Intro to Algorithmic Game Theory Mini-Course	**Spring 2009**

Lecture 1: February 5

Lecturer: Alex Fabrikant *Scribe: Wolfgang Mulzer & Shi Li*

1.1 What is Algorithmic Game Theory?

Algorithmic game theory arises from the combination of game-theoretic (or more generally economic) concepts with algorithmic ideas. The main goal of game theory is to investigate what happens if selfish, rational agents interact in a shared environment. Modelling such interactions is increasingly important to modern computer science, with the rise of the Internet and distributed computing generating a wealth of questions about algorithms for such selfish interactions. Conversely, complexity theory provides powerful tools for quantifying rational behavior, and algorithmic lower bounds on game theoretic concepts can provide broad computational critiques of models proposed by economists. Algorithmic game theory pursues both of these directions.

1.2 The Prisoner's Dilemma

In order to introduce some basic notions from game theory, we begin with the classical prisoner's dilemma: two suspects are arrested by the police and are being interrogated in separate rooms. Both of them have a choice between confessing the crime or staying silent. If both confess, they will each be sentenced to, say, four years in prison. If one confesses and the other stays silent, the whistle-blower will get by with one year, while the other prisoner will receive five years. Finally, if both stay silent, they will be sentenced to two years each for lack of evidence. What should the prisoners do?

Let us see how to model this problem: first, there is a set of *players* $N = \{1, 2\}$. Each player has a set of *strategies*, $S_1 = S_2 = \{C, S\}$, where C means "confess" and S means "silent". Of course, in general, the strategy sets of the players can differ. The Cartesian product of the strategy sets, $S = S_1 \times S_2$, is called the set of *outcomes* or *strategy profiles*. Finally, for each player there is a *cost function* $c_1, c_2 : S \to \mathbb{Q}$, representing the penalty the player incurs at a particular outcome. In general, costs can be real-valued, but in this course, as in *most* of AGT, we'll restrict ourselves to traditional computation over rationals or integers. Depending on the game, instead of the cost function we can refer to the *utility* or *payoff* of an outcome which measures the benefit a player receives, i.e. the negative of the cost. For games with two players, the cost functions are conveniently represented as a table, as in Figure 1.1. This is called the *normal form* representation of the game. Two-player normal-form games are also called *bimatrix* games.

	C	S
C	4, 4	1, 5
S	5, 1	2, 2

Figure 1.1: The prisoner's dilemma in normal form.

The first number represents the cost for the row player, the second number the cost for the column player. Looking at the table, we see that no matter what Prisoner 2 does, Prisoner 1 will always be better off confessing. We say that strategy S is *dominated* for Prisoner 1. The same holds for Prisoner 2, and hence the best option is for both to confess. This gives a simple algorithm for determining a best strategy, the *iterated dominance* algorithm: find a dominated strategy, remove it, and continue until only one strategy is left.

Iterated dominance works for the prisoner's dilemma, but in most games it does not. Consider the 2-player game called "Battle of the Sexes": Alice and Bob have to decide whether they want to go see a baseball or a softball game. Alice likes softball, Bob likes baseball, and both like each other. The payoffs are shown in Figure 1.2 (Alice is the row player, Bob the column player). Now, no strategy dominates the

	B	S
B	1,2	0,0
S	0,0	2,1

Figure 1.2: Battle of the Sexes in normal form.

other. However, the two outcomes (B,B) and (S,S) are *stable* in the following sense: neither Alice nor Bob can improve their pay-off by unilaterally changing their strategy. An outcome $s \in S$ is called a *pure Nash equilibrium* if for every player $p \in N$ and all strategies $s'_p \in S_p$ for this player we have $u_p(s) \geq u_p(s'_p, s_{-p})$, where u_p denotes p's utility function and s_{-p} are all components of s except for the one corresponding to p. Hence, both (B,B) and (S,S) are pure Nash equilibria. The notion of pure Nash equilibrium is quite robust and studied extensively in game theory. It was amongst the earliest equilibrium concepts and any new equilibrium concepts proposed are first evaluated against it.

Note that the s_{-p} shorthand for "all components of vector s except the p'th one" turns out to be very convenient for many game-theoretic discussions, and is used heavily throughout the literature.

1.3 A Load-Balancing Game

Next, we consider a load balancing game. We have n players $N = \{1, \ldots, n\}$, and n CPUs $\{C_1, \ldots, C_n\}$. Each player chooses a CPU to execute her job, thus $S_1 = \cdots = S_n = \{C_1, \ldots, C_n\}$, and the cost she incurs is the number of jobs on her chosen CPU, a measure of how long it takes for the job to be executed (under several uniformity assumptions). It is easily seen that the pure Nash equilibria are precisely those outcomes in which each player has one CPU to herself.

A key question in many systems is *how well does society fare when players are allowed to act selfishly*? The "right" way to quantify the *social cost* (or *social welfare*) of a particular outcome depends on the application. With load balancing, it's common to consider the social cost as the *maximum* of the individual costs (called "makespan" in the load-balancing literature).

It's easy to see that the makespan is minimized precisely at the Nash equilibria. This shows that in this case distributed selfish decision making leads to an optimal outcome.

The *price of anarchy* is defined as the ratio of the maximum social cost incurred by a pure Nash equilibrium to the minimum social cost of any outcome (not necessarily an equilibrium). A low price of anarchy in a game means that society (the agents taken as a whole) don't suffer much from the lack of centralized control. This is thus a common metric for evaluating decentralized systems.

In the load-balancing game, the price of anarchy is 1. However, this is not always the case: in the prisoner's dilemma, the price of anarchy is 2, since ideally both prisoners would stay silent, while in the pure Nash equilibrium both confess. Worse yet, if the sentencing discrepancies are arbitrarily large, the price of anarchy in Prisoner's Dilemma could also be made arbitrarily large.

1.4 A Network-Connection Game

In the *network-connection game*, as studied by Anshelevich, et al, we are given a directed graph $G = (V, E)$ and sets of k *sources* $\sigma_1, \ldots, \sigma_k \in V$ and k *sinks* $\tau_1, \ldots, \tau_k \in V$. Each edge e has a *cost* c_e. There are k players $N = \{1, \ldots, k\}$, and player i wants to connect his source to his sink. Thus, the strategy set S_i consists of all paths from σ_i to τ_i. The cost c_i for player i is defined as

$$c_i(s) = \sum_{e \in P_i} \frac{c_e}{k_e(s)},$$

where P_i denotes the path chosen by player i, and $k_e(s)$ is the number of paths in s going through e. We think of the cost as the investment a player has to make, with the edge users' joint contribution paying for building the connection. Hence, the costs decrease if more players want to use that an edge.

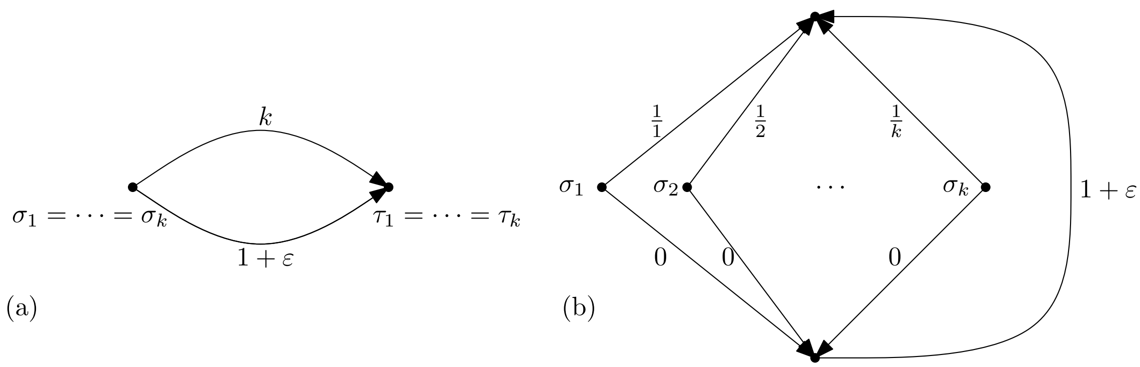

Figure 1.3: Two example graphs for the network-formation game.

How bad is the price of anarchy in this game? First we need to define the social cost, which we here define as the *sum* of the players' payments, or, equivalently, the total cost of all the edges that are used by at least one player.

Consider the example in Figure 1.3a. Let s^* be the outcome in which all players choose the lower edge. Clearly, s^* is a pure Nash equilibrium. We have $c_1(s^*) = \cdots = c_k(s^*) = (1+\varepsilon)/k$, and the social cost is $1+\varepsilon$. Now let $\tilde{s}$ be the strategy in which all players choose the upper edge. We have $c_1(\tilde{s}) = \cdots = c_k(\tilde{s}) = 1$, and $\tilde{s}$ is also a pure Nash equilibrium, since any one person would lose ε by unilaterally switching to the loewr edge. However, $\mathrm{SC}(\tilde{s}) = k$, and since it is easily seen that there are no other pure Nash equilibria, the price of anarchy is $k/(1+\varepsilon)$ and hence can be arbitrarily large.

But the other equilibrium is itself socially optimal. While total "anarchy" could result in the worst-case equilibrium, we can consider a model when a benevolent "advisor" suggests a particularly good equilibrium for everybody to start at. Such a party may, for one thing, be easier to engineer than a "benevolent dictator" party fully controlling the actions of all players.

To measure the social impact of such a scenario, we define the *price of stability* as the ratio of the social cost of the *best* equilibrium to the optimum social cost. In our example, the price of stability is 1, but this is not always true, as can be seen from the prisoner's dilemma, where there exists only one pure Nash equilibrium and the price of stability is 2. How bad can the price of stability be for the network-formation game? Consider the graph in Figure 1.3b: here, the social optimum is achieved when all players route their path downwards, giving a social cost of $1+\varepsilon$. Nonetheless, it is easily checked that this outcome is not stable, with players starting from k selfishly defecting one-by-one to their upward path. The only equilibrium state is reached when all players use their upward edge, with a social cost of $\sum_{i=1}^{k} \frac{1}{i} = H_k$, the k-th harmonic number. It follows that the price of stability can be as bad as $\Omega(\log k)$.

One might wonder whether there is an instance of the network connection game with worse price of stability. It turns out that the above is the worst that can happen:

Theorem 1.1 *For a network connection game with k players, the price of stability of the game is at most H_k.*

Proof: The key concept for the proof is a *potential function*:

Definition 1.2 *$\Psi : S \to \mathbb{R}$ is called a potential function if every move by a single player causes Ψ to change by the same amount as the moving player's payoff: $\Psi(s) - \Psi(s'_i, s_{-i}) = c_i(s) - c_i(s'_i, s_{-i})$.*

If a game has a potential function, it's called a *potential game*. The reader should verify that the local minima of a potential function are precisely the pure Nash equilibria of the game. Note that this means that a potential game always has Nash equilibria, and a sequence of strictly best-response moves by the players will always reach one of them.

In the network connection game, define a function $\Psi_e(s) = c_e \cdot H_{k_e(S)}$ for each edge e in strategy profile s, where c_e is the cost of edge e. $k_e : S \to \mathbb{N}$ is the number of players that using the edge e under the given strategy profile. We claim that the sum $\sum_{e \in E} \Psi_e(S)$ is a potential function for the game.

Lemma 1.3 *For a strategy profile $s = (s_1, s_2, \cdots, s_k) \in S$, suppose player i changes his strategy to s'_i ($s'_i \neq s_i$), changing the strategy profile to $s' = (s'_i, s_{-i})$. Then $\Psi(s') - \Psi(s) = c_i(s') - c_i(s)$.*

Proof:

$$\begin{aligned}
\Psi(s') - \Psi(s) &= \sum_{e \in E} (\Psi_e(s') - \Psi_e(s)) \\
&= \sum_{e \in E} c_e \left(h_{k_e(s')} - h_{k_e(s)} \right) \\
&= \sum_{e \in s'_i \setminus s_i} \frac{c_e}{k_e(s')} - \sum_{e \in s_i \setminus s'_i} \frac{c_e}{k_e(s)} \\
&= c_i(s') - c_i(s)
\end{aligned}$$

■

Let's return to the proof of theorem 1.1. It's easy to show $H_k SC(s) \geq \Psi(s) \geq SC(s)$. The potential function's local minima are guaranteed to be pure Nash equilibria (exercise: verify that the Nash equilibrium with the best social cost is **not** necessarily the global minimum of the Ψ). Let $\tilde{s} = \text{argmin}_{s \in S} \Psi(s)$ and $s^* = \text{argmin}_{s \in S} SC(s)$, then

$$SC(s^*) \geq \frac{\Psi(s^*)}{H_k} \geq \frac{\Psi(\tilde{s})}{H_k} \geq \frac{SC(\tilde{s})}{H_k}$$

Since $\tilde{S}$ is a possible PNE, the theorem immediately follows the above inequality. ■