a description of the games and a detailed description of the meaning of each column. Values denoted with '—' are missing due to out-of-time or out-of-memory errors.

Game {▼} G	Original game ▲ Value u^*	Nodes $\lvert\mathcal{H}\rvert$	Information $\max_{\mathcal{P}} \lvert P\rvert$	k	TB-DAG This paper (CFR) Init	ε=10^{-3}	ε=10^{-4}	EFG ZFCS22 (CG) Init	ε=10^{-3}	ε=10^{-4}
^{3}K3 {3}	0.000	151	6	6	0.00s	0.00s	0.00s	0.00s	0.00s	0.00s
^{3}K4 {3}	-0.042	601	12	8	0.01s	0.00s	0.00s	0.00s	0.01s	0.02s
^{3}K6 {3}	-0.024	3,001	30	12	1.03s	0.03s	0.12s	0.00s	0.14s	0.14s
^{3}K8 {3}	-0.019	8,401	56	16	1m6s	4.73s	32.36s	0.01s	0.23s	0.32s
^{3}K12 {3}	-0.014	33,001	132	24	—	oom	oom	0.01s	0.84s	1.39s
^{4}K5 {3,4}	-0.037	7,801	20	10	0.55s	0.03s	0.05s	—	—	—
^{4}K5 {4}	-0.030	7,801	60	15	13.71s	1.59s	6.34s	—	—	—
^{3}L133 {3}	0.215	12,688	9	6	0.49s	0.02s	0.05s	0.02s	24.89s	45.96s
^{3}L143 {3}	0.107	40,409	16	8	1.39s	0.10s	0.48s	0.05s	2m 4s	6m 3s
^{3}L151 {3}	-0.019	19,981	20	10	1.54s	0.18s	0.50s	0.04s	3.06s	13.98s
^{3}L153 {3}	0.024	98,606	25	10	16.03s	1.24s	4.94s	0.12s	7m 23s	28m 13s
^{3}L223 {3}	0.516	15,659	4	4	0.13s	0.03s	0.08s	0.05s	13.48s	18.53s
^{3}L523 {3}	0.953	1,299,005	4	4	18.02s	11.26s	24.86s	6.83s	> 6h	> 6h
^{4}L133 {3,4}	0.147	159,001	9	6	2.03s	0.21s	0.92s	—	—	—
^{3}D3 {3}	0.284	27,622	9	6	0.80s	0.11s	0.40s	0.09s	11.05s	11.05s
^{3}D4 {3}	0.284	524,225	16	8	1m3s	22.54s	1m 28s	1.57s	3h 19m	3h 19m
^{4}D3 {2,4}	0.200	663,472	9	6	27.05s	2.31s	4.70s	—	—	—
^{6}D2 {2,4,6}	0.072	524,225	8	6	10.74s	1.72s	4.26s	—	—	—
^{6}D2 {4,6}	0.265	524,225	16	8	16.55s	3.80s	11.09s	—	—	—
^{6}D2 {6}	0.333	524,225	32	10	31.00s	30.20s	1m 11s	—	—	—

Table 3.3: Runtime of our CFR-based algorithm (column '**This paper**') using the team belief DAG form, compared to the prior state-of-the-art algorithms based on linear programming and column generation by Zhang et al. [97] ('**ZFCS22**'), on several standard parametric benchmark games. See Section 3.7 for a description of the games. Column *"Init"* represents the time needed to construct the structures needed for solving the games. This corresponds to fully exploring the TB-DAG and computing its full representation in memory in the TB-DAG case. Missing or unknown values are denoted with '—'. For each row, the background color of each runtime column is set proportionally to the ratio with the best runtime for the row, according to the logarithmic color scale 

Runtimes that are more than two orders of magnitude larger than the best runtime for the row (*i.e.*, for which $R > 10^2$) are colored as if $R = 10^2$.

CFR Variant used We implemented the *Predictive CFR$^+$ (PCFR$^+$)* [29] state-of-the-art variant of CFR on the TB-DAG. PCFR$^+$ is a predictive regret minimization algorithm and uses quadratic averaging of iterates. At each time t, we use the previous utility vector for each time to predict the next. We remark that applying the CFR algorithm on the belief game and on the TB-DAG leads to identical iterations since the two representations are structurally equivalent (as proven in Section 3.4), and CFR is a deterministic algorithm. We therefore focus on the TB-DAG representation due to its efficiency. We also remark that the optimizations discussed in Sections 3.4.3 and 3.6.5 are applied during the experiments.

Baselines We use the column generation framework of Farina et al. [28] and refined by Zhang et al. [97] (henceforth "**ZFCS22**") as the prior state-of-the-algorithm to compare the performance of CFR on the team belief DAG. ZFCS22 belongs to the family of column generation approaches adopted in the past literature in team games and described in Section 3.1. ZFCS22 iteratively refines the strategy of each team by solving best-response problems using a tight integer program derived from the theory of extensive-form correlation [89]. We used the original code by the authors, which was implemented for three-player games in which a team of two players faces an opponent.

Hardware used All experiments were run on a 64-core AMD EPYC 7282 processor. Each algorithm was allocated a maximum of 4 threads, 60GBs of RAM, and a time limit of 6 hours. ZFCS22 uses the commercial solver Gurobi to solve linear and integer linear programs. All CFR implementations are single-threaded, while we allowed Gurobi to use up to four threads.

3.7.2 Discussion of the Results

We now discuss the empirical results obtained by our algorithms.

Representation vs Game size We analyze the size results from Table 3.2. The different orders of magnitude of the size of each representation and the original game highlight how the belief game construction increases the size of the game. Moreover, the striking difference between the two equivalent approaches of belief game and TB-DAG motivates the latter's introduction: the direct construction of a decision problem and the more efficient representation brought by the DAG structure allows the construction of a substantially smaller representation. The benefits of the DAG imperfect-recall structure are especially beneficial in the case of Liar's Dice instances, which have a larger depth of the game tree. Overall, this comparison confirms the results from the worst-case bounds from Sections 3.2.4, 3.4.1 and 3.6.3. The exponential factor of inefficiency between the two representations agrees with the results from the discussion in Section 3.6.2.

Some minor remarks are worth to be made. Whenever $\blacktriangledown$ is a perfect-recall player (equivalently, when the team $\triangledown$ is composed of a single player), our constructions never increase the size of its decision problem. In the case of the belief game, the adversary retains an identical number of information sets and sequences. In the case of the TB-DAG, the correspondence is $|\mathcal{D}_{\blacktriangledown}| = |\tilde{\mathcal{I}}_{\blacktriangledown}| + 1$ and $|\mathcal{S}_{\blacktriangledown}| = |\tilde{\Sigma}_{\blacktriangledown}|$

Running time We focus on the time performance of CFR applied to the games from Table 3.3. The main observation is that the TB-DAG approach combined with the CFR algorithm performs well in most games traditionally employed in the team game literature. In particular, impressive performance is achieved in games where the information complexity is low. This is the case of Leduc and Liar's Dice benchmarks (whose number of infosets and sequences in the original game are reported in Table 3.2). On the other hand, the column generation approaches struggle since the dimension of the pure strategy space depends exponentially on the number of information sets. The performance of our method depends crucially on having low information complexity. In fact, in games such as ^{3}K8 and ^{3}K12 where the information complexity is high, we observe poor performance even though the game tree is small. On the other hand, column generation techniques avoid this cost by considering an incrementally larger action space.

Chapter 4

Subgame Solving in Adversarial Team Games

In this chapter, a *subgame solving* procedure tailored to the structure of TB-DAG presented in Chapter 3 is developed. Subgame solving is a technique originally proposed in the two-player zero-sum games setting by Burch et al. [14], Brown and Sandholm [8]. It is a refinement procedure of a precomputed strategy, which is modified by performing local improvements to a strategy from any point of the game to the end. Subgame solving was one of the core techniques that allowed two-player zero-sum equilibrium finding techniques to scale to the size of poker games [65, 9, 11], and obtain superhuman performance in this setting.

For what concerns adversarial team games and imperfect recall games, the main benefit of developing a subgame-solving technique is to soften the scalability issues highlighted in Section 3.7. This is achieved by computing a low-quality strategy on the whole game and then using subgame solving while playing that strategy to compute local refinements on the parts of the game that are reached during play. We focus on extending *maxmargin* [64], the most notable subgame solving technique, to the TB-DAG setting. While the TB-DAG is already two-player zero-sum, extending maxmargin to its DAG-like structure is non-trivial. This is because the technical questions of how to generate the *gadget* game for the refinement and how to compute the counterfactual reach and best response value require careful treatment since a specific belief is reached through multiple paths.

The chapter is structured as follows. Section 4.1 describes the maxmargin technique originally developed by Moravcik et al. [64] in the setting of two-player zero-sum games and also highlights the issues that block its straightforward application to team games. Section 4.2 introduces and analyzes our *team-maxmargin* algorithm. Section 4.3 describes a column generation

framework that we propose for more efficient computation of the refinement. Section 4.4 shows the experimental performance of the proposed algorithm.

There are some important simplifications we make to ease the presentation and formalism adopted in the rest of the chapter:

- Similarly to Section 2.4 and Chapter 3, we take the equivalent perspective of two-player zero-sum imperfect-recall games. Still, our results apply to both adversarial team games and imperfect-recall timeable games.
- We fix ▲ as the *refining player* and ▼ as the *opponent*.
- In the two-player zero-sum perfect-recall case, we index sequence-form strategies $\boldsymbol{x} \in \mathcal{Q} = \{0,1\}^{\mathcal{S}}$ using nodes and information sets, instead of using the sequences associated to those. That is:

$$\boldsymbol{x}[h] \coloneqq \boldsymbol{x}[\sigma_{\blacktriangle}(h)]$$
$$\boldsymbol{x}[I] \coloneqq \boldsymbol{x}[\sigma_{\blacktriangle}(I)]$$

- Similarly, in the two-player zero-sum imperfect-recall case, we index DAG-form strategies $\boldsymbol{x} \in \mathcal{Q} = \{0,1\}^{\mathcal{S}}$ using beliefs $B \in \mathcal{D}$ and terminal nodes $z \in \mathcal{Z}$ as follows:

$$\boldsymbol{x}[B] \coloneqq \sum_{p \in P_{\{B\}}} \boldsymbol{x}[p]$$
$$\boldsymbol{x}[z] \coloneqq \boldsymbol{x}[\{z\}]$$

  where we use $\{B\} \in \mathcal{D}$ to indicate the decision node corresponding by construction to a belief $B \in \mathcal{B}$ and the correspondence of each terminal node $z \in \mathcal{Z}$ with an observation node $\{z\} \in \mathcal{S}$ in the TB-DAG.

- We differentiate ▲ and ▼'s beliefs using different letters $B \in \mathcal{B}_{\blacktriangle}$ and $C \in \mathcal{B}_{\blacktriangledown}$.

We highlight that this chapter's content has been originally published as the conference article [94].

4.1 Maxmargin

The maxmargin algorithm [64] is used in two-player zero-sum games in extensive form for the online refinement of a *blueprint strategy*[1] for the game. In particular, for every public state P reached by the players during the playthrough, maxmargin considers fixed the blueprint strategy in the *trunk* (that is, the part of the game leading to P), optimizing the player's strategy exclusively in the subgame rooted at P. Its main objective, roughly speaking, is to compute the strategy giving the largest decrease in exploitability when merged with the trunk one.

4.1.1 Value Computation in Games

This subsection focuses on the characterizations of expected utility conditioned on reaching specific nodes used to define the maxmargin algorithm. Intuitively, these values are computed considering the subset of the possible outcomes reachable from the nodes reached and renormalizing terminal reach probabilities to this support.

In the following, we consider a two-player zero-sum game with perfect recall G defined as in Definition 2.2.2. Let $\boldsymbol{x}, \boldsymbol{y}$ be some fixed strategy for ▲ and ▼ respectively.

Definition 4.1.1 (Expected Value)**.** The *expected value* $v^{\boldsymbol{x},\boldsymbol{y}}(h)$ under strategies $\boldsymbol{x}, \boldsymbol{y}$ at node $h \in \mathcal{H}$ is defined as:

$$v^{\boldsymbol{x},\boldsymbol{y}}(h) := \sum_{z \succeq h} \frac{\boldsymbol{x}[z]}{\boldsymbol{x}[h]} \frac{\boldsymbol{y}[z]}{\boldsymbol{y}[h]} \frac{\boldsymbol{p}[z]}{\boldsymbol{p}[h]} u(z)$$

This notion can be extended to an information set $I \in \mathcal{I}$ as:[2]

$$\begin{aligned} v^{\boldsymbol{x},\boldsymbol{y}}(I) &:= \sum_{z \succeq I} \frac{\boldsymbol{x}[z]}{\sum_{z \succeq I} \boldsymbol{x}[z]} \frac{\boldsymbol{y}[z]}{\sum_{z \succeq I} \boldsymbol{y}[z]} \frac{\boldsymbol{p}[z]}{\sum_{z \succeq I} \boldsymbol{p}[z]} u(z) \\ &= \sum_{h \in I} \frac{\boldsymbol{x}[h]}{\sum_{h \in I} \boldsymbol{x}[h]} \frac{\boldsymbol{y}[h]}{\sum_{h \in I} \boldsymbol{y}[h]} \frac{\boldsymbol{p}[h]}{\sum_{h \in I} \boldsymbol{p}[h]} v_{\boldsymbol{x},\boldsymbol{y}}(h) \end{aligned} \tag{4.1}$$

Definition 4.1.2 (Best Response Value)**.** The *best response value* for player ▼ against opponent's strategy $\boldsymbol{x}$ at node $h \in \mathcal{H}$ is defined as:

$$bv^{\boldsymbol{x}}_{▼}(h) := \max_{\boldsymbol{y}} v^{\boldsymbol{x},\boldsymbol{y}}(h)$$

[1] A blueprint strategy is a suboptimal strategy for the whole game being computed on an abstracted, smaller version of the game.

[2] We remark the role of each player's strategy normalization at the denominator, and the fact that $\sum_{h \in I} \boldsymbol{x}[h] = \sum_{z \succeq I} \boldsymbol{x}[z]$ due to probability conservation constraints on the strategy space from Section 2.1.3.

Similarly, this notion can be extended for infosets $I \in \mathcal{I}$ and for ▲.

It is also possible to define the per-player *counterfactual* version of the previous values, in which the player's strategies are not normalized. Intuitively, those values correspond to the case in which the player plays to reach the node h or infoset I considered, and therefore its reach $\boldsymbol{x}[h] = 1$ or $\sum_{h \in I} \boldsymbol{x}[h] = 1$ respectively.

Definition 4.1.3 (Counterfactual Value)**.** The *counterfactual value* $v_{\boldsymbol{x},\boldsymbol{y}}(h)$ for player ▼ under strategies $\boldsymbol{x}, \boldsymbol{y}$ at node $h \in \mathcal{H}$ is defined as:

$$cv^{\boldsymbol{x},\boldsymbol{y}}_{▼}(h) \coloneqq \sum_{z \sqsupseteq h} \frac{\boldsymbol{x}[z]}{\boldsymbol{x}[h]} \boldsymbol{y}[z] \frac{\boldsymbol{p}[z]}{\boldsymbol{p}[h]} u(z)$$

Similarly, this notion can be extended for infosets $I \in \mathcal{I}$ and for ▲.

Definition 4.1.4 (Counterfactual Best Response Value)**.** The *counterfactual best response value* $v_{\boldsymbol{y}}(h)$ for player ▼ against opponent's strategy $\boldsymbol{x}$ at node $h \in \mathcal{H}$ is defined as:

$$cbv^{\boldsymbol{x}}_{▼}(h) \coloneqq \max_{\boldsymbol{y}} cv_{\boldsymbol{x},\boldsymbol{y}}(h)$$

Similarly, this notion can be extended for infosets $I \in \mathcal{I}$ and for ▲.

It is to be noted that all the values defined can be efficiently computed by performing a bottom-up propagation of the values from the terminal nodes to the node or infoset considered.

4.1.2 Maxmargin Algorithm

We provide an informal description of the original maxmargin algorithm's formulation in the two-player zero-sum perfect-recall setting. The maxmargin algorithm works by finding an equilibrium on an auxiliary game called the *gadget game* and then transferring the result to the original game. The gadget game embodies the optimization problem of strategy refinement and allows one to optimize it using standard equilibrium-computation techniques. This objective is to maximize the *margin* of the refined strategy over the original strategy (and hence the name maxmargin).

Definition 4.1.5 (Margin, [64])**.** Let $\boldsymbol{x}, \boldsymbol{x}'$ be a pair of strategies for ▲ differing on a subgame S rooted at public state P. Then, the subgame margin is defined as:

$$\min_{I \subset P} cbv^{\boldsymbol{x}}_{▼}(I) - cbv^{\boldsymbol{x}'}_{▼}(I)$$

where $I \in \mathcal{I}_{\blacktriangledown}$.

Intuitively, the margin measures the difference in value at $I_{\blacktriangledown}$ a best-responding opponent would extract when $\blacktriangle$ switches from $\boldsymbol{x}$ to $\boldsymbol{x}'$. If the margin is positive, the new strategy is less exploitable than the old one. As proven in [64], the margin is proportional to a lower bound on the exploitability reduction of $\blacktriangle$'s strategy.

We now focus on the construction of the gadget game. Consider any public state P from which the blueprint is to be refined. The gadget game is obtained by extracting the subgame rooted at P from the original game and adding additional nodes above it to make it a tree. The root of the gadget game is a newly-added $\blacktriangledown$-node where the available moves—called *deviations*—allow him to choose an infoset $I \in \mathcal{I}_{\blacktriangledown}$ such that $I \subseteq P$. Each of those actions leads to a N node, where nature chooses a specific node in the infoset chosen by the opponent based on N's reach probabilities and the trunk blueprint strategy for $\blacktriangle$. Once a node has been selected by chance, the following part of the gadget game is the same as the original subgame up to the terminal nodes. Lastly, the payoffs associated with each terminal node following a specific initial choice of infoset by the opponent are decreased by the *counterfactual best response value* at that infoset. Formally, let $I_{\blacktriangledown} \in \mathcal{I}_{\blacktriangledown}$ be an infoset in P and $\boldsymbol{x}$ be $\blacktriangle$'s blueprint:

$$u(z) \leftarrow u(z) - cbv^{\boldsymbol{x}}_{\blacktriangledown}(I) \quad \forall z \in \mathcal{Z} : z \succeq I_{\blacktriangledown}$$

The change of the payoffs induces $\blacktriangledown$ to evaluate the outcomes of the gadget game with respect to the base performance of $\blacktriangle$ blueprint. Therefore, $\blacktriangledown$ restarts the gadget game in the infoset in which the current strategy of the refining player is most exploitable.

Fully solving the gadget game guarantees *safety*: if maxmargin is applied during a playthrough, then the resulting strategy is at most as exploitable as the blueprint. The rationale behind this construction is to guarantee non-increasing exploitability by offering the opponent the opportunity to perform any deviation from the trunk strategy. The payoff change allows simulating $\blacktriangledown$ switching her trunk strategy to reach the infoset in P with the lowest margin, while $\blacktriangle$ must play a balanced strategy to increase the margin on all infosets uniformly. In this way, the refining player computes its strategy considering a worst-case adversary that plays a trunk strategy that strategically counters the refinement she will make after.

The crucial issue when applying the maxmargin algorithm in the imperfect-recall/team setting is that the TB-DAG representation described in Chapter 3 separates the decision problems of the two players. On the other hand, the counterfactual values and the gadget games used by maxmargin are node-based. Unrolling the TB-DAG into a node-based extensive-form game would be a costly solution, corresponding to the belief game in Section 3.2.2. Such an

unroll is inefficient as it would require an amount of extra space exponential in the original game size Section 3.2.4. We will, therefore, develop our team-maxmargin algorithm in the TB-DAG framework and replace the gadget game with an equivalent max-min optimization problem; while this does not correspond to an EFG, Section 4.2.2 will show how to convert such problem into a TB-DAG that can be solved using state-of-the-art equilibrium computation algorithms.

4.2 Team-maxmargin

4.2.1 Linear Programming Formulation

We introduce the *team-maxmargin* algorithm, which extends maxmargin subgame solving to team games.

As aforementioned, in order to successfully apply subgame solving techniques to the TB-DAG, there is the need to relate the TB-DAG-form polytopes of the two teams to represent the counterfactual reach and counterfactual best-response values. Informally, the main problem is represented by the fact that maxmargin requires the counterfactual reach and the counterfactual best-response values *at the information set chosen by the opponent at the start of the gadget game*. While this information can be easily retrieved in a node-based extensive-form game by summing over the nodes in the information set (see Equation (4.1)), in a pure TB-DAG-based representation this is not possible anymore. The solution we propose is to compute those values from the refining player's blueprint strategy in a preprocessing phase and then combine them with the opponent's reach probabilities to select the value associated with the information set $I_{\blacktriangledown}$ chosen by the opponent.

We first show how to compute a *counterfactual reach* akin to the $\frac{1}{\sum_{z \succeq I} \boldsymbol{x}[z]} \frac{1}{\sum_{z \succeq I} \boldsymbol{p}[z]}$ term in Equation (4.1).

Definition 4.2.1 (Counterfactual reach)**.** Let $\boldsymbol{x} \in \mathcal{Q}_{\blacktriangle}$ be any strategy for $\blacktriangle$. For each $\blacktriangledown$'s belief $C \in \mathcal{B}_{\blacktriangledown}$, the counterfactual reach $\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)$ is the total N and $\blacktriangle$'s reach at C, defined as follows:

$$\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C) \coloneqq \sum_{h \in C} \boldsymbol{p}[h] \sum_{B \ni h} \boldsymbol{x}[B].$$

Similarly to the two-player zero-sum case, counterfactual reach corresponds to the probability that $\blacktriangle$ and N play to reach a specific belief C given that $\blacktriangledown$ plays a strategy $\boldsymbol{y}$ such that $\boldsymbol{y}[C] = 1$. In the definition above, the counterfactual reach $\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)$ aggregates the realization

probability of $\boldsymbol{x}$ and $\boldsymbol{p}$ on the nodes $h \in C$ so as to effectively represent the behavior of nature and ▲ in the TB-DAG of ▼.

The counterfactual reach allows us to determine which of ▼'s beliefs are actually reachable given ▲'s strategy. This set of nodes is denoted as the set of *counterfactually reachable beliefs*.

Definition 4.2.2 (Counterfactually reachable beliefs)**.** The set of all counterfactually reachable beliefs in a public state P is denoted with $\mathcal{C}^{\boldsymbol{x}}(P)$. Formally:

$$\mathcal{C}^{\boldsymbol{x}}(P) := \left\{ C \mid C \in \mathcal{C} \cap P \wedge \rho^{\boldsymbol{x}}_{\mathsf{N},▲}(C) > 0 \right\}.$$

Another concept needed to extend maxmargin to TB-DAGs is the one of *counterfactual best response value* at a ▼'s belief C, defined as the value of the best ▼'s prescription at C.

Definition 4.2.3 (Counterfactual best response value)**.** Let $\boldsymbol{x}$ be the strategy for team ▲. For each ▼ decision node C, let $u^{\boldsymbol{x}}_{▼}(C)$ be the *unnormalized* counterfactual ▼-best response value defined by the recurrences as follows:

$$u^{\boldsymbol{x}}_{▼}(C) = \begin{cases} \sum_{z \in C} \boldsymbol{x}[z]\, \boldsymbol{p}[z]\, u(z) & \text{if } C \text{ is terminal} \\ \min_{\boldsymbol{a} \in A(C)} u^{\boldsymbol{x}}_{▼}(C\boldsymbol{a}) & \text{otherwise} \end{cases},$$
$$u^{\boldsymbol{x}}_{▼}(C\boldsymbol{a}) = \sum_{C' \subseteq C\boldsymbol{a}} u^{\boldsymbol{x}}_{▼}(C'),$$

The normalized counterfactual ▼-best response value at C against $\boldsymbol{x}$ can be defined as follows:

$$cbv^{\boldsymbol{x}}_{▼}(C\boldsymbol{a}) = \frac{u^{\boldsymbol{x}}_{▼}(C\boldsymbol{a})}{\rho^{\boldsymbol{x}}_{\mathsf{N},▲}(C)}.$$

Notice that, in the definition of the unnormalized counterfactual best response value, only the realization form induced by the TB-DAG form strategy $\boldsymbol{x}$ and $\boldsymbol{p}$ is considered. On the other hand, when considering the normalized counterfactual best response value, we condition to ▲, N reaching a specific C. The division by the $\rho^{\boldsymbol{x}}_{\mathsf{N},▲}(C)$ factor introduces the normalization factors needed for $\boldsymbol{x}, \boldsymbol{p}$.

Given the above definitions, we can now provide the optimization problem the Team-Maxmargin algorithm solves to refine the strategy.

Definition 4.2.4 (Team-maxmargin linear program)**.** The maxmargin optimization problem in a TB-DAG subgame rooted at P is formulated as the following linear program:

$$\max_{\boldsymbol{x}'} \min_{\boldsymbol{y},\bar{\boldsymbol{y}}} \sum_{z \sqsupseteq P} \boldsymbol{x}'[z]\, \boldsymbol{y}[z]\, \boldsymbol{p}[z]\, u(z) - \sum_{C \in \mathcal{C}^{\boldsymbol{x}}(P)} \bar{\boldsymbol{y}}[C]\, \frac{u^{\boldsymbol{x}}_{\blacktriangledown}(C)}{\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)} \tag{4.2}$$

$$\text{s.t.} \quad \boldsymbol{x}'[B] = \boldsymbol{x}[B] \qquad \text{for all } \blacktriangle\text{-beliefs } B \in \mathcal{B}[P] \tag{4.3}$$

$$\boldsymbol{y}[C] = \frac{\bar{\boldsymbol{y}}[C]}{\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)} \qquad \text{for all } \blacktriangledown\text{-beliefs } C \in \mathcal{C}^{\boldsymbol{x}}(P) \tag{4.4}$$

$$\boldsymbol{x}' \in \mathcal{X}_P,\ \boldsymbol{y} \in \mathcal{Y}_P,\ \bar{\boldsymbol{y}} \in \Delta^{\mathcal{C}^{\boldsymbol{x}}(P)} \tag{4.5}$$

As in the original two-player formulation, $\blacktriangledown$ can choose to deviate to any of the counterfactually reachable beliefs at the public state of the subgame by choosing $\bar{\boldsymbol{y}} \in \Delta^{\mathcal{C}^{\boldsymbol{x}}(P)}$. Equation (4.2) specifies the maximization of the *margin*. This is defined similarly to Definition 4.1.5 and has a similar intuition. Equation (4.3) fixes the trunk strategy of $\blacktriangle$. Equation (4.4) rescales reach probabilities of the opponent according to the counterfactual reach $\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)$. This is equivalent to normalizing by the nature and $\blacktriangle$ reach probabilities at C. Such a normalization is crucial because it accounts for the fixed reach probabilities in the trunk for chance and $\blacktriangle$. While it may seem more intuitive to normalize by the reach probabilities of $\blacktriangle$ and chance directly on the terminal nodes, as is typically done in two-player zero-sum games, in team games it is actually necessary to do so on the beliefs in $\mathcal{C}^{\boldsymbol{x}}(P)$. This a consequence of the fact that, when $\blacktriangledown$ is composed by more than one player, the reach of $\blacktriangle$ and nature over terminal nodes depends directly on the deviation $C \in \mathcal{C}^{\boldsymbol{x}}(P)$ chosen by $\blacktriangledown$, and multiple such beliefs C can reach the same terminal node. This is a difficulty that only occurs in the team-vs-team setting. Note that only counterfactually reachable beliefs $\mathcal{C}^{\boldsymbol{x}}(P)$ of $\blacktriangledown$ are considered in this constraint since they are the only potential deviations of the opponent in the gadget game. Equation (4.5) encompasses all the TB-DAG constraints on the strategy spaces of $\blacktriangle$ and $\blacktriangledown$.

The team-maxmargin algorithm is applied similarly to the original maxmargin algorithm. If we want to locally refine a strategy $\boldsymbol{x}$ of $\blacktriangle$ while playing, team-maxmargin proceeds as follows: at each game turn, consider the public state P reached and compute an approximate solution $\boldsymbol{x}'$ to the linear program in Definition 4.2.4 (while respecting a timelimit). Then $\boldsymbol{x}$ is updated for the next turns: $\boldsymbol{x}[B] \leftarrow \boldsymbol{x}'[B]\ \forall B \in \mathcal{X}_P$. The blueprint strategy is used as the first strategy $\boldsymbol{x}$. As further explained in Section 4.3, we solve the linear program utilizing a column generation procedure in which the current strategy $\boldsymbol{x}$ is an available choice. This guarantees that the solution found is not worse than the current strategy, even if a single iteration is performed.

While we picked maxmargin as our refining procedure, other subgame solving techniques such as resolving [14] and reach solving [8] can be extended to the team setting as well.

4.2.2 Procedure for DAG Gadget

One may expect that, in parallel to the two-player case, our team-maxmargin optimization problem may also be representable as an imperfect-recall gadget game. However, the constraints we have introduced into the formulation go beyond the expressive power of the extensive-form representation, so this is not directly possible. In particular, we have that Equation (4.4) rescales reach probabilities after ▼ picked the start infoset. This affects the expected value left term in Equation (4.2). However, this scaling factor cannot be represented in an extensive-form language.[3] Despite this, the strategy spaces of both players in Definition 4.2.4 are DAGs, to which any technique for solving team games, such as CFR [108] or column generation, apply. In this way, we obtain a *DAG gadget*[4], that offers the same benefits of the perfect-recall gadget game, *i.e.* no-regret learning techniques can be applied.

The gadget DAG can be constructed from the TB-DAG of the subgame by considering, for both teams, the *root beliefs* (that is, the beliefs at the root of the subgame to be refined) and all following nodes. In each team's DAG, we introduce a root observation node followed by a single decision node, which we call the *gadget belief*. Its actions lead to as many observation nodes as beliefs considered at the root of the subgame. Each observation node has a single observation, each leading to a different root belief.

In addition, some constraints and extra payoffs have to be introduced. For ▲'s gadget DAG, each root belief B must be played with *unnormalized* "probability" $\boldsymbol{x}[B]$. (Note that $\sum \boldsymbol{x}[B] \neq 1$ in general). For ▼'s gadget DAG, the reach associated to the observation nodes following the gadget belief is divided by a factor $\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)$, where C is the belief they are reaching. Moreover, a payoff $u^{\boldsymbol{x}}_{\blacktriangledown}(C)$ is associated to those observation nodes. Algorithm 6 presents a procedure corresponding to the described construction.

Such constraints are not directly representable in the TB-DAG formalism since reaches may be greater than 1 (after division by $\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)$), and since there are decision nodes with fixed strategy and payoffs on observation nodes. These constraints cannot arise when constructing the TB-DAG of an imperfect-recall/team game

[3] The only available option would be to rescale $u(z)$ by the reach $\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)$, but this would require to dynamically rescale the terminal values accordingly to the starting information set they are reached from. There is more than one such infosets because we are dealing with a DAG decision problem rather than a tree.

[4] We avoid calling this a game because there does not exist an extensive-form representation equivalent to the TB-DAG formulation we present.

Algorithm 6 Construction of a Gadget DAG equivalent to Definition 4.2.4

Input:
    ▲-reach probabilities $\boldsymbol{x}[B]$ on beliefs $B \in \mathcal{B}[P]$
    cf. reach probabilities $\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)$ and alt values $u^{\boldsymbol{x}}_{\blacktriangledown}(C)$ for *all reachable* ▼-beliefs $C \in \mathcal{C}^{\boldsymbol{x}}(P)$
    already initialized DAGs $\mathcal{D}_{\blacktriangle}$ and $\mathcal{D}_{\blacktriangledown}$
let $\tilde{\mathcal{B}}$ be the set of ▲-beliefs in P with $\boldsymbol{x}[B] > 0$.
▷ *Considering* $\mathcal{D}_{\blacktriangle}$
$s^o_{\blacktriangle} \leftarrow$ new observation node in $\mathcal{D}_{\blacktriangle}$
$s^d_{\blacktriangle} \leftarrow$ new decision node in $\mathcal{D}_{\blacktriangle}$
add edge $s^o_{\blacktriangle} \to s^d_{\blacktriangle}$
for each belief $B \in \tilde{\mathcal{B}}$ **do**
    $s^B_{\blacktriangle} \leftarrow$ new observation node in $\mathcal{D}_{\blacktriangle}$
    add edge $s^d_{\blacktriangle} \to s^B_{\blacktriangle}$ **with fixed probability** $\boldsymbol{x}[B]$
    add edge $s'_{\blacktriangle} \to B$
▷ *Considering* $\mathcal{D}_{\blacktriangledown}$
$s^o_{\blacktriangledown} \leftarrow$ new observation node in $\mathcal{D}_{\blacktriangledown}$
$s^d_{\blacktriangledown} \leftarrow$ new decision node in $\mathcal{D}_{\blacktriangledown}$
add edge $s^o_{\blacktriangledown} \to s^d_{\blacktriangledown}$
for each belief $C \in \mathcal{C}[\boldsymbol{x}, P]$ **do**
    $s^C_{\blacktriangledown} \leftarrow$ new observation node in $\mathcal{D}_{\blacktriangledown}$
    add edge $s^d_{\blacktriangledown} \to s^C_{\blacktriangledown}$
    add edge $s'_{\blacktriangledown} \to C$
    associate a payoff to $s^C_{\blacktriangledown}$ **equal to** $-u^{\boldsymbol{x}}_{\blacktriangledown}(C)$
    divide reach along edge $s^d_{\blacktriangledown} \to s^C_{\blacktriangledown}$ **by** $\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)$
return the updated $\mathcal{D}_{\blacktriangle}$ and $\mathcal{D}_{\blacktriangledown}$

corresponds to our gadget DAG. However, such constraints maintain the scaled extension structure as specified in [99], and therefore, the same CFR algorithm used to solve DAGs can be used.

In practice for our experiments, we do not construct the DAG of ▲, since it is often too big. Instead, we use a column-generation algorithm to solve the subgame—see Algorithm 7. However, in domains where ▲ has small information complexity (see Section 3.6.3), performing subgame solving in this manner on the original DAG (or a reasonable abstraction thereof) is possible.

4.2.3 Safety of the Refinement

The team maxmargin algorithm achieves the same safety guarantees as the maxmargin algorithm for similar arguments to [64, 8]. Formally:

Theorem 4.2.1. *Applying team maxmargin subgame solving (Definition 4.2.4) to every subgame reached during play, in a nested fashion, results in playing a strategy with exploitability no worse than the blueprint.*

The following is a sketch of proof.

Proof. The blueprint has margin 0 by definition; therefore, the gadget subgame (Definition 4.2.4) always has a nonnegative value. Moreover, if any subgame strategy $\boldsymbol{x}'$ achieves nonnegative value in the gadget subgame, then the counterfactual best responses at the ▼-root beliefs cannot have improved for ▼ (by definition). Since the overall best response value for ▼ is a monotone function of these counterfactual best response values, the theorem follows. □

This is the same guarantee and argument given by Moravcik et al. [64], Brown et al. [12] in two-player zero-sum *perfect-recall* games.

4.3 Column Generation for Sparser Solutions

In this section, we argue that if the opponent ▼ is a single player (rather than a team) and the blueprint is sparse, then, by using column generation (for example Farina et al. [25]) as the solver, we can arrive at an online game-playing algorithm that runs in polynomial time (in $\mathcal{H}$, per iteration) with the exception of a best-response oracle.

The key observation is that the maxmargin formulation of subgame solving (Definition 4.2.4) has an *input* of size $|\mathcal{B}| + |\mathcal{C}|$ where $\mathcal{B}$ is the set of *played* beliefs for ▲ in the current public state, and $\mathcal{C}$ is the set of *all* beliefs for ▼ in the current public state. Therefore, to have a hope of an efficient algorithm, we need both quantities to be small. Hence, for this section, let us assume that ▼ has perfect recall or has a small TB-DAG (so that $|\mathcal{C}|$ is small) and that the blueprint is sparse (so that $|\mathcal{B}|$ is small at least on the first iteration).

The algorithm is given in Algorithm 7. At a high level, for computing best responses, it splits the problem of solving a game starting at public state P into subproblems, one for each reachable ▲-belief B. This allows a "finer" mixing of strategies than would have been allowed otherwise. Given a best-response oracle (that is, a solution method to the integer program (4.7)),

Algorithm 7 Maxmargin subgame solving with column generation, at public state P

Input:
  sparse DAG-form blueprint $\boldsymbol{x}$
  reach probabilities $\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)$ and alt values $u^{\boldsymbol{x}}_{\blacktriangledown}(C)$ for *all reachable* ▼-beliefs $C \in \mathcal{C}^{\boldsymbol{x}}(P)$
let $\tilde{\mathcal{B}}$ be the set of ▲-beliefs in P with $x[B] > 0$.
$\boldsymbol{x}'_{B,1} \leftarrow \{\textsc{GenerateArbitraryStrategy}(B)\}$ for each $B \in \tilde{\mathcal{B}}$
▷ *for each time t and belief $B \in \tilde{\mathcal{B}}$, $\boldsymbol{x}'_{B,t}$ is a pure strategy defined on the sub-DAG rooted*
▷ *at B. It represents a strategy that ▲ may take following belief B.*
for iteration $t = 1, 2, \dots$ **do**
  solve the *meta-game*:

$$\begin{aligned}
\max_{\boldsymbol{\lambda}_B, \lambda_*} \min_{\boldsymbol{y}, \bar{\boldsymbol{y}}} & \sum_{z \succeq P} \boldsymbol{x}'[z]\boldsymbol{y}[z]\boldsymbol{p}[z]u[z] - \sum_{C \in \mathcal{C}[P]} \boldsymbol{y}[C] u^{\boldsymbol{x}}_{\blacktriangledown}(C) \\
\text{s.t. } \boldsymbol{x}' &= \lambda_* \boldsymbol{x} + \sum_{B \in \tilde{\mathcal{B}}} \boldsymbol{x}[B] \sum_{1 \le \tau \le t} \lambda_{B,\tau} \boldsymbol{x}'_{B,\tau} \\
\boldsymbol{y}[C] &= \bar{\boldsymbol{y}}[C] \,/\, \rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C) && \text{for all ▼-beliefs } C \in \mathcal{C}^{\boldsymbol{x}}(P) \\
\sum_{1 \le \tau \le t} \lambda_{B,\tau} &= 1 - \lambda_* && \text{for all ▲-beliefs } B \in \tilde{\mathcal{B}} \\
\lambda_*, \boldsymbol{\lambda}_B &\ge 0 && \forall B \in \tilde{\mathcal{B}} \\
\boldsymbol{y} &\in \mathcal{Y}_P,\ \bar{\boldsymbol{y}} \in \Delta^{\mathcal{C}^{\boldsymbol{x}}(P)}
\end{aligned} \tag{4.6}$$

  for each belief $B \in \tilde{\mathcal{B}}$ **do**
    solve for a *best response* to $\boldsymbol{y}$ given belief B:

$$\begin{aligned}
\boldsymbol{x}'_{B,t+1} \leftarrow \operatorname*{argmax}_{\boldsymbol{x}' \in \{0,1\}^{\mathcal{H}_B}} & \sum_{z \succeq B} \boldsymbol{x}'[z]\boldsymbol{y}[z]\boldsymbol{p}[z]u[z] \\
\text{s.t. } \boldsymbol{x}'[ha]\boldsymbol{x}'[h'] &= \boldsymbol{x}'[h']\boldsymbol{x}'[ha] && \forall h, h' \in I \in \mathcal{I}_{\blacktriangle} \\
\boldsymbol{x}'[h] &= 1 && \forall h \in B
\end{aligned} \tag{4.7}$$

    where $\mathcal{H}_B$ is the set of nodes $h \succeq B$.
return $\boldsymbol{x}'$ converted to sparse TB-DAG form Equation (4.6)

the whole algorithm runs in time $\mathrm{poly}(|\mathcal{H}_P|, |\tilde{\mathcal{B}}|, |\mathcal{C}[P]|, t)$ where $\mathcal{H}_P$ is the set of nodes $h \succeq P$ and $\tilde{\mathcal{B}}$ is the set of ▲-beliefs in P with positive reach. While NP-hard in the general case, integer programs are practically reasonably fast to solve. In particular, if a polynomial-space algorithm such as depth-first branch-and-bound is used to solve the integer program, then the whole algorithm runs in polynomial space. This is in stark contrast to any algorithm that would expand the whole TB-DAG, as the TB-DAG takes worst-case exponential space.

The creation of sparse blueprints. Throughout this section, we have been assuming that blueprints created are sparse, in particular, that in every public state there are at most polynomially many beliefs. There are two straightforward ways to guarantee this. The first is to use a natively sparse algorithm, such as column generation, to generate the blueprint in the first place. The second is to take an arbitrary blueprint in realization form and express it as a sparse convex combination via Caratheodory's theorem[5]. Then, the blueprint will have support size at most $|\mathcal{Z}|$. On the other hand, the support size of the blueprint generated by a CG algorithm, scales linearly with the number of iterations, which, under reasonable time constraints, rarely exceeds the hundreds. Thus, the far smaller support size of the blueprint yielded by CG is a point in favor of adopting the former family of algorithms for blueprint generation.

4.4 Experimental Evaluation

4.4.1 Experimental Setting

Game instances. In our evaluation, we resort to game instances whose exact solution can be computed in practice by standard algorithms available in the literature. This choice allows us to calculate the equilibrium value, which is necessary to appropriately assess how our subgame solving method approximates the exact solution as the time spent on the strategy refinement increases. In particular, our test suite is composed of parametric versions of the adversarial team games instances customarily adopted in the literature where the adversary team is composed of a single player: Kuhn [51] and Leduc [83] poker with team collusion, team Liar's Dice [58] and Tricks [97] (a simplified Bridge endgame).

We omit the rules of these games, pointing an interested reader to the aforementioned articles. We use the following labels:

- K*nr*: *n*-player Kuhn poker with *r* ranks;
- L*nbrs*: *n*-player Leduc poker with *b* bets in each betting round, *r* ranks and *s* suits;
- D*nd*: *n*-player Liar's Dice with one *d*-sided dice per player;
- T*d*: three-player Trick-taking game with three ranks, four suits, and a limited amount of possible deals fixed to *d*.

[5]Caratheodory's theorem asserts that any convex combination of points in $\mathbb{R}^d$ can be expressed as a convex combination of at most $d + 1$ of them.

We also use a binary string of length n to denote the assignment of teams, where the ith character is 1 if and only if the ith player is on team ▲.

Blueprint and strategy refinement time limits. As usual with subgame solving methods, the performance depends on the quality of the blueprint and the time spent on its refinement after every single move in the game. To provide coherent results over multiple, different game instances, we set both of these time limits dependent on the complexity of the game. More precisely, the blueprint computation is stopped once 10 minutes have elapsed or column generation has achieved a Nash gap of $\Delta/10$, where Δ is the difference between the maximum and minimum team's payoffs, whichever comes first. We remark that the main goal of our experimental evaluation is to show that, no matter the suboptimality of the blueprint strategy and the amount of computation allocated to the subgame solving routine, the local re-optimization performed under our subgame solving framework allows us to improve the starting strategy. The choice of focusing on cases in which the blueprint strategy is suboptimal and in which the computational budget available for subgame solving is (even severely) limited goes precisely in that direction.

To create the strategy that would be played by Algorithm 7, we perform refinement at every public state in a top-down order. We use a range of time limits for the strategy refinement, defined as the average time needed by a single iteration of the CG algorithm at the root of the whole game multiplied by a number $\alpha \in \{0, 1, ..., 10\}$. Notice that the value obtained with $\alpha = 0$ is the value provided by the blueprint, while using $\alpha \geq 1$ ensures that at least one iteration of strategy refinement is done after every move in the game. Each experiment was allocated 32 CPU cores and 256 GB RAM on a cluster machine. Integer and linear programs were solved with Gurobi 9.5.

4.4.2 Discussion of the Results

For every game instance, we computed the best-response values for the opponent against the following strategies: the equilibrium strategy, the blueprint strategy, and the strategy returned by our refinement method for different values of α. We will use *gap* to refer to the difference between a value and the equilibrium value. Intuitively, it shows the relative quality of the blueprint when compared with the equilibrium. We use a performance index based on the concept of gap, showing the capability of our refinement method to reduce the gap. We call it *gap reduction*. It is defined as $1 - (E - R)/(E - B)$, where E, R, and B are the values of the equilibrium, refined strategies, and blueprint respectively.

Game instance	Team structure	Equilibrium time	Refinement time (per move)	Blueprint time	Equilibrium value	Refinement value	Blueprint value	Gap reduction
K34	110	0.05s	0.02s	0.02s	-0.042	-0.050	-0.061	58.0%
K36	110	0.55s	0.07s	0.03s	-0.024	-0.055	-0.098	58.3%
K38	110	2.58s	0.20s	0.04s	-0.019	-0.031	-0.152	91.2%
K312	110	10.73s	0.74s	0.29s	-0.014	-0.024	-0.064	78.9%
K45	1110	6.92s	0.54s	0.22s	-0.030	-0.046	-0.354	95.2%
L3133	110	6m 39s	0.57s	1.59s	0.215	0.038	-0.616	78.7%
L3143	110	>2h	2.59s	6.21s	0.107	-0.133	-0.673	69.2%
L3151	110	2m 46s	1.27s	2.80s	-0.019	-0.399	-0.624	37.3%
L3153	110	>2h	4.91s	8.84s	0.024	-0.177	-0.681	71.5%
L3223	110	7m 44s	1.02s	12.28s	0.516	-0.320	-1.299	54.0%
L3523	110	>2h	3m 21s	10m 46s	0.953	-1.151	-6.671	72.4%
D33	110	3m 55s	1.06s	9.15s	0.284	0.279	0.238	90.2%
D34	110	>2h	33.21s	10m 11s	0.284	0.241	0.139	70.0%
D62	111110	>2h	40.22s	10m 11s	0.333	0.167	-0.000	50.0%
T350	101	>2h	0.49s	1.09s	0.600	0.509	0.352	63.3%
T3100	101	>2h	1.16s	1.85s	0.710	0.514	0.495	8.5%

Table 4.1: Team's values against a best-responding opponent when playing the equilibrium strategy, blueprint, or refinement strategy with $\alpha = 5$, and corresponding algorithms running times. When the equilibrium computation requires more than 2 hours, we use the equilibrium values provided in [97]. A complete description of the columns' meaning can be found in-text.

Game instance	Team structure	Equilibrium time	Refinement time (per move)	Blueprint time	Equilibrium value	Refinement value	Blueprint value	Gap reduction
K34	110	0.05s	0.00s	0.02s	-0.042	-0.061	-0.061	0.0%
K36	110	0.55s	0.01s	0.03s	-0.024	-0.072	-0.098	34.6%
K38	110	2.58s	0.04s	0.04s	-0.019	-0.149	-0.152	2.0%
K312	110	10.73s	0.15s	0.29s	-0.014	-0.043	-0.064	42.5%
K45	1110	6.92s	0.11s	0.22s	-0.030	-0.084	-0.354	83.4%
L3133	110	6m 39s	0.12s	1.65s	0.215	-0.173	-0.616	53.3%
L3143	110	>2h	0.49s	5.84s	0.107	-0.266	-0.673	52.2%
L3151	110	2m 46s	0.26s	2.81s	-0.019	-0.442	-0.624	30.1%
L3153	110	>2h	0.99s	8.91s	0.024	-0.363	-0.681	45.1%
L3223	110	7m 44s	0.20s	12.25s	0.516	-0.525	-1.299	42.6%
L3523	110	>2h	39.95s	10m 39s	0.953	-1.953	-6.671	61.9%
D33	110	3m 55s	0.21s	9.07s	0.284	0.245	0.238	14.7%
D34	110	>2h	6.68s	10m 14s	0.284	0.208	0.139	47.7%
D62	111110	>2h	7.98s	10m 6s	0.333	-0.000	-0.000	0.0%
T350	101	>2h	0.10s	1.09s	0.600	0.509	0.352	63.3%
T3100	101	>2h	0.23s	1.85s	0.710	0.592	0.495	45.2%

Table 4.2: Team's values against a best-responding opponent when playing the equilibrium strategy, blueprint, or refinement strategy with $\alpha = 1$, and corresponding algorithms running times. When the equilibrium computation requires more than 2 hours, we use the equilibrium values provided in [97]. A complete description of the columns' meaning can be found in-text.

The values provided by the strategies when $\alpha = 5$ are provided in Table 4.1, together with the algorithms' runtimes. Similarly, Table 4.2 provides the results when $\alpha = 1$. The tables summarize the experiments for the different game instances and team structures (the i-th element of the vector equal to 1 means that player i is in team ▲). Equilibrium time, refinement time (per move), and blueprint time denote, respectively, the time needed to find an exact equilibrium via CG, the time allocated to a single iteration of subgame solving, and the time allotted to blueprint computation. The equilibrium value indicates the expected utility of the team at the equilibrium. In contrast, refinement and blueprint values represent the team's expected utility when they play against a best-responding opponent team and adopt the blueprint and refined strategies. The gap reduction column quantifies the improvement of the refined strategy over the blueprint strategy, as defined above.

We initially observe that our experimental evaluation confirms the maxmargin theoretical results as the team's exploitability never increases. Most importantly, with most game instances, the gap reduction is dramatic (on average, about 65%). These results are coherent with the results achieved in the literature for two-player zero-sum games [64, 8]. We also investigate how the performance of our strategy refinement method varies as α varies. In Figures 4.1 to 4.3, we report the results related to the full test suite.

Interestingly, we observe that, with most of the game instances, the curve describing the improvement provides the maximum increase by the first iteration of strategy refinement (corresponding to $\alpha = 1$). In particular, the average gap reduction is about 39% when $\alpha = 1$, showing that a time limit equal to the time needed for a single CG iteration at the root of the tree is sufficient to get a value very close to the equilibrium value. We also observe that the equilibrium value is never completely recovered. This is not surprising as strategy refinement methods perform a local optimization. The improvement sometimes does not monotonically increase in the number of iterations. This is due to the nature of CG algorithms.

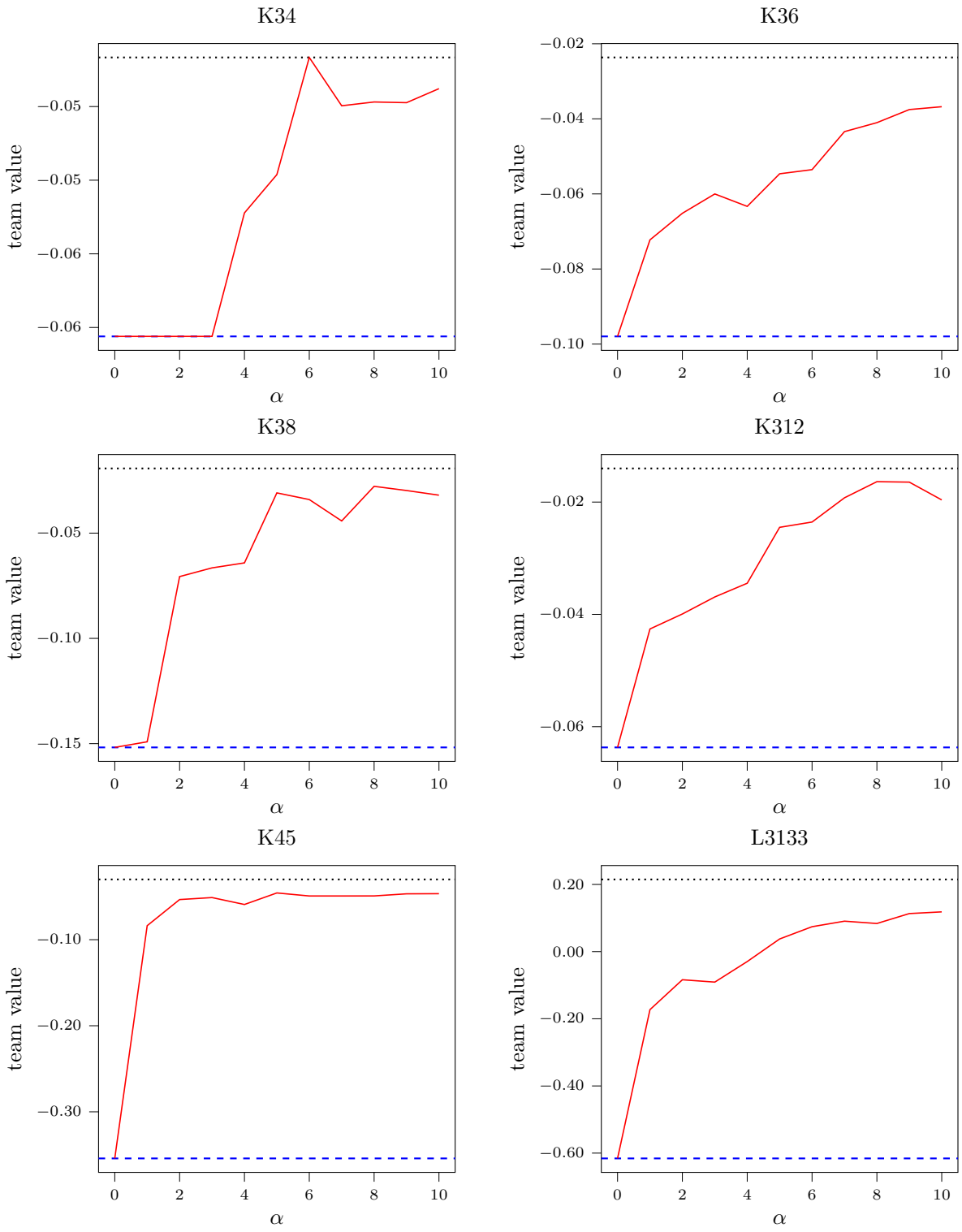


Figure 4.1: Value of the team's refined strategy as varying the refinement time limit in the K34, K36, K38, K312, K45, L3133 instances. The blue dashed line at the bottom is the blueprint value, while the black dotted line at the top is the equilibrium value.

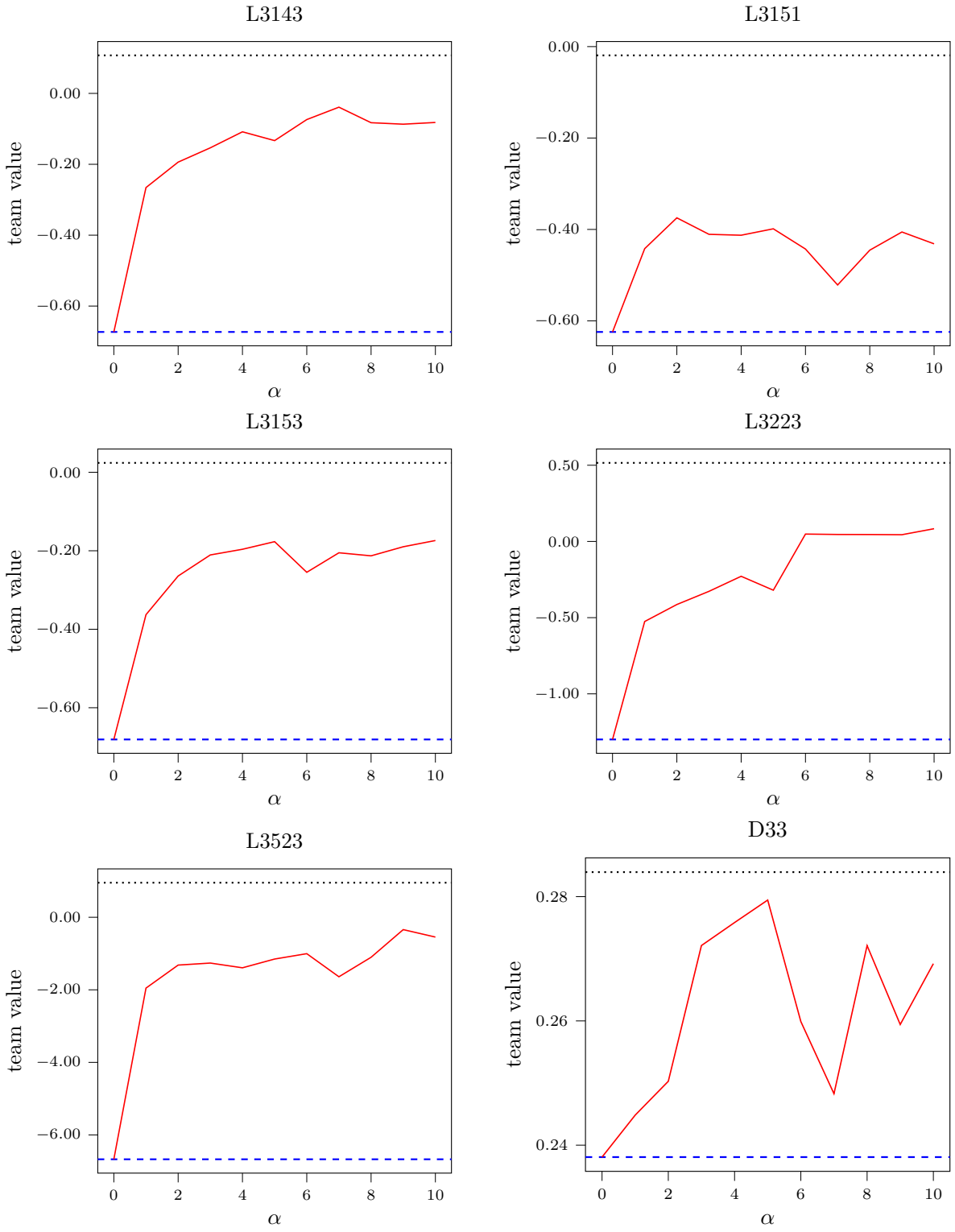


Figure 4.2: Value of the team's refined strategy as varying the refinement time limit in the L3143, L3151, L3153, L3223, L3523, D33 instances. The blue dashed line at the bottom is the blueprint value, while the black dotted line at the top is the equilibrium value.

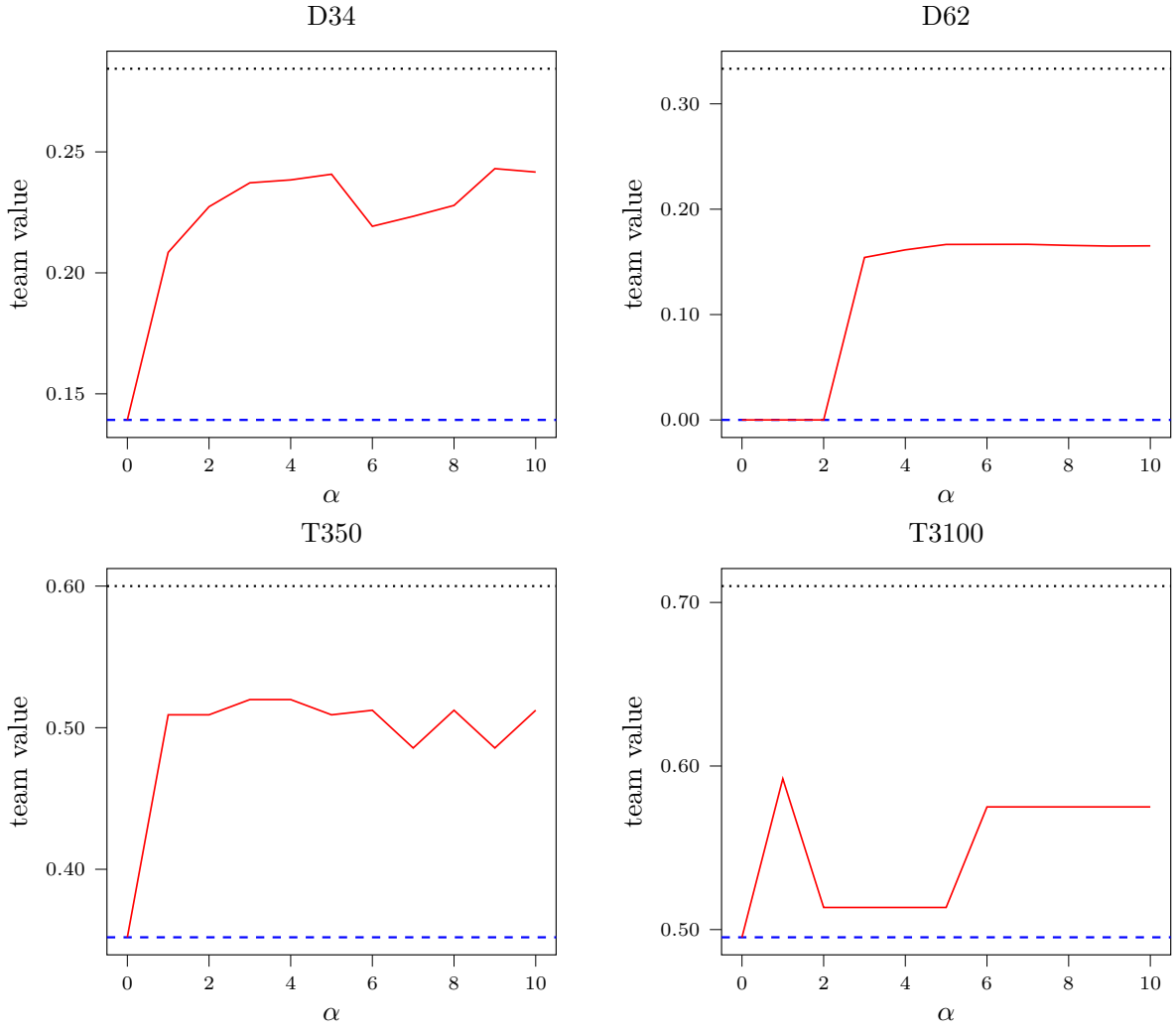


Figure 4.3: Value of the team's refined strategy as varying the refinement time limit in the D34, D62, T350, T3100 instances. The blue dashed line at the bottom is the blueprint value, while the black dotted line at the top is the equilibrium value.

Chapter 5

Hidden-Role Games: Equilibrium Concepts and Computation

In this chapter, the solution concepts from Definitions 2.4.3 and 2.4.4 originally defined for adversarial team games are used to characterize and solve a class of games called *hidden-role games*[1]. Hidden-role games model multi-agent systems in which a team of "good" agents work together to achieve some desired goal, but a subset of adversaries hidden among the agents seeks to sabotage the team. Customarily (and in our paper), the "good" agents make up a majority of the players, but they will not know who the adversaries are. On the other hand, the adversaries know each other.

Hidden-role games offer a framework for developing optimal strategies in systems and applications that face deception. They strongly emphasize communication: players need to communicate to establish trust, coordinate actions, exchange information, and distinguish teammates from adversaries.

Hidden-role games can model a wide range of recreational and real-world applications. Notable recreational examples include the popular tabletop games *Mafia* (also known as *Werewolf*) and *The Resistance*, of which *Avalon* is the best-known variant. As an example, consider the game *Mafia*. The players are split into an uninformed majority called *villagers*, and an informed minority called *mafiosi*. The game proceeds in two alternating phases, *night* and *day*. In the night phase, the mafiosi privately communicate and eliminate one of the villagers. In the day phase, players vote to eliminate a suspect through majority voting. The game ends when one of the teams is completely eliminated.

[1]These games are often commonly called *social deduction games*.

We now provide several non-recreational examples of hidden-role games. In many cybersecurity applications [32, 33, 85], an adversary compromises and controls some nodes of a distributed system whose functioning depends on cooperation and information sharing among the nodes. The system does not know which nodes have been compromised, yet it must operate in the presence of the compromised nodes.

Another instance of problems that can be modeled as hidden-role games arises in *AI alignment*, that is, the study of techniques to steer AI systems towards humans' intended goals, preferences, or ethical principles [106, 42, 39]. In this setting, there is the risk that a misaligned AI agent may attempt to deceive a human user into trusting its suggestions [72, 78]. AI debate [40] aims to steer AI agents by using an adversarial training procedure in which a judge has to decide which is the more trustworthy between two hidden agents, one of which is a deceiver trained to fool the judge. Miller et al. [63] proposes an experimental setting consisting of a chess game in which one side is controlled by a player and two advisors, which falls directly under our framework. The advisors pick action suggestions for the player to choose from, but one of the two advisors has the objective of making the team lose.

Hidden-role games also include general scenarios where agents receive inputs from other agents that may be compromised. For example, in *federated learning* (a popular category of distributed learning methods), a central server aggregates machine learning models trained by multiple distributed local agents. If some of these agents are compromised, they may send doctored input with the goal of disrupting the training process [66].

Our paper aims to characterize optimal behavior in these settings and analyze its computability. This is done by adapting the team-maxmin equilibrium solution concept to a carefully constructed representation of the game. Interestingly, the TME solution concept assumes that teams are known, but we show how to apply it to the hidden-role setting where teams are hidden by definition.

The structure of this chapter is as follows. Section 5.1 summarizes the literature on the hidden-role setting. Section 5.2 presents a high-level summary of the main contributions of this chapter, along with some examples of the setting considered.

Section 5.3 provides a formal definition of hidden-role games as a specific class of extensive-form games and of the various solution concepts distinguishing for the assumptions on the players' communication capabilities (*i.e.*, *no*, *public*, or *private* communication).

In Section 5.4, we either derive computational hardness results or present a polynomial-time algorithm finding one equilibrium for all but one of the solution concepts we define. In particular, our algorithm is based on: i) the identification of a shared mediator or its simulation

by a majority of players through the use of secure multi-party computation, and ii) a *revelation principle*, which restricts the communication between optimal players and such a mediator to either communication of private information or recommendation of actions. In the settings where our polynomial-time algorithm can be applied, the equilibrium finding problem can be cast as the computation of a Nash equilibrium in a two-player zero-sum game. Section 5.5 shows how the different solution concepts characterize players' behavior in an example game.

Section 5.6 studies some notable properties of HREs, such as the dependence on the order of commitment to strategies, the nontrivial effects of adding seemingly useless information to the game, and the *price of hidden roles* (*i.e.*, how much value a team loses due to roles being hidden).

Finally, in Section 5.7, we experimentally apply our technique to instances of *Avalon*[2], scaling up to 6-player games. The game tree of *Avalon*, even with only 5 players, contains more than 10^{56} information sets [80]. We address this by simplifying the game via manual analysis, reducing it to a size manageable for tabular solvers.

Sections B.1 to B.4 are appendices that further detail the notions presented in the previous sections and present the related proofs.

The material presented in this chapter is taken from a previous publication [17], published at the *ACM - Economics and Computation* conference as an extended abstract, and now under review to *Games and Economics Behavior (GEB)*.

5.1 Related Works

To our knowledge, no previous works have focused on the general hidden-role setting. On the other hand, there has been a limited amount of prior work on solving specific hidden-role games. Braverman et al. [7] propose an optimal strategy for *Mafia* and analyze the win probability when varying the number of players with different roles. Similarly, Christiano [21] proposes a theoretical analysis for *Avalon*, investigating the possibility of *whispering*, which consists of any two players being able to communicate without being discovered. Both papers describe game-specific strategies that players can adopt to guarantee specific utility to the teams. In contrast, to our knowledge, we provide the first rigorous definition of a reasonable solution concept for hidden-role games, an algorithm to find such equilibria, and an experimental evaluation with a wide range of parameterized instances.

[2]Avalon is more computationally challenging than Mafia thanks to a richer set of variants (Mafia admits easily computable equilibria). More details on Avalon can be found in Section 5.7.

Deep reinforcement learning techniques have also been applied to various hidden-role games [2, 47, 80], but with no theoretical guarantees and usually with no communication allowed between players. A more recent stream of works focused on investigating the deceptive capabilities of large language models (LLMs) by having them play a hidden-role game [93, 71]. The agents, being LLM-based, communicate using plain human language. However, as before, these are not grounded in any theoretical framework, and indeed, we will illustrate that optimal strategies in hidden-role games are likely to involve communication that does not resemble natural language, such as the execution of cryptographic protocols.

5.2 High-Level Contributions

5.2.1 Main Modeling Contributions

We define a (finite) *hidden-role game* as an n-player finite extensive-form game Γ in which the players are partitioned at the start of the game into two teams. Members of the same team share the same utility function, and the game is zero-sum, *i.e.*, any gain for one team means a loss for the other. We thus identify the teams as $\triangle$ and $\triangledown$, since teams share the same utility function, but have opposite objectives. At the start of the game, players are partitioned randomly into two teams. A crucial assumption is that one of the two teams is *informed*, which means that all the members of that particular team know the team assignment of all the players, while this is not true for all players belonging to the other team. Without loss of generality, we use $\triangle$ to refer to the uninformed team and $\triangledown$ to refer to the informed one.[3]

To allow our model to cover *communication* among players, we formally define the *communication extensions* of a game Γ. The communication extensions are games like Γ except that actions allowing messages to be sent between players are explicitly encoded in the game. In the *public communication extension*, players can publicly broadcast messages. In the *private communication extension*, in addition to the public broadcast channel, the players have pairwise private communication channels.[4] In all cases, communication channels are *synchronous* and *authenticated*: messages sent on one timestep are received at the next timestep and are tagged with their sender. Communication presents the main challenge of hidden role games: $\triangle$-players wish to share information with teammates, but *not* with $\triangledown$-players.

[3] For example, in *Mafia*, the villagers are $\triangle$ while mafiosi are $\triangledown$.

[4] If players are assumed to be computationally bounded, pairwise private channels can be created from the public broadcast channels through public-key cryptography. However, throughout this paper, for the sake of conceptual cleanliness, we will not assume that players are computationally bounded. Therefore, we will distinguish the public-communication case from the private-communication case.

In defining communication extensions, we must bound the *length* of the communication, that is, how many rounds of communication occur between every move of the game and how many distinct messages can be sent on each round. To do this, we fix a finite message space[5] of size M and length of communication R, and in our definition of equilibrium we will take a supremum over M and R. This will allow us to consider arbitrarily complex message spaces (with M and R arbitrarily large) while still only analyzing finite games: for any *fixed* M and R, the resulting game is a finite hidden-role game. We will show that our positive results (upper bounds) only require $\log M = R = \mathsf{polylog}(|H|, 1/\varepsilon)$, where $|H|$ is the number of nodes (histories) in the game tree and ε is the desired precision of equilibrium.

We characterize optimal behavior in the hidden-role setting by converting hidden-role games into team games in a way that preserves the strategic aspect of hidden roles. This team game is called *split-personality form* of a given hidden-role game. Given a (possibly communication-extended) hidden-role game Γ, we define and analyze two possible variants:

- the *uncoordinated split-personality form* $\textsc{USplit}(\Gamma)$ is a team games with $2n$ players, derived by splitting each player i in the original game in two distinct players, i^+ and i^-, that pick actions for i in Γ if the player is assigned to $\triangle$ or $\triangledown$ respectively. In the team game language established in Section 2.4, two players $\blacktriangle$ and $\blacktriangledown$, possibly with imperfect recall, control all the split personalities i^+ and i^- respectively.
- the *coordinated split-personality form* $\textsc{CSplit}(\Gamma)$ is the $(n+1)$-player team game in which the additional player, who we refer to simply as the *adversary* or $\triangledown$*-player*, takes control of the actions of all players who have been assigned to $\triangledown$. On the contrary, the players from 1 to n control the players as usual only if they belong to $\triangle$.

The coordinated split-personality variant encodes an extra assumption on $\triangledown$'s capabilities, namely, that $\triangledown$ is controlled by a single player $\blacktriangledown$ having perfect recall and is therefore perfectly coordinated. Trivially, when only one player is on $\triangledown$, the uncoordinated and coordinated split-personality forms coincide.

In either case, the resulting game is a team game in which each player has a fixed team assignment. We remark that the split-personality form maintains the strategic aspects of hidden roles since i^+ and i^- share identity when interacting with the environment. For example, players may observe that i has done an action but do not know if the controller was $\blacktriangle$ or $\blacktriangledown$. Similarly, messages sent by $\blacktriangle$ and $\blacktriangledown$ are signed by player i since the communication extension is applied on Γ *before* splitting personalities.

[5] Note that, if the message space is of size M, a message can be sent in $O(\log M)$ bits.

Picking which split-personality variant to use is a modeling assumption that depends on the game instance one wants to address. For example, in many recreational tabletop games, USPLIT is the more reasonable choice because ▽-players are truly distinct; however, in a network security game where a single adversary controls the corrupted nodes, CSPLIT is the more reasonable choice. The choice of which variant also affects the complexity of equilibrium computation: as we will detail later, CSPLIT yields a more tractable solution concept. In certain special cases, however, CSPLIT and USPLIT will coincide. For example, we will later show that this is the case in *Avalon*, which is key to allowing our algorithms to work in that game.

With these pieces in place, we define the *hidden-role equilibria* (HRE) of a hidden-role game Γ as the *team max-min equilibria* (TMEs) of the split-personality form of Γ. That is, the hidden-role equilibria are the optimal joint strategies for △ in the split-personality game, where optimality is judged by the expected value against a jointly-best-responding ▽. The *value* of a hidden-role game is the expected value for ▲ in any hidden-role equilibrium. If communication (private or public) is allowed, we define hidden-role equilibria and values by taking the supremum over M and R of the expected value at the equilibrium, that is, △ is allowed to set the communication parameters.

Our new solution concept encodes, by design, a pessimistic assumption for the ▲. ▲ picks M, R, and its strategy considering a worst-case ▼ adversary that knows this strategy and best responds to it. Throughout our proofs, we will heavily make use of this fact. In particular, we will often consider ▽-players that "pretend to be △-players" under certain circumstances, which is only possible if ▽-players know △-players' strategies. It is *not* allowed for △-players to know ▽-players' strategies in the same fashion. This starkly contrasts the usual zero-sum game analysis, where various versions of the *minimax theorem* promise that the game is unchanged no matter which side commits first to a strategy. Indeed, we discuss in Section 5.6.2 the fact that, for hidden-role games, the asymmetry is in some sense necessary: a minimax theorem *cannot* hold for nontrivial hidden-role games. We argue, however, that this asymmetry is natural and *inherent* in the hidden-role setting. If we assumed the contrary and inverted the order of the players in the HRE definition so that ▼ commits first to its strategy, ▲ could discover the roles immediately by agreeing to message a passphrase unknown to ▼ in the first round, thus spoiling the whole purpose of hidden-role games. This argument will be made formal in Section 5.6.2.

Existing solution concepts failures We defined our equilibrium notion as a team max-min equilibrium (TME) of the split-personality form of a communication-extended hidden role game. Here, we will argue why some other notions would be less reasonable.

- *Nash Equilibrium.* A *Nash equilibrium* [68] is a strategy profile for all players from which no player can improve its own utility by deviating. This notion is unsuitable for our purposes because it fails to capture *team coordination*. In particular, in pure coordination games (in which all players have the same utility function), which are a special case of hidden-role games (with no hidden roles and no adversary team at all), a Nash equilibrium would only be locally optimal in the sense that no player can improve the team's value alone. In contrast, our notion will lead to the optimal team strategy in such games.

- *Team-correlated equilibrium.* The *team max-min equilibrium with correlation* [90, 18] (TMECor) is a common solution concept used in team games. It arises from allowing each team the ability to communicate in private (in particular, to generate correlated randomness) *before* the game begins. For team games, TMECor is arguably a more natural notion than TME, as the corresponding optimization problem is a bilinear saddle-point problem, and therefore, in particular, the minimax theorem applies, avoiding the issue of which team ought to commit first. However, for hidden-role games, TMECor is undesirable because it does not make sense for a team to be able to correlate with teammates who have not even been assigned yet. The *team max-min equilibrium with communication* (TMECom) [18] makes an even stronger assumption about communication among team members and, therefore, suffers the same problem.

5.2.2 Main Computational Contributions

We now introduce computational results, both positive and negative, for computing the hidden-role value and hidden-role equilibria of a given game.

Polynomial-Time Algorithm Our main positive result is summarized in the following informal theorem statement.

Theorem 5.2.1 (Main result, informal; formal result in Theorem 5.4.3)**.** *If the number of players is constant, private communication is available, the* ∇ *is a strict minority (that is, strictly less than half of the players are on the* ∇*), and the adversary is coordinated, there is a polynomial-time algorithm for exactly computing the hidden-role value.*

This result should be surprising for multiple reasons. First, team games are generally hard to solve [90, 99], so any positive result for computing equilibria in team games is fairly surprising. Further, it is *a priori* not obvious that the value of any hidden-role game with private

communication and coordinated adversary is even a *rational* number[6], much less computable in polynomial time: for example, there exist adversarial team games with no communication whose TME values are irrational [90].

There are two key ingredients to the proof of Theorem 5.2.1. The first is a special type of game, which we call a *mediated* game. In a mediated game, there is a player, the *mediator*, who is always on Δ. Δ-players can, therefore, communicate with it and treat it as a trusted party. We show that when a mediator is present (and all the other assumptions of Theorem 5.2.1 also hold), the hidden-role value is computable in polynomial time. To do this, we state and prove a form of *revelation principle*. Informally, our revelation principle states that, without loss of generality, it suffices to consider $\blacktriangle$'s strategies in which, at every timestep of the game,

1. all Δ-players send their honest information to the mediator,
2. the mediator sends action recommendations to all players (regardless of their team allegiance; remember that the mediator may not know the team assignment), and
3. all Δ-players play their recommended actions.

This strategy can be implemented by $\blacktriangle$ in the split personality team game in behavioral strategies (required by the choice of TME solution concept). ∇-players are, of course, free to pretend to be Δ-players and thus send false information to the mediator; the mediator must deal with this possibility. However, ∇-players cannot just send *any* message; they must send messages that *are consistent with some* Δ*-player*, lest they be immediately revealed as ∇-players. These observations are sufficient to construct a *two-player zero-sum game* Γ_0, where the mediator is $\blacktriangle$ (since all the honest Δ-players are non-strategic since they are "controlled" by the mediator) and the coordinated adversary is $\blacktriangledown$. The value of Γ_0 is the value of the original hidden-role game, and the size of Γ_0 is at most polynomially larger than the size of the original game. Since two-player zero-sum extensive-form games can be solved in polynomial time [46, 88], mediated hidden-role games can also be solved in polynomial time.

The second ingredient is to invoke results from the literature on *secure multi-party computation* to *simulate* a mediator in the case that one is not already present. A well-known result from that literature states that so long as strictly more than half of the players are honest, essentially any interactive protocol—such as the ones used by our mediator to interact with other players—can be simulated efficiently such that the adversary can cause failure of the

[6] assuming all game values and chance probabilities are also rational numbers

protocol or leakage of information [5, 75].[7] Chaining such a protocol with the argument in the previous paragraph concludes the proof of the main theorem.

Related works on MPC and communication equilibria. The *communication equilibrium* [31, 67] is a notion of equilibrium with a mediator, in which the mediator has two-way communication with all players, and players need to be incentivized to report information honestly and follow recommendations. Communication equilibria include all Nash equilibria and, therefore, are unfit for general hidden-role games for the same reason as Nash equilibria, as discussed in the previous subsection.

However, when team ▼ has only one player, and private communication is allowed, the hidden-role equilibria coincide with the Δ-optimal communication equilibria in the original game Γ. Our main result covers this case, but an alternative way of computing a hidden-role equilibrium in this special case is to apply the optimal communication equilibrium algorithms of Zhang and Sandholm [95] or Zhang et al. [98]. However, those algorithms either involve solving linear programs, solving many zero-sum games, or solving zero-sum games with large reward ranges. This is less efficient than directly solving a single zero-sum game Γ_0.

We are not the first to observe that multi-party computation can be used to implement a mediator for use in game theory. In various settings and for various solution concepts, it is known to be possible to implement a mediator using only cheap-talk communication among players (see, for example, [86, 59, 1, 41]). For additional reading on the connections between game theory and cryptography, we refer the reader to the survey of Katz [44] and papers citing and cited by this survey.

Lower bounds. We also show *lower bounds* on the complexity of computing the hidden-role value, even for a constant number of players, when any of the assumptions in Theorem 5.2.1 is broken.

Theorem 5.2.2 (Lower bounds, informal; formal statement in Theorems 5.4.5 and 5.4.6)**.** *If private communication is disallowed, the hidden-role value problem is* NP*-hard. If* ▽ *is*

[7] In this part of the argument, the details about the communication channels become important: in particular, the MPC results that we use for our main theorem statement assume that the network is synchronous (that is, messages sent in round r arrive in round $r+1$), and that there are pairwise private channels and a public broadcast channel that are all authenticated (that is, message receivers know who sent the message). This is enough to implement MPC so long as $k < n/2$, where k is the number of adversaries and n is the number of players. Our results, however, do not depend on the specific assumptions about the communication channel, as long as said assumptions enable secure MPC with guaranteed outcome delivery. For a recent survey of MPC, see Lindell [57]. For example, if $k < n/3$, then MPC does not require a public broadcast channel, so neither do our results. We will stick to one communication model for cleanliness and to avoid introducing extra formalism.

uncoordinated, the problem is coNP*-hard. If both, the problem is* Σ_2^{P}*-hard. All hardness reductions hold even when* ∇ *is a minority, and the number of players is an absolute constant.*

Price of hidden roles. Finally, we define and compute the *price of hidden roles*. It is defined (analogously to the price of anarchy and price of stability, which are common quantities of study in game theory) as the ratio between the value of a hidden-role game and the value of the same game with team assignments made public. We show the following:

Theorem 5.2.3 (Price of hidden roles; formal statement in Theorem 5.6.1)**.** *Let* D *be a distribution of team assignments. For the class of games where teams are drawn according to distribution* D*, the price of hidden roles is equal to* $1/p$*, where* p *is the probability of the most likely team in* D*.*

Intuitively, in the worst case, Δ can be forced to guess at the beginning of the game all the members of Δ, and win if and only if its guess is correct. In particular, for the class of n-player games with k adversaries, the price of hidden roles is exactly $\binom{n}{k}$.

5.2.3 Experiments: *Avalon*

We ran experiments on the popular tabletop game *The Resistance: Avalon* (or simply *Avalon*, for short). As discussed earlier, despite the adversary team in *Avalon* not being coordinated in the sense used in the rest of the paper, we show that, at least for the 5- and 6-player variants, the adversary would not benefit from being coordinated; therefore, our polynomial-time algorithms can be used to solve the game. This observation ensures that our main result applies. Game-specific simplifications allow us to reduce the game tree from roughly 10^{56} nodes [80] to 10^8 or even fewer, enabling us to compute exact equilibria. Our experimental evaluation demonstrates the practical efficacy of our techniques in real-game instances. Our results are discussed in Section 5.7, and further detail on the game-specific reductions used, as well as a complete hand-analysis of a small *Avalon* variant, can be found in Section B.4.

5.2.4 Examples

In this section, we present three examples that will hopefully help the reader understand our notion of equilibrium and justify some choices we have made in our definition.

A hidden-role matching pennies game. Consider a n-player version of matching pennies (with $n > 2$), which we denote as $\mathsf{MP}(n)$. One player is chosen randomly as the adversary (team ▼). All n players then simultaneously choose bits $b_i \in \{0, 1\}$. △ wins (gets utility 1) if and only if all n bits match; else, △ loses (gets utility 0).

With no communication, the value of this game is $1/2^{n-1}$: it is an equilibrium for everyone to play uniformly at random. Public communication does not help because, conditioned on the public transcript, bits chosen by players must be mutually independent. Thus, the adversary can do the following: pretend to be on △, wait for all communication to finish, and then play 0 if the string of all ones is more conditionally likely than the string of all 1s, and vice-versa.

With private communication, however, the value becomes $1/(n + 1)$. Intuitively, △ should attempt to guess who the ▽-player is, and then privately discuss among the remaining $n - 1$ players what bit to play. We defer formal proofs of the above game values to Section 5.5, because they rely on results in Section 5.4.1.

Simultaneous actions. In typical formulations of extensive-form imperfect-information games, it is without loss of generality to assume that games are *turn-based*, *i.e.*, only one player acts at any given time. To simulate simultaneous actions with sequential ones, one can simply forbid players from observing each others' actions. However, when communication is allowed arbitrarily throughout the game, the distinction between simultaneous and sequential actions suddenly becomes relevant, because *players can communicate when one—but not all—the players have decided on an action*.

To illustrate this, consider the game $\mathsf{MP}(n)$ defined in the previous section, with public communication, except that the players act in sequence in order of index $(1, 2, \ldots, n)$. We claim that the value of this game is not $1/2^{n-1}$, but at least $1/2n$. To see this, consider the following strategy for △. The △-players wait for P1 to (privately) pick an action. Then, P2 publicly declares a bit $b \in \{0, 1\}$, and all remaining players play b if they are on △. If P1 was the ▽-player, this strategy wins with probability at least $1/2$, so the expected value is at least $1/2n$. This example illustrates the importance of allowing simultaneous actions in our game formulations.

Correlated randomness matters. We use our third and final example to discuss a nontrivial consequence of the definition of hidden-role equilibrium that may appear strange initially: seemingly useless information can affect strategic decisions and the game value.

To illustrate, consider the following simple game Γ: there are three players, and three role cards. Two of the three cards are marked $\triangle$, and the third is marked $\triangledown$. The cards are dealt privately and randomly to the players. Then, after some communication, all three players simultaneously cast votes to elect a winner. If no player gains a majority of votes, $\triangledown$ wins. Otherwise, the elected winner's team wins. Clearly, $\triangle$ can win no more than $2/3$ of the time in this game: $\triangledown$ can simply pretend to be on $\triangle$, and in that case $\triangle$ cannot gain information, and the best they can do is elect a random winner.

Now consider the following seemingly meaningless modification to the game. We will modify the two $\triangle$ cards so that they are distinguishable. For example, one card has $\triangle$ written on it, and the other has $\triangle'$. We argue that this, perhaps surprisingly, affects the value of the game. In fact, $\triangle$ can now win deterministically, even with only public communication. Indeed, consider the following strategy. The two players on $\triangle$ publicly declare what is written on their cards (namely, $\triangle$ or $\triangle'$). The player elected now depends on what the third player did. If one player does not declare $\triangle$ or $\triangle'$, elect either of the other two players. If two players declared $\triangle$, elect the player who declared $\triangle'$. If two players declare $\triangle'$, elect the player who declared $\triangle$. This strategy guarantees a win: no matter what the $\triangledown$-player does, any player who makes a unique declaration is guaranteed to be on $\triangle$.

What happens in the above example is that making the cards distinguishable introduces a piece of *correlated randomness* that $\triangle$ can use: the two $\triangle$-players receive cards whose labels are (perfectly negatively) correlated with each other. Since our definition otherwise prohibits the use of such correlated randomness (because players cannot communicate only with players on a specific team), introducing some into the game can have unintuitive effects. In Section 5.6.2, we expand on the effects of allowing correlated randomness: in particular, we argue that allowing correlated randomness essentially ruins the point of hidden-role games by allowing $\triangle$ to learn the entire team assignment.

5.3 Equilibrium Concepts for Hidden-Role Games

5.3.1 Additional Notation

We start with the formalization of the specific notation adopted in this chapter.

Hidden-role games vs Adversarial team games Hidden-role games are a different class of games than adversarial team games, even if both have players associated with a team. As

anticipated in Section 5.2.1, the split-personality forms USPLIT and CSPLIT allow to express a hidden-role game as an adversarial team game. The difference in the two settings implies slightly different notations that distinguish the two models from a formal perspective. We stick to the imperfect-recall notation established in Section 2.4 to characterize team play in an adversarial team game. This includes the split-personalities forms. On the other hand, there will be no imperfect-recall representation for hidden-role games before applying USPLIT and CSPLIT. We will therefore use *teams* $\triangle$ and $\triangledown$, and talk about single players $i \in \mathcal{N}$ whose team is not fixed (and whose strategy depend on the team assigned at the start of the game). This differs from $\blacktriangledown$ and $\blacktriangle$ *players* in adversarial team games.

Simultaneous actions As anticipated in Section 5.2.4, we tweak the standard extensive-form setting from Definition 2.1.1 to consider simultaneous actions. This eases the notation adopted to represent concurrent communication among players. In particular, we have that each player $i \in \mathcal{N}$ has an information set $I_i(h)$ defined at all $h \in \mathcal{H}$, and all players pick an action $a \in A_{I_i(h)}$ that the determines the transition to the next node $h\boldsymbol{a} = h'$. This change in extensive-form formalism does not impact the expressivity of the model since any simultaneous-action game can be rewritten as a game having a single active player per node as described in Footnote 8. Vice versa, all non-acting players can be described in a simultaneous actions formalism by introducing dummy information sets that propagate the previous information set received by the players and an action set with a single dummy action. The definition of sequences and imperfect recall does not change from Section 2.1.1.

Team-maxmin equilibria Team Maxmin Equilibria (TMEs) ([90, 18], defined at Definition 2.4.4) on the split-personality game are used to characterize equilibria in hidden-role games. In particular, we are interested in the *value* of the game at the equilibrium. The value of a given strategy $\boldsymbol{x}$ for $\triangle$ is the value that $\boldsymbol{x}$ achieves against a best-responding opponent $\triangledown$. The *TME value* is the value of the best strategy profile of $\triangle$. That is, the TME value of a game Γ is defined as

$$\mathsf{TMEVal}(\Gamma) := \max_{\boldsymbol{x} \in \hat{\mathcal{X}}} \min_{\boldsymbol{y} \in \hat{\mathcal{Y}}} \sum_{z \in \mathcal{Z}} \boldsymbol{x}[z]\boldsymbol{y}[z]u(z), \tag{5.1}$$

and the *TMEs* are the strategy profiles $\boldsymbol{x}$ that achieve the maximum value. To ease the exposition, we split $\blacktriangle$'s and $\blacktriangledown$'s behavioral strategies $\boldsymbol{x}$ and $\boldsymbol{y}$ into components controlling single players on their teams. We write $\boldsymbol{x} = (x_1, x_2, \dots, x_{|\triangle|}), \boldsymbol{y} = (y_1, y_2, \dots, y_{|\triangledown|})$ to describe such splits. Note that to indicate the strategies of single players, we avoid using boldface fonts, even if those strategies are still realization-form vectors.

Notice that the TME problem is nonconvex since the objective function u is nonlinear as a function of $\boldsymbol{x}$ and $\boldsymbol{y}$. As such, the minimax theorem does not apply, and swapping the teams may not preserve the solution. Computing an (approximate) TME is Σ_2^{P}-complete in extensive-form games (see Section 3.5).

5.3.2 Hidden-Role Games

While the notion of TME is well-suited for ATGs, it is not immediately clear how to generalize it to the setting of hidden-role games to characterize the behavior of uncorrelated teammates. We do so by formally defining the concepts of *hidden-role game*, *communication*, and *split-personality form* first introduced in Section 5.2.1.

Definition 5.3.1. An extensive-form game is a *zero-sum hidden-role team game*, or *hidden-role game* for short, if it satisfies the following additional properties:

1. At the root node, only chance has a nontrivial action set. Chance chooses a string $t \sim \mathcal{D} \in \Delta(\{\triangle, \triangledown\}^n)$, where t_i denotes the team to which player i has been assigned. Each player i's information set encodes (at least[8]) their team assignment t_i. In addition, $\triangledown$-players privately observe the entire team assignment t.

2. The utility of a player i is defined completely by its team: there is a $u : Z \to \mathbb{R}$ for which $u_i(z) = u(z)$ if player i is on team $\triangle$ at node z, and $-u(z)$ otherwise.[9]

In some games, players observe additional information beyond just their team assignments. For example, in *Avalon*, one $\triangle$-player is designated *Merlin*, and Merlin has additional information compared to other $\triangle$-players. In such cases, we will distinguish between the *team assignment* and *role* of a player: the team assignment is just the team that the player is on ($\triangle$ or $\triangledown$), while the role encodes the extra private information of the player as well, which may affect what actions that player is allowed to legally take. For example, the team assignment of the

[8] It is allowable for $\triangle$-players to also have more observability of the team assignment. For example, certain $\triangle$-players may know who some $\triangledown$-players are.

[9] While at a first look this condition is similar to the one in ATGs, we remark that in this case the number of players in a team depends on the roles assigned at the start. The term zero-sum is, therefore, a slight abuse of language: if the $\triangle$ and $\triangledown$ teams have different sizes, the sum of all players' utilities is not zero. However, such a game can be made zero-sum by properly scaling each player's utility. The fact that such a rescaling operation does not affect optimal strategies is a basic result for von Neumann–Morgenstern utilities [61, Chapter 2.4]. We will therefore generally ignore this detail.

player with role *Merlin* is Δ. We remark that additional imperfect information about the game may be observed after the root node.[10]

Throughout this paper, we will use k to denote the largest number of players on ∇, that is, $k = \max_{t \in \text{supp}(\mathcal{D})} |\{i : t_i = \nabla\}|$.

5.3.3 Models of Communication

The bulk of this paper concerns notions of equilibrium that allow communication between the players. We distinguish in this paper between *public* and *private communication*:

1. *Public communication*: There is an open broadcast channel on which all players can send messages.
2. *Private communication*: In addition to the open broadcast channel, each pair of players has access to a private communication channel. The private communication channel reveals to all players when messages are sent, but only reveals the message contents to the intended recipients.

Assuming that public-key cryptography is possible (for example, assuming the discrete logarithm problem is hard) and players are polynomially computationally bounded, public communication and private communication are equivalent because players can set up pairwise private channels via public-key exchange. However, in this paper, we assume that agents are computationally unbounded and thus treat the public and private communication cases as different. Our motivation for making this distinction is twofold. First, it is conceptually cleaner to explicitly model private communication because then our equilibrium notion definitions do not need to reference computational complexity. Second, perhaps counterintuitively, equilibria with public communication only may be *more* realistic to execute in practice in human play, precisely *because* public-key cryptography breaks. That is, the computationally unbounded adversary renders more "complex" strategies of Δ (involving key exchanges) useless, thus perhaps resulting in a *simpler* strategy. We emphasize that, in all of our positive results in the paper, Δ's strategy *is* efficiently computable.

To formalize these notions of communication, we now introduce the *communication extension*.

[10] This is an important difference with respect to Bayesian games [36], which assume all imperfect information to be the initial *types* of the players. Conversely, we have an imperfect information structure that evolves throughout the game, while only the teams are assigned and observed at the start.

Definition 5.3.2. The *public* and *private* (M, R)*-communication extensions* corresponding to a hidden-role game Γ are defined as follows. Informally, between every step of the original game Γ, there will be R rounds of communication; in each round, players can send a public broadcast message and private messages to each player. The communication extension starts in state $h = \varnothing \in H_\Gamma$. At each game step of Γ:

1. Each player $i \in [n]$ observes $I_i(h)$.
2. For each of R successive communication rounds:
   (a) Each player i simultaneously chooses a message $m_i \in [M]$ to broadcast publicly.
   (b) If private communication is allowed, each player i also chooses messages $m_{i\to j} \in [M] \cup \{\perp\}$ to send to each player $j \neq i$. $\perp$ denotes that the player does not send a private message at that time.
   (c) Each player j observes the messages $m_{i\to j}$ sent to it, as well as all messages m_i that were sent publicly. That is, according to the notion of communication considered, the players observe:
      - *Public*: player j observes the ordered tuple $(m_1, \ldots, m_n)$.
      - *Private*: player j also observes the ordered tuple $(m_{1\to j}, \ldots, m_{n\to j})$, and the set $\{(i, k) : m_{i\to k} \neq \perp\}$. That is, players observe messages sent to them, and players see when other players send private messages to each other (but not the contents of those messages)
3. Each player, including chance, simultaneously plays an action $a_i \in A_{I_i(h)}$. (Chance plays according to its fixed strategy.) The game state h advances accordingly.

We denote the (M, R)-extensions as $\textsc{Comm}^{M,R}_{\mathsf{priv}}(\Gamma)$, and $\textsc{Comm}^{M,R}_{\mathsf{pub}}(\Gamma)$. To unify notation, we also define $\textsc{Comm}^{M,R}_{\mathsf{none}}(\Gamma) = \Gamma$. When the type of communication allowed and the number of rounds are irrelevant, we use $\textsc{Comm}(\Gamma)$ to refer to a generic extension.

5.3.4 Split Personalities

We introduce two different *split-personality* forms $\textsc{USplit}(\Gamma)$ and $\textsc{CSplit}(\Gamma)$ of a hidden-role game Γ; the split-personality forms are adversarial team games that preserve the characteristics of Γ.

Definition 5.3.3. The *uncoordinated split-personality form*[11] of an n-player hidden-role game Γ is the $2n$-player adversarial team game $\text{USPLIT}(\Gamma)$ in which each player i is split into two players, i^+ and i^-, which control player i's actions when i is on team $\triangle$ and team $\triangledown$ respectively.

Unlike the original hidden-role game Γ, the split-personality game is an adversarial team game without hidden roles: therefore, players i^+ can now be merged in $\blacktriangle$, and i^- can be merged in $\blacktriangledown$ (equivalently, i^+ are fixed on the $\triangle$ team, and i^- are on the $\triangledown$-team). Therefore, we can apply notions of equilibrium for adversarial team games to $\text{USPLIT}(\Gamma)$. We also define the *coordinated split-personality form*:

Definition 5.3.4. The *coordinated split-personality form* of an n-player hidden-role game Γ is the $(n+1)$-player adversarial team game $\text{CSPLIT}(\Gamma)$ formed by starting with $\text{USPLIT}(\Gamma)$ and merging all $\triangledown$-players into a single adversary player, who observes all their information sets and chooses all their actions.

The previous definition guarantees that $\blacktriangledown$ has perfect-recall (that is, $\triangledown$ players can perfectly coordinate, hence the name). Assuming $\triangledown$ to be *coordinated* is a worst-case assumption for team $\triangle$, which, however, can be justified. In many common hidden-role games, such as the *Mafia* or *Werewolf* family of games and most variants of *Avalon*, such an assumption is not problematic because the $\triangledown$-team has essentially perfect information already. In Section B.4.1, we justify why this assumption is safe also in some more complex *Avalon* instances considered. The coordinated split-personality form will be substantially easier to analyze and in light of the above equivalence for games like *Avalon*, we believe that it is important to study it.

When $\triangledown$ in Γ is already coordinated, that is, if every $\triangledown$ member has the same observation at every timestep, the coordinated and uncoordinated split-personality games will, for all our purposes, coincide: in this case, any strategy of the adversary in $\text{CSPLIT}(\Gamma)$ can be matched by a joint strategy of the $\triangledown$-team members in $\text{USPLIT}(\Gamma)$. This is true in particular if there is only one $\triangledown$-team member. But, we insert here a warning: even when the base game Γ has a coordinated adversary team, the private communication extension $\text{COMM}_{\text{priv}}(\Gamma)$ will not. Thus, with private-communication extensions of Γ, we must distinguish the coordinated and uncoordinated split-personality games even if Γ itself is coordinated.

5.3.5 Equilibrium Notions

We now define the notions of equilibrium that we will primarily study in this paper.

[11] In the language of Bayesian games, the split-personality form would almost correspond to the *agent form*.

Definition 5.3.5. The *uncoordinated value* of a hidden-role game Γ with notion of communication c is defined as

$$\mathsf{UVal}_c(\Gamma) := \sup_{M,R} \mathsf{UVal}_c^{M,R}(\Gamma)$$

where $\mathsf{UVal}_c^{M,R}(\Gamma)$ is the TME value of $\textsc{USplit}(\textsc{Comm}_c^{M,R}(\Gamma))$. The *coordinated value* $\mathsf{CVal}_c(\Gamma)$ is defined analogously by using $\textsc{CSplit}$.

Definition 5.3.6. An *ε-uncoordinated hidden-role equilibrium* of Γ with a particular notion of communication $c \in \{\mathsf{none}, \mathsf{pub}, \mathsf{priv}\}$ is a tuple (M, R, x) where x is a $\triangle$-strategy profile in $\textsc{USplit}(\textsc{Comm}_c^{M,R}(\Gamma))$ of value at least $\mathsf{UVal}_c(\Gamma) - \varepsilon$. The *$\varepsilon$-coordinated hidden-role equilibria* is defined analogously, again with $\textsc{CSplit}$ and CVal instead of $\textsc{USplit}$ and UVal.

As discussed in Section 5.2.1, our notion of equilibrium is inherently asymmetric due to its max-min definition. $\triangle$ is the first to commit to a strategy and a communication scheme, and $\triangledown$ is allowed to know both how much communication will be used (that is, M and R) as well as $\triangle$'s entire strategy $\boldsymbol{x}$. As mentioned before, this asymmetry is fundamental in our setting, and we will formalize it in Section 5.6.2.

5.4 Computing Hidden-Role Equilibria

In this section, we show the main computational results regarding the complexity of computing an hidden-role equilibrium in different settings. We first provide positive results for the private-communication case in Section 5.4.1 while the negative computational results for the no/public-communications cases are presented in Section 5.4.2. The results are summarized in Table 5.1.

5.4.1 Computing Private-Communication Equilibria

In this section, we show that it is possible, under some assumptions, to compute equilibria efficiently for hidden-role games. In particular, in this section, we assume that

1. there is private communication,
2. the adversary is coordinated, and
3. the adversary is a minority ($k < n/2$).

Games with a publicly-known $\triangle$-player. First, we consider a special class of hidden-role games which we call *mediated*. In a mediated game, there is a player, the *mediator*, who is always assigned to team $\triangle$. The task of the mediator is to coordinate the actions and information transfer of $\triangle$. Our main result of this subsection is the following:

Theorem 5.4.1 (Revelation Principle)**.** *Let Γ^* be a mediated hidden-role game. Then, for $R \geq 2$ and $M \geq |H|$, there exists a coordinated private-communication equilibrium in which $\blacktriangle$ has a TME profile in which, at every step, the following events happen in sequence:*

1. *$\blacktriangle$ coordinates every player on $\triangle$ to send its information set $I_i(h)$ privately to the mediator,*
2. *$\blacktriangle$ picks a specific action for the mediator to send to each player (in both $\triangle$ and $\triangledown$), and*
3. *$\blacktriangle$ coordinates every player on team $\triangle$ to play their recommended actions.*

By operating as described, $\blacktriangle$ can overcome its imperfect recall while playing a behavioral strategy since it can coordinate all players in the team by strategically specifying the actions recommended by the mediator to the players, and the mediator has perfect recall of the team information. Players on team $\triangledown$, of course, can (and will) lie or deviate from recommendations as they wish.

We now prove that constraining $\blacktriangle$'s play in this way is effectively optimal, *i.e.*it does not reduce its value.

Proof. We show that any TME strategy $(\boldsymbol{x}, \boldsymbol{y})$ can be transformed into an outcome equivalent strategy $(\boldsymbol{x}', \boldsymbol{y}')$ as in Theorem 5.4.1. Intuitively, $\boldsymbol{x}'$ describes $\blacktriangle$ coordinating all players delegate control to the mediator, and the mediator plays in such a way that $\boldsymbol{x}$ is realized, without leaking exploitable information to $\blacktriangledown$ other than $\boldsymbol{x}$ which $\blacktriangledown$ already knew thanks to the max-min structure. Outcome equivalence guarantees that $(\boldsymbol{x}', \boldsymbol{y}')$ is a TME as well.

Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ be any strategy profile for team $\blacktriangle$ in $\textsc{CSplit}(\textsc{Comm}_{\mathsf{priv}}(\Gamma^*))$. Consider the strategy profile $\boldsymbol{x}' = (x'_1, x'_2, \ldots, x'_n)$ that operates as Theorem 5.4.1, with the mediator's strategy described as follows. For each player i, the mediator instantiates a simulated version of each player i playing according to strategy x_i. These simulated players are entirely "within the imagination of the mediator".

1. When a (real) player i sends an infoset $I_i(h)$ to the mediator, the mediator forwards infoset $I_i(h)$ to the simulated player i.

2. When a simulated player i wants to send a message to another player j, the mediator forwards the message to the *simulated* player j.

3. When a simulated player i plays an action a_i, the mediator forwards the action as a message to the real player i.

Since the strategy x_i is only well-defined on sequences that can arise in $\text{CSPLIT}(\text{COMM}_{\text{priv}}(\Gamma^*))$, the simulated player i may crash if it receives an impossible sequence of infosets. If player i's simulator has crashed, it will no longer send simulated messages, and the mediator will no longer send messages to player i.

It suffices to show that ▼ cannot exploit $\boldsymbol{x}'$ more than $\boldsymbol{x}$. Let $\boldsymbol{y}'$ be any best-response strategy profile for ▼ against $\boldsymbol{x}'$. We will show that there exists a strategy $\boldsymbol{y}$ such that $(\boldsymbol{x}', \boldsymbol{y}')$ is equivalent to $(\boldsymbol{x}, \boldsymbol{y})$. Consider the strategy $\boldsymbol{y}$ for ▼ in which each player i maintains simulators of both mediator strategy x'_M and y'_i, and acts as follows.

1. Forward the current infoset $I_i(h)$ to y'_i.

2. If y'_i wants to send any message to the mediator, forward it to x'_M.

3. If x'_M sends a message, send that message to y'_i.

4. Play the action output by y'_i whenever the communication round ends.

An alternative definition of y'_i could be proposed to make it depend directly on x_i rather than on x'_M.[12]

By definition, the profiles $(\boldsymbol{x}, \boldsymbol{y})$ and $(\boldsymbol{x}', \boldsymbol{y}')$ have the same expected utility (in fact, they are equivalent in the sense that they induce the same outcome distribution over the terminal nodes of Γ), so we are done. □

The above revelation principle implies the following algorithmic result:

Theorem 5.4.2. *Let Γ^* be a mediated hidden-role game, $R \geq 2$, and $M \geq |\mathcal{H}|$. An (exact) coordinated private-communication hidden-role equilibrium of Γ^* can be computed by solving an extensive-form zero-sum game Γ_0 with at most $|\mathcal{H}|^{k+1}$ nodes, where $\mathcal{H}$ is the node set of Γ^*.*

[12] Each player i maintains simulators of both mediator strategy x_i and y'_i, and acts as follows.
- Forward the current infoset $I_i(h)$ to y'_i.
- If y'_i wants to send any message to the mediator, forward it to x_i.
- If x_i sends a message, send that message.
- If x_i plays an action a_i, forward that action to y'_i as a message from the mediator. If x_i crashes, send empty messages to y'_i from the mediator. In either case, when y'_i outputs an action, play that action.

Proof. We describe a two-player zero-sum game Γ_0 where a mediator suggests actions to all players, and an adversary coordinates the member of the opposing team in reposting false information and picking deviating actions.

There are two players: $\blacktriangle$ is the *mediator*, controlling $\triangle$, and $\blacktriangledown$ controls the coordinated adversary, specifying actions for $\triangledown$. The game state of Γ_0 consists of a node h in Γ (initially the root), and n *sequences* σ_i. Define a sequence σ_i to be *consistent* if it is a prefix of a terminal sequence $\sigma_i(z)$ for some terminal node z of Γ. For each sequence σ_i[13] let $I(\sigma_i)$ be the set of information sets that could be the next information sets of player i in Γ.

1. For each player i on $\triangle$, let the infoset to report to the mediator be $\tilde{I}_i = I_i(h)$ (that is, the true infoset of player i). $\blacktriangledown$ has an information set uniquely identifying the set of information sets $I_i(h)$ for players i on $\triangledown$ and for each such player, $\blacktriangledown$ picks an infoset $\tilde{I}_i \in I(\sigma_i) \cup \{\bot\}$ to be a possibly untruthful or explicitly deviating infoset report.

2. Each $\tilde{I}_i$ is appended to the corresponding σ_i.

3. The mediator information set univoquely identifies $(\tilde{\sigma}_1, \dots, \tilde{\sigma}_n)$, and picks action recommendations $a_i \in A_i(\tilde{I}_i)$ to recommend to each player i, and appends each a_i to the corresponding σ_i. In case of a sequence ending in $\bot$, the mediator does not pick any action.

4. Players on $\triangle$ automatically play their recommended actions. The adversary observes all actions $(a_i : t_i = \triangledown)$ recommended to players on $\triangledown$, and selects the action played by each member of $\triangledown$.

The node h in the game state is then changed into $h' = h\boldsymbol{a}$, and the communication, recommendation, and action selection cycle continues until a terminal node $h \in \mathcal{Z}$ in Γ^* is reached.

The size of this game is given by the number of tuples of the form $(h, \sigma_1, \dots, \sigma_k)$ where σ_i is the sequence of adversary i and h is a node of the original game Γ, since the truthful sequences of players in $\triangle$ are uniquely determined by h. There are at most $|\mathcal{H}|^{k+1}$ of these, so we are done. □

The key to proving Theorem 5.4.2 is that, in the first step above, the adversary's message space is not too large. Indeed, any message sent by the adversary must be a message that *could have plausibly been sent by a* $\triangle$*-player*: otherwise, the mediator could automatically infer that

[13] We remark that sequences are made of (infoset, action) pairs and exclude the current node's information set by Definition 2.1.2.

the sender must be the adversary. It is therefore possible to exclude all other messages from the game since they belong to dominated strategies.

It is crucial in the above argument that ∇ is coordinated; indeed, otherwise, it would not be valid to model ∇ as a single adversary in Γ_0. For more elaboration on the case where ∇ is not coordinated, we refer the reader to the Avalon example from Section B.4.1.

In practice, zero-sum extensive-form games can be solved very efficiently in the tabular setting with linear programming [46], or algorithms from the counterfactual regret minimization (CFR) family [10, 29, 108]. Thus, Theorem 5.4.2 gives an efficient algorithm for solving hidden-role games with a mediator.

Simulating mediators with multi-party computation In this section, we show that the previous result essentially generalizes (up to exponentially small error) to games *without* a mediator, so long as the ∇ team is also a minority, that is, $k < n/2$. Informally, the main result of this subsection states that, when private communication is allowed, one can efficiently *simulate* the existence of a mediator using secure multi-party computation (MPC), and therefore, team Δ can achieve the same value. The form of secure MPC that we use is *information-theoretically* secure; that is, it is secure even against computationally unbounded adversaries.

Theorem 5.4.3 (Main theorem)**.** *Let Γ be a hidden-role game with $k < n/2$. Then $\mathsf{CVal}_{\mathsf{priv}}(\Gamma) = \mathsf{CVal}_{\mathsf{priv}}(\Gamma^*)$, where Γ^* is Γ with a mediator added, and moreover this value can be computed in $|H|^{O(k)}$ time by solving a zero-sum game of that size. Moreover, an ε-hidden-role equilibrium with private communication and $\log M = R = \mathsf{polylog}(|H|, 1/\varepsilon)$ can be computed and executed by the Δ-players in time $\mathsf{poly}(|H|^k, \log(1/\varepsilon))$.*

The proof uses MPC to simulate the mediator and then executes the equilibrium given by Theorem 5.4.2. The proof of Theorem 5.4.3, as well as requisite background on multi-party computation, are deferred to Section B.1. We emphasize that Theorems 5.4.2 and 5.4.3 are useful not only for algorithmically computing an equilibrium but also for manual analysis of games: instead of analyzing the infinite design space of possible messaging protocols, it suffices to analyze the finite zero-sum game Γ_0. Our experiments on *Avalon* use both manual analysis and computational equilibrium finding algorithms to solve instances.

Comparison with communication equilibria. As mentioned in Section 5.2.2, our construction simulating a mediator bears a resemblance to the construction used to define *communication*

equilibria [31, 67]. At a high level, a communication equilibrium of a game Γ is a Nash equilibrium of Γ augmented with a mediator playing according to some fixed strategy μ. Indeed, when team ∇ has only one player, it turns out that the two notions coincide:

Theorem 5.4.4. *Let Γ be a hidden-role game with $k = 1$. Then $\mathsf{CVal}_{\mathsf{priv}}(\Gamma)$ is exactly the value for Δ of the Δ-optimal communication equilibrium of Γ.*

However, in the more general case where ∇ can have more than one player, Theorem 5.4.4 does not apply: in that case, communication equilibria include all Nash equilibria in particular, and therefore fail to enforce *joint* optimality of the ∇-team, so our concepts and methods are more suitable. The proof is deferred to Section B.2.

5.4.2 Computing No/Public-Communication Equilibria

In this section, we consider games with no communication or with public communication and a coordinated minority. Conversely to the private-communication case of Section 5.4.1, in this case the problem of computing the value of a hidden-role equilibrium is, in general, NP-hard.

For the remainder of this section, when discussing the problem of "computing the value of a game", we always mean the following promise problem: given a game, a threshold v, and an allowable error $\varepsilon > 0$ (both expressed as rational numbers), decide whether the hidden-role value of Γ is $\geq v$ or $\leq v - \varepsilon$.

Theorem 5.4.5. *Even in 2-vs-1 games with public roles and $\varepsilon = 1/\mathsf{poly}(|H|)$, computing the TME value (and hence also the hidden-role value, since adversarial team games are a special case of hidden-role games) with public communication is NP-hard.*

Since there is only one ∇-player in the above reduction, the result applies regardless of whether the adversary is coordinated.

Theorem 5.4.6. *Even with a constant number of players, a minority adversary team, and $\varepsilon = 1/\mathsf{poly}(|H|)$, computing the uncoordinated value of a hidden-role game is coNP-hard with private communication and Σ_2^{P}-hard with public communication or no communication.*

Proofs of results in this section are deferred to Section B.3. Intuitively, the proofs work by constructing gadgets that prohibit any useful communication, thus reducing to the case of no communication.

Adversary Team Assumptions	Communication Type: None	Public	Private
Coordinated, Minority	NP-complete [90]	NP-hard [Thm. 5.4.3]	P [Thm. 5.4.3]
Coordinated		[Thm. 5.4.5]	*open problem*
Minority	Σ_2^{P}-complete [Thm. 5.4.6] and [99]	Σ_2^{P}-hard [Thm. 5.4.6]	coNP-hard [Thm. 5.4.6]
None			

Table 5.1: Complexity results for computing hidden-role value with a constant number of players, for various assumptions about the adversary team and notions of communication. The results shaded in green are new to our paper.

5.5 Worked Example

This section includes a worked example of value computation to illustrate the differences among the notions of equilibrium discussed in the paper and illustrates the utility of having a mediator for private communication. Consider a n-player version of matching pennies $\mathsf{MP}(n)$ as defined in Section 5.2.4.

Proposition 5.5.1. *Let $\mathsf{MP}(n)$ be the n-player matching pennies game.*

1. *The TMECor and TMECom values of* PUBLICTEAM($\mathsf{MP}(n)$) *are both* $1/2$.
2. *Without communication or with only public communication, the value of $\mathsf{MP}(n)$ is* $1/2^{n-1}$.
3. *With private communication, the value of $\mathsf{MP}(n)$ is* $1/(n+1)$.

Proof. The first claim, as well as the no-communication value, is known [4].

For the public-communication value, observe that, conditioned on the transcript, the bits chosen by the players must be mutually independent. Thus, the adversary can do the following: pretend to be on team Δ, wait for all communication to finish, and then play 0 if the string of all ones is more conditionally likely than the string of all 1s, and vice-versa[14].

Thus, only the third claim remains to be proven.

(*Lower bound*) The players simulate a mediator using multi-party computation (see Theorems 5.4.2 and 5.4.3). Consider the following strategy for the mediator. Sample a string

[14]In general, computing the conditional probabilities could take exponential time, but when defining the notion of value here, we are assuming that players have unbounded computational resources. This argument does not work for computationally-bounded adversaries. Indeed, if the adversary were computationally bounded, Δ could use cryptography to build private communication channels and thus implement a mediator, allowing our main positive result Theorem 5.4.3 to apply.

$b \in \{0,1\}^n$ uniformly at random from the set of $2n+2$ strings with at most one mismatched bit. Recommend to each player i to play b_i.

Consider the perspective of the adversary. The adversary sees only a recommended bit b_i. Assume WLOG that $b_i = 0$. Then there are $n+1$ possibilities:

1. b is all zeros (1 way)
2. All other bits of b are 1 (1 way)
3. Exactly one other bit of b is 1 ($n-1$ ways).

The adversary automatically wins in the third case (since the team has failed to coordinate), and regardless of what the adversary does, it can win only one of the first two cases. Thus, the adversary can win at most $n/(n+1)$ of the time; that is, this strategy achieves value $1/(n+1)$.

(*Upper bound*) Consider the following adversary strategy. The adversary communicates as it would if it were on team △. Let b_i be the bit the adversary would play if it were on team △. The adversary plays b_i with probability $1/(n+1)$ and $1-b_i$ otherwise. We need only show that no pure strategy of the mediator achieves value better than $1/(n+1)$ against this adversary. A strategy of the mediator is identified by a bitstring b. If b is all zeros or all ones, the team wins if and only if the adversary plays b_i (probability $1/(n+1)$). If b has a single mismatched bit, the team wins if and only if the mismatched bit is the adversary (probability $1/n$) and the adversary flips b_i (probability $n/(n+1)$). □

5.6 Properties of Hidden-role Equilibria

In the following, we discuss interesting properties of hidden-role equilibria given the definition we provided in Section 5.2.1, and that make them fairly unique relative to other notions of equilibrium in team games.

5.6.1 The Price of Hidden Roles

One interesting question arising from hidden-role games is the *price* of having them. That is, how much value does ▲ lose because roles are hidden? This section defines this quantity and derives reasonably tight bounds on it.

Definition 5.6.1. The *public-team refinement* of an n-player hidden-role game Γ is the adversarial team game $\textsc{PublicTeam}(\Gamma)$ defined by starting with the (uncoordinated) split-personality game, and adding the condition that all team assignments t_i are publicly observed by all players.

Definition 5.6.2. For a given hidden-role game Γ in which ▲ is guaranteed a nonnegative value (that is, $u_i(z) \geq 0$ whenever i is on △), the *price of hidden roles* $\mathsf{PoHR}(\Gamma)$ is the ratio between the TME value of $\textsc{PublicTeam}(\Gamma)$ and the hidden-role value of $\textsc{USplit}(\Gamma)$.

For a given class of hidden-role games $\mathcal{G}$, the price of hidden roles $\mathsf{PoHR}(\mathcal{G})$ is the supremum of the price of hidden roles across all games $\Gamma \in \mathcal{G}$.

Theorem 5.6.1. *Let $D \in \Delta(\{\triangle, \triangledown\}^n)$ be any distribution of team assignments. Let $\mathcal{G}_{n,D}$ be the class of all hidden-role games with n players and team assignment distribution D. Then the price of hidden roles of $\mathcal{G}_{n,D}$ is exactly the largest probability assigned to any team by D, that is,*

$$\mathsf{PoHR}(\mathcal{G}_{n,D}) = \max_{t \in \{\triangle,\triangledown\}^n} \Pr_{t' \sim D}[t' = t].$$

The lower bound is achieved even in the presence of private communication.

Proof. Let t^* be the team to which D assigns the highest probability, and let p^* be that probability. Our goal is to show that the price of hidden roles is $1/p^*$.

(*Upper bound*) ▲ assumes that the true △ is exactly the team t^*. Then ▲ gets utility at most a factor of $1/p^*$ worse than the TME value of $\textsc{PublicTeam}(\Gamma)$: if the assumption is correct, then ▲ gets the TME value; if the assumption is incorrect, ▲ gets value at least 0 thanks to the condition on ▲'s utilities in Definition 5.6.2.

(*Lower bound*) Consider the following game Γ. Nature first selects a team assignment $t \sim D$ and each player privately observes its team assignment. Then, all players are simultaneously asked to announce what they believe the true team assignment is. ▲ wins if every △-player announces the true team assignment. If ▲ wins, ▲ gets utility 1; otherwise ▲ gets utility 0.

Clearly, if teams are made public, ▲ wins easily. With teams not public, suppose that we add a mediator to the game so that Theorem 5.4.1 applies. This cannot decrease ▲'s value. The mediator's strategy amounts to selecting what team each player should announce. Mediator strategies in which different players announce different teams are dominated. The mediator strategy in which the mediator tells every player to announce team t wins if and only if t is the true team, which happens with probability at most p^* (if $t = t^*$). Thus, even the game with a mediator added has value at most p^*, completing the proof. □

This implies immediately:

Corollary 5.6.2. *Let $\mathcal{G}_{n,k}$ be the class of all hidden-role games where the number of players and adversaries are always exactly n and k, respectively. The price of hidden roles in $\mathcal{G}_{n,k}$ is exactly $\binom{n}{k}$.*

In particular, when $k = 1$, the price of hidden roles is at worst n. This is in sharp contrast to the *price of communication* and *price of correlation* in ATGs, both of which can be arbitrarily large even when $n = 3$ and $k = 1$ [4, 18].

5.6.2 Order of Commitment and Duality Gap

In Definition 5.3.5, when choosing the TME as our solution concept and defining the split-personality game, we explicitly choose that ▲ should pick its strategy before ▼—that is, the team committing to a strategy is the same one that has incomplete information about the roles. One may ask whether this choice is necessary or relevant: for example, what happens when the TME problem (5.1) satisfies the minimax theorem? Perhaps surprisingly, the answer to this question is that, at least with private communication, *the minimax theorem in hidden-role games only holds in "trivial" cases*, in particular, when the hidden-role game is equivalent to its public-role refinement (Definition 5.6.1).

Proposition 5.6.3. *Let Γ be any hidden-role game. Define $\mathsf{UVal}'_{\mathsf{priv}}(\Gamma)$ identically to $\mathsf{UVal}_{\mathsf{priv}}(\Gamma)$, except that ▼ commits before ▲—that is, in (5.1), the maximization and minimization are flipped. Then $\mathsf{UVal}'_{\mathsf{priv}}(\Gamma)$ is equal to the TME value of $\textsc{PublicTeam}(\Gamma)$ with communication—that is, the equilibrium value of the zero-sum game in which teams are public and intra-team communication is private and unlimited.*

Proof. It suffices to show that ▲ can always cause the teams to be revealed publicly if ▼ commits first. Let s be a long random string. All players on △ are coordinated by ▲ to broadcast s publicly at the start of the game. Since ▼ commits first, ▼ cannot know or guess s if it is sufficiently long; thus, with exponentially good probability, this completely reveals the teams publicly. Then, using the private communication channels, ▲ can play a TMECom of $\textsc{PublicTeam}(\Gamma)$. □

Therefore, the choice of having ▲ commit to a strategy before ▼ is forced upon us: flipping the order of commitment would ruin the point of hidden-role games.

Variant	5 Players	6 Players
No special roles (*Resistance*)	$3 / 10 = 0.3000$*	$1 / 3 \approx 0.3333$*
Merlin	$2 / 3 \approx 0.6667$*	$3 / 4 = 0.7500$*
Merlin & Mordred	$731 / 1782 \approx 0.4102$	$6543 / 12464 \approx 0.5250$
Merlin & 2 Mordreds	$5 / 18 \approx 0.2778$	$99 / 340 \approx 0.2912$
Merlin, Mordred, Percival, Morgana	$67 / 120 \approx 0.5583$	—

Table 5.2: Exact equilibrium values for 5- and 6-player *Avalon*. The values marked * were also manually derived by Christiano [21]; we match their results. '—': too large to solve.

5.7 Experimental Evaluation: *Avalon*

In this section, we apply Theorem 5.4.2 to instances of the popular hidden-role game *The Resistance: Avalon* (hereafter simply *Avalon*). We solve various versions of the game with up to six players.

A game of *Avalon* proceeds, generically speaking, as follows. There are n players, $\lceil n/3 \rceil$ of which are randomly assigned to ∇ and the rest to Δ. Team ∇ is informed. Some special rules allow players observe further information; for example, *Merlin* is a Δ-player who observes the identity of the players on ∇, except the ∇-player *Mordred*, and the Δ-player *Percival* knows *Merlin* and *Morgana* (who is on ∇), but does not know which is which. The game proceeds in five rounds. In each round, a *leader* publicly selects a certain number of people (defined as a function of the number of players and current round number) to go on a *mission*. Players then publicly vote on whether to accept the leader's choice. If a strict majority votes to accept, the mission begins; otherwise, leadership goes to the player to the left. If four votes pass with no mission selected, there is no vote on the fifth mission (it automatically gets accepted). If a ∇-player is sent on a mission, they can *fail* the mission. The goal of Δ is to have three missions pass. If *Merlin* is present, ∇ also wins by correctly guessing the identity of Merlin at the end of the game. *Avalon* is therefore parameterized by the number of players and the presence of the extra roles *Merlin, Mordred, Percival*, and *Morgana*.

Avalon is far too large to be written in memory: Serrino et al. [80] calculates that 5-player *Avalon* has at least 10^{56} information sets. However, in *Avalon* with ≤ 6 players, many simplifications can be made to the zero-sum game given by Theorem 5.4.2 without changing the equilibrium. These are detailed in Section B.4, but here we sketch one of them which has theoretical implications. Without loss of generality, in the zero-sum game in Theorem 5.4.2, the mediator completely dictates the choice of missions by telling everyone to propose the same mission and vote to accept missions, and $\blacktriangledown$ can do nothing to stop this. Therefore, ∇ always has symmetric information in the game: they know each others' roles (at least when

$n \leq 6$), and the mediator's recommendations to the players may as well be public. Therefore, *Avalon* is already natively, without loss of generality, a game with a coordinated adversary in the sense of Section 5.3.4, so the seemingly strong assumptions used in Definition 5.3.3 are in fact appropriate in *Avalon*. Even after our simplifications, the games are fairly large, *e.g.*, the largest instance we solve has 2.2 million infosets and 26 million terminal nodes.

Our results are summarized in Table 5.2. Games were solved using a CPU compute cluster machine with 64 CPUs and 480 GB RAM, using two algorithms:

1. A parallelized version of the PCFR+ algorithm [29], a scalable no-regret learning algorithm. PCFR+ was able to find an approximate equilibrium with exploitability $< 10^{-3}$ in less than 10 minutes in the largest game instance and complete 10,000 iterations in under two hours for each game.

2. An implementation of the simplex algorithm with exact (rational) precision, which was warm started using incrementally higher-precision solutions obtained from configurable finite-precision floating-point arithmetic implementation of the simplex algorithm, using an algorithm similar to that of Farina et al. [26]. This method incurred significantly higher runtimes (in the order of hours to tens of hours) but had the advantage of computing *exact* game values at equilibrium.

Table 5.2 shows exact game values for the instances we solved.

Findings We solve *Avalon* exactly in several instances with up to six players. In the simplest instances (*Resistance* or only Merlin), Christiano [21] previously computed equilibrium values by hand. The fact that we match those results is positive evidence of the soundness of both our equilibrium concepts and algorithms.

Curiously, as seen in Table 5.2, the game values are not "nice" fractions: this suggests that most of the equilibrium strategies will likely be inscrutable to humans. The simplest equilibrium not previously noted by Christiano [21], namely Merlin + 2 Mordreds with 5 players, is scrutable and is analyzed in detail in Section B.4.4.

Also, curiously, having Merlin and two Mordreds (that is, having a Merlin who does not actually know anything) is not the same as having no Merlin. If it were, we would expect the value of Merlin and two Mordreds to be $0.3 \times 2/3 = 0.2$ (due to the 1/3 probability of ▼ randomly guessing Merlin). But, the value is actually closer to 0.28. The discrepancy is due to the "special player" implicit correlation discussed in Section 5.2.4.

Chapter 6

Conclusion and Future Research

The contributions of this dissertation span theoretical advancements, novel algorithms, and practical insights on multi-agent decision-making under imperfect information. In particular, a novel perspective based on *learning* techniques is adopted for the computation of approximated equilibria in team games.

The main contribution concerning *adversarial team games* is an efficient and interpretable representation called TB-DAG of the optimization problem of finding a TMECor. This representation is based on the notion of grouping all the strategic reasoning of a correlating team in a single player which acts as a coordinator; equivalently, in the imperfect-recall equivalent game, the coordinator tracks all the perfect-recall information that players have while remembering her strategy. We provided both an intuitive construction of such a coordinator, accompanied by a fixed-parameter hardness result and an improved complexity analysis of computing a TMECor. The proposed construction enables the use of learning techniques already employed in the two-player zero-sum setting, leading to state-of-the-art results on most of the benchmark games commonly adopted in the literature.

The construction of a perfect-recall two-player zero-sum game equivalent to the original adversarial team game also enables techniques of *subgame solving*. We address several technical difficulties that do not arise in the two-player extensive-form setting and that instead threaten the efficiency of the *maxmargin* algorithm. The main of such challenges is that maxmargin's reach probabilities must be normalized, which requires to match the decision problems of the two teams according to the terminal nodes reached from a subgame. We circumvent the issue by redefining the maxmargin optimization problem in a way that is more suited to match those reach probabilities. Experimental evidence shows that applying any amount of subgame solving to a blueprint brings significant benefits.

Moreover, we initiated the formal study of hidden-role games from a rigorous game-theoretic perspective, by building on the previous literature on adversarial team games. We define different notions of equilibrium depending on communication capabilities and team structure, and give both positive and negative results surrounding the efficient computation of these equilibria. In experiments, we solve real-world instances of Avalon with private communication.

The connection between the contributions of this dissertation is the adoption of the same learning techniques already adopted in the two-player zero-sum case to achieve more efficient computation procedures in team games. This novel approach required to overcome challenges in the representation of team games and to adapt known algorithm to more general settings. The end results show strong empirical performance as shown on the various benchmark evaluations proposed.

We identify interesting lines of future research for the topics treated in this dissertation. In particular:

- Column generation and TB-DAG offer two different perspectives on solving the correlation problem among team members, each with complementary pros and cons. Combining the two techniques is a unique solution trading off the characteristics of the two is the interesting direction followed by [105], which achieves state-of-the-art performance on the whole spectrum of team games. *Can we design novel algorithmic solutions that scale to even larger instances, possibly targeting the game of Bridge?*

- There is a special class of games known as *triangle-free* games, in which a TMECor can be found in polynomial time [24]. At the moment, the TB-DAG construction returns a game that is worst-case exponentially bigger than the original game. *Is it possible to modify the TB-DAG construction in order to be efficient on triangle-free instances?*

- There is a lack of experimental evidence regarding what are good abstraction schemes in adversarial team games. Their development would enable to quickly approximate an optimal strategy which is then refined at play time. Similarly, there is a lack of a *depth limited* approximation schemes. *Can we design a complete subgame-limited refining pipeline similar to the one successful in Poker applications?*

- Similarly, the application of deep-learning algorithms for the computation of equilibria like ReBeL[13] is blocked by the lack of representations for beliefs. A similar question has been investigated in the Cooperative AI setting by [30, 82]. *Can we design effective and scalable representations for beliefs to be used in deep learning applications?*

- Many instances of the hidden-role solution concepts presented have computational complexity that is NP-hard. An example of this is the public communication case, whose computability would allow meaningful analysis in the many real-world scenarios falling under this category. *Can we find interesting, possibly specialized instances of hidden-role games where effective resolution techniques can be developed?*
- The evaluation of hidden-role equilibria has been performed on the recreative game Avalon. *Can we bring the computation of equilibria closer to real-world applications, for example in guiding the alignment procedures in LLMs?*

We strongly believe that any progress on the aforementioned challenges will help team games to grow over their current limitations, enabling impactful real-world applications.

The results presented on learning approaches in team games offer a solid foundation for further results. The experimental results show how this novel approach opens opportunities for efficient equilibrium computation. While the results are promising, they also highlight the complexity and richness of multi-agent decision-making, encouraging further exploration and innovation in this evolving field.

References

[1] Ittai Abraham, Danny Dolev, Rica Gonen, and Joe Halpern. Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation. In *ACM Symposium on Principles of Distributed Computing*, pages 53–62, 2006.

[2] Matthew Aitchison, Lyndon Benke, and Penny Sweetser. Learning to deceive in multi-agent hidden role games. In Stefan Sarkadi, Benjamin Wright, Peta Masters, and Peter McBurney, editors, *Deceptive AI*, pages 55–75. Springer International Publishing, 2021. ISBN 978-3-030-91779-1.

[3] Nicola Basilico, Nicola Gatti, and Francesco Amigoni. Patrolling security games: Definition and algorithms for solving large instances with single patroller and single intruder. *Artificial Intelligence*, 184-185:78–123, 2012. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2012.03.003. URL https://www.sciencedirect.com/science/article/pii/S0004370212000240.

[4] Nicola Basilico, Andrea Celli, Giuseppe De Nittis, and Nicola Gatti. Team-maxmin equilibrium: efficiency bounds and algorithms. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[5] Donald Beaver. Multiparty protocols tolerating half faulty processors. In *Advances in Cryptology—CRYPTO'89 Proceedings 9*, pages 560–572. Springer, 1990.

[6] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[7] Mark Braverman, Omid Etesami, and Elchanan Mossel. Mafia: A theoretical study of players and coalitions in a partial information environment. *Annals of Applied Probability*, 18:825–846, 2006.

[8] Noam Brown and Tuomas Sandholm. Safe and nested subgame solving for imperfect-information games. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

[9] Noam Brown and Tuomas Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2018.

[10] Noam Brown and Tuomas Sandholm. Solving imperfect-information games via discounted regret minimization. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

[11] Noam Brown and Tuomas Sandholm. Superhuman AI for multiplayer poker. *Science*, 365(6456):885–890, 2019.

[12] Noam Brown, Christian Kroer, and Tuomas Sandholm. Dynamic thresholding and pruning for regret minimization. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[13] Noam Brown, Anton Bakhtin, Adam Lerer, and Qucheng Gong. Combining deep reinforcement learning and search for imperfect-information games. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[14] Neil Burch, Michael Johanson, and Michael Bowling. Solving imperfect information games using decomposition. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2014.

[15] Luca Carminati, Federico Cacciamani, Marco Ciccone, and Nicola Gatti. Public information representation for adversarial team games. In *NeurIPS Cooperative AI Workshop*, 2021.

[16] Luca Carminati, Federico Cacciamani, Marco Ciccone, and Nicola Gatti. A marriage between adversarial team games and 2-player games: Enabling abstractions, no-regret learning, and subgame solving. In *International Conference on Machine Learning (ICML)*, pages 2638–2657. PMLR, 2022.

[17] Luca Carminati, Brian Hu Zhang, Gabriele Farina, Nicola Gatti, and Tuomas Sandholm. Hidden-role games: Equilibrium concepts and computation. *ACM Conference on Economics and Computation (EC)*, 2024. URL https://arxiv.org/pdf/2308.16017.

[18] Andrea Celli and Nicola Gatti. Computational results for extensive-form adversarial team games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[19] 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.

[20] Jianer Chen, Benny Chor, Mike Fellows, Xiuzhen Huang, David Juedes, Iyad A Kanj, and Ge Xia. Tight lower bounds for certain parameterized NP-hard problems. *Information and Computation*, 201(2):216–231, 2005.

[21] Paul F. Christiano. Solving avalon with whispering, 2018. Blog post. Available at https://sideways-view.com/2018/08/25/solving-avalon-with-whispering/.

[22] Francis Chu and Joseph Halpern. On the NP-completeness of finding an optimal strategy in games with common payoffs. *International Journal of Game Theory*, 2001.

[23] Constantinos Daskalakis, Paul Goldberg, and Christos Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM Journal on Computing*, 2008.

[24] Gabriele Farina and Tuomas Sandholm. Polynomial-time computation of optimal correlated equilibria in two-player extensive-form games with public chance moves and beyond. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[25] Gabriele Farina, Andrea Celli, Nicola Gatti, and Tuomas Sandholm. Ex ante coordination and collusion in zero-sum multi-player extensive-form games. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

[26] Gabriele Farina, Nicola Gatti, and Tuomas Sandholm. Practical exact algorithm for trembling-hand equilibrium refinements in games. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

[27] Gabriele Farina, Christian Kroer, and Tuomas Sandholm. Regret circuits: Composability of regret minimizers. In *International Conference on Machine Learning*, 2019.

[28] Gabriele Farina, Andrea Celli, Nicola Gatti, and Tuomas Sandholm. Connecting optimal ex-ante collusion in teams to extensive-form correlation: Faster algorithms and positive complexity results. In *International Conference on Machine Learning*, 2021.

[29] Gabriele Farina, Christian Kroer, and Tuomas Sandholm. Faster game solving via predictive Blackwell approachability: Connecting regret matching and mirror descent. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

[30] Jakob Foerster, Francis Song, Edward Hughes, Neil Burch, Iain Dunning, Shimon Whiteson, Matthew Botvinick, and Michael Bowling. Bayesian action decoder for deep multi-agent reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 2019.

[31] Francoise Forges. An approach to communication equilibria. *Econometrica: Journal of the Econometric Society*, pages 1375–1385, 1986.

[32] Oscar Garcia Morchon, Heribert Baldus, Tobias Heer, and Klaus Wehrle. Cooperative security in distributed sensor networks. In *2007 International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom 2007)*, pages 96–105, Nov 2007. doi: 10.1109/COLCOM.2007.4553817.

[33] Oscar Garcia-Morchon, Dmitriy Kuptsov, Andrei Gurtov, and Klaus Wehrle. Cooperative security in distributed networks. *Computer Communications*, 36(12):1284–1297, 2013. ISSN 0140-3664. doi: https://doi.org/10.1016/j.comcom.2013.04.007. URL https://www.sciencedirect.com/science/article/pii/S0140366413001072.

[34] Hugo Gimbert, Soumyajit Paul, and B. Srivathsan. A bridge between polynomial optimization and games with imperfect recall. In *Autonomous Agents and Multi-Agent Systems*, 2020.

[35] M. Grotschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimizations*. Springer-Verlag, 1993.

[36] John Harsanyi. Game with incomplete information played by Bayesian players. *Management Science*, 14:159–182; 320–334; 486–502, 1967–68.

[37] Sergiu Hart and Andreu Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68:1127–1150, 2000.

[38] Johan Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48 (4):798–859, 2001.

[39] Evan Hubinger, Carson Denison, Jesse Mu, Mike Lambert, Meg Tong, Monte MacDiarmid, Tamera Lanham, Daniel M. Ziegler, Tim Maxwell, Newton Cheng, Adam Jermyn, Amanda Askell, Ansh Radhakrishnan, Cem Anil, David Duvenaud, Deep Ganguli, Fazl Barez, Jack Clark, Kamal Ndousse, Kshitij Sachan, Michael Sellitto, Mrinank Sharma, Nova DasSarma, Roger Grosse, Shauna Kravec, Yuntao Bai, Zachary Witten, Marina Favaro, Jan Brauner, Holden Karnofsky, Paul Christiano, Samuel R. Bowman, Logan Graham, Jared Kaplan, Sören Mindermann, Ryan Greenblatt, Buck Shlegeris, Nicholas Schiefer, and Ethan Perez. Sleeper agents: Training deceptive llms that persist through safety training. *arXiv preprint arXiv: 2401.05566*, 2024.

[40] Geoffrey Irving, Paul F. Christiano, and Dario Amodei. AI safety via debate. *CoRR*, abs/1805.00899, 2018. URL http://arxiv.org/abs/1805.00899.

[41] Sergei Izmalkov, Silvio Micali, and Matt Lepinski. Rational secure computation and ideal mechanism design. In *Symposium on Foundations of Computer Science (FOCS)*, pages 585–595, Washington, DC, 2005.

[42] Jiaming Ji, Tianyi Qiu, Boyuan Chen, Borong Zhang, Hantao Lou, Kaile Wang, Yawen Duan, Zhonghao He, Jiayi Zhou, Zhaowei Zhang, Fanzhi Zeng, Kwan Yee Ng, Juntao Dai, Xuehai Pan, Aidan O'Gara, Yingshan Lei, Hua Xu, Brian Tse, Jie Fu, Stephen McAleer, Yaodong Yang, Yizhou Wang, Song-Chun Zhu, Yike Guo, and Wen Gao. AI alignment: A comprehensive survey. *CoRR*, abs/2310.19852, 2023. doi: 10.48550/ARXIV.2310.19852. URL https://doi.org/10.48550/arXiv.2310.19852.

[43] Mamoru Kaneko and J Jude Kline. Behavior strategies, mixed strategies and perfect recall. *International Journal of Game Theory*, 24(2):127–145, 1995.

[44] Jonathan Katz. Bridging game theory and cryptography: Recent results and future directions. In *Theory of Cryptography: Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008. Proceedings 5*, pages 251–272. Springer, 2008.

[45] 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.

[46] Daphne Koller, Nimrod Megiddo, and Bernhard von Stengel. Fast algorithms for finding randomized strategies in game trees. In *ACM Symposium on Theory of Computing (STOC)*, 1994.

[47] Kavya Kopparapu, Edgar A. Duéñez-Guzmán, Jayd Matyas, Alexander Sasha Vezhnevets, John P. Agapiou, Kevin R. McKee, Richard Everett, Janusz Marecki, Joel Z. Leibo, and Thore Graepel. Hidden agenda: a social deduction game with diverse learned equilibria. *CoRR*, abs/2201.01816, 2022. URL https://arxiv.org/abs/2201.01816.

[48] Vojtěch Kovařík, Martin Schmid, Neil Burch, Michael Bowling, and Viliam Lisỳ. Rethinking formal models of partially observable multiagent decision making. *Artificial Intelligence*, 303:1036–45, 2022.

[49] Mark W Krentel. The complexity of optimization problems. *Journal of computer and system sciences*, 36(3):490–509, 1988.

[50] Christian Kroer and Tuomas Sandholm. Imperfect-recall abstractions with bounds in games. In *ACM Conference on Economics and Computation (EC)*, 2016.

[51] H. W. Kuhn. Extensive games. *Proc. of the National Academy of Sciences*, 36:570–576, 1950.

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

[53] H. W. Kuhn. Extensive games and the problem of information. In H. W. Kuhn and A. W. Tucker, editors, *Contributions to the Theory of Games*, volume 2 of *Annals of Mathematics Studies, 28*, pages 193–216. Princeton University Press, 1953.

[54] 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.

[55] Mathieu Laurière, Sarah Perrin, Julien Pérolat, Sertan Girgin, Paul Muller, Romuald Élie, Matthieu Geist, and Olivier Pietquin. Learning in mean field games: A survey, 2024. URL https://arxiv.org/abs/2205.12944.

[56] Chang Lei and Huan Lei. Alphadou: High-performance end-to-end doudizhu ai integrating bidding, 2024. URL https://arxiv.org/abs/2407.10279.

[57] Yehuda Lindell. Secure multiparty computation (mpc). *Cryptology ePrint Archive*, 2020.

[58] Viliam Lisỳ, Marc Lanctot, and Michael H Bowling. Online monte carlo counterfactual regret minimization for search in imperfect information games. In *Autonomous Agents and Multi-Agent Systems*, 2015.

[59] Heng Liu. Correlation and unmediated cheap talk in repeated games with imperfect monitoring. *International Journal of Game Theory*, 46:1037–1069, 2017.

[60] Amol Mandhane, Anton Zhernov, Maribeth Rauh, Chenjie Gu, Miaosen Wang, Flora Xue, Wendy Shang, Derek Pang, Rene Claus, Ching-Han Chiang, Cheng Chen, Jingning Han, Angie Chen, Daniel J. Mankowitz, Jackson Broshear, Julian Schrittwieser, Thomas Hubert, Oriol Vinyals, and Timothy Mann. Muzero with self-competition for rate control in vp9 video compression, 2022. URL https://arxiv.org/abs/2202.06626.

[61] Michael Maschler, Shmuel Zamir, and Eilon Solan. *Game Theory*. Cambridge University Press, 2020.

[62] H. B. McMahan, Geoffrey J. Gordon, and Avrim Blum. Planning in the presence of cost functions controlled by an adversary. In *International Conference on Machine Learning*, 2003.

[63] James D. Miller, Roman Yampolskiy, Olle Häggström, and Stuart Armstrong. Chess as a testing grounds for the oracle approach to AI safety. In Huáscar Espinoza, John A. McDermid, Xiaowei Huang, Mauricio Castillo-Effen, Xin Cynthia Chen, José Hernández-Orallo, Seán Ó hÉigeartaigh, Richard Mallah, and Gabriel Pedroza, editors, *Proceedings*

of the Workshop on Artificial Intelligence Safety 2021 co-located with the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI 2021), Virtual, August, 2021, volume 2916 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2021. URL https://ceur-ws.org/Vol-2916/paper_13.pdf.

[64] Matej Moravcik, Martin Schmid, Karel Ha, Milan Hladik, and Stephen Gaukrodger. Refining subgames in large imperfect information games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2016.

[65] 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*, May 2017.

[66] Viraaji Mothukuri, Reza M Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. A survey on security and privacy of federated learning. *Future Generation Computer Systems*, 115:619–640, 2021.

[67] Roger B Myerson. Multistage games with communication. *Econometrica: Journal of the Econometric Society*, pages 323–358, 1986.

[68] John Nash. Equilibrium points in n-person games. *National Academy of Sciences*, 36: 48–49, 1950.

[69] Ashutosh Nayyar, Aditya Mahajan, and Demosthenis Teneketzis. Decentralized stochastic control with partial history sharing: A common information approach. *IEEE Transactions on Automatic Control*, 58(7):1644–1658, 2013.

[70] N. Nisan, T. Roughgarden, E. Tardos, and V.V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, 2007. ISBN 9780521872829. URL https://books.google.it/books?id=f4SFngEACAAJ.

[71] Aidan O'Gara. Hoodwinked: Deception and cooperation in a text-based game for language models. *CoRR*, abs/2308.01404, 2023. doi: 10.48550/ARXIV.2308.01404. URL https://doi.org/10.48550/arXiv.2308.01404.

[72] Peter S. Park, Simon Goldstein, Aidan O'Gara, Michael Chen, and Dan Hendrycks. AI deception: A survey of examples, risks, and potential solutions. *CoRR*, abs/2308.14752, 2023. doi: 10.48550/ARXIV.2308.14752. URL https://doi.org/10.48550/arXiv.2308.14752.

[73] Julien Perolat, Bart De Vylder, Daniel Hennes, Eugene Tarassov, Florian Strub, Vincent de Boer, Paul Muller, Jerome T. Connor, Neil Burch, Thomas Anthony, Stephen McAleer, Romuald Elie, Sarah H. Cen, Zhe Wang, Audrunas Gruslys, Aleksandra Malysheva, Mina Khan, Sherjil Ozair, Finbarr Timbers, Toby Pohlen, Tom Eccles, Mark Rowland, Marc Lanctot, Jean-Baptiste Lespiau, Bilal Piot, Shayegan Omidshafiei, Edward Lockhart, Laurent Sifre, Nathalie Beauguerlange, Remi Munos, David Silver, Satinder Singh, Demis Hassabis, and Karl Tuyls. Mastering the game of stratego with model-free multiagent reinforcement learning. *Science*, 378(6623):990–996, 2022.

[74] Tal Rabin. Robust sharing of secrets when the dealer is honest or cheating. *Journal of the ACM (JACM)*, 41(6):1089–1109, 1994.

[75] Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 73–85, 1989.

[76] I. Romanovskii. Reduction of a game with complete memory to a matrix game. *Soviet Mathematics*, 3, 1962.

[77] Marcus Schaefer and Christopher Umans. Completeness in the polynomial-time hierarchy: A compendium. *SIGACT News*, 33(3):32–49, 2002.

[78] Jérémy Scheurer, Mikita Balesni, and Marius Hobbhahn. Technical report: Large language models can strategically deceive their users when put under pressure. *CoRR*, abs/2311.07590, 2023. doi: 10.48550/ARXIV.2311.07590. URL https://doi.org/10.48550/arXiv.2311.07590.

[79] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy P. Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *CoRR*, abs/1911.08265, 2019. URL http://arxiv.org/abs/1911.08265.

[80] Jack Serrino, Max Kleiman-Weiner, David C Parkes, and Josh Tenenbaum. Finding friend and foe in multi-agent games. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[81] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484, 2016.

[82] Samuel Sokota, Edward Lockhart, Finbarr Timbers, Elnaz Davoodi, Ryan D'Orazio, Neil Burch, Martin Schmid, Michael Bowling, and Marc Lanctot. Solving common-payoff games with approximate policy iteration. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, pages 9695–9703, 2021.

[83] Finnegan Southey, Michael Bowling, Bryce Larson, Carmelo Piccione, Neil Burch, Darse Billings, and Chris Rayner. Bayes' bluff: Opponent modelling in poker. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2005.

[84] Emanuel Tewolde, Caspar Oesterheld, Vincent Conitzer, and Paul W. Goldberg. The computational complexity of single-player imperfect-recall games. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2023.

[85] Dipty Tripathi, Amit Biswas, Anil Kumar Tripathi, Lalit Kumar Singh, and Amrita Chaturvedi. An integrated approach of designing functionality with security for distributed cyber-physical systems. *J. Supercomput.*, 78(13):14813–14845, sep 2022. ISSN 0920-8542. doi: 10.1007/s11227-022-04481-9. URL https://doi.org/10.1007/s11227-022-04481-9.

[86] Amparo Urbano and Jose E Vila. Computational complexity and communication: Coordination in two–player games. *Econometrica*, 70(5):1893–1927, 2002.

[87] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

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

[89] Bernhard von Stengel and Françoise Forges. Extensive-form correlated equilibrium: Definition and computational complexity. *Mathematics of Operations Research*, 33(4): 1002–1022, 2008.

[90] Bernhard von Stengel and Daphne Koller. Team-maxmin equilibria. *Games and Economic Behavior*, 21(1):309–321, 1997.

[91] Martin J Wainwright and Michael Irwin Jordan. *Graphical models, exponential families, and variational inference*. Now Publishers Inc, 2008.

[92] Kevin Waugh. Abstraction in large extensive games. Master's thesis, University of Alberta, 2009.

[93] Zelai Xu, Chao Yu, Fei Fang, Yu Wang, and Yi Wu. Language agents with reinforcement learning for strategic play in the werewolf game. *CoRR*, abs/2310.18940, 2023. doi: 10.48550/ARXIV.2310.18940. URL https://doi.org/10.48550/arXiv.2310.18940.

[94] Brian H. Zhang, Luca Carminati, Federico Cacciamani, Gabriele Farina, Pierriccardo Olivieri, Nicola Gatti, and Tuomas Sandholm. Subgame solving in adversarial team games. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[95] Brian Hu Zhang and Tuomas Sandholm. Polynomial-time optimal equilibria with a mediator in extensive-form games. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[96] Brian Hu Zhang and Tuomas Sandholm. Team correlated equilibria in zero-sum extensive-form games via tree decompositions. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2022.

[97] Brian Hu Zhang, Gabriele Farina, Andrea Celli, and Tuomas Sandholm. Optimal correlated equilibria in general-sum extensive-form games: Fixed-parameter algorithms, hardness, and two-sided column-generation. In *ACM Conference on Economics and Computation (EC)*, 2022.

[98] Brian Hu Zhang, Gabriele Farina, Ioannis Anagnostides, Federico Cacciamani, Stephen Marcus McAleer, Andreas Alexander Haupt, Andrea Celli, Nicola Gatti, Vincent Conitzer, and Tuomas Sandholm. Computing optimal equilibria and mechanisms via learning in zero-sum extensive-form games. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

[99] Brian Hu Zhang, Gabriele Farina, and Tuomas Sandholm. Team belief dag: Generalizing the sequence form to team games for fast computation of correlated team max-min equilibria via regret minimization. In *International Conference on Machine Learning (ICML)*, 2023.

[100] Brian Hu Zhang, Ioannis Anagnostides, Gabriele Farina, and Tuomas Sandholm. Efficient Φ-regret minimization with low-degree swap deviations in extensive-form games, 2024. arXiv.

[101] Brian Hu Zhang, Gabriele Farina, and Tuomas Sandholm. Mediator interpretation and faster learning algorithms for linear correlated equilibria in general extensive-form games. In *International Conference on Learning Representations*, 2024.

[102] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of reinforcement learning and control*, pages 321–384, 2021.

[103] Youzhi Zhang and Bo An. Converging to team-maxmin equilibria in zero-sum multiplayer games. In *International Conference on Machine Learning (ICML)*, 2020.

[104] Youzhi Zhang, Bo An, and Jakub Černỳ. Computing ex ante coordinated team-maxmin equilibria in zero-sum multiplayer extensive-form games. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

[105] Youzhi Zhang, Bo An, and Daniel Dajun Zeng. DAG-based column generation for adversarial team games. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 58563–58578. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/zhang24b.html.

[106] Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul F. Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *CoRR*, abs/1909.08593, 2019. URL http://arxiv.org/abs/1909.08593.

[107] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *International Conference on Machine Learning (ICML)*, 2003.

[108] Martin Zinkevich, Michael Bowling, Michael Johanson, and Carmelo Piccione. Regret minimization in games with incomplete information. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2007.

Appendix A

Additional Results from Chapter 4

A.1 Other Subgame Solving Techniques

A.1.1 Gifts

Considering counterfactual best response values as the maximum counterfactual values allowed for the opponent in any of his root infosets is a sufficient condition to guarantee that the exploitability of the refined strategy will not be higher than that of the blueprint.

In [8], a relaxation of this condition is proposed. For each action a of the opponent in an infoset in the trunk, the *gift* is defined as the difference between the counterfactual best response value of the best action and the one of a. If an action has a positive gift, that means that the action is suboptimal. Therefore, the refining player can increase the counterfactual best response value of any action having a positive gift without incurring any exploitability increase. In reach-maxmargin [8], the sum of gifts over actions played before reaching an infoset is used to increase the associated counterfactual best response values.

Brown and Sandholm [8] originally proposed two definitions of gifts:

- a *lower bound* gift definition which considers differences in cbv between the action on the path and sibling actions immediately leading to a terminal. This avoids the overestimation of gifts due to considering cbv changing due to further refinements happening in other subgames;
- a *safe* gift definition, which maximizes the magnitude of gifts while preserving the guarantee of safety;

While the lower bound definition has wider guarantees of reducing exploitability, the safe definition is more performant empirically [8]. We, therefore, pick the safe definition for our exposition.

Definition A.1.1 (Gifts)**.** The gift associated to an action $\boldsymbol{a}$ played in a belief C is defined as:

$$g^{\boldsymbol{x}}_{\blacktriangledown}(C\boldsymbol{a}) \coloneqq (u^{\boldsymbol{x}}_{\blacktriangledown}(C) - u^{\boldsymbol{x}}_{\blacktriangledown}(C\boldsymbol{a}))\,\frac{1}{\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)}.$$

The updated team-maxmargin objective can be formulated as:

$$\max_{\boldsymbol{x}'}\min_{\boldsymbol{y},\bar{\boldsymbol{y}}}\sum_{z\succeq P}\boldsymbol{x}'[z]\,\boldsymbol{y}[z]\,\boldsymbol{p}[z]\,u[z] - \sum_{C\in\mathcal{C}^{\boldsymbol{x}}(P)}\boldsymbol{y}[C]\left(u^{\boldsymbol{x}}_{\blacktriangledown}(C) + \rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)\min_{\pi\text{ path }\varnothing\to C}\sum_{C'\boldsymbol{a}'\in\pi} g^{\boldsymbol{x}}_{\blacktriangledown}(C'\boldsymbol{a}')\right).$$

The minimization over paths $\varnothing \to C$ is taken over $\blacktriangledown$'s DAG, and deals with the possibility of multiple ways of reaching the same belief. Such a minimization can be computed efficiently (in the size of $\blacktriangledown$'s DAG) via dynamic programming.

Preliminary experiments showed that combining gifts with the column-generation procedure from Section 4.3 showed no improvement over team-maxmargin without gifts. This is in line with the experiments in Brown and Sandholm [8, see Maxmargin vs Reach-Maxmargin], where the application of gifts brought a minimal improvement.

A.1.2 Resolving

Similarly to maxmargin, *resolving* [14] can be formulated for team games as well by relaxing team maxmargin from Definition 4.2.4. In particular, we consider any strategy achieving a positive margin for every $\blacktriangledown$ belief.

Definition A.1.2 (Resolving linear program)**.** Resolving optimization problem in a TB-DAG subgame rooted at P can be expressed as the following linear program.

$$
\begin{aligned}
&\max_{\boldsymbol{x}'} 0 \\
&\text{s.t.} \quad \min_{\boldsymbol{y},\bar{\boldsymbol{y}}} \sum_{z \succeq P} \boldsymbol{x}'[z]\, \boldsymbol{y}[z]\, \boldsymbol{p}[z]\, u(z) - \sum_{C \in \mathcal{C}^{\boldsymbol{x}}(P)} \bar{\boldsymbol{y}}[C]\, \frac{u^{\boldsymbol{x}}_{\blacktriangledown}(C)}{\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)} \geq 0 \\
&\qquad \boldsymbol{x}'[B] = \boldsymbol{x}[B] \qquad \text{for all } \blacktriangle\text{-beliefs } B \in \mathcal{B}[P] \\
&\qquad \boldsymbol{y}[C] = \frac{\bar{\boldsymbol{y}}[C]}{\rho^{\boldsymbol{x}}_{\mathsf{N},\blacktriangle}(C)} \qquad \text{for all } \blacktriangledown\text{-beliefs } C \in \mathcal{C}^{\boldsymbol{x}}(P) \\
&\qquad \boldsymbol{x}' \in \mathcal{X}_P,\ \boldsymbol{y} \in \mathcal{Y}_P,\ \bar{\boldsymbol{y}} \in \Delta^{\mathcal{C}^{\boldsymbol{x}}(P)}
\end{aligned}
$$

Appendix B

Additional Results from Chapter 5

B.1 Multi-Party Computation and Proof of Theorem 5.4.3

We first formalize the usual setting of multi-party computation (MPC). Let X be the set of binary strings of length ℓ, and let λ be a security parameter. Rabin and Ben-Or [75] claims in their Theorem 4 that essentially any protocol involving a mediator can be efficiently simulated without a mediator so long as more than half the players follow the protocol and we allow some exponentially small error. However, they do not include a proof of this result. In the interest of completeness, we prove the version of their result that is needed for our setting, based only on the primitives of *secure multi-party computation* and *verifiable secret sharing*.

B.1.1 Secure MPC

In *secure MPC*, there is a (possibly randomized) function $f : X^n \to X^n$ defined by a circuit with N nodes. A subset $K \subset [n]$ of size $< n/2$ has been *corrupted*. Each *honest* player $i \in [n] \setminus K$ holds an input $x_i \in X$. The goal is to design a randomized messaging protocol, with *private* communication, such that, regardless of what the corrupted players do, there exist inputs $\{x_j : j \in K\}$ such that:

1. (Output delivery) At the end of the protocol, each player i learns its own output, $y_i := f(x_1, \ldots, x_n)_i$, with probability $1 - 2^{-\lambda}$.

2. (Privacy) No subset of $< n/2$ players can learn any information except their own output y_i. That is, the players cannot infer any extra information from analyzing the transcripts of the message protocol than what they already know.

This can be intuitively modeled as follows: if any minority of colluded players were to analyze the empirical distributions of protocol transcripts for any input-output tuple, then such distribution would be fully explained in terms of their input-outputs, leaving no conditional dependence on other players' input-outputs. Formally, for any such subset $K \subseteq [n]$ of size $< n/2$, there exists a randomized algorithm Sim_K that takes the inputs and outputs of the players in set K, and reconstructs transcripts T, such that for all $x \in X^n$, we have

$$\sum_T |\Pr[\mathsf{Sim}_K(x_K, y_K) = T] - \Pr[\mathsf{View}_K(x) = T]| \leq 2^{-\lambda}$$

where $\mathsf{View}_K(x)$ is the distribution of transcripts observed by players in set K when running the protocol with input x.

In other words, the players in set K cannot do anything except pass to f inputs of their choice. For our specific application, this implies that introducing an MPC protocol to simulate the mediator is equivalent to having a mediator because no extra information aside from the intended function will be leaked to the players.

Theorem B.1.1 ([5, 75]). *Secure MPC is possible with polytime (in ℓ, λ, n, N) algorithms that take at most polynomially many rounds and send at most polynomially many bits in each round.*

B.1.2 Verifiable Secret Sharing

In the *verifiable secret sharing* (VSS) problem, the goal is to design a function $\mathsf{Share} : X \to \bar{X}^n$, where $\bar{X} = \{0,1\}^{\mathsf{poly}(\ell,\lambda,n)}$, that *shares* a secret $x \in X$ by privately informing each player i of its piece $\mathsf{Share}(x)_i$, such that:

1. (Reconstructibility) Any subset of $> n/2$ players can recover the secret fully, even if the remaining players are adversarial. That is, there exists a function $\mathsf{Reconstruct} : X^n \to X$ such that, where $(x_1, \ldots, x_n) \leftarrow \mathsf{Share}(x)$, we have $\mathsf{Reconstruct}(x'_1, \ldots, x'_n) = x$ so long as $x'_i = x_i$ for $> n/2$ players i.

2. (Privacy) No subset of $< n/2$ players can learn any information about the secret x. That is, for any such subset K and any secret x, there exists a distribution $\mathsf{Sim}_K \in \Delta(X^n)$ such that

$$\|\mathsf{Share}(x)_K - \mathsf{Sim}_K\|_1 \leq 2^{-\lambda},$$

where $\|\cdot\|_1$ denotes the ℓ_1-norm on probability distributions.

Theorem B.1.2 ([74, 75])**.** *There exist algorithms* Share *and* Reconstruct*, with runtime* $\mathsf{poly}(\ell, \lambda, n)$*, that implement robust secret sharing.*

VSS is a primitive used in a fundamental way to build secure MPC protocols; to be formally precise in this paper, we will require VSS as a separate primitive as well to maintain the state of the mediator throughout the game.

B.1.3 Simulating a Mediator

We will assume that the game Γ_0 in Theorem 5.4.2 has been solved, and that its solution is given by a (possibly randomized) function

$$f : \Sigma \times O^n \to \Sigma \times A^n \tag{B.1}$$

where Σ is the set of information states of the mediator in Γ_0. At each step, the mediator takes n received infosets $I_1, \ldots, I_n$ and its current sequence σ as input, and updates its sequence and outputs action recommendations according to the function f.

Consider the function $\hat{f} : (\bar{X} \times \mathcal{I})^n \to (\bar{X} \times A)^n$ defined by

$$\hat{f}((x_1, o_1), \ldots, (x_n, o_n)) = ((x'_1, a_1), \ldots, (x'_n, a_n))$$

where

$$(s', a_1, \ldots, a_n) = f(\mathsf{Reconstruct}(x_1, \ldots, x_n), o_1, \ldots, o_n)$$

$$(x'_1, \ldots, x'_n) = \mathsf{Share}(s').$$

That is, $\hat{f}$ operates the mediator with its state secret-shared across the various players. The players will run secure MPC on $\hat{f}$ at every timestep. By the properties of MPC and secret sharing, this securely implements the mediator in such a way that the players on ∇ can neither break privacy nor cause the protocol to fail, with probability better than $O(|\mathcal{H}| \cdot 2^{-\lambda})$, where $\mathcal{H}$ is the set of nodes of the game.

We have thus shown our main theorem, which is more formally stated as follows:

Theorem B.1.3 (Formal version of Theorem 5.4.3)**.** *Let:*

- Γ *be a hidden-role game with a* ∇ *of size* $k < n/2$,
- Γ^* *be identical to* Γ *except that there is an additional player who takes no nontrivial actions but is always on* Δ;
- Γ_0 *be the zero-sum game defined by Theorem 5.4.2 based on* Γ^*;
- $\boldsymbol{x}$ *be any strategy of* Δ*'s mediator player in* Γ_0, *represented by an arithmetic circuit* f *as in* (B.1) *with* $N = \mathsf{poly}(|\mathcal{H}|^k)$ *gates; and*
- λ *be a security parameter.*

Then there exists a strategy profile $\boldsymbol{x}'$ *of* $\blacktriangle$ *in* $\textsc{CSplit}(\textsc{Comm}^{M,R}_{\mathsf{priv}}(\Gamma))$, *where* $\log M = R = \mathsf{poly}(\lambda, N, \log|\mathcal{H}|)$ *such that:*

1. (Equivalence of value) *the value of* $\boldsymbol{x}'$ *is within* $2^{-\lambda}$ *of the value of* x *in* Γ_0, *and*
2. (Efficient execution) *there is a* $\mathsf{poly}(r)$*-time randomized algorithm* $\mathcal{A}_\Gamma$ *that takes as input an information set* I_i *of* $\textsc{CSplit}(\textsc{Comm}^{M,R}_{\mathsf{priv}}(\Gamma))$ *and returns the (possibly random) action that player* i *should play at* I_i.

B.2 Connection to Communication Equilibria

In this section, we prove Theorem 5.4.4, recalled below.

Theorem 5.4.4. *Let* Γ *be a hidden-role game with* $k = 1$. *Then* $\mathsf{CVal}_{\mathsf{priv}}(\Gamma)$ *is exactly the value for* Δ *of the* Δ*-optimal communication equilibrium of* Γ.

Before proving this result, we must first formally define a communication equilibrium. Given an arbitrary game Γ with n players, consider the $(n+1)$-player game in which the extra player is the mediator. Consider a private-communication extension $\tilde{\Gamma}$ of that game, with $R = 2$ rounds and $M = |\mathcal{H}|$, in which only communication with the mediator is allowed. In Γ^*, each player has a *direct strategy* x_i^* in which, at every timestep, the player sends its honest information to the mediator, interprets the mediator's message in reply as an action recommendation, and plays that action recommendation.

Definition B.2.1. A *communication equilibrium* of Γ is a strategy profile $\mu = (x_1, \ldots, x_n)$ for the mediator of $\tilde{\Gamma}$ such that, with μ held fixed, the profile $(x_1, \ldots, x_n)$ is a Nash equilibrium of the resulting n-player game.

The *revelation principle for communication equilibria* [31, 67] states that, without loss of generality in the above definition, it can be assumed that $\boldsymbol{x} = \boldsymbol{x}^*$, that is, players are direct in equilibrium.

We now prove the theorem. Let Γ^* be Γ with a mediator who is always on $\triangle$, and let Γ_0 be the zero-sum game constructed by Theorem 5.4.2. Because $\mathsf{CVal}_{\mathsf{priv}}(\Gamma)$ is the zero-sum value of Γ_0 by Theorem 5.4.3, it is enough to prove the inequality chain

$$\mathsf{CVal}_{\mathsf{priv}}(\Gamma) \leq \mathsf{CommVal}(\Gamma) \leq \mathsf{Val}(\Gamma_0),$$

where $\mathsf{CommVal}$ is the value of the $\blacktriangle$-optimal communication equilibrium and Val is the zero-sum game value.

By Theorem 5.4.3, the hidden-role equilibria of Γ are (up to an arbitrarily small error $\varepsilon > 0$) TMEs of $\textsc{CSplit}(\Gamma^*)$. By von Stengel and Koller [90], since $\blacktriangledown$ has only one player, the TMEs of $\textsc{CSplit}(\Gamma^*)$ are precisely the $\blacktriangle$-team-optimal Nash equilibria of $\textsc{CSplit}(\Gamma^*)$. Thus, for a TME $(\mu, x_1, \ldots, x_n)$ of that game (where μ is a strategy of the mediator), there exists an adversary strategy y such that $(\mu, x_1, \ldots, x_n, y)$ is a Nash equilibrium. But then $(\mu, x_1, \ldots, x_n, y)$ is also a communication equilibrium. Thus $\mathsf{TMEVal}(\textsc{CSplit}(\Gamma^*)) \leq \mathsf{CommVal}(\Gamma)$.

We now show the second inequality. If Γ is an adversarial hidden-role game, each player i's strategy can be expressed as a tuple (x_i, y_i) where x_i is player i's strategy in the subtree where i is on $\triangle$, and y_i is the same on $\triangledown$. In that case, the problem of finding a $\blacktriangle$-optimal communication equilibrium can be expressed as:

$$\begin{aligned} \max_{\mu} \quad & u(\mu, \boldsymbol{x}^*, \boldsymbol{y}^*) \\ \text{s.t.} \quad & \forall i \quad \max_{x_i} u(\mu, x_i, \boldsymbol{x}^*_{-i}, \boldsymbol{y}^*) \leq u(\mu, \boldsymbol{x}^*, \boldsymbol{y}^*) \\ & \forall i \quad \min_{y_i} u(\mu, \boldsymbol{x}^*, y_i, \boldsymbol{y}^*_{-i}) \geq u(\mu, \boldsymbol{x}^*, \boldsymbol{y}^*) \end{aligned}$$

where, with an abuse of notation, u is $\blacktriangle$'s expected utility function, formally defined as $u(\mu, \boldsymbol{x}, \boldsymbol{y}) := \sum_{z \in \mathcal{Z}} \mu[z]\boldsymbol{x}[z]\boldsymbol{y}[z]u(z)$. That is, no player can increase their team's utility by deviating from the direct profile $(\boldsymbol{x}^*, \boldsymbol{y}^*)$, regardless of which team they are assigned to. Now, using the fact that $\blacktriangledown$ has only one player, we can write $\blacktriangle$'s utility function u as a sum $u = \sum_i u_i$ where u_i is the utility of $\blacktriangle$ when the $\triangledown$-player is player i (weighted by the probability of that happening). Each term u_i depends only on $\boldsymbol{x}$ and the y_is. Thus, the above program can be

rewritten as

$$\begin{aligned} \max_{\mu} \quad & \sum_i u_i(\mu, \boldsymbol{x}^*, y_i^*) \\ \text{s.t.} \quad \forall i \quad & \max_{x_i} u(\mu, x_i, \boldsymbol{x}^*_{-i}, y^*) \leq u(\mu, \boldsymbol{x}^*, \boldsymbol{y}^*) \\ \forall i \quad & \min_{y_i} u_i(\mu, \boldsymbol{x}^*, y_i) \geq u_i(\mu, \boldsymbol{x}^*, y_i^*) \end{aligned}$$

or, equivalently,

$$\begin{aligned} \max_{\mu} \quad & \sum_i \min_{y_i} u_i(\mu, \boldsymbol{x}^*, y_i) \\ \text{s.t.} \quad \forall i \quad & \max_{x_i} u(\mu, x_i, \boldsymbol{x}^*_{-i}, \boldsymbol{y}^*) \leq u(\mu, \boldsymbol{x}^*, \boldsymbol{y}^*). \end{aligned}$$

or, equivalently,

$$\begin{aligned} \max_{\mu} \min_{y} \quad & u(\mu, \boldsymbol{x}^*, y) \\ \text{s.t.} \quad \forall i \quad & \max_{x_i} u(\mu, x_i, \boldsymbol{x}^*_{-i}, \boldsymbol{y}^*) \leq u(\mu, \boldsymbol{x}^*, \boldsymbol{y}^*). \end{aligned}$$

This is precisely the problem of computing a zero-sum equilibrium in the game Γ_0, except with an extra constraint, so the inequality follows. □

B.3 Complexity Bounds and Proofs

Here, we state and prove the various lower bounds in Table 5.1. Before proceeding, we make several remarks about the conventions used in this section.

- Utilities will be given *unnormalized* by chance probability. That is, if we say that a player gets utility 1, what we really mean is that the player gets utility $1/p$, where p is the probability that chance sampled all actions on the path to z. Thus, the contribution to the expected value from this terminal node will be 1. This makes calculations easier.

- The utility range of the games used in the reductions will usually be of the form $[-M, M]$ where M is large but polynomial in the size of the game. Our definition of an extensive-form game allows only games with reward range $[-1, 1]$. This discrepancy is easily remedied by dividing all utility values in the proofs by M.

- $\Theta(\cdot)$ hides only an absolute constant.

To recap, we show hardness of *approximating* the value of the game to a desired precision $\varepsilon > 0$. Our hardness results will hold even when $\varepsilon = 1/\mathsf{poly}(|H|)$.

Theorem 5.4.5. *Even in* 2*-vs-*1 *games with public roles and* $\varepsilon = 1/\mathsf{poly}(|H|)$*, computing the TME value (and hence also the hidden-role value, since adversarial team games are a special case of hidden-role games) with public communication is* NP*-hard.*

Proof. We show that given any graph G, it is possible to construct a hidden-role game based on G, whose value corresponds to the size of the graph's max-cut. This reduces MAX-CUT to the TME value problem.

Let G be an arbitrary graph with n nodes and m edges, and consider the following team game (no hidden roles). There are 3 players, 2 of whom are on △. The game progresses as follows.

1. Chance chooses two vertices v_1, v_2 in G, independently and uniformly at random. The two players on △ observe v_1 and v_2 respectively.

2. ▲ independently (that is, with imperfect recall) select bits b_1, b_2 in place of the two players on △, and ▼ selects a pair (v_1', v_2'). There are now several things that can happen: (L is a large number to be picked later)

   (a) (*Agreement of players*) If $v_1 = v_2$ and $b_1 \neq b_2$, ▲ gets utility $-L$.

   (b) (*Objective*) If $v_1 \neq v_2$, $b_1 \neq b_2$, and (v_1, v_2) is an edge in G, then ▲ gets utility 1.

   (c) (*Non-leakage*) If $(v_1', v_2') = (v_1, v_2)$ then ▲ gets utility $-(n^2 - 1)L$. Otherwise, ▲ gets utility L.[1]

Consider a sufficiently large $L = \mathsf{poly}(m)$. The game is designed so that ▼'s objective is to guess the vertices v_1, v_2 sampled by chance, but she has no information apart from the communication transcript to guess it. Therefore, ▼'s optimal strategy is to punish any communication attempt between the players and play the most likely pair of vertices v_1, v_2. If no communication happens, her best strategy is to play a random pair of vertices. On the other hand, ▲'s optimal strategy must ensure that under no circumstance players play the same bit when assigned to the same vertex (lest they incur the large penalty L). Therefore, the strategy of both ▲ players is to play a fixed bit in each vertex, and the optimal strategy is the one that assigns a bit to the vertices in such a way that the number of edges connecting vertices with

[1] Note that ▼ playing uniformly at random means that the expected utility of this term is 0.

different bits is maximized. This corresponds to finding a max cut, and therefore, the value of the game is (essentially) c^* where c^* is the true size of the maximum cut. Moreover, any communication attempt would be immediately shut down by ▼ strategy since any leak of the observation received on the public channel implies receiving a fraction of the large penalty L.

We now formalize this intuition.

First, note that ▲ can achieve utility exactly c^* by playing according to a maximum cut. To see that ▲ cannot do significantly better, consider the following strategy for ▼. Observe the entire transcript τ of messages shared between the two ▲ players. Pick (v_1, v_2) maximizing the probability $p(\tau|v_1, v_2)$ that the players would have produced τ.[2]

First, suppose that ▲ does not communicate. Then ▼'s choice is independent of ▲'s, so ▲ can WLOG play a pure strategy. If $b_1 \neq b_2$ for any pair (v_1, v_2), then ▲ loses utility L in expectation. For $L > n^2$, this makes any such strategy certainly inferior to playing the maximum cut. Therefore, ▲ should play the maximum cut, achieving value c^*.

Now suppose that ▲ uses communication. Fix a transcript τ, and let

$$\delta := \max_{v_1,v_2} p(v_1, v_2|\tau) - \frac{1}{n^2}.$$

Now note that we can write $p(\cdot|\tau) = (1-\alpha)q_0 + \alpha q_1$, where $q_0, q_1 \in \Delta(V^2)$, q_0 is uniform, and $\alpha \leq \Theta(n^4\delta)$. Now consider any strategy that ▲ could play, given transcript τ. Such a strategy has the form $x_1(b_1|v_1)$ and $x_2(b_2|v_2)$. Since q is α-close to uniform, the utility of ▲ under this strategy conditioned on τ must be bounded above by

$$(1-\alpha)u_0 + \alpha \leq (1-\alpha)c^* + \alpha \leq c^* + \alpha \leq c^* + \Theta(n^4\delta)$$

where u_0 is the expected value of profile (x_1, x_2) given τ if $v_1, v_2|\tau$ were truly uniform. But now, in expectation over τ, team ▼ can gain utility $Ln^2\delta$ by playing $\operatorname{argmax}_{v_1,v_2} p(v_1, v_2|\tau)$. So, ▲'s utility is bounded above by

$$c^* + \Theta(n^4\delta) - Ln^2\delta \leq c^*$$

by taking L sufficiently large. □

The next result illustrates the difference between the uncoordinated hidden-role value and the coordinated hidden-role value, which is the focus of the positive results of our paper.

[2]Note again, as in Section 5.5, that this computation may take time exponential in r and the size of G, but we allow the players to perform unbounded computations.

Whereas the coordinated hidden-role value with private communication can be computed in polynomial time when k is constant (Theorem 5.4.3), the uncoordinated hidden-role value cannot, even when $k = 2$:

Theorem B.3.1. *Even in* 3*-vs-*2 *hidden-role games, the uncoordinated hidden-role value problem with private communication is* coNP*-hard.*

Proof. We reduce from UNSAT. Let ϕ be any 3-CNF-SAT formula, and consider the following 5-player hidden-role game. Two players are chosen uniformly at random to be on ∇; the rest are on Δ. The players on ∇ know each other. The players on ∇ play the SAT gadget game described by Koller and Megiddo [45]. Namely:

1. The players on ∇ are numbered P1 and P2, at random, by chance.
2. Chance selects a clause C in ϕ and tells P1.
3. P1 selects a variable x_i in C, and that variable (but not its sign in C, nor the clause C itself) is revealed to P2.
4. P2 selects an assignment $b_i \in \{0, 1\}$ to x_i. $\blacktriangledown$ wins the gadget game if the assignment b_i matches the sign of x_i in C.

The value of this game is decreasing with M and R since $\blacktriangle$ does nothing, so it is in the best interest of $\blacktriangle$ to select $R = 0$ (that is, forbid communication). In that case, the best probability with which $\blacktriangledown$ can win the game is exactly the maximum fraction of clauses satisfied by any assignment, which completes the proof. □

The above result is fairly straightforward: it is known that optimizing the mixed-strategy space of an imperfect-recall player (or equivalently a team with *asymmetric information* across its members) is hard [45], and private communication does not help if $\blacktriangle$ does not allow its use. However, next, we will show that the result even applies when $\blacktriangledown$ has symmetric information, that is when the original game Γ is coordinated. This may seem mysterious at first, but the intuition is the following. Just because Γ has symmetric information for ∇, it does not mean $\text{USPLIT}(\text{COMM}_{\text{priv}}(\Gamma))$ does. Indeed, Δ-players can send different private messages to different ∇-players, resulting in imperfect recall for $\blacktriangledown$ in the split-personality game. This result illustrates precisely the reason that we define two different split-personality games rather than simply dealing with the special case where the original game Γ has symmetric information for ∇.

Theorem B.3.2. *Even in* 3*-vs-*2 *hidden-role games with a mediator where no* ∇*-player has any information beyond the team assignment, the uncoordinated hidden-role value problem with private communication is* coNP*-hard.*

Proof. We will reduce from (the negation of) MAX-CUT. Let G be an arbitrary graph with n nodes and m edges, and consider the following 5-player hidden-role game with 2 players on ∇ and 3 players on Δ. Player 5 is always on Δ and is the mediator. The other four players are randomly assigned teams so that two are on Δ and two are on ∇. The game proceeds as follows.

1. Chance chooses vertices $v_1, \ldots, v_4$ uniformly at random from G. The mediator privately observes the whole tuple $(v_1, \ldots, v_4)$.

2. For notational purposes, call the players on Δ 3 and 4, and ∇ 1 and 2. (The mediator does not know these numbers.) After some communication, the following actions happen simultaneously:

   (a) P1 and P2 select bits $b_1, b_2 \in \{0, 1\}$;

   (b) P3 and P4 select vertices v_3', v_4' of G and players $i_3, i_4 \in \{1, 2, 3, 4\}$.

3. The following utilities are given: (L is a large number to be picked later, and each item in the list represents an additive term in the utility function)

   (a) (*Correct vertex identification*) For each player $i \in \{3, 4\}$, if $v_i \neq v_i'$ then Δ gets utility $-L^4$

   (b) (*Agreement of* ∇*-players*) If $v_1 = v_2$ and $b_1 \neq b_2$ then Δ gets utility L.

   (c) (*Objective*) If $v_1 \neq v_2$, $b_1 \neq b_2$, and (v_1, v_2) is an edge in G, then Δ gets utility -1.

   (d) (*Privacy*) For each $j \in \{3, 4\}$, if i_j^* is on ∇, then Δ gets utility L. Otherwise, Δ gets utility $-L/2$.

We claim that this game has value (essentially) $-c^*$, where c^* is the actual size of the maximum cut of G. To see this, observe first that the mediator *must* tell all players their true vertices v_i, lest it risk incurring the large negative utility $-L^2$. Further, any player except the mediator who sends a message must be on ∇. This prevents $\blacktriangledown$ from communicating. Thus, the mediator's messages force $\blacktriangledown$ to play an imperfect-recall game, which is hard.

We now formalize this intuition. First, consider the following strategy for ▲: The mediator sends all players their true types, and △-players play their types. If any △-player sees a message sent from anyone except the mediator, the △-player guesses that that player is on ▽.

Now consider any (pure) strategy profile of ▼. First, ▼ achieves utility $-c^*$ by observing the mediator's message and playing bits according to a maximum cut. We now show that this is the best that ▼ can do. Sending messages is, as before, a bad idea. Thus, a pure strategy profile of ▼ is given by four $f_1, f_2, f_3, f_4 : V \to \{0,1\}$ denoting how player i should pick its bits. But then $f_1 = f_2 = f_3 = f_4$; otherwise, the agreement of ▽-players would guarantee that ▼ is not playing optimally for large enough L.

Now, for any (possibly mixed) strategy profile of ▲, consider the following strategy profile for each ▽-player. Let $f : V \to \{0,1\}$ be a maximum cut. Pretend to be a △-player, and let v_i' be the vertex that would be played by that △-player. Play $f(v_i')$.

First, consider any △-player strategy profile for which, for some player i and some v_i, the probability that $v_i' \neq v_i$ exceeds $1/L^2$. Then ▲ gets a penalty of roughly L^2 in expectation, but now setting L large enough would force ▼ to have utility worse than -1, so that ▲ would rather simply play v_i with probability 1.

Now, condition on the event that $v_i = v_i'$ for all i (probability at least $1 - \Theta(1/L^2)$). In that case, the utility of ▲ is exactly $-c^*$, because ▼ is playing according to the maximum cut. Thus, the utility of ▲ is bounded by $-(1-\Theta(1/L^2))c^* + \Theta(1/L^2 \cdot L) \leq -c^*/n^2 + \Theta(n/L) < c^* + 1/2$ for sufficiently large L. Thus, solving the hidden-role game to sufficient precision and rounding the result would give the maximum cut, completing the proof. □

Theorem B.3.3. *Even in* 5*-vs-*4 *hidden-role games, the uncoordinated hidden-role value problem with public communication is* Σ_2^{P}*-hard.*

Proof. We reduce from ∃∀3-DNF-SAT, which is Σ_2^{P}-complete [38]. The ∃∀3-DNF-SAT problem is the following. Given a 3-DNF formula $\phi(x, y)$ with k clauses, where $x \in \{0,1\}^m$ and $y \in \{0,1\}^n$, decide whether $\exists x \forall y\ \phi(x, y)$. Consider the following game. There are 9 players, 5 on △ and 4 on ▽. One designated player, who we will call P0, is always on △ and has no role in the game. (The sole purpose of this player is so that △ is a majority.) The other players are randomly assigned teams. These other players are randomly assigned teams. For the sake of analysis, we number the remaining players P1 through P8 such that P1, P3, P4, P5 are on △ and P2, P6, P7, P8 are on ▽. ▼ knows the entire team assignment, whereas ▲ does not. We will call P3–P8 "regular players", and P1–P2 "guessers". The game proceeds as follows.

1. For each regular Δ-player, chance selects a literal (either x_j or $\neg x_j$), uniformly at random. For each regular ∇-player (P6–8), chance selects a literal (either y_j or $\neg y_j$), also uniformly at random. Each player privately observes the *variable* (index j), but not the sign of that variable.

2. After some communication, the following actions happen simultaneously.

   (a) P3–P8 select assignments (0 or 1) to their assigned variables.

   (b) P1 (who observes nothing) guesses a player (among the six players P3–P8) that P1 believes is on ∇.

   (c) P2 (who observes nothing) guesses one literal for each Δ-player (there are $K := (2m)^3$ such possible guesses.)

3. The following utilities are assigned. (L is a large number to be picked later, and each item in the list represents an additive term in the utility function)

   (a) (*Satisfiability*) Chance selects three regular players at random. If the three literals given to those players form a clause in ϕ, and the clause is satisfied, $\blacktriangle$ gets utility 1.

   (b) (*Consistency*) If chance selected the same variable three times, if the three players did not give the same assignment, then either $\blacktriangle$ or $\blacktriangledown$ gets utility $-L^2$ depending on if the variable was an x_i or a y_j.

   (c) (*Secrecy of* $\blacktriangle$) If P2 guesses the three literals correctly, $\blacktriangledown$ gets utility L^3K; otherwise, $\blacktriangledown$ gets utility $-L^3$.[3]

   (d) (*Secrecy of* $\blacktriangledown$) If P1 guesses a ∇-player, $\blacktriangle$ gets utility L; otherwise, $\blacktriangle$ gets $-L$.

We claim that the value of this game with public communication is at least 1 if and only if ϕ is $\exists\forall$-satisfiable. Intuitively, the rest of the proof goes as follows:

1. $\blacktriangle$ will not use the public communication channels if ϕ is $\exists\forall$-satisfiable. By the *Secrecy of* $\blacktriangledown$ term, ∇-players, therefore, cannot do so either without revealing themselves immediately. Thus, $\blacktriangle$ will get utility at least 1 if ϕ is $\exists\forall$-satisfiable.

2. If ϕ is not $\exists\forall$-satisfiable, then consider any Δ's 'strategy profile. By the *Secrecy of* $\blacktriangle$ term, $\blacktriangle$ cannot make nontrivial use of the public communication channel without

[3] These utilities are once again selected so that a uniformly random guess gets utility 0.

leaking information to P9. By the *Consistency* term, P1 through P3 must play the same assignment or else incur a large penalty. So, ▲ must play essentially an assignment to the variables in x, but such an assignment cannot achieve positive utility because there will exist a y that makes ϕ unsatisfied.

We now formalize this intuition. Suppose first that ϕ is $\exists\forall$-satisfiable, and let x be the satisfying assignment. Suppose that △-players never communicate and assign according to x, and P4 guesses any player that sends a message. Then any ▼-strategy that sends a message is bad for sufficiently large L because it guarantees a correct guess from ▲; any ▼-strategy that is inconsistent is bad because it will lose utility at least L; and any ▼-strategy that is consistent will cause ▲ to satisfy at least one clause. Thus ▲ guarantees utility at least 1.

Now suppose that ϕ is not $\exists\forall$-satisfiable. Consider any strategy profile for ▲. Suppose that ▼ plays as follows. During the public communication phase, each ▽-player samples a literal from the set $\{x_1, \neg x_1, \ldots, x_m, \neg x_m\}$ uniformly at random and pretends to be a △-player given that literal. P8 observes the public transcript and selects the triplet of literals that is conditionally most likely given the transcript. By an identical argument to that used in Theorem 5.4.5, ▲ then cannot profit from using the communication channel for sufficiently large L. Therefore, we can assume that ▲ does not use the communication channels, and therefore, by the argument in the previous paragraph, neither does ▼.

Now, the strategy of each △-player i can be described by a vector $s_i \in [0,1]^m$, where s_{ij} is the probability that player i assigns 1 to variable x_t. For sufficiently large L, we have $s_i \in [0, \varepsilon] \cup [1-\varepsilon, 1]$ where $\varepsilon = 1/L$, because otherwise ▲ would incur a penalty proportional to $L^2\varepsilon > k$ and would rather just play (for example) the all-zeros profile, which guarantees value 0. Condition on the event that every player at every variable chooses to play the most likely assignment according to the s_is. This happens with probability at least $1 - \Theta(m\varepsilon)$. These assignments must be consistent (that is, every player must have the same most-likely assignment), or else the players would once again incur a large penalty proportional to L^2. Call that assignment x, and let y be such that $\phi(x, y)$ is unsatisfied. Suppose ▼ plays according to y. Then ▲'s expected utility is bounded above by $\Theta(m\varepsilon k)$: with probability $1 - \Theta(m\varepsilon)$ it is bounded above by 0; otherwise it is bounded above by k. For $\varepsilon < \Theta(1/mk)$ this completes the proof. □

B.4 The Game *Avalon*

B.4.1 Equivalence of Split-Personality Games

In this section, we show that the choice of whether to use USPLIT or CSPLIT—that is, whether ▼ is coordinated—is irrelevant for the instantiations of *Avalon* that we investigate in this paper. We refer to Section 5.7 for a complete description of the *Avalon* game.

For this section, we assume that all ▼ players know the roles of all other ▼ players. This is always true in the instances considered in the experiments. In particular, we considered instances with six or fewer players and no *Oberon* role, which is a ▼ member, which is not revealed as such to both *Merlin* and the other ▼ members. This guarantees that both the ▼-players can deduce their respective roles. Conversely this is not guaranteed, as for example with 7 players (and hence $k = 3$) and Mordred, the two non-Mordred ▼-players do not know the identity of Mordred.

The main observation we make in this subsection states that the choice of whether to use USPLIT or CSPLIT does not matter.

Theorem B.4.1. *In* Avalon *with private communication, assuming that all ▼ players know each others' roles, every r-round uncoordinated hidden-role equilibrium is also an r-round coordinated hidden-role equilibrium.*

Before proving the result, we first observe that we cannot *a priori* assume the use of the results in Section 5.4.1 for this proof because they do not apply to the symmetric hidden-role equilibrium.

Fortunately, a variant of the revelation principle for mediated games (Theorem 5.4.1) and the resulting zero-sum game theorem (Theorem 5.4.2) still holds *almost* verbatim: the lone change is that the zero-sum game formed from symmetric splitting, which we will denote $\tilde{\Gamma}_0$, is a zero-sum *team* game, where the ▼ players cannot perfectly coordinate with each other, and the equality of value becomes an inequality:

Proposition B.4.2. *Let Γ^* be a mediated hidden-role game, and let Γ_0 be the zero-sum version posed by Theorem 5.4.2. Let $\tilde{\Gamma}_0$ be the game that is identical to Γ_0, except that the ▼-players are not coordinated and thus cannot communicate except as permitted in Γ. That is, $\tilde{\Gamma}_0$ is a* team game *with the mediator as the ▲-player and the adversaries as the ▼-players. Then $\mathsf{UVal}_{\mathsf{priv}}(\Gamma^*) \leq \mathsf{TMEVal}(\tilde{\Gamma}_0)$, where TMEVal is the TME value function (with ▲ committing first).*[4]

[4]Since ▼ is the nontrivial team and ▲ commits first, the TME and TMECor values of $\tilde{\Gamma}_0$ coincide here.

Proof. The revelation principle (Theorem 5.4.1) applies verbatim in this setting, so we may assume that ▲ in Γ^* uses the mediator as specified in the revelation principle. With this assumption, the only difference that remains between Γ^* and $\tilde{\Gamma}_0$ is that, in the latter, ▼-players cannot coordinate *at all*, whereas in Γ^*, ▼-players can somewhat coordinate by sending private messages (albeit at the cost of revealing themselves as adversaries). But this is a strict disadvantage for ▼ in $\tilde{\Gamma}_0$. □

We, therefore, have the inequality chain

$$\mathsf{Val}(\Gamma_0) = \mathsf{CVal}_{\mathsf{priv}}(\Gamma) \leq \mathsf{UVal}_{\mathsf{priv}}(\Gamma) \leq \mathsf{UVal}_{\mathsf{priv}}(\Gamma^*) \leq \mathsf{TMEVal}(\tilde{\Gamma}_0)$$

where Γ_0 is the zero-sum game that appears in Theorem 5.4.2, with asymmetric splitting, Val is the zero-sum game value, and Val is the zero-sum value.

So far, this inequality chain would hold for *any* hidden-role game Γ. We will now show that, specifically for *Avalon*, we have $\mathsf{TMEVal}(\tilde{\Gamma}_0) = \mathsf{Val}(\Gamma_0)$, which would complete the proof of Theorem B.4.1. This part of the proof depends on the special structure of *Avalon*.

The only difference between Γ_0 and $\tilde{\Gamma}_0$ is that, in the latter, ▼-players cannot communicate among each other because Theorem 5.4.2 only consider communication with the mediator. But what would ▼-players even need to communicate? Their information is entirely symmetric, except that they do not know the recommendations that were given to their teammates in the mission proposal and voting phases. We will show the following result.

Lemma B.4.3. *For both $\tilde{\Gamma}_0$ and Γ_0, the game value does not change if we assume that the mediator unilaterally dictates missions (circumventing the mission proposal and voting process).*

Proof Sketch. Make the following restrictions to strategy spaces.

- The mediator picks a single mission proposal in each round and recommends that proposal and that everyone vote in favor of it until it is approved, and ignores any information gained during this phase.
- *Spies* follow the above recommendations and ignore all information except the eventually approved mission.

Consider any TMECor of the restricted game. We claim that it also must be a TMECor of the full game:

- The mediator cannot do better because *Spies* are following all recommendations and ignoring any information gained except the eventual actual mission.
- *Spies* cannot do better because deviations are ignored by the mediator and will ultimately not affect the approved mission. (because the number of mission proposals and the number of *Resistance* players both exceed the number of *Spies*).

This completes the proof. □

But, having restricted the strategy spaces in the above manner, $\tilde{\Gamma}_0$ and Γ_0 become identical because there is no other source of asymmetric information. Therefore, $\mathsf{TMEVal}(\tilde{\Gamma}_0) = \mathsf{Val}(\Gamma_0)$, and we are done.

B.4.2 Abstractions

In this section, we describe the abstractions employed on the original game to transform it into a smaller version, which can then be solved tabularly. The abstractions employed are lossless in the sense that they do not change the game's value and that any Nash equilibrium in the abstracted game corresponds to a Nash equilibrium in the original one.

In particular, our main interest in the experiments is to compute one strategy belonging to a Nash equilibria for each player. This allows us to iteratively remove many actions without loss of generality.

Note that throughout this section, we interchangeably use the standard names *Resistance* and *Spies* to refer to the teams instead of ▲ and ▼.

We apply Theorems 5.4.1 to 5.4.3 to an Avalon game instance to compute a hidden-communication equilibria. In the first communication round, anybody can claim to the mediator to have a specific role and have specific knowledge about other players' roles. (For example, a ▼-player, that is, a *Spy*, could claim to be Merlin and lie about who is on team ▲.) Thanks to the revelation principle (Theorem 5.4.1), we can assume that each player in the *Resistance* team truthfully communicates their information, while each *Spies* player may decide to correlate with the others to emit a specific fake claim. During the game, the mediator recommends an action to each player anytime it is their move. In this case, the revelation principle allows us to assume WLOG that *Resistance* players always obey the recommendations received from the mediator, while *Spies* can decide to deviate. Overall, we obtain a two-player zero-sum game in which the *Resistance* team is represented by a mediator and the *Spies* team is a single player.

In the following, we list the abstractions we applied to our Avalon instances, and for each, we sketch a proof of correctness. Each proof will have the same structure: we will restrict the strategy spaces of both teams, resulting in a smaller zero-sum game; we will then show that any equilibrium of this smaller zero-sum game cannot have a profitable deviation in the full game.

1. *The recommendations emitted by the mediator to the player in charge of the mission proposal should never be rejected. That is, the mediator has unilateral power to dictate who goes on missions.*

   This is Lemma B.4.3.

2. *Call the following information* common knowledge*:*

   - *The sets of good players claimed by players claiming to be Merlin, but* not *the identities of the claimants themselves (because ▼ does not know the true Merlin)*
   - *Mission proposals and results*
   - *Any claimant whose claim is disproven by common-knowledge information is a* Spy

   Call a subset $S \subseteq [n]$ plausible *if, based on common-knowledge information, it is possible that S contains only good players. Every mission proposal should be plausible.*

   Proof sketch. Make the following restrictions to strategy spaces.

   - Every mission proposal by the mediator is plausible.
   - If the mediator proposes an implausible set, the *Spies* pick a plausible set at random, announce it publicly, and act as if the mediator proposed that set instead.

   Consider any Nash equilibrium of the restricted game. We claim it must also be a Nash equilibrium of the full game.

   - The mediator cannot improve because *Spies* will pretend that the mediator played a plausible set anyway, and the mediator can simulate how the spies will do this.
   - *Spies* cannot improve because against a mediator who always proposes a plausible set, the strategy space of the spies is not limited.

3. *If any player is contained in* every *plausible set S, then that player should always be included on missions if possible. We call such players* safe*.*

   Proof sketch. Make the following restrictions to strategy spaces.

- The mediator always sends all *safe* players on missions, if possible.
- *Spies* pretend that all *safe* players are always sent on missions. More precisely, if i is safe, not sent on a mission, and at least one unsafe player j is sent on that mission, *Spies* pick such a pair (i, j) at random, announce both i and j publicly, and pretend that i was sent in place of j. (If there are multiple such replacements that can be done, this process is simply repeated.)

Consider any Nash equilibrium of the restricted game. We claim it must also be a Nash equilibrium of the full game.

- The mediator cannot improve because *spies* will pretend that all safe players are always sent regardless of how the mediator actually proposes missions, and the mediator can simulate how *spies* will perform the replacement.
- *Spies* cannot improve because against a mediator who always sends all safe players on missions, the strategy space of the spies is not limited.

4. *If a mission of size s passes, all missions will pass until the next mission whose size is $> s$.* (In particular, any maximum-size mission passing implies that all remaining missions pass.)

   Proof sketch. Suppose that a mission M_0 of size s_0 has passed, and suppose the next t missions have sizes $s_1, \ldots, s_t \leq s_0$. Make the following restrictions to strategy spaces.

   - The mediator randomly[5] selects subsets $M'_i \subseteq M_0$ to send on missions s_i for $i = 1, \ldots, t$. If any of them (say, M_i) fails, the mediator pretends that M_0 failed instead. Then, the mediator generates and publicly announces the missions $M'_1, \ldots, M'_i$ that it *would have* proposed had M_0 failed, assuming that each M'_j passes. The mediator then pretends that that is what happened. The mediator does not use the M_is to inform future decisions.
   - For each $i = 1, \ldots, t$, *Spies* automatically pass M_i and ignore what missions are proposed by the mediator.

   Consider any Nash equilibrium of the restricted game. We claim it must also be a Nash equilibrium of the full game.

[5]The fact that this is *random* instead of dictated by the mediator allows us to avoid issues of imperfect recall.

- The mediator has no incentive to propose a mission different from a random $M_i \subseteq M_0$ in the unrestricted game, because against spies who ignore and always pass missions $M_1, \ldots, t$, any mediator mission would lead to the same outcome of having a mission passing, no change in behavior from *Spies*, and no information gained.

- The *Spies* have no incentive to fail mission M_i for $i > 0$ if they pass mission M_0. This is true because the mediator strategy is such that *Spies* passing missions $M_0, \ldots, M_{i-1}$ and failing mission M_i would be equivalent to *Spies* failing M_0 and passing missions $M'_1, \ldots M'_i$.

Christiano [21] used a very strong reduction that allowed the assumption that the smallest two missions are guaranteed to pass. The reduction is sound in the base game *Resistance*, as well as when the mediator's strategy is manually constrained to *never* reveal information about Merlin, as is the case in Christiano's analysis. It turns out, however, that this reduction is *not* sound in general because its proof assumes that all mission proposals—even those that occur after *Resistance* have already reached the required number of missions—are public information. However, in our setting, each mission proposal may leak more and more information about Merlin, so there is an incentive to reveal as few missions as possible. Indeed, in the 6-player variants with Mordred, we observe that removing the two smallest missions causes a small but nonzero change in the game value, whether or not future mission proposals are published. However, a more refined analysis can be used to partially recover a reduction in the number of missions.

With five players, the refined analysis allows the same result Christiano [21]: the first two missions should automatically pass.

In 5-player *Avalon*, the mission sizes are 2, 2, 3, 3, 3 in that order.

Proposition B.4.4. *In 5-player Avalon with private communication, the hidden-role value is the same if the two missions of size 2 are automatically passed (hence skipped).*

Proof Sketch. Make the following restrictions to strategy spaces.

- The mediator privately picks three missions M_1, M_2, M_3 of size 3. Then the mediator proposes M_1 (or a subset thereof) until it fails, then M_2 or a subset thereof until that fails, and finally M_3 until that fails.

- *Spies* ignore and automatically pass the first two mission proposals and fail the last three missions if possible.

Consider any Nash equilibrium of the restricted game. We claim it must also be a Nash equilibrium of the full game.

- The mediator has no incentive to behave differently because *Spies* are ignoring the first two missions regardless.
- *Spies* win if either three missions are failed or they guess Merlin. But if at least one mediator guesses is the true set of good players, only two missions can fail, and so *Spies* cannot increase their probability of failing three missions. As for guessing Merlin, *Spies* cannot increase the amount of information gained. Let i be the index such that M_i is the true good set and $i = 4$ if all mediator guesses were wrong. Spies are currently gaining the information $M_{\leq i}$, and Spies cannot gain more information from deviating because the mediator always plays the missions in order.

This completes the proof. □

Our analysis is refined compared to Christiano [21] in that we must consider the possibility that *Spies* can gain more information about Merlin by deviating from the restricted strategy space. With five players, this turns out not to have an effect, but with six players, it will. With six players, the refined analysis only allows the first mission to be removed.

In six-player Avalon, the mission sizes are 2, 3, 4, 3, 4. The problem in adapting the analysis of the previous result is that the final size-3 mission comes *between* the size-4 missions, preventing its removal.

Proposition B.4.5. *In 6-player Avalon with private communication, the hidden-role value is the same if the mission of size 2 is automatically passed (hence skipped).*

Proof Sketch. Consider any strategy x for the mediator in the restricted game in which the first mission is ignored. We will extend x to a full-game strategy x'. Strategy x' works as follows. The mediator maintains a simulator of strategy x.

1. The mediator queries the x-simulator for its second mission M_2 (of size 3), and plays a random size-2 subset of M_2. If this mission passes, the mediator plays M_2 itself on the second mission, then plays the rest of the game according to x.
2. If the first mission fails, the mediator tells the x-simulator that M_2 failed. The x-simulator will then output its third mission M_3 (of size 3). The mediator picks a random size-3 subset of M_3 to send on the second mission. If this mission passes, the mediator plays M_3 again on the third mission and once again plays the rest of the game according to x.

3. If the second mission fails, the mediator tells the x-simulator that M_3 failed. The x-simulator will then output its fourth mission M_4 (of size 3). The mediator immediately tells the x-simulator that M_4 passed. The x-simulator will then output its fifth mission M_5, of size 4. The mediator plays random subsets of M_5 until the game ends.

We claim that, against this mediator, the adversary cannot do better by failing the first mission than by passing it.

- If the adversary fails the first mission and passes the second, then the adversary will have, in the first three missions, observed a random subset of M_2 and the whole set M_3. But the adversary could have accomplished more by passing the first mission and failing M_2—the behavior of the mediator would be the same in both cases (by construction of the mediator), and the adversary would observe the whole set M_2 instead of merely a random subset.
- If the adversary fails the first two missions, the adversary (regardless of what else happens in the game) will have observed a random subset of M_2, a random subset of M_3, and M_5 by the end of the game. Further, three missions will pass if and only if M_5 is the true good set. The adversary, however, could have accomplished more by passing the first mission, failing the second and third (M_2 and M_3), and passing the fourth—the adversary induces the same outcome in the game but instead observes all four missions M_2, M_3, M_4, M_5.

Thus, the adversary is always better off failing the first mission, and therefore x and x' have the same value. □

B.4.3 A Description of *Avalon* in Reduced Representation

The optimized *Avalon* game instances can be succinctly represented as follows:

1. At the start of the game, Chance player deals the roles.
2. *[if Merlin is present in the game]* Merlin reports who the *Spies* are (except *Mordred*). Spies may also (falsely) do the same.
3. *[if Percival is present in the game]* Percival reports who Merlin is. *Spies* may also falsely do the same.

Variant	5 Players $\lvert Z\rvert$	5 Players RW	5 Players MG	6 Players $\lvert Z\rvert$	6 Players RW	6 Players MG
No special roles (*Resistance*)	5.9×10^3	0.3000	n/a	1.4×10^6	0.3333	n/a
Merlin only	1.5×10^4	1.0000	0.3333	6.0×10^5	1.0000	0.2500
Merlin + Mordred	4.9×10^5	0.6691	0.3869	1.8×10^8	0.7529	0.3046
Merlin + 2 Mordreds	9.0×10^4	0.5000	0.4444	2.8×10^7	0.4088	0.2886
Merlin+Mordred+Percival+Morgana	2.4×10^6	0.9046	0.3829	unknown	unknown	unknown

Table B.1: Breakdown of equilibrium outcome probabilities for each variant of *Avalon*. $|Z|$: number of terminal nodes in the reduced game tree. 'RW': probability of Resistance passing three missions. 'MG': probability of Spies guessing Merlin correctly, conditioned on Resistance passing three missions. (The game value, which appears in Table 5.2, is RW $\cdot$ $(1 - \text{MG})$).

4. Until the number of missions passed is less than 3, the mediator sends a mission, and the *Spies* are offered the option to fail it if any of them is on it. The space of possible missions to be proposed is constrained according to the previous optimizations

5. If the number of missions failed is 3, the game ends with a win for the *Spies*.

6. If the number of missions won is 3, and Merlin is present, the *Spies* have to guess the player with the Merlin role. If they guess correctly, they win, otherwise they lose. If Merlin is absent, the game ends with a win for the *Resistance*.

B.4.4 Example of Optimal Play in *Avalon*

In this section, we fully describe one of the equilibria claimed in Table 5.2: the case of 5 players with Merlin and both ▼ players "Mordred" (hidden from Merlin). We choose this version because it is by far the easiest to describe the equilibrium: the other new values listed in Table 5.2 are very highly mixed and difficult to explain.

Similarly to what happens in the example provided in Section 5.2.4, this case is different from the case with no Merlin (even if Merlin does not have any useful knowledge), due to the effect of added correlation. In particular, if Merlin were not known to the mediator, then the equilibrium value would be $3/10 \cdot 2/3 = 2/10$: $3/10$ from the value of pure *Resistance* (no Merlin), and the factor of $2/3$ from a blind Merlin guess.

By the discussion above, it suffices to analyze the reduced game with three missions, each of size 3, where ▲ need only win one mission to win the game.

The following strategy pair constitutes an equilibrium for this *Avalon* variant:

Strategy for ▲ The mediator's strategy is different depending on how many players claim to it to be Merlin.

One claim. Randomly label the players A through E such that A is Merlin. Send missions ABC, ABD, and ABE.

There are six possible correct sets (ABC, ABD, ABE, ACD, ACE, ADE), and ▲ has three guesses, so ▲ wins the regular game with probability $1/2$. If the first mission succeeds, ▼ learns nothing about Merlin, so Merlin is guessed correctly with probability $1/3$. Otherwise, ▼ knows that Merlin is either A or B, so Merlin is guessed correctly with probability $1/2$. Thus, this strategy attains value $(1/6)(2/3) + (1/3)(1/2) = 5/18$.

Two claims. Randomly label the players A through E such that A and B are the Merlin claimers. Send missions ACD, BCE, and ADE.

Three claims. The non-claimants are guaranteed to be good. Pick a non-claimant randomly, pretend they are Merlin, and execute the strategy from the one-claim case. This is strictly better than the one-claim case because ▼ learns only information about who is *not* Merlin instead of who *is* Merlin. (This strategy may not be subgame-perfect, but for the sake of this analysis, it is enough for this strategy to achieve value *at least* $5/18$).

The analysis of the one-claim case applies nearly verbatim: there are six possible correct sets, and the second mission narrows down ▼'s list of possible Merlins from 3 to 2 (namely, A or D), so again the value for ▼ is $5/18$.

Strategy for ▼ Emulate the behavior of a non-Merlin ▲-player (*i.e.*, do not claim to be Merlin).

1. If the first guess by ▲ is correct, guess Merlin randomly from the good players.
2. If the second guess by ▲ is correct, guess Merlin randomly from the good players included on both guesses.
3. If the third guess by ▲ is correct, guess Merlin at random among all good players sent on at least two missions, weighting any player sent on all three missions double. (for example, if there was one player sent on all three missions and one player sent on two, guess the former with probability $2/3$ and the latter with probability $1/3$).

Against this strategy, the mediator's only decision is what three distinct missions to send.

1. If Merlin is sent on *one* mission, the probability of winning is at most $1/6$ because only that mission has a chance of being the correct one.

2. If Merlin is sent on *two* missions, then the probability of one mission passing is at most $1/3$. If the first mission passes, Merlin is guessed correctly with probability $1/3$. If the second mission passes, Merlin may not be guessed correctly at all (it is possible for Merlin to only have been sent on the second mission, but not the first). If the third mission passes, Merlin is guessed correctly with a probability of at least $1/5$. Thus, the probability of winning for good is at most $(1/3)(1 - (1/3 + 0 + 1/5)/3) = 37/135 < 5/18$.

3. If Merlin is sent on *all three* missions, the probability of a mission passing is at most $1/2$. If the first mission passes, Merlin is guessed correctly with probability $1/3$. Otherwise, Merlin is guessed correctly with probability at least $1/2$, since the only way to decrease this probability under $1/2$ would be for the last mission to pass *and* for another player to also have been sent on all three missions, but in that case, it is impossible for anyone to have been sent on two missions. Therefore, the probability of winning for good is at most $(1/2)(1 - (1/3 + 1/2 + 1/2)/3) = 5/18$.