CHAPTER 7

Graphical Games

Michael Kearns

Abstract

In this chapter we examine the representational and algorithmic aspects of a class of graph-theoretic models for multiplayer games. Known broadly as *graphical games*, these models specify restrictions on the direct payoff influences among the player population. In addition to a number of nice computational properties, these models have close connections to well-studied graphical models for probabilistic inference in machine learning and statistics.

7.1 Introduction

Representing multiplayer games with large player populations in the normal form is undesirable for both practical and conceptual reasons. On the practical side, the number of parameters that must be specified grows exponentially with the size of the population. On the conceptual side, the normal form may fail to capture structure that is present in the strategic interaction, and which can aid understanding of the game and computation of its equilibria. For this reason, there have been many proposals for parametric multiplayer game representations that are more succinct than the normal form, and attempt to model naturally arising structural properties. Examples include congestion and potential games and related models (Monderer and Shapley, 1996; Rosenthal, 1973).

Graphical games are a representation of multiplayer games meant to capture and exploit locality or sparsity of direct influences. They are most appropriate for large population games in which the payoffs of each player are determined by the actions of only a small subpopulation. As such, they form a natural counterpart to earlier parametric models. Whereas congestion games and related models implicitly assume a large number of weak influences on each player, graphical games are suitable when there is a small number of strong influences.

Graphical games adopt a simple graph-theoretic model. A graphical game is described at the first level by an undirected graph G in which players are identified with

vertices. The semantics of the graph are that a player or vertex i has payoffs that are entirely specified by the actions of i and those of its neighbor set in G. Thus G alone may already specify strong qualitative constraints or structure over the direct strategic influences in the game. To fully describe a graphical game, we must additionally specify the numerical payoff functions to each player – but now the payoff to player i is a function only of the actions of i and its neighbors, rather than the actions of the entire population. In the many natural settings where such local neighborhoods are much smaller than the overall population size, the benefits of this parametric specification over the normal form are already considerable.

But several years of research on graphical games has demonstrated that the advantages of this model extend well beyond simple parsimony – rather, they are computational, structural, and interdisciplinary as well. We now overview each of these in turn.

Computational. Theoretical computer science has repeatedly established that strong but naturally occurring constraints on optimization and other problems can be exploited algorithmically, and game theory is no exception. Graphical games provide a rich language in which to state and explore the computational benefits of various restrictions on the interactions in a large-population game. As we shall see, one fruitful line of research has investigated topological restrictions on the underlying graph G that yield efficient algorithms for various equilibrium computations.

Structural. In addition to algorithmic insights, graphical games also provide a powerful framework in which to examine the relationships between the network structure and strategic outcomes. Of particular interest is whether and when the local interactions specified by the graph G alone (i.e., the topology of G, regardless of the numerical specifications of the payoffs) imply nontrivial structural properties of equilibria. We will examine an instance of this phenomenon in some detail.

Interdisciplinary. Part of the original motivation for graphical games came from earlier models familiar to the machine learning, AI and statistics communities – collectively known as graphical models for probabilistic inference, which include Bayesian networks, Markov networks, and their variants. Broadly speaking, both graphical models for inference and graphical games represent complex interactions between a large number of variables (random variables in one case, the actions of players in a game in the other) by a graph combined with numerical specification of the interaction details. In probabilistic inference the interactions are stochastic, whereas in graphical games they are strategic (best response). As we shall discuss, the connections to probabilistic inference have led to a number of algorithmic and representational benefits for graphical games.

In this chapter we will overview graphical games and the research on them to date. We will center our discussion around two main technical results that will be examined in some detail, and are chosen to illustrate the computational, structural, and interdisciplinary benefits discussed above. These two case studies will also serve as natural vehicles to survey the broader body of literature on graphical games.

The first problem we shall examine is the computation of Nash equilibria in graphical games in which the underlying graph G is a tree (or certain generalizations of trees). Here we will discuss a natural two-pass algorithm for computing Nash equilibria requiring only the local exchange of "conditional equilibrium" information over the

edges of G. This algorithm comes in two variations – one that runs in time polynomial in the representation size of the graphical game and computes (a compact representation of) *approximations* of all Nash equilibria, and another that runs in exponential time but computes (a compact representation of) all Nash equilibria *exactly*. We will discuss a number of generalizations of this algorithm, including one known as **NashProp**, which has close ties to the well-known belief propagation algorithm in probabilistic inference. Together these algorithms provide examples of the algorithmic exploitation of structural restrictions on the graph.

The second problem we shall examine is the representation and computation of the correlated equilibria of a graphical game. Here we will see that there is a satisfying and natural connection between graphical games and the probabilistic models known as Markov networks, which can succinctly represent high-dimensional multivariate probability distributions. More specifically, we shall show that any graphical game with graph G can have all of its correlated equilibria (up to payoff equivalence) represented by a Markov network with the same network structure. If we adopt the common view of correlated equilibria as permitting "shared" or "public" randomization (the source of the correlations) – whereas Nash equilibria permit only "private" randomization or mixed strategies – this result implies that the shared randomization can actually be distributed locally throughout the graph, and that distant parties need not be (directly) correlated. From the rich tools developed for independence analysis in Markov networks, it also provides a compact representation of a large number of independence relationships between player actions that may be assumed at (correlated) equilibrium. The result thus provides a good example of a direct connection between graph structure and equilibrium properties, as well as establishing further ties to probabilistic inference. We shall also discuss the algorithmic benefits of this result.

After studying these two problems in some detail, we will briefly overview recent research incorporating network structure into other game-theoretic and economic settings, such as exchange economies (Arrow-Debreu, Fischer and related models). Again the emphasis will be on computational aspects of these models, and on the relationship between graph structure and equilibrium properties.

7.2 Preliminaries

In this section we shall provide formal definitions for graphical games, along with other needed definitions, terminology, and notation. We begin with notions standard to classical multiplayer game theory.

A multiplayer game consists of n players, each with a finite set of *pure strategies* or actions available to them, along with a specification of the *payoffs* to each player. Throughout the chapter, we use a_i to denote the action chosen by player i. For simplicity we will assume a binary action space, so $a_i \in \{0, 1\}$. (The generalization of the results examined here to the multiaction setting is straightforward.) The payoffs to player i are given by a table or matrix M_i, indexed by the joint action $\vec{a} \in \{0, 1\}^n$. The value $M_i(\vec{a})$, which we assume without loss of generality to lie in the interval $[0, 1]$, is the payoff to player i resulting from the joint action $\vec{a}$. Multiplayer games described in this way are referred to as *normal form* games.

The actions 0 and 1 are the *pure strategies* of each player, while a *mixed* strategy for player i is given by the probability $p_i \in [0, 1]$ that the player will play 0. For any joint mixed strategy, given by a product distribution $\vec{p}$, we define the expected payoff to player i as $M_i(\vec{p}) = \mathbf{E}_{\vec{a}\sim\vec{p}}[M_i(\vec{a})]$, where $\vec{a} \sim \vec{p}$ indicates that each a_j is 0 with probability p_j and 1 with probability $1 - p_j$ independently. When we introduce correlated equilibria below, we shall allow the possibility that the distribution over $\vec{a}$ is not a product distribution, but has correlations between the a_i.

We use $\vec{p}[i : p_i']$ to denote the vector (product distribution) which is the same as $\vec{p}$ except in the ith component, where the value has been changed to p_i'. A *Nash equilibrium (NE)* for the game is a mixed strategy $\vec{p}$ such that for any player i, and for any value $p_i' \in [0, 1]$, $M_i(\vec{p}) \geq M_i(\vec{p}[i : p_i'])$. (We say that p_i is a *best response* to the rest of $\vec{p}$.) In other words, no player can improve their expected payoff by deviating unilaterally from an NE. The classic theorem of Nash (1951) states that for any game, there exists an NE in the space of joint mixed strategies.

We will also use a straightforward (additive) definition for *approximate* Nash equilibria. An *ϵ-Nash equilibrium* is a mixed strategy $\vec{p}$ such that for any player i, and for any value $p_i' \in [0, 1]$, $M_i(\vec{p}) + \epsilon \geq M_i(\vec{p}[i : p_i'])$. (We say that p_i is an *ϵ-best response* to the rest of $\vec{p}$.) Thus, no player can improve their expected payoff by more than ϵ by deviating unilaterally from an approximate NE.

We are now ready to introduce the graphical game model. In a *graphical game*, each player i is represented by a vertex in an undirected graph G. We use $N(i) \subseteq \{1, \ldots, n\}$ to denote the *neighborhood* of player i in G – that is, those vertices j such that the edge (i, j) appears in G. By convention $N(i)$ always includes i itself as well. If $\vec{a}$ is a joint action, we use $\vec{a}^{\,i}$ to denote the projection of $\vec{a}$ onto just the players in $N(i)$.

Definition 7.1 A *graphical game* is a pair $(G, \mathcal{M})$, where G is an undirected graph over the vertices $\{1, \ldots, n\}$, and $\mathcal{M}$ is a set of n *local game matrices*. For any joint action $\vec{a}$, the local game matrix $M_i \in \mathcal{M}$ specifies the payoff $M_i(\vec{a}^{\,i})$ for player i, which depends only on the actions taken by the players in $N(i)$.

Remarks. Graphical games are a (potentially) more compact way of representing games than standard normal form. In particular, rather than requiring a number of parameters that is exponential in the number of players n, a graphical game requires a number of parameters that is exponential only in the size d of the largest local neighborhood. Thus if $d \ll n$ – that is, the number of direct influences on any player is much smaller than the overall population size – the graphical game representation is dramatically smaller than the normal form. Note that we can represent any normal form game as a graphical game by letting G be the complete graph, but the representation is only useful when a considerably sparser graph can be found. It is also worth noting that although the payoffs to player i are determined only by the actions of the players in $N(i)$, *equilibrium* still requires *global* coordination across the player population – if player i is connected to player j who is in turn connected to player k, then i and k indirectly influence each other via their mutual influence on the payoff of j. How local influences propagate to determine global equilibrium outcomes is one of the computational challenges posed by graphical games.

In addition to Nash equilibrium, we will also examine graphical games in the context of *correlated equilibria (CE)*. CE (Aumann, 1974) generalize NE, and can be viewed as

(possibly arbitrary) distributions $P(\vec{a})$ over joint actions satisfying a certain conditional expectation property.

The intuition behind CE can be described as follows. Imagine that there is a trusted party that faithfully draws a joint action $\vec{a}$ according to distribution P, and distributes to each player i only their private component a_i. If P is a product distribution, as in the NE case, then due to the independence between all players the revelation of a_i does not condition player i's beliefs over the play of others. For general P, however, this is not true. The CE condition asks that the expected payoff to i if he is "obedient" and plays a_i be at least as great the amount i could earn by "cheating" and deviating to play a different action. In other words, in Bayesian terms, despite the observation of a_i updating the posterior distribution over the other player actions from i's perspective, it is still payoff-optimal for i to play a_i. This leads to the formal definition below, in which for any given joint distribution $P(\vec{a})$ over player actions and $b \in \{0, 1\}$, we let $P_{a_i=b}$ denote the distribution on $\vec{a}$ conditioned on the event that $a_i = b$.

Definition 7.2 A *correlated equilibrium (CE)* for a two-action normal form game is a distribution $P(\vec{a})$ over actions satisfying

$$\forall i \in \{1, ..., n\}, \forall b \in \{0, 1\} : \mathbf{E}_{\vec{a} \sim P_{a_i=b}}[M_i(\vec{a})] \geq \mathbf{E}_{\vec{a} \sim P_{a_i=b}}[M_i(\vec{a}[i : \neg b])]$$

The expectation $\mathbf{E}_{\vec{a} \sim P_{a_i=b}}[M_i(\vec{a})]$ is over those cases in which the value $a_i = b$ is revealed to player i, who proceeds to "honestly" play $a_i = b$. The expectation $\mathbf{E}_{\vec{a} \sim P_{a_i=b}}[M_i(\vec{a}[i : \neg b])]$ is over the same cases, but now player i unilaterally deviates to play $a_i = \neg b$, whereas the other players faithfully play from the conditional distribution $P_{a_i=b}$. It is straightforward to generalize this definition to the multiaction case – again, we demand that it be optimal for each player to take the action provided by the trusted party, despite the conditioning information revealed by this action.

Remarks. CE offers a number of conceptual and computational advantages over NE, including the facts that new and sometimes more "fair" payoffs can be achieved, that CE can be computed efficiently for games in standard normal form (though recall that "efficiently" here means exponential in the number of players, an issue we shall address), and that CE are the convergence notion for several natural "no-regret" learning algorithms (Foster and Vohra, 1999). Furthermore, it has been argued that CE is the natural equilibrium concept consistent with the Bayesian perspective (Aumann, 1987; Foster and Vohra, 1997). One of the most interesting aspects of CE is that they broaden the set of "rational" solutions for normal form games without the need to address often difficult issues such as stability of coalitions and payoff imputations (Aumann, 1987). The traffic signal is often cited as an informal everyday example of CE, in which a single bit of shared information allows a fair split of waiting times (Owen, 1995). In this example, no player stands to gain greater payoff by unilaterally deviating from the correlated play, for instance by "running a light." This example also illustrates a common alternative view of CE, in which correlations arise as a result of "public" or "shared" random bits (in addition to the "private" random bits allowed in the standard mixed strategies or product distributions of NE). Here the state of the traffic light itself (which can be viewed as a binary random variable, alternately displayed as red and green to orthogonal streets) provides the shared randomization.

7.3 Computing Nash Equilibria in Tree Graphical Games

In this section, we describe the first and perhaps most basic algorithm exploiting the advantages of graphical game representation for the purposes of equilibrium computation. The case considered is that in which the underlying graph G is a tree. While obviously a strong restriction on the topology, we shall see that this case already presents nontrivial computational challenges, which in turn force the development of algorithmic tools that can be generalized beyond trees to obtain a more general heuristic known as.

NashProp. We first describe the algorithm **TreeNash** at a high level, leaving certain important implementation details unspecified, because it is conceptually advantageous to do so. We then describe two instantiations of the missing details – yielding one algorithm that runs in polynomial time and provably computes approximations of all equilibria, and another algorithm that runs in exponential time and provably computes all exact equilibria.

We begin with some notation and concepts needed for the description of **TreeNash**. In order to distinguish parents from children in the tree, it will be convenient to treat players/vertices symbolically (such as U, V, and W) rather than by integer indices, so we use M_V to denote the local game matrix for the player identified with player/vertex V. We use capital letters to denote vertex/players to distinguish them from their chosen actions, for which we shall use lower case. If G is a tree, we choose an arbitrary vertex as the root (which we visualize as being at the bottom, with the leaves at the top). Any vertex on the path from a vertex V to the root will be called *downstream* from V, and any vertex on a path from V to any leaf will be called *upstream* from V. Thus, each vertex other than the root has exactly one downstream neighbor (or child), and perhaps many upstream neighbors (or parents). We use $\mathrm{UP}_G(V)$ to denote the set of all vertices in G that are upstream from V, including V by definition.

Suppose that V is the child of U in G. We let G^U denote the subgraph induced by the vertices in $\mathrm{UP}_G(U)$ – that is, the subtree of G rooted at U. If $v \in [0, 1]$ is a mixed strategy for player (vertex) V, $\mathcal{M}^U_{V=v}$ will denote the subset of payoff matrices in $\mathcal{M}$ corresponding to the vertices in $\mathrm{UP}_G(U)$, with the modification that the game matrix M_U is collapsed by one index by fixing $V = v$. We can think of an NE for the graphical game $(G^U, \mathcal{M}^U_{V=v})$ as a *conditional* equilibrium "upstream" from U (inclusive) – that is, an equilibrium for G^U *given* that V plays v. Here we are simply exploiting the fact that since G is a tree, fixing a mixed strategy v for the play of V isolates G^U from the rest of G.

Now suppose that vertex V has k parents $U_1, \ldots, U_k$, and the single child W. We now describe the data structures sent from each U_i to V, and in turn from V to W, on the downstream pass of **TreeNash**. Each parent U_i will send to V a binary-valued "table" $T(v, u_i)$. The table is indexed by the *continuum* of possible values for the mixed strategies $v \in [0, 1]$ of V and $u_i \in [0, 1]$ of U_i, $i = 1, \ldots, k$. The semantics of this table will be as follows: for any pair (v, u_i), $T(v, u_i)$ will be 1 if and only if there exists an NE for $(G^{U_i}, \mathcal{M}^{U_i}_{V=v})$ in which $U_i = u_i$. Note that we will slightly abuse notation by letting $T(v, u_i)$ refer to both the entire table sent from U_i to V, and the particular value associated with the pair (v, u_i), but the meaning will be clear from the context.

Algorithm **TreeNash**
Inputs: Graphical game $(G, \mathcal{M})$ in which G is a tree.
Output: A Nash equilibrium for $(G, \mathcal{M})$.

(i) Compute a depth-first ordering of the vertices of G.
(ii) (**Downstream Pass**) For each vertex V in depth-first order:
  (a) Let vertex W be the child of V (or nil if V is the root).
  (b) For all $w, v \in [0, 1]$, initialize $T(w, v)$ to be 0 and the witness list for $T(w, v)$ to be empty.
  (c) If V is a leaf (base case):
    1. For all $w, v \in [0, 1]$, set $T(w, v)$ to be 1 if and only if $V = v$ is a best response to $W = w$ (as determined by the local game matrix M_V).
  (d) Else (inductive case, V is an internal vertex):
    1. Let $\vec{U} = (U_1, \ldots, U_k)$ be the parents of V; let $T(v, u_i)$ be the table passed from U_i to V on the downstream pass.
    2. For all $w, v \in [0, 1]$ and for all joint mixed strategies $\vec{u} = (u_1, \ldots, u_k)$ for $\vec{U}$: If $V = v$ is a best response to $W = w$, $\vec{U} = \vec{u}$ (as determined by the local game matrix M_V), and $T(v, u_i) = 1$ for $i = 1, \cdots, k$, set $T(w, v)$ to be 1 and add $\vec{u}$ to the witness list for $T(w, v)$.
  (e) Pass the table $T(w, v)$ from V to W.
(iii) (**Upstream Pass**) For each vertex V in reverse depth-first ordering (starting at the root):
  (a) Let $\vec{U} = (U_1, \ldots, U_k)$ be the parents of V (or the empty list if V is a leaf); let W be the child of V (or nil if V is the root), and (w, v) the values passed from W to V on the upstream pass.
  (b) Label V with the value v.
  (c) (Non-deterministically) Choose any witness $\vec{u}$ to $T(w, v) = 1$.
  (d) For $i = 1, \ldots, k$, pass (v, u_i) from V to U_i.

Figure 7.1. Algorithm **TreeNash** for computing NE of tree graphical games.

Since v and u_i are continuous variables, it is not obvious that the table $T(v, u_i)$ can be represented compactly, or even finitely, for arbitrary vertices in a tree. For now we will simply assume a finite representation, and shortly discuss how this assumption can be met in two different ways.

The initialization of the downstream pass of the algorithm begins at the leaves of the tree, where the computation of the tables is straightforward. If U is a leaf and V its only child, then $T(v, u) = 1$ if and only if $U = u$ is a best response to $V = v$ (Step (ii) (c) of Figure 7.1).

Assuming for induction that each U_i sends the table $T(v, u_i)$ to V, we now describe how V can compute the table $T(w, v)$ to pass to its child W (Step (ii) (d)2 of Figure 7.1). For each pair (w, v), $T(w, v)$ is set to 1 if and only if there exists a vector of mixed strategies $\vec{u} = (u_1, \ldots, u_k)$ (called a *witness*) for the parents $\vec{U} = (U_1, \ldots, U_k)$ of V such that

(i) $T(v, u_i) = 1$ for all $1 \leq i \leq k$; and
(ii) $V = v$ is a best response to $\vec{U} = \vec{u}$, $W = w$.

Note that there may be more than one witness for $T(w, v) = 1$. In addition to computing the value $T(w, v)$ on the downstream pass of the algorithm, V will also keep a list of the witnesses $\vec{u}$ for each pair (w, v) for which $T(w, v) = 1$ (Step ii(d)2 of Figure 7.1). These witness lists will be used on the upstream pass.

To see that the semantics of the tables are preserved by the computation just described, suppose that this computation yields $T(w, v) = 1$ for some pair (w, v), and let $\vec{u}$ be a witness for $T(w, v) = 1$. The fact that $T(v, u_i) = 1$ for all i (condition (7.3) above) ensures by induction that if V plays v, there are upstream NE in which each $U_i = u_i$. Furthermore, v is a best response to the local settings $U_1 = u_1, \ldots, U_k = u_k$, $W = w'$ (condition (7.3) above). Therefore, we are in equilibrium upstream from V. On the other hand, if $T(w, v) = 0$, it is easy to see there can be no equilibrium in which $W = w$, $V = v$. Note that the existence of an NE guarantees that $T(w, v) = 1$ for at least one (w, v) pair.

The downstream pass of the algorithm terminates at the root Z, which receives tables $T(z, y_i)$ from each parent Y_i. Z simply computes a one-dimensional table $T(z)$ such that $T(z) = 1$ if and only if for some witness $\vec{y}$, $T(z, y_i) = 1$ for all i, and z is a best response to $\vec{y}$.

The upstream pass begins by Z choosing any z for which $T(z) = 1$, choosing any witness $(y_1, \ldots, y_k)$ to $T(z) = 1$, and then passing both z and y_i to each parent Y_i. The interpretation is that Z will play z, and is "instructing" Y_i to play y_i. Inductively, if a vertex V receives a value v to play from its downstream neighbor W, and the value w that W will play, then it must be that $T(w, v) = 1$. So V chooses a witness $\vec{u}$ to $T(w, v) = 1$, and passes each parent U_i their value u_i as well as v (Step (iii) of Figure 7.1). Note that the semantics of $T(w, v) = 1$ ensure that $V = v$ is a best response to $\vec{U} = \vec{u}$, $W = w$.

We have left the choices of each witness in the upstream pass unspecified or nondeterministic to emphasize that the tables and witness lists computed represent *all* the NE. The upstream pass can be specialized to find a number of specific NE of interest, including player optimum (NE maximizing expected reward to a chosen player), social optimum (NE maximizing total expected reward, summed over all players), and welfare optimum (NE maximizing expected reward to the player whose expected reward is smallest).

Modulo the important details regarding the representation of the tables $T(w, v)$, which we discuss next, the arguments provided above establish the following formal result.

Theorem 7.3 *Let $(G, \mathcal{M})$ be any graphical game in which G is a tree. Algorithm* **TreeNash** *computes a Nash equilibrium for $(G, \mathcal{M})$. Furthermore, the tables and witness lists computed by the algorithm represent all Nash equilibria of $(G, \mathcal{M})$.*

7.3.1 An Approximation Algorithm

In this section, we sketch one instantiation of the missing details of algorithm **TreeNash** that yields a polynomial-time algorithm for computing *approximate* NE for the tree

game $(G, \mathcal{M})$. The approximation can be made arbitrarily precise with greater computational effort.

Rather than playing an arbitrary mixed strategy in $[0, 1]$, each player will be constrained to play a *discretized* mixed strategy that is a multiple of τ, for some τ to be determined by the analysis. Thus, player i plays $q_i \in \{0, \tau, 2\tau, \ldots, 1\}$, and the joint strategy $\vec{q}$ falls on the discretized τ-grid $\{0, \tau, 2\tau, \ldots, 1\}^n$. In algorithm **TreeNash**, this will allow each table $T(v, u)$ (passed from vertex U to child V) to be represented in discretized form as well: only the $1/\tau^2$ entries corresponding to the possible τ-grid choices for U and V are stored, and all computations of best responses in the algorithm are modified to be approximate best responses.

To quantify how the choice of τ will influence the quality of the approximate equilibria found (which in turn will determine the computational efficiency of the approximation algorithm), we appeal to the following lemma. We note that this result holds for arbitrary graphical games, not only trees.

Lemma 7.4 *Let G be a graph of maximum degree d, and let $(G, \mathcal{M})$ be a graphical game. Let $\vec{p}$ be a Nash equilibrium for $(G, \mathcal{M})$, and let $\vec{q}$ be the nearest (in L_1 metric) mixed strategy on the τ-grid. Then $\vec{q}$ is a $d\tau$-NE for $(G, \mathcal{M})$.*

The proof of Lemma 7.4, which we omit, follows from a bound on the L_1 distance for product distributions along with an argument that the strategic properties of the NE are preserved by the approximation. We note that the original paper (Kearns et al., 2001) used a considerably worse L_1 bound that was exponential in d. However, the algorithm remains exponential in d simply due to the representational complexity of the local product distributions. The important point is that τ needs to depend only on the *local neighborhood* size d, not the total number of players n.

It is now straightforward to describe **ApproximateTreeNash**. This algorithm is identical to algorithm **TreeNash** with the following exceptions:

- The algorithm now takes an additional input ϵ.
- For any vertex U with child V, the table $T(u, v)$ will contain only entries for u and v multiples of τ.
- All computations of best responses in algorithm **TreeNash** become computations of ϵ-best responses in algorithm **ApproximateTreeNash**.

For the running time analysis, we simply note that each table has $(1/\tau)^2$ entries, and that the computation is dominated by the downstream calculation of the tables (Step (ii)(d) of algorithm **TreeNash**). This requires ranging over all table entries for all k parents, a computation of order $((1/\tau)^2)^k$. By appropriately choosing the value of τ in order to obtain the required ϵ-approximations, we obtain the following theorem.

Theorem 7.5 *Let $(G, \mathcal{M})$ be a graphical game in which G is a tree with n vertices, and in which every vertex has at most d parents. For any $\epsilon > 0$, let $\tau = O(\epsilon/d)$. Then* **ApproximateTreeNash** *computes an ϵ-Nash equilibrium for $(G, \mathcal{M})$. Furthermore, for* every *exact Nash equilibrium, the tables and witness lists computed by the algorithm contain an ϵ-Nash equilibrium that is within τ of this exact equilibrium (in L_1 norm). The running time of the algorithm is*

polynomial in $1/\epsilon$, n *and* 2^d, *which is polynomial in the size of the representation* $(G, \mathcal{M})$.

7.3.2 An Exact Algorithm

By approximating the continuously indexed tables $T(u, v)$ in discretized form, the algorithm developed in Section 7.3.1 side-stepped not only the exact computation but also a fundamental question about the $T(u, v)$ – namely, do the regions $\{(u, v) \in [0, 1]^2 : T(u, v) = 1\}$ have any interesting geometric structure? It turns out the answer in the case of trees is affirmative, and can be used in developing an alternate instantiation of the general **TreeNash** algorithm of Section 7.3 – one that yields an algorithm for computing (all) *exact* equilibria, but in time that is exponential in the number of players n rather than only the degree d.

Although the details are beyond our scope, it is possible to show via an inductive argument (where again the leaves of G serve as the base cases) that in any tree graphical game, for any of the tables $T(u, v)$ defined by **TreeNash**, the region $\{(u, v) \in [0, 1]^2 : T(u, v) = 1\}$ can be represented by a finite union of (axis-aligned) rectangular regions in $[0, 1]^2$ (i.e., regions that are defined as products of closed intervals $[a, a'] \times [b, b']$ for some $0 \leq a \leq a' \leq 1, 0 \leq b \leq b' \leq 1$). The induction shows that the number of such regions multiplies at each level as we progress downstream toward the root, yielding a worst-case bound on the number of rectangular regions that is exponential in n.

This simple (if exponential in n) geometric representation of the tables $T(u, v)$ permits the development of an alternative algorithm **ExactTreeNash**, which is simply the abstract algorithm **TreeNash** with the tables represented by unions of rectangles (and with associated implementations of the necessary upstream and downstream computations).

Theorem 7.6 *There is an algorithm* **ExactTreeNash** *that computes an exact Nash equilibrium for any tree graphical game* $(G, \mathcal{M})$. *Furthermore, the tables computed by the algorithm represent all Nash equilibria of* $(G, \mathcal{M})$. *The algorithm runs in time exponential in the number of vertices of* G.

7.3.3 Extensions: NashProp and Beyond

At this point it is of course natural to ask what can be done when the underlying graph of a graphical game is not a tree. Remaining close to the development so far, it is possible to give an *heuristic* generalization of algorithm **ApproximateTreeNash** to the setting in which the graph G is arbitrary. This algorithm is known as **NashProp**, which we will now briefly sketch. By heuristic we mean that the algorithm is well-defined and will terminate on any graphical game; but unlike **ApproximateTreeNash**, the running time is not guaranteed to be polynomial in the size of the input graphical game. (In general, we should expect provably efficient algorithms for equilibrium computation to require *some* topological restriction, since allowing G to be the complete graph reduces to the classical normal form representation.)

Recall that the main computation at vertex V in **ApproximateTreeNash** was the computation of the downstream table $T(w, v)$ from the upstream tables $T(v, u_i)$. This

assumed an underlying orientation to the tree that allowed V to know which of its neighbors were in the direction of the leaves (identified as the U_i) and which single neighbor was in the direction of the root (identified as W). The easiest way to describe **NashProp** informally is to say that each V does this computation once for *each* of its neighbors, each time "pretending" that this neighbor plays the role of the downstream neighbor W in **ApproximateTreeNash**, and the remaining neighbors play the roles of the upstream U_i. If all discretized table entries are initialized to the value of 1,[1] it easy to show that the only possible effect of these local computations is to change table values from 1 to 0, which in effect refutes conditional NE assertions when they violate best-response conditions. In other words, the tables will all *converge* (and in fact, in time polynomial in the size of the graphical game) – however, unlike in **ApproximateTreeNash**, the tables do not represent the set of all approximate NE, but a superset. This necessitates a second phase to the algorithm that employs more traditional search heuristics in order to find a true equilibrium, and it is this second phase that may be computationally expensive.

One of the merits of **NashProp** is that the first (table computation) phase can be viewed as an instance of constraint satisfaction programming (CSP), which in turn plays an important role in many algorithms for probabilistic inference in Bayesian networks, Markov networks, and related models. The **NashProp** algorithm was also inspired by, and bears a fair similarity to, the well-known belief propagation algorithm for Bayesian network inference. We shall see other connections to these models arise in our examination of correlated equilibria in graphical games, which we turn to now.

7.4 Graphical Games and Correlated Equilibria

Our second case study is an examination of graphical games and correlated equilibrium. As has already been noted, if we are fortunate enough to be able to accurately represent a multiplayer game we are interested in as a graphical game with small degree, the representational benefits purely in terms of parameter reduction may be significant. But this is still a rather cosmetic kind of parsimony. We shall see a much deeper variety in the context of correlated equilibrium.

The first issue that arises in this investigation is the problem of *representing* correlated equilibria. Recall that NE may be viewed as a special case of CE in which the distribution $P(\vec{a})$ is a product distribution. Thus, however computationally difficult it may be to find an NE, at least the object itself can be succinctly represented – it is simply a mixed strategy profile $\vec{p}$, whose length equals the number of players n. Despite their aforementioned advantages, in moving to CE we open a representational Pandora's Box, since even in very simple graphical games there may be correlated equilibria of essentially arbitrary complexity. For example, the CE of a game always include all mixture distributions of NE, so any game with an exponential number of NE can yield extremely complex CE. Such games can be easily constructed with very simple graphs.

[1] Note that in the description of **TreeNash** in Figure 7.1 it was more convenient to initialize the table values to 0, but the change is cosmetic.

More generally, whereas by definition in an NE all players are independent, in a CE there may be arbitrary high-order correlations.

In order to maintain the succinctness of graphical games, some way of addressing this distributional complexity is required. For this we turn to another, older graph-theoretic formalism – namely, undirected graphical models for probabilistic inference, also known as *Markov networks* (Lauritzen, 1996). We will establish a natural and powerful relationship between a graphical game and a certain associated Markov network. Like the graphical game, the associated Markov network is a graph over the players. While the interactions between vertices in the graphical game are entirely *strategic* and given by local payoff matrices, the interactions in the associated Markov network are entirely *probabilistic* and given by local potential functions. The graph of the associated Markov network retains the parsimony of the graphical game.

We will show that the associated Markov network is sufficient for representing *any* correlated equilibria of the graphical game (up to expected payoff equivalence). In other words, the fact that a multiplayer game can be succinctly represented by a graph implies that its entire space of CE outcomes can be represented graphically with comparable succinctness. This result establishes a natural relationship between graphical games and modern probabilistic modeling. We will also briefly discuss the computational benefits of this relationship.

7.4.1 Expected Payoff and Local Neighborhood Equivalence

Our effort to succinctly model the CE of a graphical game consists of two steps. In the first step, we argue that it is not necessary to model *all* the correlations that might arise in a CE, but only those required to represent all of the possible (expected payoff) outcomes for the players. In the second step, we show that the remaining correlations can be represented by a Markov network. For these two steps we respectively require two equivalence notions for distributions – *expected payoff equivalence* and *local neighborhood* equivalence. We shall show that there is a natural *subclass* of the set of all CE of a graphical game, based on expected payoff equivalence, whose representation size is linearly related to the representation size of the graphical game.

Definition 7.7 Two distributions P and Q over joint actions $\vec{a}$ are *expected payoff equivalent*, denoted $P \equiv_{\text{EP}} Q$, if P and Q yield the same expected payoff vector: for each i, $\mathbf{E}_{\vec{a}\sim P}[M_i(\vec{a})] = \mathbf{E}_{\vec{a}\sim Q}[M_i(\vec{a})]$.

Note that merely finding distributions giving the same payoffs as the CE is not especially interesting *unless those distributions are themselves CE* – we want to preserve the strategic properties of the CE, not only its payoffs. Our primary tool for accomplishing this goal will be the notion of local neighborhood equivalence, or the preservation of local marginal distributions. Below we establish that local neighborhood equivalence both implies payoff equivalence and preserves the CE property. In the following subsection, we describe how to represent this natural subclass in a certain Markov network whose structure is closely related to the structure of the graphical game.

Expected payoff equivalence of two distributions is, in general, dependent upon the reward matrices of a graphical game. Let us consider the following (more stringent) equivalence notion, which is based only on the graph G of a game.

Definition 7.8 For a graph G, two distributions P and Q over joint actions $\vec{a}$ are *local neighborhood equivalent* with respect to G, denoted $P \equiv_{\text{LN}} Q$, if for all players i, and for all settings $\vec{a}^{\,i}$ of $N(i)$, $P(\vec{a}^{\,i}) = Q(\vec{a}^{\,i})$.

In other words, the marginal distributions over all local neighborhoods defined by G are identical. Since the graph is always clear from context, we shall just write $P \equiv_{\text{LN}} Q$. The following lemma establishes that local neighborhood equivalence is indeed a stronger notion of equivalence than expected payoff.

Lemma 7.9 *For all graphs G, for all joint distributions P and Q on actions, and for all graphical games with graph G, if $P \equiv_{LN} Q$ then $P \equiv_{EP} Q$. Furthermore, for any graph G and joint distributions P and Q, there exist payoff matrices $\mathcal{M}$ such that for the graphical game $(G, \mathcal{M})$, if $P \not\equiv_{LN} Q$ then $P \not\equiv_{EP} Q$.*

PROOF The first statement follows from the observation that the expected payoff to player i depends only on the marginal distribution of actions in $N(i)$. To prove the second statement, if $P \not\equiv_{\text{LN}} Q$, then there must exist a player i and a joint action $\vec{a}^{\,i}$ for its local neighborhood which has a different probability under P and Q. Simply set $M_i(\vec{a}^{\,i}) = 1$ and $M_i = 0$ elsewhere. Then i has a different payoff under P and Q, and so $P \not\equiv_{\text{EP}} Q$. □

Thus local neighborhood equivalence implies payoff equivalence, but the converse is not true in general (although there exists some payoff matrices where the converse is correct). We now establish that local neighborhood equivalence also preserves CE. It is important to note that this result does *not* hold for expected payoff equivalence.

Lemma 7.10 *For any graphical game $(G, \mathcal{M})$, if P is a CE for $(G, \mathcal{M})$ and $P \equiv_{LN} Q$ then Q is a CE for $(G, \mathcal{M})$.*

PROOF The lemma follows by noting that the CE expectation equations are dependent only upon the marginal distributions of local neighborhoods, which are preserved in Q. □

While explicitly representing *all* CE is infeasible even in simple graphical games, we next show that we *can* concisely represent, in a single model, all CE *up to local neighborhood (and therefore payoff) equivalence*. The amount of space required is comparable to that required to represent the graphical game itself, and allows us to explore or enumerate the different outcomes achievable in the space of CE.

7.4.2 Correlated Equilibria and Markov Nets

In the same way that graphical games provide a concise language for expressing local interaction in game theory, *Markov networks* exploit undirected graphs for expressing local interaction in probability distributions. It turns out that (a special case of) Markov networks are a natural and powerful language for expressing the CE of a graphical game, and that there is a close relationship between the graph of the game and its associated Markov network graph. We begin with the necessary definitions.

Definition 7.11 A *local Markov network* is a pair $M \equiv (G, \Psi)$, where

- G is an undirected graph on vertices $\{1, \ldots, n\}$;
- Ψ is a set of *potential functions*, one for each local neighborhood $N(i)$, mapping binary assignments of values of $N(i)$ to the range $[0, \infty)$:

$$\Psi \equiv \{\psi_i : i = 1, \ldots, n; \psi_i : \{\vec{a}^{\,i}\} \to [0, \infty)\},$$

where $\{\vec{a}^{\,i}\}$ is the set of all $2^{|N(i)|}$ settings to $N(i)$.

A local Markov network M defines a probability distribution P_M as follows. For any binary assignment $\vec{a}$ to the vertices, we define

$$P_M(\vec{a}) \equiv \frac{1}{Z}\left(\prod_{i=1}^{n} \psi_i(\vec{a}^{\,i})\right),$$

where $Z = \sum_{\vec{a}} \prod_{i=1}^{n} \psi_i(\vec{a}^{\,i}) > 0$ is the normalization factor.

Note that any joint distribution can be represented as a local Markov network on a sufficiently dense graph: if we let G be the complete graph then we simply have a single potential function over the entire joint action space $\vec{a}$. However, if d is the size of the maximal neighborhood in G, then the representation size of a distribution in this network is $O(n2^d)$, a considerable savings over a tabular representation if $d \ll n$.

Local Markov networks are a special case of Markov networks, a well-studied probabilistic model in AI and statistics (Lauritzen, 1996; Pearl, 1988). A Markov network is typically defined with potential functions ranging over settings of *maximal cliques* in the graph, rather than local neighborhoods. Another approach we could have taken is to transform the graph G to a graph G', which forms cliques of the local neighborhoods $N(i)$, and then used standard Markov networks over G' as opposed to local Markov networks over G. However, this can sometimes result in an unnecessary exponential blow-up of the size of the model when the resulting maximal cliques are much larger than the original neighborhoods. For our purposes, it is sufficient to define the potential functions over just local neighborhoods (as in our definition) rather than maximal cliques in G', which avoids this potential blow-up.

The following technical lemma establishes that a local Markov network always suffices to represent a distribution up to local neighborhood equivalence.

Lemma 7.12 *For all graphs G, and for all joint distributions P over joint actions, there exists a distribution Q that is representable as a local Markov network with graph G such that $Q \equiv_{LN} P$ with respect to G.*

PROOF The objective is to find a single distribution Q that is consistent with the players' local neighborhood marginals under P and is also a Markov network with graph G. For this we shall sketch the application of methods from probabilistic inference and maximum entropy models to show that the maximum entropy distribution Q^*, subject to $P \equiv_{\mathrm{LN}} Q^*$, is a local Markov network. The sketch below follows the classical treatment of this topic (Berger et al., 1996; Lauritzen and Spiegelhalter, 1998; Dawid and Lauritzen, 1993) and is included for completeness.

Formally, we wish to show that the solution to the following constrained maximum entropy problem is representable in G:

$$Q^* = \mathrm{argmax}_Q H(Q) \equiv \mathrm{argmax}_Q \sum_{\vec{a}} Q(\vec{a}) \log(1/Q(\vec{a}))$$

subject to

(i) $Q(\vec{a}^{\,i}) = P(\vec{a}^{\,i})$, for all $i, \vec{a}^{\,i}$.

(ii) Q is a proper probability distribution.

Note first that this problem always has a unique answer since $H(Q)$ is strictly concave and all constraints are linear. In addition, the feasible set is nonempty, as it contains P itself.

To get the form of Q^*, we solve the optimization problem by introducing Lagrange multipliers $\lambda_{i,\vec{a}^{\,i}}$ (for all i and $\vec{a}^{\,i}$) for the neighborhood marginal constraints (Condition 7.4.2 above); let us call $\vec{\lambda}$ the resulting vector of multipliers. We also introduce a single Lagrange multiplier β for the normalization constraint (Condition (ii) above). The optimization then becomes

$$\begin{aligned} Q^* &= \mathrm{argmax}_{Q,\vec{\lambda},\beta}\{L(Q, \vec{\lambda}, \beta)\} \\ &\equiv \mathrm{argmax}_{Q,\vec{\lambda},\beta} \left\{ H(Q) + \sum_{i\in[n]} \sum_{\vec{a}^{\,i}} \lambda_{i,\vec{a}^{\,i}} (Q(\vec{a}^{\,i}) - P(\vec{a}^{\,i})) \right. \\ &\quad \left. + \beta \left(\sum_{\vec{a}} Q(\vec{a}) - 1 \right) \right\}, \end{aligned}$$

where $Q(\vec{a})$ is constrained to be positive. Here, L is the Lagrangian function.

A necessary condition for Q^* is that $\partial L/\partial Q(\vec{a})|_{Q=Q^*} = 0$, for all $\vec{a}$ such that $P(\vec{a}) > 0$. After taking derivatives and some algebra, this condition implies, for all $\vec{a}$,

$$Q^*_{\vec{\lambda}}(\vec{a}) = (1/Z_{\vec{\lambda}}) \prod_{v=1}^{n} I[P(\vec{a}^{\,i}) \neq 0] \exp(\lambda_{i,\vec{a}^{\,i}}),$$

where $I[P(\vec{a}^{\,i}) \neq 0]$ is an indicator function that evaluates to 1 iff $P(\vec{a}^{\,i}) \neq 0$. We use the subscript $\vec{\lambda}$ on $Q^*_{\vec{\lambda}}$ and $Z_{\vec{\lambda}}$ to explicitly denote that they are parameterized by the Lagrange multipliers.

It is important to note at this point that regardless of the value of the Lagrange multipliers, each $\lambda_{i,\vec{a}^{\,i}}$ is only a function of the $\vec{a}^{\,i}$. Let the dual function $F(\vec{\lambda}) \equiv L(Q^*_{\vec{\lambda}}(\vec{a}), \vec{\lambda}, 0)$, and let $\vec{\lambda}^*$ maximize this function. Note that those $\lambda_{i,\vec{a}^{\,i}}$ that

correspond to $P(\vec{a}^{\,i}) = 0$ are irrelevant parameters since $F(\vec{\lambda})$ is independent of them. So for all i and $\vec{a}^{\,i}$ such that $P(\vec{a}^{\,i}) = 0$, we set $\lambda^*_{i,\vec{a}^{\,i}} = 0$. For all $i, \vec{a}^{\,i}$, we define the functions $\psi^*_i(\vec{a}^{\,i}) \equiv I[P(\vec{a}^{\,i}) \neq 0] \exp(\lambda^*_{i,\vec{a}^{\,i}})$. Hence, we can express the maximum entropy distribution Q^* as, for all $\vec{a}$,

$$Q^*_{\vec{\lambda}^*}(\vec{a}) = (1/Z_{\vec{\lambda}^*}) \prod_{i=1}^{n} \psi^*_i(\vec{a}^{\,i}),$$

which completes the proof. □

The main result of this section follows, and shows that we can represent any correlated equilibria of a graphical game $(G, \mathcal{M})$, up to payoff equivalence, with a local Markov network (G, Ψ). The proof follows from Lemmas 7.9, 7.10, and 7.12.

Theorem 7.13 *For all graphical games $(G, \mathcal{M})$, and for any correlated equilibrium P of $(G, \mathcal{M})$, there exists a distribution Q such that*

(**i**) *Q is also correlated equilibrium for $(G, \mathcal{M})$;*

(**ii**) *Q gives all players the same expected payoffs as P: $Q \equiv_{EP} P$; and*

(**iii**) *Q can be represented as a local Markov network with graph G.*

Note that the representation size for any local Markov network with graph G is linear in the representation size of the graphical game, and thus we can represent the CE of the game parsimoniously.

Remarks. Aside from simple parsimony, Theorem 7.13 allows us to import a rich set of tools from the probabilistic inference literature (Pearl, 1988; Lauritzen, 1996). For example, it is well known that for any vertices i and j and vertex set S in a (local) Markov network, i and j are conditionally independent given values for the variables in S if and only if S separates i and j – that is, the removal of S from G leaves i and j in disconnected components. This, together with Theorem 7.13, immediately implies a large number of conditional independences that must hold in any CE outcome. Also, as mentioned in the Introduction, Theorem 7.13 can be interpreted as strongly limiting the nature of the public randomization needed to implement any given CE outcome – namely, only "local" sources of random bits (as defined by G) are required.

7.4.3 Algorithms for Correlated Equilibria in Graphical Games

Having established in Theorem 7.13 that a concise graphical game yields an equally concise representation of its CE up to payoff equivalence, we now turn our attention to algorithms for *computing* CE. In the spirit of our results thus far, we are interested in algorithms that can efficiently exploit the compactness of graphical games.

It is well known that it is possible to compute CE via linear programming in time polynomial in the standard *noncompact* normal form. In this approach, one variable is introduced for every possible joint action probability $P(\vec{a})$, and the constraints enforce both the CE condition and the fact that the variables must define a probability distribution. It is not hard to verify that the constraints are all linear and there are $O(2^n)$ variables and constraints in the binary action case. By introducing any linear

optimization function, one can get an algorithm based on linear programming for computing a single exact CE that runs in time polynomial in the size of the normal-form representation of the game (i.e., polynomial in 2^n).

For graphical games this solution is clearly unsatisfying, since it may require time exponential in the size of the graphical game. What is needed is a more concise way to express the CE and distributional constraints – ideally, linearly in the size of the graphical game representation. As we shall now sketch, this is indeed possible for tree graphical games. The basic idea is to express both the CE and distributional constraints entirely in terms of the local marginals, rather than the global probabilities of joint actions.

For the case in which the game graph is a tree, it suffices to introduce linear distributional constraints over only the local marginals, along with *consistency* constraints on the *intersections* of local marginals. We thus define the following three categories of local constraints defining a linear program:

Variables: For every player i and assignment $\vec{a}^{\,i}$, there is a variable $P(\vec{a}^{\,i})$.
LP Constraints:

(i) *CE Constraints:* For all players i and actions a, a',

$$\sum_{\vec{a}^{\,i}:a_i^i=a} P(\vec{a}^{\,i})M_i(\vec{a}^{\,i}) \geq \sum_{\vec{a}^{\,i}:a_i^i=a} P(\vec{a}^{\,i})M_i([\vec{a}^{\,i}[i:a']).$$

(ii) *Neighborhood Marginal Constraints:* For all players i,

$$\forall \vec{a}^{\,i}: \quad P(\vec{a}^{\,i}) \geq 0; \ \sum_{\vec{a}^{\,i}} P(\vec{a}^{\,i}) = 1.$$

(iii) *Intersection Consistency Constraints:* For all players i and j, and for any assignment $\vec{y}^{\,ij}$ to the *intersection neighborhood* $N(i) \cap N(j)$,

$$\begin{aligned} P(\vec{a}^{\,ij}) &\equiv \sum_{\vec{a}^{\,i}:\vec{a}^{\,ij}=\vec{y}^{\,ij}} P(\vec{a}^{\,i}) \\ &= \sum_{\vec{a}^{\,j}:\vec{a}^{\,ij}=\vec{y}^{\,ij}} P_j(\vec{a}^{\,j}) \\ &\equiv P_j(\vec{a}^{\,ij}). \end{aligned}$$

Note that if d is the size of the largest neighborhood, this system involves $O(n2^d)$ variables and $O(n2^d)$ linear inequalities, which is linear in the representation size of the original graphical game, as desired. This leads to the following algorithmic result.

Theorem 7.14 *For all tree graphical games $(G, \mathcal{M})$, any solution to the linear constraints given above is a correlated equilibrium for $(G, \mathcal{M})$.*

Thus, for instance, we may choose any linear objective function $F(\{P(\vec{a}^{\,i})\})$ and apply standard efficient linear programming algorithms in order to find a CE maximizing F in time polynomial in the size of the graphical game. One natural choice for F is the social welfare, or the total expected payoff over all players:

$$F(\{P_i(\vec{a}^{\,i})\}) = \sum_i \sum_{\vec{a}^{\,i}} P_i(\vec{a}^{\,i})M_i(\vec{a}^{\,i}).$$

7.5 Graphical Exchange Economies

In the same way that the graph of a graphical game represents restrictions on which pairs of players directly influence each other's payoffs, it is natural to examine similar restrictions in classical exchange economies and other market models. In such models, there is typically some number k of goods available for trade, and n players or consumers in the economy. Each consumer has a utility function mapping bundles or amounts of the k goods to a subjective utility. (Settings in which the utility functions obey certain constraints, such as concavity or linearity, are often assumed.) Each consumer also has an endowment – a particular bundle of goods that they are free to trade. It is assumed that if prices $\vec{p} \in (\Re^+)^k$ are posted for the k goods, each consumer will attempt to sell their initial endowment at the posted prices, and then attempt to buy from other consumers that bundle of goods which maximizes their utility, subject to the amount of cash received in the sale of their endowment. A celebrated result of Arrow and Debreu (1954) established very general conditions under which an *equilibrium* price vector exists – prices at which all consumers are able to sell all of their intial endowments (no excess supply) and simultaneously able to purchase their utility-maximizing bundles (no excess demand). The result depends crucially on the fact that the model permits exchange of goods between any pair of consumers in the economy.

A natural graph- or network-based variant of such models again introduces an undirected graph G over the n consumers, with the interpretation that trade is permitted between consumers i and j if and only if the edge (i, j) is present in G.[2] The classical equilibrium existence result can be recovered – but only if we now allow for the possibility of *local* prices, that is, prices for each good–consumer pair. In other words, at equilibrium in such a *graphical economy*, the price per unit of wheat may differ when purchased from different consumers, due to the effects of network topology. In this model, rationality means that consumers must always purchase goods from the neighboring consumers offering the lowest prices.

As with graphical games, there are at least two compelling lines of research to pursue in such models. The first is computational: What graph topologies permit efficient algorithms for computing price equilibria? The second is structural: What can we say about how network structure influences properties of the price equilibria, such as the amount of price variation? Here we briefly summarize results in these two directions.

On the computational side, as with the **TreeNash** algorithm for computing NE in graphical games, it is possible to develop a provably correct and efficient algorithm for computing approximate price equilibria in tree graphical economies with fairly general utility functions. Like **ApproxTreeNash**, this algorithm is a two-pass algorithm in which information regarding *conditional* price equilibria is exchanged between neighboring nodes, and a discrete approximation scheme is introduced. It is complementary to other recent algorithms for computing price equilibria in the classical non-graphical (fully connected) setting under linearity restrictions on the utility functions (discussed in detail in Chapter 5.

[2] In the models considered to date, resale of purchased goods is not permitted – rather, we have "one-shot" economies.

On the structural side, it can be shown that different stochastic models of network formation can result in radically different price equilibrium properties. For example, consider the simplified setting in which the graph G is a bipartite graph between two types of parties, *buyers* and *sellers*. Buyers have an endowment of 1 unit of an abstract good called *cash*, but have utility only for *wheat*; sellers have an endowment of 1 unit of wheat but utility only for cash. Thus the only source of asymmetry in the economy is the structure of G. If G is a random bipartite graph (i.e., generated via a bipartite generalization of the classical Erdos–Renyi model), then as n becomes large there will be essentially *no* price variation at equilibrium (as measured, for instance, by the ratio of the highest to lowest prices for wheat over the entire graph). Thus random graphs behave "almost" like the fully connected case. In contrast, if G is generated according to a stochastic process such as preferential attachment (Barabasi and Albert, 1999), the price variation at equilibrium is unbounded, growing as a root of the economy size n.

7.6 Open Problems and Future Research

There are a number of intriguing open areas for further research in the broad topics discussed in this chapter, including the following.

- **Efficient Algorithms for Exact Nash Computation in Trees.** Perhaps the most notable technical problem left unresolved by the developments described here is that of efficiently (i.e., in time polynomial in the graphical game description) computing exact Nash equilibria for trees. This class falls between the positive results of Elkind et al. (2006) for unions of paths and cycles, and the recent PPAD-completeness results for bounded treewidth graphs (see Chapter 2).
- **Strategy-Proof Algorithms for Distributed Nash Computation.** The NashProp algorithm described here and its variants are clearly not strategy-proof, in the sense that players may have incentive to deviate from the algorithm if they are to actually realize the Nash equilibrium they collectively compute. It would be interesting to explore the possibilities for strategy-proof algorithms for graphical games.
- **Cooperative, Behavioral, and Other Equilibrium Notions.** Here we have described algorithms and structural results for graphical games under noncooperative equilibrium notions. It would be interesting to develop analogous theory for cooperative equilibria, such as how the coalitions that might form depend on graph topology. The recent explosion of work in behavioral game theory and economics (Camerer, 2003) is also ripe for integration with graphical games (and many other aspects of algorithmic game theory as well). For instance, one could investigate how the behavioral phenomenon of inequality aversion might alter the relationship between network structure and equilibrium outcomes.

7.7 Bibliographic Notes

Graphical games were introduced by Kearns et al. (2001) (abbreviated KLS henceforth). Related models were introduced at approximately the same time by Koller and Milch (2003) and La Mura (2000). Graph-theoretic or network models of interaction

have a long history in economics and game theory, as surveyed by Jackson (2005); these models tend to be less general than graphical games, and there is naturally less explicit emphasis on computational issues.

The original KLS paper contained the algorithm and analyses of the tree-based algorithms examined in Section 7.1. The **NashProp** generalization of these algorithms is due to Ortiz and Kearns (2003). A follow-up to the KLS paper by the same authors (Littman et al., 2002) erroneously claimed an efficient algorithm for computing an *exact* NE in tree graphical games (recall that the KLS paper gave an efficient algorithm only for approximate NE in trees). The error was recently discovered and discussed by Elkind et al. (2006), who proved that in fact no two-pass algorithm can compute an exact equilibrium. The problem of efficiently computing an exact equilibrium in time polynomial in the size of a tree graphical game remains open.

The study of correlated equilibria in graphical games given in Section 7.4 is adapted from Kakade et al. (2003). Roughgarden and Papadimitriou (2005) and Papadimitriou (2005) gave more general algorithms for computing correlated equilibria in graphical games and other compact representations. It is interesting to note that while the Kakade et al. results show how all correlated equilibria (up to payoff equivalance) can be succinctly *represented* as a Markov networks, Papadimitriou's algorithm (2005) computes correlated equilibria that are mixtures of Nash equilibria and thus can be efficiently sampled. Intractability results for certain correlated equilibrium computations are given by Gilboa and Zemel (1989), as well as by Roughgarden and Papadimitriou (2005).

Other papers providing algorithms for equilibrium computation in graphical games include those of Vickrey and Koller (2002), who examine hill-climbing algorithms for approximate NE, as well as constraint satisfaction generalizations of **NashProp**; and Daskalakis and Papadimitriou (2006), who show close connections between the computation of pure NE and probabilistic inference on the Markov network models discussed in the context of correlated equilibria in Section 7.4.

Graphical games have also played a central role in striking recent developments establishing the intractability of NE computations in general multiplayer games, including the work by Daskalakis et al. (2006) and Goldberg and Papadimitriou (2006); these developments are discussed in detail in Chapter 29. Daskalakis and Papadimitriou also proved intractability results for computing NE in graphical games on highly regular graphs (Daskalakis and Papadimitriou, 2005), while Schoenebeck and Vadhan (2006) systematically characterize the complexity of a variety of equilibrium-related computations, including NE verification and existence of pure equilibria.

The formulation of the graphical exchange economy model summarized in Section 7.5, as well as the price equilibrium proof and algorithms mentioned, is due to Kakade et al. (2004). The result on price variation in different stochastic graph generation models is due to Kakade et al. (2005).

Recently a graph-theoretic generalization of classical evolutionary game theory has been introduced, and it has been shown that random graphs generally preserve the classical evolutionary stable strategies (Kearns and Suri, 2006); these results are discussed in some detail in Chapter 29.

Acknowledgments

I would like to give warm thanks to Michael Littman, Satinder Singh, Sham Kakade, John Langford, and Luis Ortiz for their permission to adapt material from our joint publications (Kakade et al., 2003; Kearns et al., 2001) for presentation in this chapter.

Bibliography

K. Arrow and G. Debreu. Existence of an equilibrium for a competitive economy. *Econometrica*, 22(3):265–290, 1954.

R.J. Aumann. Subjectivity and correlation in randomized strategies. *J. Math. Econ.*, 1, 1974.

R.J. Aumann. Correlated equilibrium as an expression of Bayesian rationality. *Econometrica*, 55, 1987.

A. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

A. Berger, S.D. Pietra, and V.D. Pietra. A maximum entropy approach to natural language processing. *Comp. Ling.*, 22(1), March 1996.

C. Camerer. *Behavioral Game Theory*. Princeton University Press, 2003.

C. Daskalakis, P. Goldberg, and C. Papadimitriou. The complexity of computing a Nash equilibrium. In *Proc. 38th ACM Symp. Theory of Computing*, pp. 71–78. ACM Press, 2006.

C. Daskalakis and C. Papadimitriou. The complexity of games on highly regular graphs. In *Proc. 13th Annual Euro. Symp. Algo.*, p. 71. Springer, Berlin, 2005.

C. Daskalakis and C. Papadimitriou. Computing pure Nash equilibria in graphical games via Markov random fields. In *Proc. 7th ACM Conf. on Electronic Commerce*, pp. 91–99. ACM Press, 2006.

A.P. Dawid and S.L. Lauritzen. Hyper Markov laws in the statistical analysis of decomposable graphical models. *Ann. Stat.*, 21(3):1271–1317, September 1993.

E. Elkind, L. Goldberg, and P. Goldberg. Graphical games on trees revisited. In *Proc. 7th ACM Conf. on Electronic Commerce*, pp. 100–109. ACM Press, 2006.

D. Foster and R. Vohra. Calibrated learning and correlated equilibrium. *Games Economic Behav.*, 1997.

D. Foster and R. Vohra. Regret in the on-line decision problem. *Games Econ. Behav.*, pp. 7–36, 1999.

I. Gilboa and E. Zemel. Nash and correlated equilibria: Some complexity considerations. *Games Econ. Behav.*, 1:80–93, 1989.

P. Goldberg and C. Papadimitriou. Reducibility among equilibrium problems. In *Proc. 38th ACM Symp. Theo. Comp.*, pp. 61–70. ACM Press, 2006.

M. Jackson. The economics of social networks. In *Adv. Economics and Econometrics, Theo. Appl.: Ninth World Congr. Econo. Soc*. Cambridge University Press, 2005.

S. Kakade, M. Kearns, J. Langford, and L. Ortiz. Correlated equilibria in graphical games. In *Proc. 4th ACM Conf. on Electronic Commerce*, pp. 42–47. ACM Press, 2003.

S. Kakade, M. Kearns, and L. Ortiz. Graphical economics. In *Proc. 17th Annual Conf. on Learning Theo.*, pp. 17–32. Springer, Berlin, 2004.

S. Kakade, M. Kearns, L. Ortiz, R. Pemantle, and S. Suri. Economic properties of social networks. In L. Saul, Y. Weiss, and L. Bottou, editors, *Adv. Neural Infor. Proc. Syst., 17*, pp. 633–640. MIT Press, 2005.

M. Kearns, M. Littman, and S. Singh. Graphical models for game theory. In *Proc. 17th Annual Conf. on Uncertainty in Artificial Intelligence*, pp. 253–260. Morgan Kaufmann, 2001.

M. Kearns and S. Suri. Networks preserving evolutionary equilibria and the power of randomization. In *Proc. 7th ACM Conf. Electronic Commerce*, pp. 200–207. ACM Press, 2006.

D. Koller and B. Milch. Multi-agent influence diagrams for representing and solving games. *Games Econ. Behav.*, 45(1):181–221, 2003.

P. La Mura. Game networks. In *Proc. 16th Conf. Uncertainty in Artificial Intelligence*, pp. 335–342. Morgan Kaufmann, 2000.

S. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

S. Lauritzen and D. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *J. Royal Stat. Soc. B*, 50(2):157–224, 1988.

M. Littman, M. Kearns, and S. Singh. An efficient exact algorithm for singly connected graphical games. In *Adv. in Neural Inf. Proc. Syst. 14*. MIT Press, 2002.

D. Monderer and L. Shapley. Potential games. *Games Econ. Behav.*, 14:124–143, 1996.

J.F. Nash. Non-cooperative games. *Ann. Math.*, 54:286–295, 1951.

L. Ortiz and M. Kearns. Nash propagation for loopy graphical games. In S. Becker, S. Thrun, and K. Obermayer, editors, *Adv. Neural Inf. Proc. Syst. 15*, pp. 793–800. MIT Press, 2003.

G. Owen. *Game Theory*. Academic Press, UK, 1995.

C. Papadimitriou. Computing correlated equilibria in multi-player games. In *Proc. 37th ACM Symp. Theo. Comp.*, pp. 49–56. ACM Press, 2005.

C. Papadimitriou and T. Roughgarden. Computing equilibria in multi-player multi-player games. In *Proc. 16th ACM-SIAM Symp. Disc. Algo.*, pp. 82–91. SIAM, 2005.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.

R. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *Intl. J. Game Theory*, 2:65–67, 1973.

G. Schoenebeck and S. Vadhan. The computational complexity of Nash equilibria in concisely represented games. In *Proc. 7th ACM Conf. Electronic Commerce*, pp. 270–279. ACM Press, 2006.

D. Vickrey and D. Koller. Multi-agent algorithms for solving graphical games. In *Proc. 18th Ntl. Conf. on Artificial Intelligence*, pp. 345–351. AAAI Press, 2002.