Constraint Satisfaction Algorithms for Graphical Games

Vishal Soni, Satinder Singh, and Michael P. Wellman
University of Michigan, Computer Science & Engineering
2260 Hayward St, Ann Arbor, MI 48109-2121, USA
{soniv,baveja,wellman}@umich.edu

ABSTRACT

We formulate the problem of computing equilibria in multiplayer games represented by arbitrary undirected graphs as a constraint satisfaction problem and present two algorithms. The first is PureProp: an algorithm for computing approximate Nash equilibria in complete information one-shot games and approximate Bayes-Nash equilibria in one-shot games of incomplete information. PureProp unifies existing message-passing based algorithms for solving these classes of games. We also address *repeated graphical games*, and present a second algorithm, PureProp-R, for computing approximate Nash equilibria in these games.

Categories and Subject Descriptors

I.2 [**Artificial Intelligence**]; F.2 [**Analysis of Algorithms**]

General Terms

Algorithms, Economics, Theory

Keywords

graphical games, constraint satisfaction

1. INTRODUCTION

Graphical models for representing games were introduced by Kearns et al. [3] to capture locality of interactions among players in a static game. Under this model, nodes represent players and (undirected) edges represent interactions between players. A missing edge between two nodes implies that the payoffs of the respective players do not directly depend on the other's action. The central idea is to view the game as being composed of several interacting local games and to exploit this locality by iteratively computing local equilibria and patching them together to obtain global equilibria efficiently. Games with sparsely connected graphs arise quite commonly in various settings, for example those with network topologies based on computer connectivity, social networks, business relationships, ad hoc networks of mobile devices, etc.

The original work by Kearns et al. [3] considered *acyclic* graphical games of *complete information* and presented a message-passing algorithm (hereafter, the KLS algorithm) for computing approximate Nash equilibria (NE) efficiently. Ortiz and Kearns [6] later presented the NashProp algorithm, which extended the KLS algorithm to *arbitrary graph structures* in complete information games. In other recent work, Vickrey and Koller [10] reformulated the problem of finding approximate equilibria in games of complete information as a constraint satisfaction problem (CSP) and derived a generalization of the KLS algorithm as a constraint satisfaction algorithm. Singh et al. [8] considered *acyclic* games of *incomplete information* and presented a message-passing algorithm to compute approximate Bayes-Nash equilibria (BNE) efficiently. In games of incomplete information, the payoff to a player depends not only on the actions of the other players but also on its own private type. Agents have to choose actions without explicit knowledge of the private types of other players. Thus, a BNE strategy has to be an equilibrium with respect to the known distribution over the types of the other players.

In this paper, we (1) define a new constraint satisfaction formulation of the problem of finding approximate equilibria in graphical games, (2) derive a constraint satisfaction algorithm, PureProp(ϵ), from the application of straightforward CSP techniques, (3) show that PureProp(ϵ) unifies all of the above mentioned special-purpose results on graphical games and extends them to *incomplete information games* and *repeated games* with *arbitrary graph structures*, and (4) demonstrate the generality of our CSP formulation by deriving an algorithm, PureProp-R(ϵ), that computes ϵ-NE in repeated graphical games under the average payoff criterion.

Both PureProp(ϵ) and PureProp-R(ϵ) accept an approximation parameter ϵ as an input argument and compute pure-strategy ϵ-equilibria in their respective games. (Hereafter, we drop ϵ from the names for ease of exposition.) In each case, we show how to induce a restricted game solvable by the corresponding algorithm from all the classes of games described above. The construction of the induced game is such that a pure-strategy ϵ-NE always exists, and furthermore, the equilibria found are ϵ-equilibria in the original game.

We describe one-shot games and the corresponding PureProp algorithm first, then describe repeated games and the

AAMAS'07 May 14–18 2007, Honolulu, Hawai'i, USA.

corresponding PureProp-R algorithm, and finally conclude with some illustrative empirical results.

2. ONE-SHOT GAMES

2.1 Complete Information Games

A complete information game $G = \langle I, \mathbf{A}, \mathbf{U} \rangle$ is specified by a set of players I, a set of actions $\mathbf{A} = \times_{i \in I} A_i$ where A_i is the set of actions of player i, and a set of payoff functions $\mathbf{U} = \times_i u_i$ where $u_i : \mathbf{A} \rightarrow \Re$ for every player $i \in I$. Players have complete information in the sense that they have full knowledge of the payoff functions of the other players. Any action a of player i is also referred to as a *pure strategy*, whereas a *mixed strategy* $\sigma_i : A_i \rightarrow [0, 1]$ with $\sum_{a \in A_i} \sigma_i(a) = 1$ is a probability distribution over the pure strategies of i. A strategy profile $\sigma = \langle \sigma_1 \cdots \sigma_I \rangle$ is a strategy assignment to each player. The expected payoff to i under profile σ is

$$u_i(\sigma) = \sum_{\mathbf{a} \in \mathbf{A}} u_i(\mathbf{a}) \prod_{j \in I} \sigma_j(a_j). \quad (1)$$

We use the symbol $-i$ to refer to the set of all players other than i. Similarly, the symbols a_{-i} and σ_{-i} refer to a pure strategy profile and mixed strategy profile over $-i$.

A mixed strategy profile σ^* is a Nash equilibrium if, for all players i,

$$u_i(\sigma^*_{-i}, \sigma^*_i) \geq u_i(\sigma^*_{-i}, \sigma_i), \forall \sigma_i.$$

The implication here is that no player can benefit in expectation by deviating unilaterally from a Nash equilibrium. An ϵ-Nash equilibrium is a mixed strategy profile σ^* such that for any player i,

$$u_i(\sigma^*_{-i}, \sigma^*_i) + \epsilon \geq u_i(\sigma^*_{-i}, \sigma'_i), \forall \sigma'_i.$$

Thus, no player can gain more than ϵ in expectation by unilateral deviation from an ϵ-Nash equilibrium.

2.2 Incomplete Information Games

In games of incomplete information, each player's payoff function is defined $u_i : \mathbf{A} \times T_i \rightarrow \Re$, where T_i is the type space of i. Thus, the payoff is a function of not only the action profile but also of a realized type $t_i \in T_i$ of player i. For games with discrete types, the type of each player is assumed to be chosen independently from some commonly known distribution over a finite set of types $T_i = \langle T_i^1 \cdots T_i^k \rangle$. For games with continuous types, types are assumed to be chosen independently from a known density over a finite range $T_i = [T_i^l, T_i^u]$ of types. Players have incomplete information in that they do not have knowledge of the realized types of other players. However, players do know the distributions (or densities) from which the types of other players are drawn.

For each player i, a mixed strategy $\sigma_i : A_i \times T_i \rightarrow [0, 1]$ specifies a probability distribution over the actions of i for each type t_i such that $\forall t_i \in T_i, \sum_{a \in A_i} \sigma(a_i|t_i) = 1$ for all $i \in I$. The expected payoff to player i when its revealed type is t_i under a strategy profile σ is

$$u_i(\sigma_{-i}, \sigma_i|t_i) = \sum_{t_{-i}} Pr(t_{-i}) u_i(\sigma_{-i}, \sigma_i|t_{-i}, t_i), \quad (2)$$

where

$$u_i(\sigma_{-i}, \sigma_i|t_{-i}, t_i) = \sum_{\mathbf{a} \in \mathbf{A}} \sigma_i(a_i|t_i) \sigma_{-i}(a_{-i}|t_{-i}) u_i(\mathbf{a}|t_i).$$

A strategy profile σ^* is a Nash equilibrium of the incomplete information game (also called a *Bayes Nash equilibrium* or BNE for short) if for every player i and every type $t_i \in T_i$,

$$u_i(\sigma^*_{-i}, \sigma^*_i|t_i) \geq u_i(\sigma^*_{-i}, \sigma'_i|t_i), \forall \sigma'_i(\cdot|t_i).$$

Thus for each of its types, no player can benefit in expectation by unilateral deviation from its type-contingent mixed strategy. A profile σ^* is an ϵ-BNE if for every one of their types, players do not gain more than ϵ by unilateral deviation from their type contingent mixed strategy, that is, for all players i and every type $t_i \in T_i$,

$$u_i(\sigma^*_{-i}, \sigma^*_i|t_i) + \epsilon \geq u_i(\sigma^*_{-i}, \sigma'_i|t_i), \forall \sigma'_i(\cdot|t_i).$$

Note that for continuous types, the summations and type probabilities in the above equations will be replaced by integrals and type densities.

Complete information games are incomplete information games with just one type per player. It is well known that exact ($\epsilon = 0$) mixed strategy NE and BNE always exist in games of complete and incomplete information.

2.3 Graphical models for Games

In contrast to the representations of games described above, an n player graphical game has nodes representing players and edges representing direct interaction between players. The payoff function of each node is defined as a function of the actions of its neighbors. The neighbors of i are denoted $\mathcal{N}^i$ (this includes i) and $\mathcal{N}^i_{-i}$ denotes the set $\mathcal{N}^i - \{i\}$ (i.e. all the neighbors of i excluding i). Let k be the largest number of neighbors of any node in the graph.

It is important to note that this is a representational departure and doesn't change the underlying game - a graphical game will have the same equilibria as the underlying game. Any normal form game can be trivially converted into a graphical game by representing it as a fully connected graph. If the connectivity of the graph is sparse, i.e $n >> k$, the graphical representation leads to a substantially more compact representation of the game than the usual normal-form (matrix) representation. The locality of the game also reduces the computation required to determine the expected rewards in equations 1 and 2 because only the neighbors of players have to be considered in the summations.

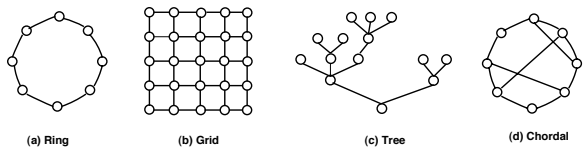

Figure 1: Example graph topologies.

Figure 1 illustrates a few graph topologies that capture different types of locality between players. In our experiments, we examine the effect these topologies have on the run-time of our algorithms.

We now describe the PureProp algorithm, for solving complete and incomplete information one-shot games, and the PureProp-R algorithm for solving infinitely repeated games. We start with PureProp.

2.4 PureProp

In Section 2.4.1 describe how to convert the problem of finding pure strategy ϵ-NE in games of complete information with a finite number of actions into a constraint satisfaction problem (the game-CSP). We then present our algorithm for solving game-CSPs (in Section 2.4.2). Finally, in Section 2.5 we show how to use our algorithm to compute approximate mixed-strategy equilibria in general complete and incomplete information one-shot graphical games.

2.4.1 Game-CSP

A CSP problem is defined by a set of variables, the domain of each variable (i.e. the values each variable can assume), and a set of constraints between variables. Each constraint involves some subset of variables and defines allowable combinations of values for that subset. A game-CSP consists of a set of variables V such that each variable corresponds to a player in the underlying game. The domain D_i of each variable i is a the set of all strategy profiles of the player's local neighborhood. The game-CSP imposes two types of constraints over domain values:

- **Unary Constraints**. These constraints are defined over the strategy profiles in the domain of every player. A profile σ in the domain of i is said to satisfy the unary constraint if

  $$u_i(\sigma_{\mathcal{N}^i_{-i}}, \sigma_i) + \epsilon \geq u_i(\sigma_{\mathcal{N}^i_{-i}}, \sigma'_i), \forall \sigma'_i$$

  That is, i's strategy is an ϵ-best response to the strategy profile $\sigma_{\mathcal{N}^i_{-i}}$ of its neighbors (recall that ϵ is an input argument to PureProp).

- **Binary Constraints**. These constraints are defined over pairs of profiles in the domain of neighboring players. Let i and j be neighbors. A profile $\sigma \in D_i$ satisfies the binary constraint with another profile $\rho \in D_j$ if both σ and ρ assign the same strategy to the common neighbors of i and j.

CSPs are often visualized using a constraint graph which places an undirected edge between any two nodes that have a constraint between them. Thus, the constraint graph of the game-CSP for a graphical game will have exactly the same edges as the graphical game itself.

A solution to the game-CSP is an assignment of profiles to each variable such that none of the constraints are violated. If there is no such assignment of profiles to variables the game-CSP has no solution.

The approach of formulating a graphical game as a CSP has also been examined by Vickrey and Koller [10]. In that work, each player in the game is a variable in the CSP and the domain of each variable is the strategy space of the corresponding player. Constraints over variables ensure that no player can benefit from deviating from a strategy profile of its neighborhood. This approach transforms the problem into a CSP with non-binary constraints. One of the motivations behind our formulation, however, is to unify graphical game algorithms from the literature [3, 6, 8]. It turns out that our game-CSP is what is known as the *dual transform* [2] of the CSP presented by Vickrey and Koller [10]. A dual transform of a CSP with non-binary constraints is an equivalent CSP with binary constraints.

The following theorem follows from the construction of the game-CSP above.

Theorem 1 *Any solution to the game-CSP for a graphical game of complete information is an ϵ-Nash equilibrium of the underlying game. Also, all pure-strategy ϵ-Nash equilibria of a complete information game with a finite number of actions are solutions to the game-CSP generated by the game.*

2.4.2 Solving the game-CSP

Once we have the game-CSP, it is possible to perform a backtracking search like algorithm to find a globally consistent solution. However, there are numerous search techniques for CSPs that take advantage of the graph structure and that use general purpose heuristics to improve search performance. The algorithm we provide here utilizes some of these techniques.

The PureProp algorithm takes in an input argument ϵ and returns an ϵ-NE of the game. It proceeds in three consecutive phases:

1. **Node consistency:** This phase considers each variable in turn and removes the profiles from its domain that violate the unary constraint for that variable. The end result is that for any profile $\sigma_{\mathcal{N}^i}$ in the domain of i, the strategy σ_i is an ϵ-best response to $\sigma_{\mathcal{N}^i_{-i}}$. The complexity of this step, while exponential in the size of the local neighborhood, is linear in the total number of players. If the connectivity of the graph is of order k, this step runs in $O(nd^k)$ where d is the maximum number of actions any player has. We denote the domain of i post arc consistency as D^0.

2. **Arc Consistency:** Each undirected edge in the constraint graph of the CSP is treated as two directed arcs in this phase. So for an edge between neighbors i and j, there is an arc $[i, j]$ as well as an arc $[j, i]$. Given current domains D_i and D_j, the arc $[i, j]$ is said to be *arc-consistent* if for every profile in the domain of i there is some profile in the domain of j with which the binary constraint is satisfied. We denote the domain of i post arc consistency as D^*. The goal of the arc consistency algorithm is to return domains for the variables such that all arcs are arc-consistent.

   While there are a number of algorithms for arc consistency in the CSP literature, we implemented the AC-3 algorithm as presented in [7]. This algorithm maintains a queue of arcs to be examined (initialized to contain all the arcs in the graph) and at each step examines the arc at the head of the queue. For every arc $[i, j]$ examined AC-3 reduces the domain of node i so that $[i, j]$ is arc-consistent. If the domain of i is reduced then all incoming arcs to i are inserted into the queue. The algorithm stops either when every arc is arc-consistent (the queue is empty) or when the domain of some node is empty. We denote the domains returned by AC-3 as D^*.

   If the domain of any node is empty at any step in either of the above two phases, the game-CSP has no solution. The next Lemma shows that all pure strategy ϵ-NE in the underlying game are present in D^*.

Lemma 2 *Let σ be a global profile. Then, for all players i, $\sigma_{\mathcal{N}^i} \in D_i^*$ if and only if σ is an ϵ-NE in the underlying game.*

Proof In the forward direction, if for all i $\sigma_{i,\mathcal{N}^i_{-i}} \in D_i^*$, node consistency implies that $u_i(\sigma_i, \sigma_{\mathcal{N}^i_{-i}}) + \epsilon \geq u_i(\sigma'_i, \sigma_{\mathcal{N}^i_{-i}}), \forall \sigma'_i$ Thus σ must be an ϵ-NE.

In the reverse direction, if σ is an ϵ-NE, then for all $i \in I$,

$$u_i(\sigma_i, \sigma_{-i}) + \epsilon \geq u_i(\sigma'_i, \sigma_{-i}), \forall \sigma'_i$$
$$\Rightarrow \sigma_{i,\mathcal{N}^i_{-i}} \in D_i^o$$

Also, for any two neighbors i and j, $\sigma_{i,\mathcal{N}^i_{-i}}$ and $\sigma_{j,\mathcal{N}^j_{-j}}$ satisfy the binary constraint for i and j in the CSP. Thus $\sigma_{i,\mathcal{N}^i_{-i}} \in D_i^*$. □

3. **Value Propagation** This phase executes a standard backtracking search over the variable domains (post node and arc consistency) in the constraint graph. Following the "minimum values remaining" heuristic, nodes are visited in ascending order of their domain size. At any stage of the search, a node i picks from its domain a profile σ that is consistent with the current action assignments of i and its neighbors. If any of i's neighbors have not been assigned an action, i instructs them to play an action according to σ. If i is not able to find such a profile, the algorithm backtracks. It is known that backtracking search is a complete algorithm that in the worst-case amounts to an exhaustive search over variable domains. The algorithm can stop after finding the first solution or can be extended to find all solutions to the game-CSP.

Treating the conversion to the game-CSP as part of PureProp the following result holds:

Theorem 3 *PureProp(ϵ) computes (all) pure-strategy ϵ-NE in complete information games with a finite number of actions.*

Proof The result is directly implied by Theorem 1, Lemma 2, and the completeness of the value propagation phase. □

2.5 Computing mixed strategy approximate equilibria

So far we have shown that PureProp is able to find pure strategy ϵ-NE in complete information games. While this seems limited in applicability, we show next how PureProp can be used to solve classes of complete and incomplete information games for approximate mixed-strategy equilibria. In each case, the space of strategies is infinite and hence PureProp cannot be directly applied. We thus have to construct a new reduced game with a finite strategy space such that (a) all pure-strategy ϵ-NE in the reduced game are ϵ-equilibria in the original game, and (b) at least one ϵ-equilibrium of the original game is a pure-strategy ϵ-NE of the reduced game. We refer to such a reduced game as an *ϵ-induced* game.

In each case, we construct the ϵ-induced game by restricting a player's action probabilities to multiples of a discretization parameter τ. The result is a finite set of τ discretized strategies that can be used as pure actions of the ϵ-induced game.

We consider two classes of incomplete information games: games with a discrete type space, and games with a continuous type space. Note that since complete information games are a special case of incomplete information games with one type per player, the construction of the ϵ-induced game for discrete types presented below can be applied directly to complete information games.

2.5.1 Discrete Types

In such games, a strategy is a mapping from the discrete type space to probability distributions over actions. Recall that k is the maximum number of neighbors of any node in the graph and d is the maximum number of actions available to any player in the game.

Theorem 4 *For arbitrary graphical games of incomplete information with discrete types, PureProp applied to the ϵ-induced game defined above with discretization parameter $\tau \leq \frac{\epsilon}{d^k(4k \log k)}$ will converge to an ϵ-BNE of the game.*

Proof (Sketch) Singh et al.([8]) show that for any ϵ, there is a $\tau \leq \frac{\epsilon}{d^k(4k \log k)}$ such that if the mixed strategy space for every type restricted to multiples of τ, there exists an ϵ-BNE in the τ-discretized strategy space. Thus the number of pure actions in the resulting ϵ-induced game is of $O\left[\left(\frac{1}{\tau}\right)^{md}\right]$ where m is the number of discrete types. PureProp applied to the ϵ-induced game is thus guaranteed to converge to an ϵ-BNE of the original game. □

2.5.2 Continuous types

In games with continuous types, it is reasonable to require that types have some structured effects on players' payoffs. Otherwise just representing the effect of types exactly can lead to unbounded game descriptions. Quite naturally, the desire is to identify common effects that can be exploited for computational efficiency. Singh et al. [8] considered a class of games with continuous types that have what is called the single crossing point property (examples of such games include certain kinds of first price auctions, all-pay auctions, noisy signaling games, oligopoly games, etc. [1], [5], [9]) — such games have the interesting property that there exist BNE that are *monotone threshold pure strategies*. In a threshold pure strategy each type is mapped to a pure action and furthermore each action is played in at most one contiguous region of the continuous type interval. The points in the type interval at which the strategies switch to new actions thus constitute thresholds. In monotone threshold pure strategies there is an ordering over actions such that the actions appear in that order in the strategy of all players. Even though distributions over the pure actions don't need to be considered in such games, the number of strategies in such games is infinite because of the continuous types.

Theorem 5 *For graphical games of incomplete information with continuous types that satisfy the single crossing point property, PureProp applied to the ϵ-induced game defined above with discretization parameter $\tau \leq \frac{\epsilon}{4dk\rho_t}$ will converge to an ϵ-BNE of the original game.*

Proof Singh et al. [8] showed that for any ϵ, there exists a $\tau = \frac{\epsilon}{4dk\rho_t}$, where ρ_t is the maximum probability density over type space, such that there exists a τ-discretized strategy that is an ϵ-BNE. A τ-discretized strategy only allows thresholds at the τ intervals in type space. There will be

$O\left[\left(\frac{1}{\tau}\right)^d\right]$ many τ-discretized strategies and these will be the pure actions of the ϵ-induced game. As before, the existence of an ϵ-BNE in the τ-discretized strategies of the original game will guarantee the existence of an ϵ-NE in the ϵ-induced game. PureProp applied to the ϵ-induced game with the τ-discretized strategies as actions will find ϵ- BNE in the original game. □

2.5.3 Extension and Connection with other algorithms

PureProp extends the results of Singh etal. [8] on tree-games to general graphical games without requiring any changes in the discretization parameters. The ability to compute approximate equilibria in the above classes of incomplete information games with arbitrary graph structure is one of the contributions of this work.

Furthermore, PureProp also unifies all the previous results on graphical games mentioned in the introduction; for lack of space we only briefly discuss this aspect. For tree structured games of incomplete information, it can be shown that with an appropriate ordering of variables during arc-consistency and value assignment, PureProp applied to the ϵ-induced games implements the algorithm of [8]. The discretization technique employed for discrete type games can be applied here to obtain an ϵ-induced game solvable by PureProp for ϵ-NE. This is the same discretization scheme as that used by the algorithms of Ortiz and Kearns [6] and Kearns etal. [3]. Thus, the resulting algorithm implements the NashProp algorithm of [6] for arbitrary graphs, and, with an appropriate variable ordering heuristic, the algorithm of [3] for tree structured games.

3. REPEATED GAMES

We have shown how formulating a one-shot graphical game as a CSP allows us to effectively solve the game by applying standard constraint satisfaction algorithms. We now demonstrate the generality of this approach by extending the CSP formulation to graphical repeated games. Specifically, we consider infinitely repeated complete information graphical games under the average payoff criterion. We present PureProp-R, a relatively straightforward algorithm resulting from our CSP formulation, for computing ϵ-equilibria in these games.

An infinitely repeated game is one in which the same *stage game* $G = \langle I, \mathbf{A}, \mathbf{U} \rangle$ is played at every time step. Players simultaneously pick an action at every round, and at the end of every period all players observe the realized action profile. A player's strategy is a probability distribution over it's actions for every stage of the game, and is in general a function of the history of interactions. Players seek to maximize their expected average future payoff. Under the average payoff criterion, a Nash Equilibrium is a strategy profile in which no player can improve their average expected future payoff by unilateral deviation.

We use σ_i to denote a probability distribution over player i's actions, and $\sigma_{N^i_{-i}}$ to denote a probability distribution over the actions of i's neighbors. We use $u_i(\sigma_{N^i_{-i}}, \sigma_i)$ to denote the stage payoff to i when i plays according to σ_i and its neighbors play according to $\sigma_{N^i_{-i}}$. Following Littman and Stone [4], let $\alpha_{\mathcal{N}^i_{-i}}$ denote the profile of i's neighborhood that minimizes i's expected payoff (the *attack profile*) in any stage of the game, i.e.

$$\alpha_{\mathcal{N}^i_{-i}} = \arg\min_{\sigma_{\mathcal{N}^i_{-i}}} \max_{a_i} u_i(a_i, \sigma_{\mathcal{N}^i_{-i}})$$

Let δ_i denote the strategy that i should play to maximize its expected stage payoff (its *defend strategy*), i.e.

$$\delta_i = \arg\max_{\sigma_i} \min_{a_{\mathcal{N}^i_{-i}}} u_i(\sigma_i, a_{\mathcal{N}^i_{-i}})$$

The expected stage payoff that i can guarantee itself by playing its defend strategy, i.e. its *minmax payoff* is:

$$\nu_i = \sum_{a_{\mathcal{N}^i_{-i}} \in A_{\mathcal{N}^i_{-i}}} \prod_{j \in \mathcal{N}^i_{-i}} \alpha_j(a_j) \sum_{a_i \in A_i} \delta_i(a_i) u_i(a_i, a_{\mathcal{N}^i_{-i}}),$$

where by abuse of notation $\alpha_j(a_j)$ denotes the probability of i's neighbor j taking action a under the strategy profile $\alpha_{\mathcal{N}^i_{-i}}$. This is the lowest payoff that player i's opponents can restrict i to in any stage of the game.

A payoff profile $u(\sigma)$ is said to be *enforceable* if for every player i, $u_i(\sigma) \geq \nu_i$, and is further said to be *feasible* if it lies in the convex hull of pure strategy profiles. The "folk theorem" for repeated games tells us that any stage payoff profile that is enforceable and feasible is the payoff of a NE of the repeated game. Indeed, the folk theorem gives us a prescriptive way to establish a feasible and enforceable profile as an equilibrium as follows: every player plays according to a fixed feasible and enforceable profile at every stage of the game. In the event of unilateral deviation by any player, the other players *punish* it by restricting the deviating player to its minmax payoff. These punishments strategies are off-equilibrium strategies and need to be defined so as to disincentivize deviation. For the approximate setting, a profile is an ϵ-Nash equilibrium of the repeated game if no player can expect to gain more than ϵ by unilateral deviation (again assuming punishment by the other players after a first deviation).

We present PureProp-R as an algorithm to compute *pure strategy* feasible and enforceable profiles of the stage game. We will then show how with an appropriate discretization of the mixed strategy space, PureProp-R applied to the discretized game will return a profile that with the appropriate off-equilibrium punishment strategy can be established as an ϵ-NE of the original undiscretized repeated game.

Below, let the symbols α', δ', and ν' denote the attack strategies, defend strategies, and minmax payoffs computed over the pure strategy space.

3.1 The game-CSP for repeated games

As before, the game-CSP is defined by a constraint network over a of a set of nodes V with each node representing a player in the game. The domain D_i of each player i is the set of pure strategy profiles over the nodes in i's neighborhood. The game-CSP imposes two types of constraints over domain values:

- **Unary Constraints.** A profile σ in the domain of i satisfies the unary constraint if $u_i(\sigma_i, \sigma_{\mathcal{N}^i_{-i}}) > \nu'_i$ where ν'_i denotes i's minmax payoff computed over the pure strategy profiles in its domain.
- **Binary Constraints.** As for the game-CSP for one-shot games, these constraints are defined over pairs of profiles in the domain of neighboring players. Let i and

j be neighbors. A profile $\sigma \in D_i$ satisfies the binary constraint with another profile $\rho \in D_j$ if both σ and ρ assign the same strategy to the common neighbors of i and j.

Theorem 6 follows by construction of the game-CSP above:

Theorem 6 *Any solution to the game-CSP for a repeated graphical game is a feasible and enforceable pure strategy profile of the stage game. Also, all pure-strategy feasible and enforceable profiles of a stage game with a finite number of actions are solutions to the corresponding game-CSP.*

3.2 Solving the game-CSP

As before, the algorithm proceeds in three consecutive phases:

- **Node Consistency** Every player first computes the attack strategy of its neighborhood (α'), its defend strategy (δ'), and its minmax payoff (ν') over the pure strategy profiles in its domain. To achieve node consistency, a player retains all the profiles in its domain that satisfy the unary constraint. The complexity of this step is $O(nd^k)$ where d is the maximum number of actions of any player and k is the size of the largest neighborhood. We use D^0 to denote the domain of player i after node consistency.

- **Arc Consistency and Value Propagation** Both phases execute in the same manner as in PureProp. Since these steps do not depend on the actual constraints defining the graph or on the type of equilibrium, the same algorithm can be used to find a globally consistent profile. We use D^* to denote the domain of player i after arc consistency.

The next Lemma shows that all pure strategy feasible and enforceable profiles in the game are present in D^*.

Lemma 7 *Let σ be a global profile. Then, for all players $i \in I$, $\sigma_{i,\mathcal{N}^i_{-i}} \in D^*(i)$ if and only if σ is a feasible and enforceable pure strategy profile of the stage game.*

Proof In the forward direction, if for all $i \in I$ $\sigma_{i,\mathcal{N}^i_{-i}}, \in D^*_i$, then node consistency implies that $u_i(\sigma_i, \sigma_{\mathcal{N}^i_{-i}}) \geq \nu'_i$. Thus σ must be a feasible and enforceable pure strategy.

In the reverse direction, if σ is a feasible and enforceable profile, then for all $i \in I$,

$$u_i(\sigma_{i,\mathcal{N}^i_{-i}}) \geq \nu'_i \Rightarrow \sigma_{i,\mathcal{N}^i_{-i}} \in D^0_i$$

Also, for any neighbors i and j, $\sigma_{i,\mathcal{N}^i_{-i}}$ and $\sigma_{j,\mathcal{N}^j_{-j}}$ satisfy the binary constraint for i and j in the CSP. Thus $\sigma_{i,\mathcal{N}^i_{-i}} \in D^*_i$. □

The output of the value propagation step will thus be a feasible and enforceable strategy profile of the stage game.

3.3 Computing mixed strategy ϵ-NE

Since PureProp-R solves for pure strategy profiles, we need to reduce the repeated game into a game solvable by PureProp-R. Given an ϵ, we show next how to derive a discretization parameter τ so that a profile returned by PureProp-R in the τ-discretized game can be established as an ϵ-NE of the underlying game.

Lemma 8 (from KLS) *Let $A_i \in [0,1]$ for all i. Let σ and ρ satisfy $|\sigma_i - \rho_i| < \tau$ for all $i \in I$, and let $\tau \leq 2/(k \log^2(k/2))$. Then $|\prod_{i=1}^k \sigma_i - \prod_{i=1}^k \rho_i| \leq (2klogk)\tau$ where k is the size of the largest neighborhood.*

Lemma 9 *Let ρ and ρ' be profiles such that for all players i and actions $a_i \in A_i$, $|\rho_j(a_j) - \rho'_j(a_j)| \leq \tau$. Then,*

$$\left|u_i(\rho) - u_i(\rho')\right| \leq d^k(2k \log k)\tau$$

provided $\tau \leq 2/(k \log^2(k/2))$

Proof

$$\left|u_i(\rho) - u_i(\rho')\right| \quad = \quad \sum_{\mathbf{a} \in \mathbf{A}} \left|\rho(\mathbf{a}) - \rho'(\mathbf{a})\right| u_i(\mathbf{a})$$

Let d be the maximum number of actions of any player. From Lemma 8 it follows that

$$\left|u_i(\rho) - u_i(\rho')\right| \leq \sum_{\mathbf{a} \in \mathbf{A}} u_i(\mathbf{a})(2k \log k)\tau \leq d^k(2k \log k)\tau$$

where we have assumed that the payoff function is bounded in absolute value by 1. □

For every player i, PureProp-R computes the minmax profile $\langle \delta'_i, \alpha'_{\mathcal{N}^i_{-i}} \rangle$ over the discretized strategy space. Let γ_i be an *arbitrary* strategy of i that maximizes its reward against $\alpha'_{\mathcal{N}^i_{-i}}$. We now bound the difference $u_i(\gamma_i, \alpha'_{\mathcal{N}^i_{-i}}) - u_i(\delta'_i, \alpha'_{\mathcal{N}^i_{-i}})$. Here, $u_i(\gamma_i, \alpha'_{\mathcal{N}^i_{-i}})$ is the payoff the neighbors of i can hold i to when they play the computed attack strategy $\alpha'_{\mathcal{N}^i_{-i}}$.

Lemma 10 *Let $\langle \delta'_i, \alpha'_{\mathcal{N}^i_{-i}} \rangle$ be a defend strategy of i and the attack profile of its neighborhood in the τ-discretized strategy space. Let $\gamma_i = \arg\max_{\sigma_i} u_i(\sigma_i, \alpha'_{\mathcal{N}^i_{-i}})$. Then, provided $\tau \leq 2/(k \log^2(k/2))$,*

$$u_i(\gamma_i, \alpha'_{\mathcal{N}^i_{-i}}) - u_i(\delta'_i, \alpha'_{\mathcal{N}^i_{-i}}) \leq d^k(2k \log k)\tau$$

Proof Let σ_i be an arbitrary (not restricted to discretized) strategy of player i. Let σ'_i be any corresponding τ-discretized strategy. Then from Lemma 9 it follows that

$$\left|u_i(\sigma_i, \alpha'_{\mathcal{N}^i_{-i}}) - u_i(\sigma'_i, \alpha'_{\mathcal{N}^i_{-i}})\right| \quad \leq \quad d^k(2k \log k)\tau.$$

Since the above equation holds for arbitrary strategy σ_i,

$$\left|\max_{\sigma_i} u_i(\sigma_i, \alpha'_{\mathcal{N}^i_{-i}}) - \max_{\sigma'_i} u_i(\sigma'_i, \alpha'_{\mathcal{N}^i_{-i}})\right| \quad \leq \quad d^k(2k \log k)\tau$$

$$\Rightarrow u_i(\gamma_i, \alpha'_{\mathcal{N}^i_{-i}}) - u_i(\delta'_i, \alpha'_{\mathcal{N}^i_{-i}}) \quad \leq \quad d^k(2k \log k)\tau$$

□

In any profile σ returned by PureProp-R, $u_i(\sigma) \geq u_i(\delta'_i, \alpha'_{\mathcal{N}^i_{-i}})$ for all i. We show next that such a profile an be established as an ϵ-NE of the repeated game where $\epsilon = d^k(2k \log k)\tau$.

Theorem 11 *For arbitrary repeated graphical games under the average payoff criterion, PureProp-R applied to the ϵ-induced game derived using a discretization parameter $\tau = \frac{\epsilon}{d^k 2k \log k}$ will return a profile that can be established as an ϵ-NE of the repeated game.*

Proof Let $\epsilon = d^k(2k \log k)\tau$. Let $\nu'_i = u_i(\delta'_i, \alpha'_{\mathcal{N}^i_{-i}})$ be the minmax payoff to i computed over discretized strategy profiles. Let σ be a profile returned by PureProp-R, i.e. $u_i(\sigma) > \nu'_i$ for all i. From Lemma 10, $\nu'_i + \epsilon$ is the lowest payoff $\mathcal{N}^i_{-i}$ can restrict it to by playing their attack profile $\alpha'_{\mathcal{N}^i_{-i}}$. We consider two possibilities for the payoff to i:

Case 1: For all i, $u_i(\sigma) > \nu'_i + \epsilon$. Let p_i be the largest payoff player i receives for deviating from this profile. The neighbors of i can punish i for this deviation by restricting i to $\nu'_i + \epsilon$. Let n be the number of rounds of the game for which i is punished. Player i will not gain anything from deviating if

$$u_i(\sigma_i, \sigma_{\mathcal{N}^i_{-i}}) \geq \frac{p_i + n(\nu'_i + \epsilon)}{n+1}$$

Thus, the punishment strategy of the neighbors of i is to play their attack profile against i for $\geq n$ rounds.

Case 2: For some player i, $u_i(\sigma) \in [\nu'_i, \nu'_i + \epsilon]$. Define the punishment strategy as one in which $\mathcal{N}^i_{-i}$ play their attack profile for all rounds following defection (known as a *grim strategy*). Then, the expected future payoff to i is

$$\lim_{n \to \infty} \frac{p_i + n(\nu'_i + \epsilon)}{n+1}$$

which goes to $\nu'_i + \epsilon$ as $n \to \infty$. Thus the maximum expected gain to i from deviating is $\leq \epsilon$.

A profile σ returned by PureProp-R is thus an ϵ-NE of the repeated game as follows: in every stage of the game, every player plays according to σ. If a player deviates, its neighbors punish it according to the punishment strategies described above. In this manner, no player will gain more than ϵ in expectation by deviating from σ. □

4. EXPERIMENTAL RESULTS

For our experiments, we evaluated PureProp on the *party game with threshold pure strategies* described in [8], and PureProp-R on the repeated prisoners dilemma game.

In the party game, players have been invited to a party and must decide whether to attend or not (so two actions). Each player has a social-network of players it is acquainted with and these constitute its neighbors in the graphical game. A player's reward for attending is an additive function of its like/dislike for its attending acquaintances. Each player also incurs a private cost associated with attending the party which has an additive effect on its reward. In the corresponding graph game, nodes represent players while edges connect a player to its social network.

We used the τ-discretization idea described above to construct an ϵ-induced game and then used PureProp to compute approximate equilibria in party games for the following neighborhood topologies (see Figure 1): (a) ring, (b) grid, (c) acyclic with a branching factor of 2 (i.e. each node has 3 neighbors), (d) chordal (10% of edges are chords), (e) chordal (20% of edges are chords). For each topology, we varied the number of players and measured the effect on the run-time[1] of various phases of our algorithm.

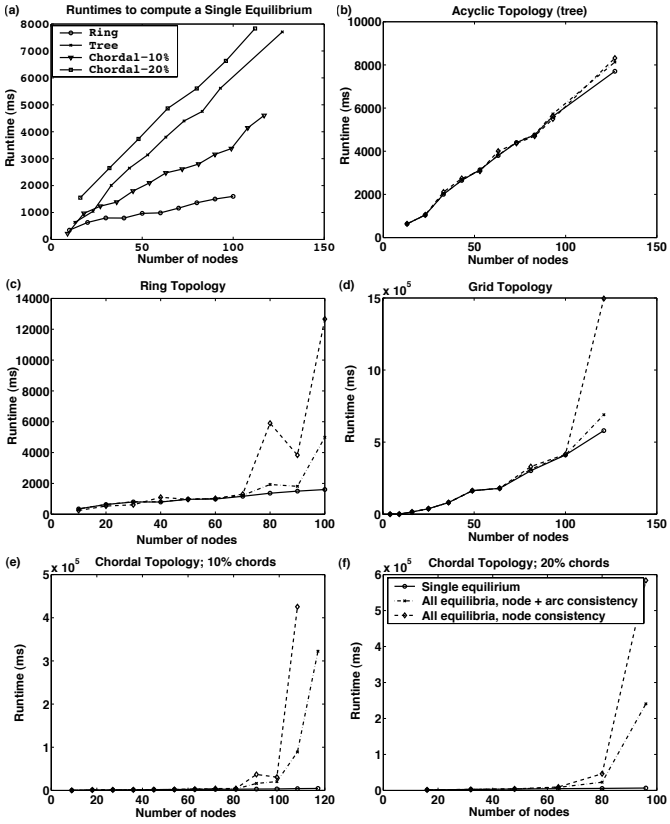

Figure 2: Party Game Results. (a) Scaling with number of players. (b) – (f) Run-time analysis of the PureProp algorithm on the following topologies : acyclic (tree), ring, grid, chordal (10%), and chordal (20%). In (b) – (f), solid lines indicate the time to compute a single ϵ-BNE, dashed-dot lines indicate the time to compute all ϵ-BNE, and dashed lines indicate the time to compute all ϵ-BNE when the arc consistency step is skipped.

Figure 2(a) compares the run-time of the algorithm on various topologies as a function of the number of players. We observe that regardless of topology, the run-time to compute a single ϵ-BNE scales linearly with the number of nodes. Additionally, we see that run-time is strongly affected by the sizes of local neighborhoods. For instance, computation is quickest for ring topologies. For the two chordal graphs, we observe that the graph with fewer chords, and hence fewer cycles, results in faster run-times.

Each plot in Figures 2(b - f) shows three curves that indicate (1) the time to compute a single ϵ-BNE, (2) the time to compute all ϵ-BNE, and (3) the time to compute all ϵ-BNE when the arc consistency phase is skipped, that is, node consistency is immediately followed by value assignment.

In Figure 2(b), we see that for acyclic graphs, the time required for all three computations is pretty much the same, suggesting that most of the work is done by the node consistency step and very little by arc consistency and value

[1]We measured run-time for all our experiments on a machine with a 2.80GHz Intel(R) Pentium(R) CPU and 1GB of RAM.

propagation – this is exactly the behavior we would expect since the number of backtracking steps during value propagation will be minimal. In Figure 2(c), we notice that for ring topologies, circumventing arc consistency results in a big performance hit. This serves to demonstrate the effectiveness of the arc consistency step in reducing domain sizes. In Figure 2(d), however, it is interesting to note that for the grid topology, arc consistency does not prove very beneficial for graphs with fewer than 100 nodes. We speculate that the reason for this is the high degree of connectivity in this graph. For smaller graphs, this would cause the node consistency phase to eliminate a considerable number of profiles. However, as the ratio of node connectivity to graph size decreases, so does the effectiveness of node consistency. At the same time, the benefits of arc consistency start to become apparent. In Figures 2(e) and 2(f) we observe similar effects, and note again that the algorithm generally ran faster on graphs with fewer chords due to the presence of fewer cycles.

Next we illustrate the performance of PureProp-R on a simple repeated prisoners' dilemma game, in which each player has two actions: whether to cooperate with its neighbors or to defect by testifying to the police against its neighbors in the graph. Each player gets a payoff of $+1$ for each of its cooperating neighbors, and a payoff of -1 for each of its defecting neighbors. A player also gets a payoff of 0 for cooperating and a payoff of $+1$ for defecting. The total payoff to a player is thus the sum of these quantities. We discretized the mixed strategy space using the technique described in the previous section and applied PureProp-R to the resulting game. Figure 3 illustrates the effect of number of players and graph topology on PureProp-R. Once again, we observe that algorithm run-time scales linearly with the number of players regardless of topology.

Thus, we see that both PureProp and PureProp-R are able to exploit locality of interaction in their respective classes of games for compuational benefit.

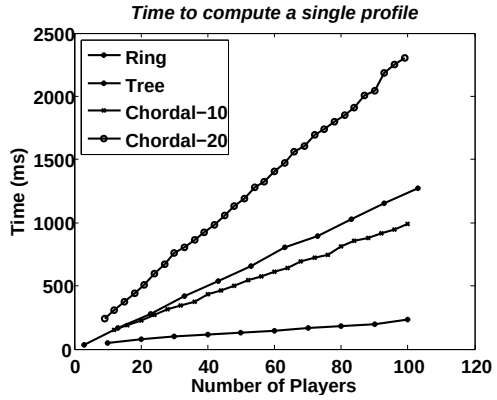

Figure 3: Repeated prisoner's dilemma results. This graph shows the time taken by PureProp-R to find a single feasible and ϵ-enforceable profile. In the repeated setting too, the complexity is roughly linear in the number of players and is more strongly affected by graph density.

5. CONCLUSION

We have formulated the problem of finding approximate equilibria in graphical games as a constraint satisfaction problem. Our resulting algorithm, PureProp, unifies several of the previous results on graphical games and extends then to incomplete information games with arbitrary graph structure. We have also shown that the approach of formulating a graphical game as a CSP is not just limited to one-shot games by deriving a new constraint satisfaction algorithm, PureProp-R, for infinitely repeated games under the average payoff criterion. Empirically, we have verified our claim that the complexity of our algorithms, while an exponential function of the size of the neighborhoods in the graph, is linear in the total number of players in the game.

6. ACKNOWLEDGEMENTS

Vishal Soni and Satinder Singh were supported by the NSF under Grant Number CCF-0432027 and by a grant from DARPAs IPTO program. Michael Wellman was supported by NSF grant IIS-0205435, and by the STIET program under NSF IGERT grant 0114368. Any opinions, findings, and conclusions or recommenda-tions expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or DARPA.

7. REFERENCES

[1] S. Athey. Single crossing properties and the existence of pure strategy equilibria in games of incomplete information. *Econometrica*, 69:861–869, 2001.

[2] R. Dechter and J. Pearl. Tree clustering schemes for constraint processing. In *Seventh National Conference on Artificial Intelligence (AAAI-88)*, pages 37–42, 1988.

[3] M. Kearns, M. Littman, and S. Singh. Graphical models for game theory. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 253–260, San Francisco, CA, 2001. Morgan Kaufmann Publishers.

[4] M. L. Littman and P. Stone. A polynomial-time Nash equilibrium algorithm for repeated games. *Decision Support Systems*, 39:55–66, 2005.

[5] P. Milgrom and C. Shannon. Monotone comparative statics. *Econometrica*, 62:157–180, 1994.

[6] L. E. Ortiz and M. Kearns. Nash propagation for loopy graphical games. In S. T. S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 793–800. MIT Press, Cambridge, MA, 2003.

[7] S. Russel and P. Norvig. *Artificial Intelligence - A Modern Approach*. Prentice Hall, 2003.

[8] S. Singh, V. Soni, and M. Wellman. Computing approximate bayes nash equilibria in tree-games of incomplete information. In *ACM Conference on Electronic Commerce*, 2004.

[9] D. Topkis. *Supermodularity and Complementarity*. Princeton University Press, 2002.

[10] D. Vickrey and D. Koller. Multi-agent algorithms for solving graphical games. In *Eighteenth national conference on Artificial intelligence*, pages 345–351. American Association for Artificial Intelligence, 2002.