CS 684 Algorithmic Game Theory	Scribe: Dan Sheldon
Instructor: Eva Tardos	October 19, 2004

1 Price of Anarchy Without Convergence

1.1 An Alternative to Nash

Last time we discussed the problem of finding Nash equilibria in generalized congestion (potential) games. Recall that every discrete potential game is a generalized congestion game, i.e., it can be written in a form similar to that of the discrete routing game. In such games, any solution minimizing Φ is a Nash equilibrium, so we asked the natural question: can we find a Nash by allowing users to make best response moves? The answer was yes, but the process may converge very slowly, not necessarily in polynomial time. Then we suggested an alternative: only allow users to move if Φ decreases by a sufficient amount. This process is guaranteed to terminate in polynomial time, but may not converge to a Nash (or even to an *approximate* Nash).

These problems led us to an alternate question. If we allow users to make a certain number of best response moves, will we arrive at a "good" solution, even if it's not a Nash? In this setting we're willing to tolerate instability in a solution due to individual players' desire to switch, as long as the social value is close enough to the optimum. We'll explore this idea in the context of a simple load balancing game.

1.2 Simple Load Balancing

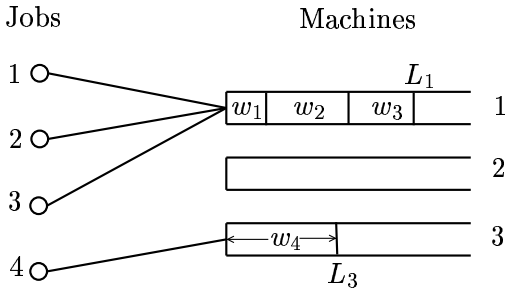

Figure 1: Illustration of the game

The game consists of

- jobs $j \in \{1, \ldots, n\}$,
- weights w_j for each job j (**Note**: this is a change in notation. Previously we used p_j, but we'd like to reserve p for probabilities, which will come soon!),
- machines $i \in \{1, \ldots, m\}$, all *identical*,

- and load L_i on each machine i,

$$L_i = \sum_{\substack{\text{jobs } j \text{ assigned} \\ \text{to machine } i}} w_j.$$

We'll illustrate solutions as above, often omitting the jobs and just drawing the corresponding loads on each machine. The players are jobs, and each job wishes to minimize the load of the machine to which it's assigned.

Job j on machine i will switch to machine i if

$$L_k + w_j < L_i$$

Our notion of social value of a solution is the maximum load on any machine, also known as the *makespan*.

1.3 The Result

Theorem 1 *For any starting assignment and any sequence of best response selfish moves such that each player gets a chance to move, the resulting solution satisfies*

$$\textit{max load} \leq 2 \cdot (\textit{minimum max load possible})$$

Proof. After such a sequence of moves, let i be the machine with the maximum load, that is, $L_i = \max_k L_k$. Consider the job j on machine i that was last given a chance to move — either j moved onto i, or j was already on i and chose not to move.

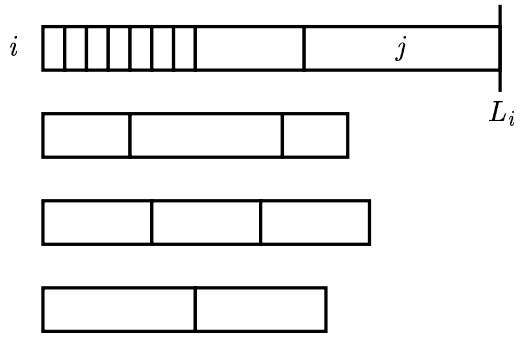

Figure 2: Proof setting

Now consider the point in time when j last moved, and say that $\hat{L}_k$ is the load on a given machine k at this point in time, *excluding the contribution from job j*. Since job j chooses machine i, and no more jobs will move from machine i, we know that $\hat{L}_i + w_j = L_i$. Furthermore, machine i must have been the best possible choice, so for each machine k,

$$\hat{L}_k + w_j \geq \hat{L}_i = L_i.$$

Reversing the inequality and summing over all machines k,

$$\begin{aligned} mL_i &\leq \sum_k \hat{L}_k + mw_j \\ &\leq \sum_\ell w_\ell + mw_j, \end{aligned}$$

or, dividing by m,

$$L_i \leq \left(\frac{1}{m}\sum_\ell w_\ell\right) + w_j. \tag{1}$$

But we know that OPT is at least as big as the average load on all machines, so $OPT \geq \frac{1}{m}\sum_\ell w_\ell$. Also, every job j is assigned to some machine, so for all jobs j, $OPT \geq w_j$. Using these facts in (1), we get $L_i \leq 2 \cdot OPT$.

■

We would like to have similar results for other problems

2 Randomized Nash Equilibria

We now introduce the notion of randomized (mixed) strategies in games, and similarly, randomized (mixed) Nash equilibria. These have long been studied in classical game theory — they were introduced to algorithmic game theory by Koutsoupias and Papadimitriou in a study of a load balancing with randomized strategies [1]. Randomized Nash equilibria are a nice concept because they *always* exist. On the negative side, many of our nice price of anarchy results will be replaced with negative results when considering randomized equilibria.

We'll continue the discussion of randomized games in the context of the simple load balancing game from earlier. First, what is a randomized strategy? Instead of requiring job j to pick a machine i deterministically, job j may choose among all machines according to some probability distribution. A strategy for job j is an assignment of probabilities p_{ij} for choosing each machine i satisfying

$$p_{ij} \geq 0, \quad \sum_i p_{ij} = 1.$$

Now how can we discuss the outcome of a game, the experience of a user, and the quality of a solution? We'll need probabilistic machinery, that is, we'll deal with random variables, expectation and conditional expectation.

Let X_{ij} be an indicator random variable for the assignment of job j to machine i,

$$X_{ij} = \begin{cases} 1 & \text{if } j \text{ chooses } i \\ 0 & \text{otherwise} \end{cases}.$$

Similarly, let L_i be a random variable representing the load on machine i,

$$L_i = \sum_j X_{ij} w_j.$$

What is the experience of job j? We clearly want some form of expected load, but a simple expectation is not quite right. For example, consider a very simple instance of the balancing game with a single job j that chooses among all machines uniformly at random, so $p_{ij} = \frac{1}{m}$ for all machines i. Then,

$$L_i = \begin{cases} w_j & \text{with probability } \frac{1}{m} \\ 0 & \text{otherwise} \end{cases} .$$

So for all machines i, the expected load is $\text{Exp}(L_i) = \frac{1}{m} w_j$. But this is not an accurate depiction of user j's experience! We know for sure that user j will experience a load of w_j. Instead, we'll define a random variable C_j to be the load on the machine *actually* selected by job j, and evaluate $\text{Exp}(C_j)$ using conditional expectation. Then, the goal of user j will be to minimize $Exp(C_j)$, and he will adjust his strategy accordingly.

The next lecture will introduce more precise definitions. The reader is encouraged to note the similarities and differences between randomized strategies and splittable jobs in the load balancing game.

References

[1] E. Koutsoupias, C. H. Papadimitriou "Worst-case equilibria," STACS 99.