Lossless abstraction of imperfect information games

ANDREW GILPIN and TUOMAS SANDHOLM
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA, USA

Finding an equilibrium of an extensive form game of imperfect information is a fundamental problem in computational game theory, but current techniques do not scale to large games. To address this, we introduce the *ordered game isomorphism* and the related *ordered game isomorphic abstraction transformation*. For a multi-player sequential game of imperfect information with observable actions and an ordered signal space, we prove that any Nash equilibrium in an abstracted smaller game, obtained by one or more applications of the transformation, can be easily converted into a Nash equilibrium in the original game. We present an algorithm, *GameShrink*, for abstracting the game using our isomorphism exhaustively. Its complexity is $\tilde{O}(n^2)$, where n is the number of nodes in a structure we call the signal tree. It is no larger than the game tree, and on nontrivial games it is drastically smaller, so *GameShrink* has time and space complexity *sublinear* in the size of the game tree. Using *GameShrink*, we find an equilibrium to a poker game with 3.1 billion nodes—over four orders of magnitude more than in the largest poker game solved previously. To address even larger games, we introduce approximation methods that do not preserve equilibrium, but nevertheless yield (*ex post*) provably close-to-optimal strategies.

Categories and Subject Descriptors: I.2 [**Artificial Intelligence**]: General; J.4 [**Social and Behavioral Sciences**]: Economics; F.2 [**Analysis of Algorithms and Problem Complexity**]: General

General Terms: Algorithms, Economics, Theory

Additional Key Words and Phrases: Automated abstraction, computer poker, equilibrium finding, game theory, sequential games of imperfect information

1. INTRODUCTION

In environments with more than one agent, an agent's outcome is generally affected by the actions of the other agent(s). Consequently, the optimal strategy of one agent can depend on the others. Game theory provides a normative framework for analyzing such strategic situations. In particular, it provides solution concepts that define what rational behavior is in such settings. The most famous and important solution concept is that of *Nash equilibrium* [Nash 1950]. It is a strategy profile (one strategy for each agent) in which no agent has incentive to deviate to a different strategy. However, for the concept to be operational, we need algorithmic techniques for finding an equilibrium.

Games can be classified as either games of *perfect information* or *imperfect infor-*

This material is based upon work supported by the National Science Foundation under ITR grants IIS-0121678 and IIS-0427858, and a Sloan Fellowship. A short version of this paper appeared at the ACM Conference on Electronic Commerce (EC), Ann Arbor, MI, June, 2006.

mation. Chess and Go are examples of the former, and, until recently, most game playing research has been on games of this type. To compute an optimal strategy in a perfect information game, an agent traverses the game tree and evaluates individual nodes. If the agent is able to traverse the entire game tree, she simply computes an optimal strategy from the bottom-up, using the principle of *backward induction*.[1] In computer science terms, this is done using *minimax search* (often in conjunction with α-β-*pruning* to reduce the search tree size and thus enhance speed). Minimax search runs in linear time in the size of the game tree.[2]

The differentiating feature of games of imperfect information, such as poker, is that they are not fully observable: when it is an agent's turn to move, she does not have access to all of the information about the world. In such games, the decision of what to do at a point in time cannot generally be optimally made without considering decisions at all other points in time (including ones on other paths of play) because those other decisions affect the probabilities of being at different states at the current point in time. Thus the algorithms for perfect information games do not solve games of imperfect information.

For sequential games with imperfect information, one could try to find an equilibrium using the normal (matrix) form, where every contingency plan of the agent is a pure strategy for the agent.[3] Unfortunately (even if equivalent strategies are replaced by a single strategy [Kuhn 1950a]) this representation is generally exponential in the size of the game tree [von Stengel 1996].

By observing that one needs to consider only sequences of moves rather than pure strategies [Romanovskii 1962; Selten 1988; Koller and Megiddo 1992; von Stengel 1996], one arrives at a more compact representation, the *sequence form*, which is linear in the size of the game tree.[4] For 2-player games, there is a polynomial-sized (in the size of the game tree) linear programming formulation (linear complemen-

[1]This actually yields a solution that satisfies not only the Nash equilibrium solution concept, but a stronger solution concept called *subgame perfect Nash equilibrium* [Selten 1965].

[2]This type of algorithm still does not scale to huge trees (such as in chess or Go), but effective game-playing agents can be developed even then by evaluating intermediate nodes using a heuristic evaluation and then treating those nodes as leaves.

[3]There has been significant recent work on Nash equilibrium finding for normal (matrix) form games. An ϵ-equilibrium in a normal form game with any constant number of agents can be constructed in quasi-polynomial time [Lipton et al. 2003], but finding an exact equilibrium is PPAD-complete even in a 2-player game [Chen and Deng 2006]. The most prevalent algorithm for finding an equilibrium in a 2-agent game is *Lemke-Howson* [Lemke and Howson 1964], but it takes exponentially many steps in the worst case [Savani and von Stengel 2004]. For a survey of equilibrium computation in 2-player games, see [von Stengel 2002]. Equilibrium-finding algorithms that enumerate supports (i.e., sets of pure strategies that are played with positive probability) have been shown efficient on many games [Porter et al. 2004], and efficient mixed integer programming algorithms that search in the space of supports have been developed [Sandholm et al. 2005]. For more than two players, many algorithms have been proposed, but they currently only scale to very small games [Govindan and Wilson 2003; McKelvey and McLennan 1996; Porter et al. 2004]. Progress has also been made on algorithms for finding equilibria in restricted and/or structured games (e.g., [Papadimitriou and Roughgarden 2005; Bhat and Leyton-Brown 2004; Leyton-Brown and Tennenholtz 2003; Blum et al. 2003; Singh et al. 2004]).

[4]There were also early techniques that capitalized in different ways on the fact that in many games the vast majority of pure strategies are not played in equilibrium [Wilson 1972; Koller and Megiddo 1996].

tarity in the non-zero-sum case) based on the sequence form such that strategies for players 1 and 2 correspond to primal and dual variables. Thus, the equilibria of reasonable-sized 2-player games can be computed using this method [von Stengel 1996; Koller et al. 1996; Koller and Pfeffer 1997].[5] However, this approach still yields enormous (unsolvable) optimization problems for many real-world games, such as poker.

1.1 Our approach

In this paper, we take a different approach to tackling the difficult problem of equilibrium computation. Instead of developing an equilibrium-finding method *per se*, we instead develop a methodology for automatically abstracting games in such a way that any equilibrium in the smaller (abstracted) game corresponds directly to an equilibrium in the original game. Thus, by computing an equilibrium in the smaller game (using any available equilibrium-finding algorithm), we are able to construct an equilibrium in the original game. The motivation is that an equilibrium for the smaller game can be computed drastically faster than for the original game.

To this end, we introduce *games with ordered signals* (Section 2), a broad class of games that has enough structure which we can exploit for abstraction purposes. Instead of operating directly on the game tree (something we found to be technically challenging), we instead introduce the use of *information filters* (Section 2.2), which coarsen the information each player receives. They are used in our analysis and abstraction algorithm. By operating only in the space of filters, we are able to keep the strategic structure of the game intact, while abstracting out details of the game in a way that is lossless from the perspective of equilibrium finding. We introduce the *ordered game isomorphism* to describe strategically symmetric situations and the *ordered game isomorphic abstraction transformation* to take advantange of such symmetries (Section 3). As our main equilibrium result we have the following:

Theorem 3.4 *Let Γ be a game with ordered signals, and let F be an information filter for Γ. Let F' be an information filter constructed from F by one application of the ordered game isomorphic abstraction transformation, and let σ' be a Nash equilibrium strategy profile of the induced game $\Gamma_{F'}$ (i.e., the game Γ using the filter F'). If σ is constructed by using the corresponding strategies of σ', then σ is a Nash equilibrium of Γ_F.*

The proof of the theorem uses an equivalent characterization of Nash equilibria: σ is a Nash equilibrium if and only if there exist beliefs μ (players' beliefs about unknown information) at all points of the game reachable by σ such that σ is sequentially rational (i.e., a best response) given μ, where μ is updated using Bayes' rule. We can then use the fact that σ' is a Nash equilibrium to show that σ is a Nash equilibrium considering only local properties of the game.

We also give an algorithm, *GameShrink*, for abstracting the game using our isomorphism exhaustively (Section 4). Its complexity is $\tilde{O}(n^2)$, where n is the number of nodes in a structure we call the signal tree. It is no larger than the game

[5]Recently this approach was extended to handle computing *sequential equilibria* [Kreps and Wilson 1982] as well [Miltersen and Sørensen 2006].

tree, and on nontrivial games it is drastically smaller, so *GameShrink* has time and space complexity *sublinear* in the size of the game tree. We present several algorithmic and data structure related speed improvements (Section 4.1), and we demonstrate how a simple modification to our algorithm yields an approximation algorithm (Section 5).

1.2 Applications

Sequential games of imperfect information are ubiquitous, for example in negotiation and in auctions. Often aspects of a player's knowledge are not pertinent for deciding what action the player should take at a given point in the game. On the trivial end, some aspects of a player's knowledge are never pertinent (e.g., whether it is raining or not has no bearing on the bidding strategy in an art auction), and such aspects can be completely left out of the model specification. However, more generally, some aspects can be pertinent in certain states of the game while they are not pertinent in other states, and thus cannot be left out of the model completely. Furthermore, it may be highly non-obvious which aspects are pertinent in which states of the game. Our algorithm automatically discovers which aspects are irrelevant in different states, and eliminates those aspects of the game, resulting in a more compact, equivalent game representation.

One broad application area that has this property is sequential negotiation (potentially over multiple issues). Another broad application area is sequential auctions (potentially over multiple goods). For example, in those states of a 1-object auction where bidder A can infer that his valuation is greater than that of bidder B, bidder A can ignore all his other information about B's signals, although that information would be relevant for inferring B's exact valuation. Furthermore, in some states of the auction, a bidder might not care which exact other bidders have which valuations, but cares about which valuations are held by the other bidders in aggregate (ignoring their identities). Many open-cry sequential auction and negotiation mechanisms fall within the game model studied in this paper (specified in detail later), as do certain other games in electronic commerce, such as sequences of take-it-or-leave-it offers [Sandholm and Gilpin 2006].

Our techniques are in no way specific to an application. The main experiment that we present in this paper is on a recreational game. We chose a particular poker game as the benchmark problem because it yields an extremely complicated and enormous game tree, it is a game of imperfect information, it is fully specified as a game (and the data is available), and it has been posted as a challenge problem by others [Shi and Littman 2002] (to our knowledge no such challenge problem instances have been proposed for electronic commerce applications that require solving sequential games).

1.3 Rhode Island Hold'em poker

Poker is an enormously popular card game played around the world. The 2005 World Series of Poker had over \$103 million dollars in total prize money, including \$56 million for the main event. Increasingly, poker players compete in online casinos, and television stations regularly broadcast poker tournaments. Poker has been identified as an important research area in AI due to the uncertainty stemming from opponents' cards, opponents' future actions, and chance moves, among other reasons [Billings et al. 2002].

Almost since the field's founding, game theory has been used to analyze different aspects of poker [Kuhn 1950b; Nash and Shapley 1950; Bellman and Blackwell 1949; von Neumann and Morgenstern 1947, pp. 186–219]. However, this work was limited to tiny games that could be solved by hand. More recently, AI researchers have been applying the computational power of modern hardware to computing game theory-based strategies for larger games. Koller and Pfeffer determined solutions to poker games with up to 140,000 nodes using the sequence form and linear programming [Koller and Pfeffer 1997]. Large-scale approximations have been developed [Billings et al. 2003], but those methods do not provide any guarantees about the performance of the computed strategies. Furthermore, the approximations were designed manually by a human expert. Our approach yields an automated abstraction mechanism along with theoretical guarantees on the strategies' performance.

Rhode Island Hold'em was invented as a testbed for computational game playing [Shi and Littman 2002]. It was designed so that it was similar in style to Texas Hold'em, yet not so large that devising reasonably intelligent strategies would be impossible. (The rules of Rhode Island Hold'em are given in Section 2.1. That section also shows how Rhode Island Hold'em can be modeled as a game with ordered signals, that is, it fits in our model.) We applied the techniques developed in this paper to find an exact (minimax) solution to Rhode Island Hold'em, which has a game tree exceeding 3.1 billion nodes.

Applying the sequence form to Rhode Island Hold'em directly without abstraction yields a linear program with 91,224,226 rows, and the same number of columns. This is much too large for (current) linear programming algorithms to handle. We used our *GameShrink* algorithm to reduce this through lossless abstraction, and it yielded a linear program with 1,237,238 rows and columns—with 50,428,638 non-zero coefficients. We then applied iterated elimination of dominated strategies, which further reduced this to 1,190,443 rows and 1,181,084 columns. (Applying iterated elimination of dominated strategies without *GameShrink* yielded 89,471,986 rows and 89,121,538 columns, which still would have been prohibitively large to solve.) *GameShrink* required less than one second to perform the shrinking (i.e., to compute all of the ordered game isomorphic abstraction transformations). Using a 1.65GHz IBM eServer p5 570 with 64 gigabytes of RAM (the linear program solver actually needed 25 gigabytes), we solved it in 7 days and 17 hours using the interior-point barrier method of CPLEX version 9.1.2. We demonstrated our optimal Rhode Island Hold'em poker player at the AAAI-05 conference [Gilpin and Sandholm 2005], and it is available for play on-line at `http://www.cs.cmu.edu/~gilpin/gsi.html`.

While others have worked on computer programs for playing Rhode Island Hold'em [Shi and Littman 2002], no optimal strategy has been found before. This is the largest poker game solved to date by over four orders of magnitude.

2. GAMES WITH ORDERED SIGNALS

We work with a slightly restricted class of games, as compared to the full generality of the extensive form.[6] This class, which we call *games with ordered signals*, is highly structured, but still general enough to capture a wide range of strategic situations. A game with ordered signals consists of a finite number of rounds. Within a round, the players play a game on a directed tree (the tree can be different in different rounds). The only uncertainty players face stems from private signals the other players have received and from the unknown future signals. In other words, players observe each others' actions, but potentially not nature's actions. In each round, there can be public signals (announced to all players) and private signals (confidentially communicated to individual players). For simplicity, we assume—as is the case in most recreational games—that within each round, the number of private signals received is the same across players (this could quite likely be relaxed). We also assume that the legal actions that a player has are independent of the signals received. For example, in poker, the legal betting actions are independent of the cards received. Finally, the strongest assumption is that there is a partial ordering over sets of signals, and the payoffs are increasing (not necessarily strictly) in these signals. For example, in poker, this partial ordering corresponds exactly to the ranking of card hands.

Definition 2.1. A *game with ordered signals* is a tuple $\Gamma = \langle I, G, L, \Theta, \kappa, \gamma, p, \succeq, \omega, u\rangle$ where:

(1) $I = \{1, \ldots, n\}$ is a finite set of players.

(2) $G = \langle G^1, \ldots, G^r\rangle$, $G^j = \left(V^j, E^j\right)$, is a finite collection of finite directed trees with nodes V^j and edges E^j. Let Z^j denote the leaf nodes of G^j and let $N^j(v)$ denote the outgoing neighbors of $v \in V^j$. G^j is the *stage game* for round j.

(3) $L = \langle L^1, \ldots, L^r\rangle$, $L^j : V^j \setminus Z^j \to I$ indicates which player acts (chooses an outgoing edge) at each internal node in round j.

(4) Θ is a finite set of *signals*.

(5) $\kappa = \langle \kappa^1, \ldots, \kappa^r\rangle$ and $\gamma = \langle \gamma^1, \ldots, \gamma^r\rangle$ are vectors of nonnegative integers, where κ^j and γ^j denote the number of public and private signals (per player), respectively, revealed in round j. Each signal $\theta \in \Theta$ may only be revealed once, and in each round every player receives the same number of private signals, so we require $\sum_{j=1}^{r} \kappa^j + n\gamma^j \le |\Theta|$. The public information revealed in round j is $\alpha^j \in \Theta^{\kappa^j}$ and the public information revealed in all rounds up through round j is $\tilde{\alpha}^j = \left(\alpha^1, \ldots, \alpha^j\right)$. The private information revealed to player $i \in I$ in round j is $\beta_i^j \in \Theta^{\gamma^j}$ and the private information revaled to player $i \in I$ in all rounds up through round j is $\tilde{\beta}_i^j = \left(\beta_i^1, \ldots, \beta_i^j\right)$. We also write $\tilde{\beta}^j = \left(\tilde{\beta}_1^j, \ldots, \tilde{\beta}_n^j\right)$ to represent all private information up through round j, and $\left(\tilde{\beta'}_i^j, \tilde{\beta}_{-i}^j\right) = \left(\tilde{\beta}_1^j, \ldots, \tilde{\beta}_{i-1}^j, \tilde{\beta'}_i^j, \tilde{\beta}_{i+1}^j, \ldots, \tilde{\beta}_n^j\right)$ is $\tilde{\beta}^j$ with $\tilde{\beta}_i^j$ replaced with $\tilde{\beta'}_i^j$.

[6]For readers unfamiliar with extensive form games, we provide a complete definition in Appendix A.

The total information revealed up through round j, $\left(\tilde{\alpha}^j, \tilde{\beta}^j\right)$, is said to be *legal* if no signals are repeated.

(6) p is a probability distribution over Θ, with $p(\theta) > 0$ for all $\theta \in \Theta$. Signals are drawn from Θ according to p without replacement, so if X is the set of signals already revealed, then

$$p(x \mid X) = \begin{cases} \frac{p(x)}{\sum_{y \notin X} p(y)} & \text{if } x \notin X \\ 0 & \text{if } x \in X. \end{cases}$$

(7) $\succeq$ is a partial ordering of subsets of Θ and is defined for at least those pairs required by u.

(8) $\omega : \bigcup\limits_{j=1}^{r} Z^j \to \{\mathit{over}, \mathit{continue}\}$ is a mapping of terminal nodes within a stage game to one of two values: *over*, in which case the game ends, or *continue*, in which case the game continues to the next round. Clearly, we require $\omega(z) = \mathit{over}$ for all $z \in Z^r$. Note that ω is independent of the signals. Let $\omega^j_{over} = \left\{z \in Z^j \mid \omega(z) = \mathit{over}\right\}$ and $\omega^j_{cont} = \left\{z \in Z^j \mid \omega(z) = \mathit{continue}\right\}$.

(9) $u = (u^1, \ldots, u^r)$, $u^j : \bigtimes\limits_{k=1}^{j-1} \omega^k_{cont} \times \omega^j_{over} \times \bigtimes\limits_{k=1}^{j} \Theta^{\kappa^k} \times \bigtimes\limits_{i=1}^{n} \bigtimes\limits_{k=1}^{j} \Theta^{\gamma^k} \to \mathbb{R}^n$ is a utility function such that for every j, $1 \leq j \leq r$, for every $i \in I$, and for every $\tilde{z} \in \bigtimes\limits_{k=1}^{j-1} \omega^k_{cont} \times \omega^j_{over}$, at least one of the following two conditions holds:

   (a) Utility is signal independent: $u^j_i(\tilde{z}, \vartheta) = u^j_i(\tilde{z}, \vartheta')$ for all legal $\vartheta, \vartheta' \in \bigtimes\limits_{k=1}^{j} \Theta^{\kappa^k} \times \bigtimes\limits_{i=1}^{n} \bigtimes\limits_{k=1}^{j} \Theta^{\gamma^k}$.

   (b) $\succeq$ is defined for all legal signals $(\tilde{\alpha}^j, \tilde{\beta}^j_i)$ and $(\tilde{\alpha}^j, \tilde{\beta}'^j_i)$ through round j and a player's utility is increasing in her private signals, everything else equal:

$$\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right) \succeq \left(\tilde{\alpha}^j, \tilde{\beta}'^j_i\right) \Longrightarrow u_i\left(\tilde{z}, \tilde{\alpha}^j, \left(\tilde{\beta}^j_i, \tilde{\beta}^j_{-i}\right)\right) \geq u_i\left(\tilde{z}, \tilde{\alpha}^j, \left(\tilde{\beta'}^j_i, \tilde{\beta}^j_{-i}\right)\right).$$

We will use the term *game with ordered signals* and the term *ordered game* interchangeably.

2.1 Rhode Island Hold'em modeled as an ordered game

In this section we describe how Rhode Island Hold'em can be defined as an ordered game in accordance with Definition 2.1. First, we describe the rules of Rhode Island Hold'em.

(1) Each player pays an *ante* of 5 chips which is added to the *pot*. Both players initially receive a single card, face down; these are known as the *hole cards*.

(2) After receiving the hole cards, the players participate in one betting round. Each player may *check* (not placing any money in the pot and passing) or *bet* (placing 10 chips into the pot) if no bets have been placed. If a bet has been placed, then the player may *fold* (thus forfeiting the game along with any money they have put into the pot), *call* (adding chips to the pot equal to the last player's bet), or *raise* (calling the current bet and making an additional

Table I. Ranking of three-card poker hands, from highest to lowest.

Hand	**Prob.**	**Description**	**Example**
Straight flush	0.00217	3 cards w/ consecutive rank & same suit	K♠, Q♠, J♠
Three of a kind	0.00235	3 cards of the same rank	Q♠, Q♡, Q♣
Straight	0.03258	3 cards w/ consecutive rank	3♣, 4♠, 5♡
Flush	0.04959	3 cards of the same suit	2♢, 5♢, 8♢
Pair	0.16941	2 cards of the same rank	2♢, 2♠, 3♡
High card	0.74389	None of the above	J♣, 9♡, 2♠

bet). In Rhode Island Hold'em, the players are limited to three bets each per betting round. (A raise equals two bets.) In the first betting round, the bets are equal to 10 chips.

(3) After the first betting round, a community card is dealt face up. This is called the *flop* card. Another betting round take places at this point, with bets equal to 20 chips.

(4) Following the second betting round, another community card is dealt face up. This is called the *turn* card. A final betting round takes place at this point, with bets again equal to 20 chips.

(5) If neither player folds, then the *showdown* takes place. Both players turn over their cards. The player who has the best 3-card poker hand takes the pot. In the event of a draw, the pot is split evenly.

Hands in 3-card poker games are ranked slightly differently than 5-card poker hands. The main differences are that the order of flushes and straights are reversed, and a three of a kind is better than straights or flushes. Table I describes the rankings. Within ranks, ties are broken by by ordering hands according to the rank of cards that make up the hand. If players are still tied after applying this criterion, *kickers* are used to determine the winner. A kicker is a card that is not used to make up the hand. For example, if player 1 has a pair of eights and a five, and player 2 has a pair of eights and a six, player 2 wins.

To make the definition of ordered games concrete, here we define each of the components of the tuple $\Gamma = \langle I, G, L, \Theta, \kappa, \gamma, p, \succeq, \omega, u\rangle$ for Rhode Island Hold'em. There are two players so $I = \{1, 2\}$. There are three rounds, and the stage game is the same in each round so we have $G = \langle G_{RI}, G_{RI}, G_{RI}\rangle$ where G_{RI} is given in Figure 1, which also specifies the player label L.

Θ is the standard deck of 52 cards. The community cards are dealt in the second and third rounds, so $\kappa = \langle 0, 1, 1\rangle$. Each player receives a since face down card in the first round only, so $\gamma = \langle 1, 0, 0\rangle$. p is the uniform distribution over Θ. $\succeq$ is defined for three card hands and is defined using the ranking given in Table I. The game-ending nodes ω are denoted in Figure 1 by ω. u is defined as in the above description; it is easy to verify that it satisfies the necessary conditions.

2.2 Information filters

In this subsection, we define an *information filter* for ordered games. Instead of completely revealing a signal (either public or private) to a player, the signal first passes through this filter, which outputs a *coarsened* signal to the player. By varying

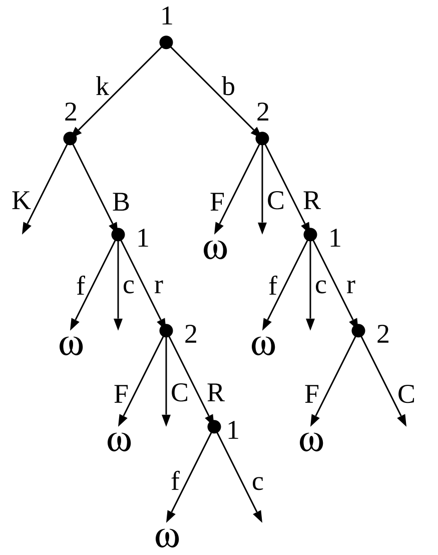

Fig. 1. Stage game G_{RI}, player label L, and game-ending nodes ω for Rhode Island Hold'em. The action labels denote which action the player is taking: k (check), b (bet), f (fold), c (call), and r (raise). Lower case letters indicate player 1 actions and upper case letters indicate player 2 actions.

the filter applied to a game, we are able to obtain a wide variety of games while keeping the underlying action space of the game intact. We will use this when designing our abstraction techniques. Formally, an information filter is as follows.

Definition 2.2. Let $\Gamma = \langle I, G, L, \Theta, \kappa, \gamma, p, \succeq, \omega, u\rangle$ be an ordered game. Let $S^j \subseteq \bigtimes_{k=1}^{j} \Theta^{\kappa^k} \times \bigtimes_{k=1}^{j} \Theta^{\gamma^k}$ be the set of legal signals (i.e., no repeated signals) for one player through round j. An *information filter* for Γ is a collection $F = \langle F^1, \ldots, F^r\rangle$ where each F^j is a function $F^j : S^j \to 2^{S^j}$ such that each of the following conditions hold:

(1) (Truthfulness) $(\tilde{\alpha}^j, \tilde{\beta}^j_i) \in F^j(\tilde{\alpha}^j, \tilde{\beta}^j_i)$ for all legal $(\tilde{\alpha}^j, \tilde{\beta}^j_i)$.
(2) (Independence) The range of F^j is a partition of S^j.
(3) (Information preservation) If two values of a signal are distinguishable in round k, then they are distinguishable fpr each round $j > k$. Let $m^j = \sum_{l=1}^{j} \kappa^l + \gamma^l$. We require that for all legal $(\theta_1, \ldots, \theta_{m^k}, \ldots, \theta_{m^j}) \subseteq \Theta$ and $(\theta'_1, \ldots, \theta'_{m^k}, \ldots, \theta'_{m^j}) \subseteq \Theta$:

$$(\theta'_1, \ldots, \theta'_{m^k}) \notin F^k(\theta_1, \ldots, \theta_{m^k}) \Longrightarrow (\theta'_1, \ldots, \theta'_{m^k}, \ldots, \theta'_{m^j}) \notin F^j(\theta_1, \ldots, \theta_{m^k}, \ldots, \theta_{m^j}).$$

A game with ordered signals Γ and an information filter F for Γ defines a new game Γ_F. We refer to such games as *filtered ordered games.* We are left with the original game if we use the identity filter $F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right) = \left\{\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right\}$. We have the following simple (but important) result:

PROPOSITION 2.3. *A filtered ordered game is an extensive form game satisfying perfect recall. (For the unfamiliar reader, the definition of games with perfect recall is given in Appendix A.)*

A simple proof proceeds by constructing an extensive form game directly from the ordered game, and showing that it satisfies perfect recall. In determining the payoffs in a game with filtered signals, we take the average over all real signals in the filtered class, weighted by the probability of each real signal occurring.

2.3 Strategies and Nash equilibrium

We are now ready to define behavior strategies in the context of filtered ordered games.

Definition 2.4. A *behavior strategy* for player i in round j of $\Gamma = \langle I, G, L, \Theta, \kappa, \gamma, p, \succeq, \omega, u \rangle$ with information filter F is a probability distribution over possible actions, and is defined for each player i, each round j, and each $v \in V^j \setminus Z^j$ for $L^j(v) = i$:

$$\sigma_{i,v}^j : \bigtimes_{k=1}^{j-1} \omega_{cont}^k \times Range\left(F^j\right) \to \Delta\left\{w \in V^j \mid (v, w) \in E^j\right\}.$$

($\Delta(X)$ is the set of probability distributions over a finite set X.) A behavior strategy for player i in round j is $\sigma_i^j = (\sigma_{i,v_1}^j, \ldots, \sigma_{i,v_m}^j)$ for each $v_k \in V^j \setminus Z^j$ where $L^j(v_k) = i$. A behavior strategy for player i in Γ is $\sigma_i = \left(\sigma_i^1, \ldots, \sigma_i^r\right)$. A *strategy profile* is $\sigma = (\sigma_1, \ldots, \sigma_n)$. A strategy profile with σ_i replaced by σ_i' is $(\sigma_i', \sigma_{-i}) = (\sigma_1, \ldots, \sigma_{i-1}, \sigma_i', \sigma_{i+1}, \ldots, \sigma_n)$.

By an abuse of notation, we will say player i receives an expected payoff of $u_i(\sigma)$ when all players are playing the strategy profile σ. Strategy σ_i is said to be player i's *best response* to σ_{-i} if for all other strategies σ_i' for player i we have $u_i(\sigma_i, \sigma_{-i}) \geq u_i(\sigma_i', \sigma_{-i})$. σ is a *Nash equilibrium* if, for every player i, σ_i is a best response for σ_{-i}. A Nash equilibrium always exists in finite extensive form games [Nash 1950], and one exists in behavior strategies for games with perfect recall [Kuhn 1953]. Using these observations, we have the following corollary to Proposition 2.3:

Corollary 2.5. *For any filtered ordered game, a Nash equilibrium exists in behavior strateges.*

3. EQUILIBRIUM-PRESERVING ABSTRACTIONS

In this section, we present our main technique for reducing the size of games. We begin by defining a *filtered signal tree* which represents all of the chance moves in the game. The bold edges (i.e. the first two levels of the tree) in the game trees in Figure 2 correspond to the filtered signal trees in each game.

Definition 3.1. Associated with every ordered game $\Gamma = \langle I, G, L, \Theta, \kappa, \gamma, p, \succeq, \omega, u \rangle$ and information filter F is a *filtered signal tree*, a directed tree in which each node corresponds to some revealed (filtered) signals and edges correspond to revealing specific (filtered) signals. The nodes in the filtered signal tree represent the set of all possible revealed filtered signals (public and private) at some point in time. The filtered public signals revealed in round j correspond to the nodes in the κ^j levels beginning at level $\sum_{k=1}^{j-1}\left(\kappa^k + n\gamma^k\right)$ and the private signals revealed in round j correspond to the nodes in the $n\gamma^j$ levels beginning at level $\sum_{k=1}^{j} \kappa^k + \sum_{k=1}^{j-1} n\gamma^k$. We denote children of a node x as $N(x)$. In addition, we associate weights with the

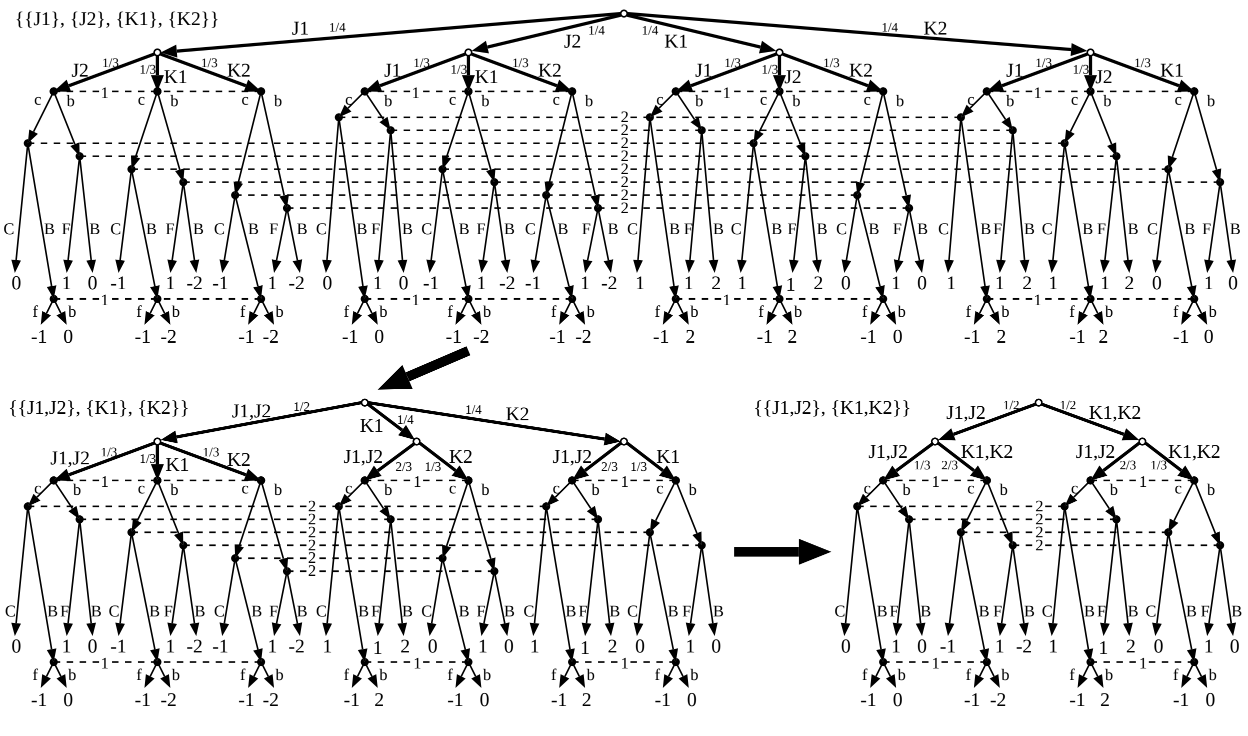

Fig. 2. *GameShrink* applied to a tiny two-person four-card (two Jacks and two Kings) poker game. Next to each game tree is the range of the information filter F. Dotted lines denote information sets, which are labeled by the controlling player. Open circles are chance nodes with the indicated transition probabilities. The root node is the chance node for player 1's card, and the next level is for player 2's card. The payment from player 2 to player 1 is given below each leaf. In this example, the algorithm reduces the game tree from 113 nodes to 39 nodes.

edges corresponding to the probability of the particular edge being chosen given that its parent was reached.

In many games, there are certain situations in the game that can be thought of as being strategically equivalent to other situations in the game. By melding these situations together, it is possible to arrive at a strategically equivalent smaller game. The next two definitions formalize this notion via the introduction of the *ordered game isomorphic* relation and the *ordered game isomorphic abstraction transformation.*

Definition 3.2. Two subtrees beginning at internal nodes x and y of a filtered signal tree are *ordered game isomorphic* if x and y have the same parent and there is a bijection $f : N(x) \to N(y)$, such that for $w \in N(x)$ and $v \in N(y)$, $v = f(w)$ implies the weights on the edges (x, w) and (y, v) are the same and the subtrees beginning at w and v are ordered game isomorphic. Two leaves (corresponding to filtered signals ϑ and ϑ' up through round r) are ordered game isomorphic if for all $\tilde{z} \in \bigtimes_{j=1}^{r-1} \omega_{cont}^{j} \times \omega_{over}^{r}$, $u^r(\tilde{z}, \vartheta) = u^r(\tilde{z}, \vartheta')$.

Definition 3.3. Let $\Gamma = \langle I, G, L, \Theta, \kappa, \gamma, p, \succeq, \omega, u \rangle$ be an ordered game and let F be an information filter for Γ. Let ϑ and ϑ' be two information structures where the subtrees in the induced filtered signal tree corresponding to the nodes ϑ and ϑ' are ordered game isomorphic, and ϑ and ϑ' are at either level $\sum_{k=1}^{j-1} \left(\kappa^k + n\gamma^k\right)$ or $\sum_{k=1}^{j} \kappa^k + \sum_{k=1}^{j-1} n\gamma^k$ for some round j. The *ordered game isomorphic abstraction transformation* is given by creating a new information filter F':

$$F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right) = \begin{cases} F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right) & \text{if } \left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right) \notin \vartheta \cup \vartheta' \\ \vartheta \cup \vartheta' & \text{if } \left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right) \in \vartheta \cup \vartheta'. \end{cases}$$

Figure 2 shows the ordered game isomorphic abstraction transformation applied twice to a tiny poker game. Theorem 3.4, our main equilibrium result, shows how the ordered game isomorphic abstraction transformation can be used to compute equilibria faster.

THEOREM 3.4. *Let* $\Gamma = \langle I, G, L, \Theta, \kappa, \gamma, p, \succeq, \omega, u \rangle$ *be an ordered game and* F *be an information filter for* Γ*. Let* F' *be an information filter constructed from* F *by one application of the ordered game isomorphic abstraction transformation. Let* σ' *be a Nash equilibrium of the induced game* $\Gamma_{F'}$*. If we take* $\sigma_{i,v}^j\left(\tilde{z}, F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)\right) = \sigma_{i,v}'^j\left(\tilde{z}, F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)\right)$*,* σ *is a Nash equilibrium of* Γ_F*.*

PROOF. For an extensive form game, a *belief system* μ assigns a probability to every decision node x such that $\sum_{x \in h} \mu(x) = 1$ for every information set h. A strategy profile σ is *sequentially rational at* h *given belief system* μ if

$$u_i(\sigma_i, \sigma_{-i} \,|\, h, \mu) \geq u_i(\tau_i, \sigma_{-i} \,|\, h, \mu)$$

for all other strategies τ_i, where i is the player who controls h. A basic result [Mas-Colell et al. 1995, Proposition 9.C.1] characterizing Nash equilibria dictates that σ is a Nash equilibrium if and only if there is a belief system μ such that for every information set h with $\Pr(h \,|\, \sigma) > 0$, the following two conditions hold: (C1) σ is sequentially rational at h given μ; and (C2) $\mu(x) = \frac{\Pr(x \,|\, \sigma)}{\Pr(h \,|\, \sigma)}$ for all $x \in h$. Since σ' is a Nash equilibrium of Γ', there exists such a belief system μ' for $\Gamma_{F'}$. Using μ', we will construct a belief system μ for Γ and show that conditions C1 and C2 hold, thus supporting σ as a Nash equilibrium.

Fix some player $i \in I$. Each of i's information sets in some round j corresponds to filtered signals $F^j\left(\tilde{\alpha}^{*j}, \tilde{\beta}_i^{*j}\right)$, history in the first $j-1$ rounds $(z_1, \ldots, z_{j-1}) \in \bigtimes_{k=1}^{j-1} \omega_{cont}^k$, and history so far in round j, $v \in V^j \setminus Z^j$. Let $\tilde{z} = (z_1, \ldots, z_{j-1}, v)$ represent all of the player actions leading to this information set. Thus, we can uniquely specify this information set using the information $\left(F^j\left(\tilde{\alpha}^{*j}, \tilde{\beta}_i^{*j}\right), \tilde{z}\right)$.

Each node in an information set corresponds to the possible private signals the other players have received. Denote by $\tilde{\beta}$ some legal

$$(F^j(\tilde{\alpha}^j, \tilde{\beta}_1^j), \ldots, F^j(\tilde{\alpha}^j, \tilde{\beta}_{i-1}^j), F^j(\tilde{\alpha}^j, \tilde{\beta}_{i+1}^j), \ldots, F^j(\tilde{\alpha}^j, \tilde{\beta}_n^j)).$$

In other words, there exists $(\tilde{\alpha}^j, \tilde{\beta}_1^j, \ldots, \tilde{\beta}_n^j)$ such that $(\tilde{\alpha}^j, \tilde{\beta}_i^j) \in F^j(\tilde{\alpha}^{*j}, \tilde{\beta}_i^{*j})$, $(\tilde{\alpha}^j, \tilde{\beta}_k^j) \in F^j(\tilde{\alpha}^j, \tilde{\beta}_k^j)$ for $k \neq i$, and no signals are repeated. Using such a set of signals $(\tilde{\alpha}^j, \tilde{\beta}^j_1, \ldots, \tilde{\beta}^j_n)$, let $\hat{\beta}'$ denote $(F'^j(\tilde{\alpha}^j, \tilde{\beta}^j_1), \ldots, F'^j(\tilde{\alpha}^j, \tilde{\beta}^j_{i-1}), F'^j(\tilde{\alpha}^j, \tilde{\beta}^j_{i+1}), \ldots, F'^j(\tilde{\alpha}^j, \tilde{\beta}^j_n))$. (We will abuse notation and write $F'^j_{-i}\left(\hat{\beta}\right) = \hat{\beta}'$.) We can now compute μ directly from μ':

$$\mu\left(\hat{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) = \begin{cases} \mu'\left(\hat{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) \\ \quad \text{if } F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right) \neq F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right) \text{ or } \hat{\beta} = \hat{\beta}' \\ p^*\mu'\left(\hat{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) \\ \quad \text{if } F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right) = F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right) \text{ and } \hat{\beta} \neq \hat{\beta}' \end{cases}$$

where $p^* = \frac{\Pr\left(\hat{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)}{\Pr\left(\hat{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)}$. The following three claims show that μ as calculated above supports σ as a Nash equilibrium.

CLAIM 3.5. *μ is a valid belief system for Γ_F.*

PROOF OF CLAIM 3.5. Let h be player i's information set after some history $\left(F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right)$. Clearly $\mu\left(\hat{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) \geq 0$ for all $\hat{\beta} \in h$. We need to show

$$\sum_{\hat{\beta} \in h} \mu\left(\hat{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) = 1.$$

CASE 1. $F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right) \neq F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)$. From the construction of F', $F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)$ is ordered game isomorphic to some $F^j\left(\tilde{\alpha}'^j \tilde{\beta}'^j_i\right)$ with $F^j\left(\tilde{\alpha}'^j \tilde{\beta}'^j_i\right) \neq F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)$. Let h' be player i's information set corresponding to the history $\left(F^j\left(\tilde{\alpha}'^j, \tilde{\beta}'^j_i\right), \tilde{z}\right)$. By the definition of the ordered game isomorphism, there exists a perfect matching between the nodes in the information set h and h', where each matched pair of nodes corresponds to a pair of ordered game isomorphic information structures. Since we have that $F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right) = F'^j\left(\tilde{\alpha}'^j, \tilde{\beta}'^j_i\right)$, each edge in the matching corresponds to a node in the information set corresponding to the history $\left(F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right)$ in $\Gamma_{F'}$; denote this information set by h''. (See Figure 3.)

Thus, there is a bijection between h and h'' defined by the perfect matching. Using this matching:

$$\begin{aligned} \sum_{\hat{\beta} \in h} \mu\left(\hat{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) &= \sum_{\hat{\beta} \in h} \mu'\left(F'^j_{-i}\left(\hat{\beta}\right) \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) \\ &= \sum_{\hat{\beta}' \in h''} \mu'\left(\hat{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) \\ &= 1. \end{aligned}$$

CASE 2. $F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right) = F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)$. We need to treat members of h differently depending on if they map to the same set of signals in $\Gamma_{F'}$ or not. Let $h_1 = \left\{\hat{\beta} \in h \mid \hat{\beta} = F'^j_{-i}\left(\hat{\beta}\right)\right\}$ and let $h_2 = \left\{\hat{\beta} \in h \mid \hat{\beta} \subset F'^j_{-i}\left(\hat{\beta}\right)\right\}$. Clearly (h_1, h_2)

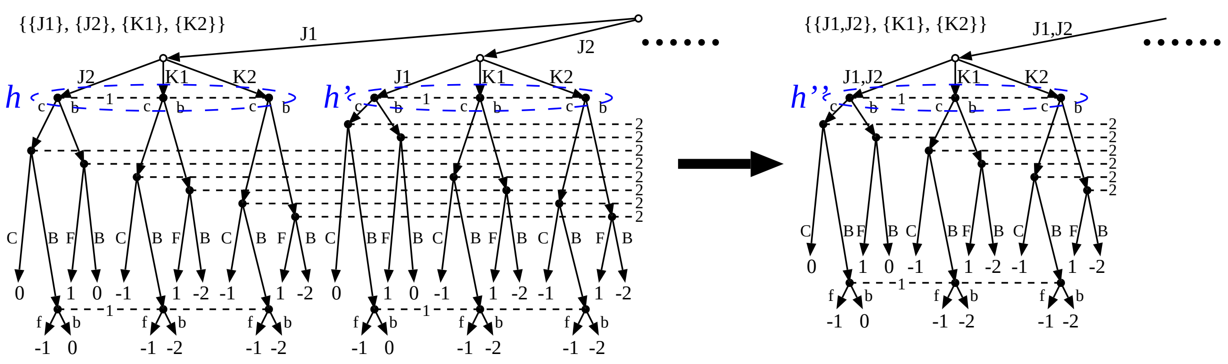

Fig. 3. Illustration of Case 1 of Claim 3.5.

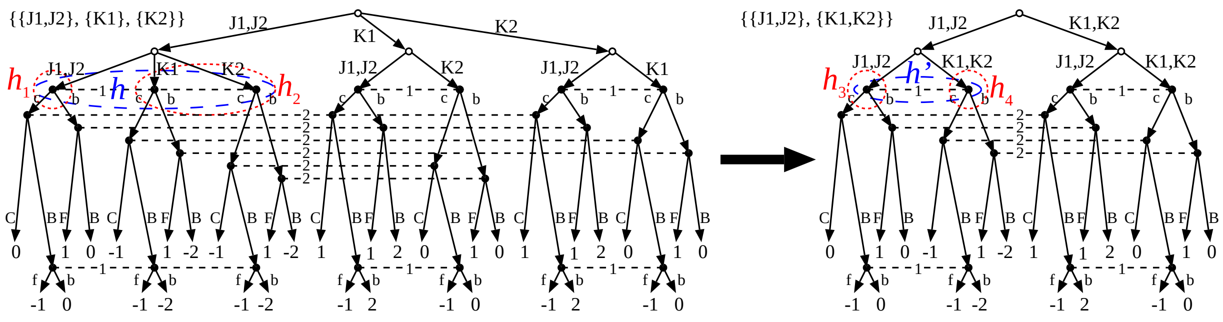

Fig. 4. Illustration of Case 2 of Claim 3.5.

is a partition of h. Let h' be player i's information set corresponding to the history $\left(F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}\right)$ in $\Gamma_{F'}$. We can create a partition of h' by letting $h_3 = \left\{F'^j_{-i}\left(\hat{\beta}\right) \mid \hat{\beta} \in h_1\right\}$ and $h_4 = \left\{F'^j_{-i}\left(\hat{\beta}\right) \mid \hat{\beta} \in h_2\right\}$. Cleary (h_3, h_4) partitions h'. (See Figure 4.) The rest of the proof for this case proceeds in three steps.
Step 1. In this step we show the following relationship between h_1 and h_3:

$$\begin{aligned}\sum_{\hat{\beta}\in h_1} \mu\left(\hat{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}\right) &= \sum_{\hat{\beta}\in h_1} \mu'\left(F'^j_{-i}\left(\hat{\beta}\right) \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}\right) \\ &= \sum_{\hat{\beta}'\in h_3} \mu'\left(\hat{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}\right) \qquad (1)\end{aligned}$$

Step 2. In this step we want to show a similar relationship between h_2 and h_4. In doing so, we use the following fact: $\hat{\beta} \subset \hat{\beta}' \rightarrow F'^j_{-i}\left(\hat{\beta}\right) = \hat{\beta}'$. With this in mind, we can write:

$$\begin{aligned}
\sum_{\hat{\beta}\in h_2} \mu\left(\hat{\beta}|F^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right),\tilde{z}\right) &= \sum_{\hat{\beta}\in h_2} \frac{\Pr\left(\hat{\beta}|F^j(\tilde{\alpha}^j,\tilde{\beta}^j_i)\right)}{\Pr\left(F'^j_{-i}(\hat{\beta})|F'^j(\tilde{\alpha}^j,\tilde{\beta}^j_i)\right)}\mu'\left(F'^j_{-i}(\hat{\beta})|F'^j(\tilde{\alpha}^j,\tilde{\beta}^j_i),\tilde{z}\right)\\
&= \sum_{\hat{\beta}'\in h_4}\sum_{\substack{\hat{\beta}\in h_2\\ \hat{\beta}\subset\hat{\beta}'}} \frac{\Pr\left(\hat{\beta}|F^j(\tilde{\alpha}^j,\tilde{\beta}^j_i)\right)}{\Pr\left(F'^j_{-i}(\hat{\beta})|F'^j(\tilde{\alpha}^j,\tilde{\beta}^j_i)\right)}\cdot\mu'\left(F'^j_{-i}(\hat{\beta})|F'^j(\tilde{\alpha}^j,\tilde{\beta}^j_i),\tilde{z}\right)\\
&= \sum_{\hat{\beta}'\in h_4}\sum_{\substack{\hat{\beta}\in h_2\\ \hat{\beta}\subset\hat{\beta}'}} \frac{\Pr\left(\hat{\beta}|F^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right)\right)}{\Pr\left(\hat{\beta}'|F^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right)\right)}\mu'\left(\hat{\beta}'|F'^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right),\tilde{z}\right)\\
&= \sum_{\hat{\beta}'\in h_4}\mu'\left(\hat{\beta}'|F'^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right),\tilde{z}\right)\sum_{\substack{\hat{\beta}\in h_2\\ \hat{\beta}\subset\hat{\beta}'}} \frac{\Pr\left(\hat{\beta}|F^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right)\right)}{\Pr\left(\hat{\beta}'|F^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right)\right)}\\
&= \sum_{\hat{\beta}'\in h_4}\mu'\left(\hat{\beta}'|F'^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right),\tilde{z}\right) \qquad (2)
\end{aligned}$$

STEP 3. Using (1) and (2):

$$\begin{aligned}
\sum_{\hat{\beta}\in h}\mu\left(\hat{\beta}\mid F^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right),\tilde{z}\right) &= \sum_{\hat{\beta}\in h_1}\mu\left(\hat{\beta}\mid F^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right),\tilde{z}\right)+\sum_{\hat{\beta}\in h_2}\mu\left(\hat{\beta}\mid F^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right),\tilde{z}\right)\\
&= \sum_{\hat{\beta}'\in h_3}\mu'\left(\hat{\beta}'\mid F'^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right),\tilde{z}\right)+\sum_{\hat{\beta}'\in h_4}\mu'\left(\hat{\beta}'\mid F'^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right),\tilde{z}\right)\\
&= \sum_{\hat{\beta}'\in h'}\mu'\left(\hat{\beta}'\mid F'^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right),\tilde{z}\right)\\
&= 1
\end{aligned}$$

In both cases we have shown $\sum_{\hat{\beta}\in h}\mu\left(\hat{\beta}\mid F^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right),\tilde{z}\right)=1.$ □

CLAIM 3.6. *For all information sets h with $\Pr(h\mid\sigma)>0$, $\mu(x)=\frac{\Pr(x\mid\sigma)}{\Pr(h\mid\sigma)}$ for all $x\in h$.*

PROOF OF CLAIM 3.6. Let h be player i's information set after some history $\left(F^j\left(\tilde{\alpha}^j,\tilde{\beta}^j_i\right),\tilde{z}\right)$, and fix some $\hat{\beta}\in h$. Let $\hat{\beta}'=F'^j_{-i}\left(\hat{\beta}\right)$. We need to show that $\mu(\hat{\beta}|F^j(\tilde{\alpha}^j,\tilde{\beta}^j_i),\tilde{z})=\frac{\Pr(\hat{\beta}\mid\sigma)}{\Pr(h\mid\sigma)}$. Let h' be player i's information set after history $(F'^j(\tilde{\alpha}^j,\tilde{\beta}^j_i),\tilde{z})$.

CASE 1. $F^j(\tilde{\alpha}^j, \tilde{\beta}^j_i) \neq F'^j(\tilde{\alpha}^j, \tilde{\beta}^j_i)$.

$$\begin{aligned}
\mu\left(\hat{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) &= \mu'\left(\hat{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) \\
&= \frac{\Pr\left(\hat{\beta}' \mid \sigma'\right)}{\Pr\left(h' \mid \sigma'\right)} \\
&= \frac{\frac{\Pr\left(\hat{\beta}, F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)}{\Pr\left(\hat{\beta}', F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)} \Pr\left(\hat{\beta}' \mid \sigma'\right)}{\frac{\Pr\left(\hat{\beta}, F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)}{\Pr\left(\hat{\beta}', F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)} \Pr\left(h' \mid \sigma'\right)} \\
&= \frac{\Pr\left(\hat{\beta} \mid \sigma\right)}{\Pr\left(h \mid \sigma\right)}
\end{aligned}$$

CASE 2. $F^j(\tilde{\alpha}^j, \tilde{\beta}^j_i) = F'^j(\tilde{\alpha}^j, \tilde{\beta}^j_i)$ and $\hat{\beta} \neq \hat{\beta}'$.

$$\begin{aligned}
\mu\left(\hat{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) &= \frac{\Pr\left(\tilde{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)}{\Pr\left(\tilde{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)} \mu'\left(\hat{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) \\
&= \frac{\Pr\left(\tilde{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)}{\Pr\left(\tilde{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)} \frac{\Pr\left(\hat{\beta}' \mid \sigma'\right)}{\Pr\left(h' \mid \sigma'\right)} \\
&= \frac{\Pr\left(\tilde{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)}{\Pr\left(\tilde{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)} \frac{\frac{\Pr\left(\tilde{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)}{\Pr\left(\tilde{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right)\right)} \Pr\left(\hat{\beta} \mid \sigma\right)}{\Pr\left(h \mid \sigma\right)} \\
&= \frac{\Pr\left(\hat{\beta} \mid \sigma\right)}{\Pr\left(h \mid \sigma\right)}
\end{aligned}$$

CASE 3. $F^j(\tilde{\alpha}^j, \tilde{\beta}^j_i) = F'^j(\tilde{\alpha}^j, \tilde{\beta}^j_i)$ and $\hat{\beta} = \hat{\beta}'$.

$$\begin{aligned}
\mu\left(\hat{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) &= \mu'\left(\hat{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}^j_i\right), \tilde{z}\right) \\
&= \frac{\Pr\left(\hat{\beta}' \mid \sigma'\right)}{\Pr\left(h' \mid \sigma'\right)} \\
&= \frac{\Pr\left(\hat{\beta} \mid \sigma\right)}{\Pr\left(h \mid \sigma\right)}
\end{aligned}$$

Thus we have $\mu(x) = \frac{\Pr(x \mid \sigma)}{\Pr(h \mid \sigma)}$ for all information sets h with $\Pr(h \mid \sigma) > 0$. $\square$

CLAIM 3.7. *For all information sets h with $\Pr(h \mid \sigma) > 0$, σ is sequentially rational at h given μ.*

PROOF OF CLAIM 3.7. Suppose, by way of contradiction, that σ is not sequentially rational given μ. Then, there exists a strategy τ_i such that, for some $(F^j(\tilde{\alpha}^j, \tilde{\beta}^j_i), \tilde{z})$,

$$u^j_i(\tau_i, \sigma_{-i} | F^j \tilde{\alpha}^j, \tilde{\beta}^j_i), \tilde{z}, \mu) > u^j_i(\sigma_i, \sigma_{-i} | F^j(\tilde{\alpha}^j, \tilde{\beta}^j_i), \tilde{z}, \mu). \qquad (3)$$

We will construct a strategy τ'_i for player i in $\Gamma_{F'}$ such that

$$u_i^j(\tau'_i, \sigma'_{-i}|F'^j(\tilde{\alpha}^j, \tilde{\beta}_i^j), \tilde{z}, \mu') > u_i^j(\sigma'_i, \sigma'_{-i}|F'^j(\tilde{\alpha}^j, \tilde{\beta}_i^j), \tilde{z}, \mu'),$$

thus contradicting the fact that σ' is a Nash equilibrium. The proof proceeds in four steps.

Step 1. We first construct τ'_i from τ_i. For a given $F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)$, let

$$\Upsilon = \left\{F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right) \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right) \subseteq F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)\right\} \tag{4}$$

and let

$$\tau'^j_{i,v}(F'^j(\tilde{\alpha}^j, \tilde{\beta}_i^j), \tilde{z}) = \sum_{\vartheta \in \Upsilon} \Pr\left(\vartheta \mid F'^j(\tilde{\alpha}^j, \tilde{\beta}_i^j)\right) \tau^j_{i,v}\left(\vartheta, \tilde{z}\right).$$

In other words, the strategy τ'_i is the same as τ_i except in situations where only the filtered signal history is different, in which case τ'_i is a weighted average over the strategies at the corresponding information sets in Γ_F.

Step 2. We need to show that $u_i^j(\tau'_i, \sigma'_{-i} \mid F'^j(\tilde{\alpha}^j, \tilde{\beta}_i^j), \tilde{z}, \mu') = u_i^j(\tau_i, \sigma_{-i} \mid F^j(\tilde{\alpha}^j, \tilde{\beta}_i^j), \tilde{z}, \mu)$ for all histories $(F^j(\tilde{\alpha}^j, \tilde{\beta}_i^j), \tilde{z})$. Fix $(F^j(\tilde{\alpha}^j, \tilde{\beta}_i^j), \tilde{z})$, and assume, without loss of generality, the equality holds for all information sets coming after this one in Γ.

Case 1. $F^j(\tilde{\alpha}^j, \tilde{\beta}_i^j) \neq F'^j(\tilde{\alpha}^j, \tilde{\beta}_i^j)$. Let z^j denote the current node of G^j and let Υ as in (4).

$$
\begin{aligned}
&u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \mu'\right) = \sum_{\hat{\beta}' \in h'} \mu'\left(\hat{\beta}'\right) u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}'\right) \\
&= \sum_{\hat{\beta} \in h} \mu'\left(F_{-i}'^j\left(\hat{\beta}\right)\right) u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, F_{-i}'^j\left(\hat{\beta}\right)\right) \\
&= \sum_{\hat{\beta} \in h} \mu\left(\hat{\beta}\right) u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, F_{-i}'^j\left(\hat{\beta}\right)\right) \\
&= \sum_{\hat{\beta} \in h} \mu\left(\hat{\beta}\right) \sum_{v \in N^j(z^j)} \tau_{i,v}'^j\left(\tilde{z}, F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)\right) \cdot u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), (\tilde{z}, v), F_{-i}'^j\left(\hat{\beta}\right)\right) \\
&= \sum_{\hat{\beta} \in h} \mu\left(\hat{\beta}\right) \sum_{v \in N^j(z^j)} \sum_{\vartheta \in \Upsilon} \Pr\left(\vartheta \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)\right) \tau_{i,v}^j\left(\tilde{z}, \vartheta\right) \cdot \\
&\qquad\qquad \left[u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), (\tilde{z}, v), F_{-i}'^j\left(\hat{\beta}\right)\right)\right] \\
&= \sum_{\hat{\beta} \in h} \mu\left(\hat{\beta}\right) \sum_{v \in N^j(z^j)} \sum_{\vartheta \in \Upsilon} \Pr\left(\vartheta \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)\right) \tau_{i,v}^j\left(\tilde{z}, \vartheta\right) \cdot \\
&\qquad\qquad \left[u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), (\tilde{z}, v), \hat{\beta}\right)\right] \\
&= \sum_{\hat{\beta} \in h} \mu\left(\hat{\beta}\right) \sum_{v \in N^j(z^j)} u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), (\tilde{z}, v), \hat{\beta}\right) \cdot \\
&\qquad\qquad \left[\sum_{\vartheta \in \Upsilon} \Pr\left(\vartheta \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)\right) \tau_{i,v}^j\left(\tilde{z}, \vartheta\right)\right] \\
&= \sum_{\hat{\beta} \in h} \mu\left(\hat{\beta}\right) \sum_{v \in N^j(z^j)} \tau_{i,v}^j\left(\tilde{z}, F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)\right) \cdot u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), (\tilde{z}, v), \hat{\beta}\right) \\
&= \sum_{\hat{\beta} \in h} \mu\left(\hat{\beta}\right) u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}\right) \\
&= u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \mu\right)
\end{aligned}
$$

CASE 2. $F^j(\tilde{\alpha}^j, \tilde{\beta}_i^j) = F'^j(\tilde{\alpha}^j, \tilde{\beta}_i^j)$. Let h_1, h_2, h_3, and h_4 as in the proof of Case 2 of Claim 3.5. We can show

$$
\sum_{\hat{\beta}' \in h_3} \mu'\left(\hat{\beta}'\right) u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}'\right) = \sum_{\hat{\beta} \in h_1} \mu\left(\hat{\beta}\right) u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}\right) \tag{5}
$$

using a procedure similar to that in Case 1. We can show the following relationship between h_2 and h_4:

$$\begin{aligned}
&\sum_{\hat{\beta}' \in h_4} \mu'\left(\hat{\beta}'\right) u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}'\right) \\
&\quad = \sum_{\hat{\beta}' \in h_4} \sum_{\substack{\hat{\beta} \in h_2 \\ \hat{\beta} \subset \hat{\beta}'}} \frac{\Pr\left(\hat{\beta} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)\right)}{\Pr\left(\hat{\beta}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)\right)} \cdot \mu'\left(\hat{\beta}'\right) u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}'\right) \\
&\quad = \sum_{\hat{\beta}' \in h_4} \sum_{\substack{\hat{\beta} \in h_2 \\ \hat{\beta} \subset \hat{\beta}'}} \mu\left(\hat{\beta}\right) u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}'\right) \\
&\quad = \sum_{\hat{\beta}' \in h_4} \sum_{\substack{\hat{\beta} \in h_2 \\ \hat{\beta} \subset \hat{\beta}'}} \mu\left(\hat{\beta}\right) \sum_{v \in N^j(z^j)} \tau'^j_{i,v}\left(\tilde{z}, F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)\right) \cdot u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), (\tilde{z}, v), \hat{\beta}'\right) \\
&\quad = \sum_{\hat{\beta}' \in h_4} \sum_{\substack{\hat{\beta} \in h_2 \\ \hat{\beta} \subset \hat{\beta}'}} \mu\left(\hat{\beta}\right) \sum_{v \in N^j(z^j)} \tau^j_{i,v}\left(\tilde{z}, F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right)\right) \cdot u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), (\tilde{z}, v), \hat{\beta}\right) \\
&\quad = \sum_{\hat{\beta}' \in h_4} \sum_{\substack{\hat{\beta} \in h_2 \\ \hat{\beta} \subset \hat{\beta}'}} \mu\left(\hat{\beta}\right) u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}\right) \\
&\quad = \sum_{\hat{\beta} \in h_2} \mu\left(\hat{\beta}\right) u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}\right) \qquad (6)
\end{aligned}$$

Using (5) and (6):

$$\begin{aligned}
&u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \mu'\right) = \sum_{\tilde{\beta}' \in h'} \mu'\left(\tilde{\beta}'\right) u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \tilde{\beta}'\right) \\
&= \sum_{\hat{\beta}' \in h_3} \mu'\left(\hat{\beta}'\right) u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}'\right) + \sum_{\hat{\beta}' \in h_4} \mu'\left(\hat{\beta}'\right) u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}'\right) \\
&= \sum_{\hat{\beta} \in h_1} \mu\left(\hat{\beta}\right) u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}\right) + \sum_{\hat{\beta} \in h_2} \mu\left(\hat{\beta}\right) u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}\right) \\
&= \sum_{\hat{\beta} \in h} \mu\left(\hat{\beta}\right) u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \hat{\beta}\right) \\
&= u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \mu\right)
\end{aligned}$$

In both cases we have shown:

$$u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \mu'\right) = u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \mu\right). \qquad (7)$$

STEP 3. We can show that

$$u_i^j\left(\sigma_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \mu\right) = u_i^j\left(\sigma_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \mu'\right). \qquad (8)$$

using a procedure similar to the previous step.

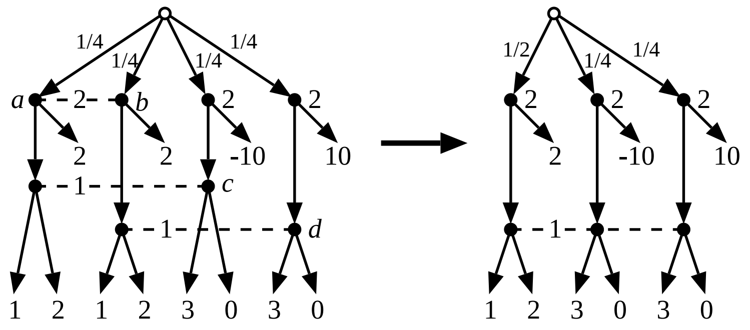

Fig. 5. Example illustrating difficulty in developing a theory of equilibrium-preserving abstractions for general extensive form games.

Step 4. Combining (3), (7), and (8), we have:

$$u_i^j\left(\tau_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \mu'\right) = u_i^j\left(\tau_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \mu\right)$$
$$> u_i^j\left(\sigma_i, \sigma_{-i} \mid F^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \mu\right) = u_i^j\left(\sigma_i', \sigma_{-i}' \mid F'^j\left(\tilde{\alpha}^j, \tilde{\beta}_i^j\right), \tilde{z}, \mu'\right).$$

Thus, σ' is not a Nash equilibrium. Therefore, by contradiction, σ is sequentially rational at all information sets h with $\Pr(h \mid \sigma) > 0$. $\square$

We can now complete the proof of Theorem 3.4. By Claims 3.5 and 3.6, we know that condition C2 holds. By Claim 3.7, we know that condition C1 holds. Thus, σ is a Nash equilibrium.

3.1 Nontriviality of generalizing beyond this model

Our model does not capture general sequential games of imperfect information because it is restricted in two ways (as discussed above): 1) there is a special structure connecting the player actions and the chance actions (for one, the players are assumed to observe each others' actions, but nature's actions might not be publicly observable), and 2) there is a common ordering of signals. In this subsection we show that removing either of these conditions can make our technique invalid.

First, we demonstrate a failure when removing the first assumption. Consider the game in Figure 5.[7] Nodes a and b are in the same information set, have the same parent (chance) node, have isomorphic subtrees with the same payoffs, and nodes c and d also have similar structural properties. By merging the subtrees beginning at a and b, we get the game on the right in Figure 5. In this game, player 1's only Nash equilibrium strategy is to play left. But in the original game, player 1 knows that node c will never be reached, and so should play right in that information set.

Removing the second assumption (that the utility functions are based on a common ordering of signals) can also cause failure. Consider a simple three-card game with a deck containing two Jacks (J1 and J2) and a King (K), where player 1's utility function is based on the ordering K $\succeq$ J1 $\sim$ J2 but player 2's utility function

[7]We thank Albert Xin Jiang for providing this example.

is based on the ordering J2 $\succeq$ K $\succeq$ J1. It is easy to check that in the abstracted game (where Player 1 treats J1 and J2 as being "equivalent") the Nash equilibrium does not correspond to a Nash equilibrium in the original game.[8]

4. GAMESHRINK: AN EFFICIENT ALGORITHM FOR COMPUTING ORDERED GAME ISOMORPHIC ABSTRACTION TRANSFORMATIONS

In this section we present an algorithm, *GameShrink*, for conducting the abstractions. The algorithm only needs to analyze the signal tree discussed above, rather than the entire game tree.

We first present a subroutine that *GameShrink* uses. It is a dynamic program for computing the ordered game isomorphic relation.[9] Again, it operates on the signal tree.

```
ALGORITHM 1. OrderedGameIsomorphic? $(\Gamma, \vartheta, \vartheta')$
(1) If $\vartheta$ and $\vartheta'$ are both leaves of the signal tree:
    (a) If $u^r(\vartheta \mid \tilde{z}) = u^r(\vartheta' \mid \tilde{z})$ for all $\tilde{z} \in \bigtimes_{j=1}^{r-1} \omega_{cont}^j \times \omega_{over}^r$, then return true.
    (b) Otherwise, return false.
(2) Create a bipartite graph $G_{\vartheta,\vartheta'} = (V_1, V_2, E)$ with $V_1 = N(\vartheta)$ and $V_2 = N(\vartheta')$.
(3) For each $v_1 \in V_1$ and $v_2 \in V_2$:
        If OrderedGameIsomorphic? $(\Gamma, v_1, v_2)$
            Create edge $(v_1, v_2)$
(4) Return true if $G_{\vartheta,\vartheta'}$ has a perfect matching; otherwise, return false.
```

By evaluating this dynamic program from bottom to top, Algorithm 1 determines, in time polynomial in the size of the signal tree, whether or not any pair of equal depth nodes x and y are ordered game isomorphic. The test in step 1(a) can be computed in $O(1)$ time by consulting the $\succeq$ relation from the specification of the game. Each call to *OrderedGameIsomorphic?* performs at most one perfect matching computation on a bipartite graph with $O(|\Theta|)$ nodes and $O(|\Theta|^2)$ edges (recall that Θ is the set of signals). Using the Ford-Fulkerson algorithm [Ford, Jr. and Fulkerson 1962] for finding a maximal matching, this takes $O(|\Theta|^3)$ time. Let S be the maximum number of signals possibly revealed in the game (e.g., in Rhode Island Hold'em, $S = 4$ because each of the two players has one card in the hand plus there are two cards on the table). The number of nodes, n, in the signal tree is $O(|\Theta|^S)$. The dynamic program visits each node in the signal tree, with each visit requiring $O(|\Theta|^2)$ calls to the *OrderedGameIsomorphic?* routine. So, it takes $O(|\Theta|^S|\Theta|^3|\Theta|^2) = O(|\Theta|^{S+5})$ time to compute the entire ordered game isomorphic relation.

While this is exponential in the number of revealed signals, we now show that it is polynomial in the size of the signal tree—and thus polynomial in the size of the

[8]We thank an anonymous person for providing this example.

[9]Actually, this is computing a slightly relaxed notion since it allows nodes with different parents to be considered ordered game isomorphic. However, the *GameShrink* algorithm only calls it with sibling nodes as the arguments.

game tree because the signal tree is smaller than the game tree. The number of nodes in the signal tree is

$$n = 1 + \sum_{i=1}^{S} \prod_{j=1}^{i} (|\Theta| - j + 1)$$

(Each term in the summation corresponds to the number of nodes at a specific depth of the tree.) The number of leaves is

$$\prod_{j=1}^{S} (|\Theta| - j + 1) = \binom{|\Theta|}{S} S!$$

which is a lower bound on the number of nodes.[10] For large $|\Theta|$ we can use the relation $\binom{n}{k} \sim \frac{n^k}{k!}$ to get

$$\binom{|\Theta|}{S} S! \sim \left(\frac{|\Theta|^S}{S!}\right) S! = |\Theta|^S$$

and thus the number of leaves in the signal tree is $\Omega(|\Theta|^S)$. Therefore, $O(|\Theta|^{S+5}) = O(n|\Theta|^5)$, which proves that we can indeed compute the ordered game isomorphic relation in time polynomial in the number of nodes, n, of the signal tree.

The algorithm often runs in *sublinear* time (and space) in the size of the game tree because the signal tree is significantly smaller than the game tree in most nontrivial games. (Note that the input to the algorithm is not an explicit game tree, but a specification of the rules, so the algorithm does not need to read in the game tree.) In general, if an ordered game has r rounds, and each round's stage game has at least b nonterminal leaves, then the size of the signal tree is at most $\frac{1}{b^r}$ of the size of the game tree. For example, in Rhode Island Hold'em, the game tree has 3.1 billion nodes while the signal tree only has 6,632,705.

Given the *OrderedGameIsomorphic?* routine for determining ordered game isomorphisms in an ordered game, we are ready to present the main algorithm, *GameShrink*.

[10]Using the inequality $\binom{n}{k} \geq \left(\frac{n}{k}\right)^k$, we get the lower bound $\binom{|\Theta|}{S} S! \geq \left(\frac{|\Theta|}{S}\right)^S S! = |\Theta|^S \frac{S!}{S^S}$.

Algorithm 2. *GameShrink* (Γ)

```
(1) Initialize $F$ to be the identity filter for $\Gamma$.
(2) For $j$ from 1 to $r$:
      For each pair of sibling nodes $\vartheta, \vartheta'$ at either level $\sum_{k=1}^{j-1}\left(\kappa^k + n\gamma^k\right)$ or $\sum_{k=1}^{j}\kappa^k + \sum_{k=1}^{j-1} n\gamma^k$ in the filtered (according to $F$) signal tree:
          If $OrderedGameIsomorphic?(\Gamma, \vartheta, \vartheta')$, then $F^j(\vartheta) \leftarrow F^j(\vartheta') \leftarrow F^j(\vartheta) \cup F^j(\vartheta')$.
(3) Output $F$.
```

Given as input an ordered game $\Gamma = \langle I, G, L, \Theta, \kappa, \gamma, p, \succeq, \omega, u\rangle$, *GameShrink* applies the shrinking ideas presented above as aggressively as possible. Once it finishes, there are no contractible nodes (since it compares every pair of nodes at each level of the signal tree), and it outputs the corresponding information filter F. The correctness of *GameShrink* follows by a repeated application of Theorem 3.4. Thus, we have the following result:

Theorem 4.1. GameShrink *finds all ordered game isomorphisms and applies the associated ordered game isomorphic abstraction transformations. Furthermore, for any Nash equilibrium, σ', of the abstracted game, the strategy profile constructed for the original game from σ' is a Nash equilibrium.*

The dominating factor in the run time of *GameShrink* is in the r^{th} iteration of the main for-loop. There are at most $\binom{|\Theta|}{S} S!$ nodes at this level, where we again take S to be the maximum number of signals possibly revealed in the game. Thus, the inner for-loop executes $O\left(\left(\binom{|\Theta|}{S} S!\right)^2\right)$ times. As discussed in the next subsection, we use a union-find data structure to represent the information filter F. Each iteration of the inner for-loop possibly performs a union operation on the data structure; performing M operations on a union-find data structure containing N elements takes $O(\alpha(M, N))$ amortized time per operation, where $\alpha(M, N)$ is the inverse Ackermann's function [Ackermann 1928; Tarjan 1975] (which grows extremely slowly). Thus, the total time for *GameShrink* is $O\left(\left(\binom{|\Theta|}{S} S!\right)^2 \alpha\left(\left(\binom{|\Theta|}{S} S!\right)^2, |\Theta|^S\right)\right)$. By the inequality $\binom{n}{k} \le \frac{n^k}{k!}$, this is $O\left((|\Theta|^S)^2 \alpha\left((|\Theta|^S)^2, |\Theta|^S\right)\right)$. Again, although this is exponential in S, it is $\tilde{O}(n^2)$, where n is the number of nodes in the signal tree. Furthermore, *GameShrink* tends to actually run in *sublinear* time and space in the size of the game tree because the signal tree is significantly smaller than the game tree in most nontrivial games, as discussed above.

4.1 Efficiency enhancements

We designed several speed enhancement techniques for *GameShrink*, and all of them are incorporated into our implementation. One technique is the use of the union-find data structure [Cormen et al. 2001, Chapter 21] for storing the information filter F. This data structure uses time almost linear in the number of operations [Tarjan 1975]. Initially each node in the signalling tree is its own set (this corresponds to the identity information filter); when two nodes are contracted they are joined into a new set. Upon termination, the filtered signals for the abstracted game correspond exactly to the disjoint sets in the data structure. This is an efficient method of recording contractions within the game tree, and the memory requirements are only linear in the size of the signal tree.

Determining whether two nodes are ordered game isomorphic requires us to determine if a bipartite graph has a perfect matching. We can eliminate some of these computations by using easy-to-check necessary conditions for the ordered game isomorphic relation to hold. One such condition is to check that the nodes have the same number of chances as being ranked (according to $\succeq$) higher than, lower than, and the same as the opponents. We can precompute these frequencies for every game tree node. This substantially speeds up *GameShrink*, and we can leverage this database across multiple runs of the algorithm (for example, when trying different abstraction levels; see next section). The indices for this database depend on the private and public signals, but not the *order* in which they were revealed, and thus two nodes may have the same corresponding database entry. This makes the database significantly more compact. (For example in Texas Hold'em, the database is reduced by a factor $\binom{50}{3}\binom{47}{1}\binom{46}{1}/\binom{50}{5} = 20$.) We store the histograms in a 2-dimensional database. The first dimension is indexed by the private signals, the second by the public signals. The problem of computing the index in (either) one of the dimensions is exactly the problem of computing a bijection between all subsets of size r from a set of size n and integers in $\left[0, \dots, \binom{n}{r} - 1\right]$. We efficiently compute this using the subsets' *colexicographical ordering* [Bollobás 1986]. Let $\{c_1, \dots, c_r\}$, $c_i \in \{0, \dots, n-1\}$, denote the r signals and assume that $c_i < c_{i+1}$. We compute a unique index for this set of signals as follows:

$$index(c_1, \dots, c_r) = \sum_{i=1}^{r} \binom{c_i}{i}.$$

5. APPROXIMATION METHODS

Some games are too large to compute an exact equilibrium, even after using the presented abstraction technique. In this section we discuss general techniques for computing approximately optimal strategy profiles. For a two-player game, we can always evaluate the worst-case performance of a strategy, thus providing some objective evaluation of the strength of the strategy. To illustrate this, suppose we know player 2's planned strategy for some game. We can then fix the probabilities of player 2's actions in the game tree as if they were chance moves. Then player 1 is faced with a single-agent decision problem, which can be solved bottom-up, maximizing expected payoff at every node. Thus, we can objectively determine the expected worst-case performance of player 2's strategy. This will be most useful when we want to evaluate how well a given strategy performs when we know that it is not an equilibrium strategy. (A variation of this technique may also be applied in n-person games where only one player's strategies are held fixed.) This technique provides *ex post* guarantees about the worst-case performance of a strategy, and can be used independently of the method that is used to compute the strategies in the first place.

5.1 State-space approximations

By slightly modifying the *GameShrink* algorithm we can obtain an algorithm that yields even smaller game trees, at the expense of losing the equilibrium guarantees of Theorem 3.4. Instead of requiring the payoffs at terminal nodes to match exactly, we can instead compute a penalty that increases as the difference in utility between two nodes increases.

There are many ways in which the penalty function could be defined and implemented. One possibility is to create edge weights in the bipartite graphs used in Algorithm 1, and then instead of requiring perfect matchings in the unweighted graph we would instead require perfect matchings with low cost (i.e., only consider two nodes to be ordered game isomorphic if the corresponding bipartite graph has a perfect matching with cost below some threshold). Thus, with this threshold as a parameter, we have a knob to turn that in one extreme (threshold $= 0$) yields an optimal abstraction and in the other extreme (threshold $= \infty$) yields a highly abstracted game (this would in effect restrict players to ignoring all signals, but still observing actions). This knob also begets an *anytime* algorithm. One can solve increasingly less abstracted versions of the game, and evaluate the quality of the solution at every iteration using the *ex post* method discussed above.

5.2 Algorithmic approximations

In the case of two-player zero-sum games, the equilibrium computation can be modeled as a linear program (LP), which can in turn be solved using the simplex method. This approach has inherent features which we can leverage into desirable properties in the context of solving games.

In the LP, primal solutions correspond to strategies of player 2, and dual solutions correspond to strategies of player 1. There are two versions of the simplex method: the primal simplex and the dual simplex. The primal simplex maintains primal feasibility and proceeds by finding better and better primal solutions until the dual solution vector is feasible, at which point optimality has been reached. Analogously, the dual simplex maintains dual feasibility and proceeds by finding increasingly better dual solutions until the primal solution vector is feasible. (The dual simplex method can be thought of as running the primal simplex method on the dual problem.) Thus, the primal and dual simplex methods serve as *anytime* algorithms (for a given abstraction) for players 2 and 1, respectively. At any point in time, they can output the best strategies found so far.

Also, for any feasible solution to the LP, we can get bounds on the quality of the strategies by examining the primal and dual solutions. (When using the primal simplex method, dual solutions may be read off of the LP tableau.) Every feasible solution of the dual yields an upper bound on the optimal value of the primal, and vice versa [Chvátal 1983, p. 57]. Thus, without requiring further computation, we get lower bounds on the expected utility of each agent's strategy against that agent's worst-case opponent.

One problem with the simplex method is that it is not a primal-dual algorithm, that is, it does not maintain both primal and dual feasibility throughout its execution. (In fact, it only obtains primal and dual feasibility at the very end of execution.) In contrast, there are interior-point methods for linear programming that maintain primal and dual feasibility throughout the execution. For example, many interior-point path-following algorithms have this property [Wright 1997, Ch. 5]. We observe that running such a linear programming method yields a method for finding ϵ-equilibria (i.e., strategy profiles in which no agent can increase her expected utility by more than ϵ by deviating). A threshold on ϵ can also be used as a termination criterion for using the method as an anytime algorithm. Furthermore, interior-point methods in this class have polynomial-time worst-case run time, as opposed to the simplex algorithm, which takes exponentially many steps in the worst case.

6. RELATED RESEARCH

The main technique applied in this paper is that of transforming large extensive form games into smaller extensive form games for which an equilibrium can be computed. Then, the equilibrium strategies of the smaller game are mapped back into the original larger game. One of the first pieces of research addressing functions which transform extensive form games into other extensive form games, although not for the purpose of making the game smaller, was in an early paper [Thompson 1952], which was later extended [Elmes and Reny 1994]. In these papers, several distinct *transformations*, now known as Thompson-Elmes-Reny transformations, are defined. The main result is that one game can be derived from another game by a sequence of those transformations if and only if the games have the same *pure reduced normal form*. The pure reduced normal form is the extensive form game represented as a game in normal form where duplicates of pure strategies (i.e., ones with identical payoffs) are removed and players essentially select equivalence classes of strategies [Kuhn 1950a]. An extension to this work shows a similar result, but for slightly different transformations and *mixed reduced normal form* games [Kohlberg and Mertens 1986]. Modern treatments of this previous work on game transformations have also been written [Perea 2001, Ch. 6], [de Bruin 1999].

The notion of *weak isomorphism* in extensive form games [Casajus 2003] is related to our notion of restricted game isomorphism. The motivation of that work was to justify solution concepts by arguing that they are invariant with respect to isomorphic transformations. Indeed, the author shows, among other things, that many solution concepts, including Nash, perfect, subgame perfect, and sequential equilibrium, are invariant with respect to weak isomorphisms. However, that definition requires that the games to be tested for weak isomorphism are of the same size. Our focus is totally different: we find strategically equivalent *smaller* games. Another difference is that their paper does not provide any algorithms.

Abstraction techniques have been used in artificial intelligence research before. In contrast to our work, most (but not all) research involving abstraction has been for single-agent problems (e.g. [Knoblock 1994; Liu and Wellman 1996]). Furthermore, the use of abstraction typically leads to sub-optimal solutions, unlike the techniques presented in this paper, which yield optimal solutions. A notable exception is the use of abstraction to compute optimal strategies for the game of Sprouts [Applegate et al. 1991]. However, a significant difference to our work is that Sprouts is a game of perfect information.

One of the first pieces of research to use abstraction in multi-agent settings was the development of *partition search*, which is the algorithm behind *GIB*, the world's first expert-level computer bridge player [Ginsberg 1996; 1999]. In contrast to other game tree search algorithms which store a particular game position at each node of the search tree, partition search stores *groups* of positions that are similar. (Typically, the similarity of two game positions is computed by ignoring the less important components of each game position and then checking whether the abstracted positions are similar—in some domain-specific expert-defined sense—to each other.) Partition search can lead to substantial speed improvements over α-β-search. However, it is not game theory-based (it does not consider information sets in the game tree), and thus does not solve for the equilibrium of a game of imperfect information, such as poker.[11] Another difference is that the abstraction is defined by an expert human while our abstractions are determined automatically.

There has been some research on the use of abstraction for imperfect information games. Most notably, Billings *et al* [Billings et al. 2003] describe a manually constructed abstraction for the game of Texas Hold'em poker, and include promising results against expert players. However, this approach has significant drawbacks. First, it is highly specialized for Texas Hold'em. Second, a large amount of expert knowledge and effort was used in constructing the abstraction. Third, the abstraction does not preserve equilibrium: even if applied to a smaller game, it might not yield a game-theoretic equilibrium. Promising ideas for abstraction in the context of general extensive form games have been described in an extended abstract [Pfeffer et al. 2000], but to our knowledge, have not been fully developed.

7. CONCLUSIONS AND DISCUSSION

We introduced the ordered game isomorphic abstraction transformation and gave an algorithm, *GameShrink*, for abstracting the game using the isomorphism exhaustively. We proved that in games with ordered signals, any Nash equilibrium in the smaller abstracted game maps directly to a Nash equilibrium in the original game.

The complexity of *GameShrink* is $\tilde{O}(n^2)$, where n is the number of nodes in the signal tree. It is no larger than the game tree, and on nontrivial games it is drastically smaller, so *GameShrink* has time and space complexity *sublinear* in the size of the game tree. Using *GameShrink*, we found a minimax equilibrium to Rhode Island Hold'em, a poker game with 3.1 billion nodes in the game tree—over four orders of magnitude more than in the largest poker game solved previously.

To further improve scalability, we introduced an approximation variant of *GameShrink*, which can be used as an anytime algorithm by varying a parameter that controls the coarseness of abstraction. We also discussed how (in a two-player zero-sum game), linear programming can be used in an anytime manner to generate approximately optimal strategies of increasing quality. The method also yields

[11]Bridge is also a game of imperfect information, and partition search does not find the equilibrium for that game either. Instead, partition search is used in conjunction with statistical sampling to simulate the uncertainty in bridge. There are also other bridge programs that use search techniques for perfect information games in conjunction with statistical sampling and expert-defined abstraction [Smith et al. 1998]. Such (non-game-theoretic) techniques are unlikely to be competitive in poker because of the greater importance of information hiding and bluffing.

bounds on the suboptimality of the resulting strategies. We are currently working on using these techniques for full-scale 2-player limit Texas Hold'em poker, a highly popular card game whose game tree has about 10^{18} nodes. That game tree size has required us to use the approximation version of *GameShrink* discussed in Section 5 [Gilpin and Sandholm 2006]. More recently we have also applied other lossy abstraction techniques [Gilpin and Sandholm 2007; Gilpin et al. 2007] and custom equilibrium-finding algorithms [Gilpin et al. 2007] to that problem. These techniques have yielded highly competitive software programs for that game.

While our main motivation was games of private information, our abstraction method can also be used in games where there is no private information. The method can be helpful even if all signals that are revealed during the game are public (such as public cards drawn from a deck, or throws of dice). However, in such games, *expectiminimax search* [Michie 1966] (possibly supplemented with α-β-pruning) can be used to solve the game in linear time in n. In contrast, solving games with private information takes significantly longer: the time to solve an $O(n) \times O(n)$ linear program in the two-person zero-sum setting, and longer in more general games. Therefore, our abstraction method will pay off as a preprocessor in games with no private information only if the signal tree of the game is significantly smaller than the game tree.

Furthermore, while our method applies our isomorphism-based abstraction—which abstracts away the unnecessary aspects of the signals—exhaustively, there exist "abstractions" that our method does not capture, such as transpositions (that is, different sequences of the same moves leading to equivalent positions, as in tic-tac-toe, checkers, chess, and Go) and symmetries (e.g., rotating the pieces by 90 degrees on a Go board or in a chess endgame with no pawns). To address this, one could use our method first, and then use classic techniques—such as transposition tables and mapping symmetries to a canonical orientation—for dealing with such, more traditional, "abstraction" opportunities. While we do not know of any popular games that have both types of abstraction opportunities, we can construct such games to prove existence. For example, consider playing a game of tic-tac-toe to determine who goes first in a poker game that follows (a tie in the tic-tac-toe game could be broken, for example, in favor of player 1).

APPENDIX

A. EXTENSIVE FORM GAMES AND PERFECT RECALL

Our model of an extensive form game is defined as usual.

Definition A.1. An *n-person game in extensive form* is a tuple $\Gamma = (I, V, E, P, H, A, u, p)$ satisfying the following conditions:

(1) $I = \{0, 1, \ldots, n\}$ is a finite set of players. By convention, player 0 is the *chance* player.

(2) The pair (V, E) is a finite directed tree with nodes V and edges E. Z denotes the leaves of the tree, called *terminal nodes*. $V \setminus Z$ are *decision nodes*. $N(x)$ denotes x's children and $N^*(x)$ denotes x's descendants.

(3) $P : V \setminus Z \to I$ determines which player moves at each decision node. P induces a partition of $V \setminus Z$ and we define $P_i = \{x \in V \setminus Z \mid P(x) = i\}$.

(4) $H = \{H_0, \ldots, H_n\}$ where each H_i is a partition of P_i. For each of player i's *information sets* $h \in H_i$ and for $x, y \in h$, we have $|N(x)| = |N(y)|$. We denote the information set of a node x as $h(x)$ and the player who controls h is $i(h)$.

(5) $A = \{A_0, \ldots, A_n\}$, $A_i : H_i \to 2^E$ where for each $h \in H_i$, $A_i(h)$ is a partition of the set of edges $\{(x, y) \in E \mid x \in h\}$ leaving the information set h such that the cardinalities of the sets in $A_i(h)$ are the same and the edges are disjoint. Each $a \in A_i(h)$ is called an *action* at h.

(6) $u : Z \to \mathbb{R}^N$ is the *payoff* function. For $x \in Z$, $u_i(x)$ is the payoff to player i in the event that the game ends at node x.

(7) $p : H_0 \times \{a \in A_0(h) \mid h \in H_0\} \to [0, 1]$ where

$$\sum_{a \in A_0(h)} p(h, a) = 1$$

for all $h \in H_0$ is the transition probability for chance nodes.

In this paper we restrict our attention to games with *perfect recall* (formally defined in [Kuhn 1953]), which means that players never forget information.

Definition A.2. An n-person game in extensive form satisfies *perfect recall* if the following two constraints hold:

(1) Every path in (V, E) intersects h at most once.

(2) If v and w are nodes in the same information set and there is a node u that preceeds v and $P(u) = P(v)$, then there must be some node x that is in the same information set as u and preceeds v and the paths taken from u to v is the same as from x to w.

A straightforward representation for strategies in extensive form games is the *behavior strategy* representation. This is without loss of generality since Kuhn's theorem [Kuhn 1953] states that for any mixed strategy there is a payoff-equivalent behavioral strategy in games with perfect recall. For each information set $h \in H_i$, a behavior strategy is $\sigma_i(h) \in \Delta(A_i(h))$ where $\Delta(A_i(h))$ is the set of all probability distributions over actions available at information set h. A group of strategies $\sigma = (\sigma_1, \ldots, \sigma_n)$ consisting of strategies for each player is a *strategy profile*. We sometimes write

$$\sigma_{-i} = (\sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots, \sigma_n)$$

and

$$(\sigma_i', \sigma_{-i}) = (\sigma_1, \ldots, \sigma_{i-1}, \sigma_i', \sigma_{i+1}, \ldots, \sigma_n).$$

REFERENCES

Ackermann, W. 1928. Zum Hilbertschen Aufbau der reellen Zahlen. *Math. Annalen 99*, 118–133.

Applegate, D., Jacobson, G., and Sleator, D. 1991. Computer analysis of sprouts. Tech. Rep. CMU-CS-91-144, Carnegie Mellon University.

Bellman, R. and Blackwell, D. 1949. Some two-person games involving bluffing. *Proceedings of the National Academy of Sciences 35*, 600–605.

Bhat, N. A. R. and Leyton-Brown, K. 2004. Computing Nash equilibria of action-graph games. In *Proceedings of the 20th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*. AUAI Press, Banff, Canada.

Billings, D., Burch, N., Davidson, A., Holte, R., Schaeffer, J., Schauenberg, T., and Szafron, D. 2003. Approximating game-theoretic optimal strategies for full-scale poker. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*. Morgan Kaufmann, Acapulco, Mexico, 661–668.

Billings, D., Davidson, A., Schaeffer, J., and Szafron, D. 2002. The challenge of poker. *Artificial Intelligence 134,* 1-2, 201–240.

Blum, B., Shelton, C. R., and Koller, D. 2003. A continuation method for Nash equilibria in structured games. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*. Morgan Kaufmann, Acapulco, Mexico.

Bollobás, B. 1986. *Combinatorics*. Cambridge University Press, Cambridge.

Casajus, A. 2003. Weak isomorphism of extensive games. *Mathematical Social Sciences 46*, 267–290.

Chen, X. and Deng, X. 2006. Settling the complexity of 2-player Nash equilibrium. In *Proceedings of the Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society, Berkeley, CA, 261–272.

Chvátal, V. 1983. *Linear Programming*. W. H. Freeman and Company, New York, NY.

Cormen, T., Leiserson, C., Rivest, R., and Stein, C. 2001. *Introduction to Algorithms*, Second ed. MIT Press, Cambridge, MA.

de Bruin, B. P. 1999. Game transformations and game equivalence. Technical note x-1999-01, University of Amsterdam, Institute for Logic, Language, and Computation.

Elmes, S. and Reny, P. J. 1994. On the strategic equivalence of extensive form games. *Journal of Economic Theory 62*, 1–23.

Ford, Jr., L. R. and Fulkerson, D. R. 1962. *Flows in Networks*. Princeton University Press, Princeton, NJ.

Gilpin, A., Hoda, S., Peña, J., and Sandholm, T. 2007. Gradient-based algorithms for finding Nash equilibria in extensive form games. In *3rd International Workshop on Internet and Network Economics (WINE)*. San Diego, CA.

Gilpin, A. and Sandholm, T. 2005. Optimal Rhode Island Hold'em poker. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*. AAAI Press / The MIT Press, Pittsburgh, PA, 1684–1685. Intelligent Systems Demonstration.

Gilpin, A. and Sandholm, T. 2006. A competitive Texas Hold'em poker player via automated abstraction and real-time equilibrium computation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*. AAAI Press, Boston, MA.

Gilpin, A. and Sandholm, T. 2007. Better automated abstraction techniques for imperfect information games, with application to Texas Hold'em poker. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. ACM, Honolulu, HI.

Gilpin, A., Sandholm, T., and Sørensen, T. B. 2007. Potential-aware automated abstraction of sequential games, and holistic equilibrium analysis of Texas Hold'em poker. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*. AAAI Press, Vancouver, Canada, 50–57.

Ginsberg, M. L. 1996. Partition search. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*. AAAI Press, Portland, OR, 228–233.

Ginsberg, M. L. 1999. GIB: Steps toward an expert-level bridge-playing program. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI)*. Morgan Kaufmann, Stockholm, Sweden.

Govindan, S. and Wilson, R. 2003. A global Newton method to compute Nash equilibria. *Journal of Economic Theory 110*, 65–86.

Knoblock, C. A. 1994. Automatically generating abstractions for planning. *Artificial Intelligence 68,* 2, 243–302.

Kohlberg, E. and Mertens, J.-F. 1986. On the strategic stability of equilibria. *Econometrica 54*, 1003–1037.

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

KOLLER, D. AND MEGIDDO, N. 1996. Finding mixed strategies with small supports in extensive form games. *International Journal of Game Theory 25*, 73–92.

KOLLER, D., MEGIDDO, N., AND VON STENGEL, B. 1996. Efficient computation of equilibria for extensive two-person games. *Games and Economic Behavior 14,* 2, 247–259.

KOLLER, D. AND PFEFFER, A. 1997. Representations and solutions for game-theoretic problems. *Artificial Intelligence 94,* 1 (July), 167–215.

KREPS, D. M. AND WILSON, R. 1982. Sequential equilibria. *Econometrica 50,* 4, 863–894.

KUHN, H. W. 1950a. Extensive games. *Proc. of the National Academy of Sciences 36*, 570–576.

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

KUHN, H. W. 1953. Extensive games and the problem of information. In *Contributions to the Theory of Games*, H. W. Kuhn and A. W. Tucker, Eds. Annals of Mathematics Studies, 28, vol. 2. Princeton University Press, Princeton, NJ, 193–216.

LEMKE, C. AND HOWSON, J. 1964. Equilibrium points of bimatrix games. *Journal of the Society of Industrial and Applied Mathematics 12*, 413–423.

LEYTON-BROWN, K. AND TENNENHOLTZ, M. 2003. Local-effect games. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI).* Morgan Kaufmann, Acapulco, Mexico.

LIPTON, R., MARKAKIS, E., AND MEHTA, A. 2003. Playing large games using simple strategies. In *Proceedings of the ACM Conference on Electronic Commerce (ACM-EC).* ACM, San Diego, CA, 36–41.

LIU, C.-L. AND WELLMAN, M. 1996. On state-space abstraction for anytime evaluation of Bayesian networks. *SIGART Bulletin 7,* 2, 50–57. Special issue on Anytime Algorithms and Deliberation Scheduling.

MAS-COLELL, A., WHINSTON, M., AND GREEN, J. R. 1995. *Microeconomic Theory.* Oxford University Press, New York, NY.

MCKELVEY, R. D. AND MCLENNAN, A. 1996. Computation of equilibria in finite games. In *Handbook of Computational Economics*, H. Amann, D. Kendrick, and J. Rust, Eds. Vol. 1. Elsevier, 87–142.

MICHIE, D. 1966. Game-playing and game-learning automata. In *Advances in Programming and Non-Numerical Computation*, L. Fox, Ed. Pergamon, New York, NY, 183–200.

MILTERSEN, P. B. AND SØRENSEN, T. B. 2006. Computing sequential equilibria for two-player games. In *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA).* SIAM, Miami, FL, 107–116.

NASH, J. 1950. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences 36*, 48–49.

NASH, J. F. AND SHAPLEY, L. S. 1950. A simple three-person poker game. In *Contributions to the Theory of Games*, H. W. Kuhn and A. W. Tucker, Eds. Vol. 1. Princeton University Press, Princeton, NJ, 105–116.

PAPADIMITRIOU, C. AND ROUGHGARDEN, T. 2005. Computing equilibria in multi-player games. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA).* SIAM, Vancouver, BC, Canada, 82–91.

PEREA, A. 2001. *Rationality in extensive form games.* Kluwer Academic Publishers, Dordrecht, The Netherlands.

PFEFFER, A., KOLLER, D., AND TAKUSAGAWA, K. 2000. State-space approximations for extensive form games. Talk given at the First International Congress of the Game Theory Society, Bilbao, Spain.

PORTER, R., NUDELMAN, E., AND SHOHAM, Y. 2004. Simple search methods for finding a Nash equilibrium. In *Proceedings of the National Conference on Artificial Intelligence (AAAI).* AAAI Press, San Jose, CA, 664–669.

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

SANDHOLM, T. AND GILPIN, A. 2006. Sequences of take-it-or-leave-it offers: Near-optimal auctions without full valuation revelation. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. ACM, Hakodate, Japan, 1127–1134.

SANDHOLM, T., GILPIN, A., AND CONITZER, V. 2005. Mixed-integer programming methods for finding Nash equilibria. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*. AAAI Press / The MIT Press, Pittsburgh, PA, 495–501.

SAVANI, R. AND VON STENGEL, B. 2004. Exponentially many steps for finding a Nash equilibrium in a bimatrix game. In *Proceedings of the Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society Press, Rome, Italy, 258–267.

SELTEN, R. 1965. Spieltheoretische behandlung eines oligopolmodells mit nachfrageträgheit. *Zeitschrift für die gesamte Staatswissenschaft 12*, 301–324.

SELTEN, R. 1988. Evolutionary stability in extensive two-person games – correction and further development. *Mathematical Social Sciences 16*, 223–266.

SHI, J. AND LITTMAN, M. 2002. Abstraction methods for game theoretic poker. In *CG '00: Revised Papers from the Second International Conference on Computers and Games*. Springer-Verlag, London, UK, 333–345.

SINGH, S. P., SONI, V., AND WELLMAN, M. P. 2004. Computing approximate Bayes-Nash equilibria in tree-games of incomplete information. In *Proceedings of the ACM Conference on Electronic Commerce (ACM-EC)*. ACM, New York, NY, 81–90.

SMITH, S. J. J., NAU, D. S., AND THROOP, T. 1998. Computer bridge: A big win for AI planning. *AI Magazine 19,* 2, 93–105.

TARJAN, R. E. 1975. Efficiency of a good but not linear set union algorithm. *Journal of the ACM 22,* 2, 215–225.

THOMPSON, F. 1952. Equivalence of games in extensive form. RAND Memo RM-759, The RAND Corporation. Jan.

VON NEUMANN, J. AND MORGENSTERN, O. 1947. *Theory of Games and Economic Behavior.* Princeton University Press, Princeton, NJ.

VON STENGEL, B. 1996. Efficient computation of behavior strategies. *Games and Economic Behavior 14,* 2, 220–246.

VON STENGEL, B. 2002. Computing equilibria for two-person games. In *Handbook of game theory*, R. Aumann and S. Hart, Eds. Vol. 3. North Holland, Amsterdam, The Netherlands.

WILSON, R. 1972. Computing equilibria of two-person games from the extensive form. *Management Science 18,* 7, 448–460.

WRIGHT, S. J. 1997. *Primal-Dual Interior-Point Methods.* SIAM, Philadelphia, PA.

Received March 2006; accepted April 2007