University of Washington Lecturer: Abraham Flaxman/Vahab S. Mirrokni

CSE599m: Algorithms and Economics of Networks

April 6, 2007 Scribe: Chih-Wei Huang

Lecture 4: Coordination Mechanism Design

1 Summary

The coordination mechanism is studied to influence on the quality of the equilibrium. We verify the PoA of $R||C_{max}$ as $O(m)$ and review the full tables of PoA and NE results under various scheduling policies and models. In an unrelated machine, we show that the lower bound m for strongly local policy. Furthermore, if we use efficiency based algorithm and the modified version, we get PoAs as $O(\log m)$ and $O(\log^2 m)$ respectively.

2 Coordination Mechanism

A common goal of mechanism design is to design system-wide rules which, given the selfish decisions of the users, maximize the total social welfare. The degree to which these rules approximate the social welfare is represented by the worst ratio of *NE* cost to *OPT* cost known as *price of anarchy* (PoA) of the mechanism.

Possible solutions for this purpose including changing the system (add tolls, payments), Stackelberg strategy (centralized), and Coordination mechanisms. A coordination mechanism is a local policy that assigns a cost to each strategy s, where the cost of s is a function of the agents who have chosen s. It has advantages of local decision making and using the same type of cost.

2.1 Selfish Scheduling Game

We consider the selfish scheduling game as the example. There are n jobs must be processed on m machines. Job i has processing time p_{ij} on machine j. If the maximum completion time (makespan) is C_{max}, the social objective is to minimize C_{max}. Therefore

$$PoA = \frac{\text{Makespan of worst NE}}{\text{OPT makeSpan}}$$

2.2 Local Scheduling Policies

In selfish scheduling games, the coordination mechanism for this game is a local policy that determines how to schedule jobs assigned to that machine. Each policy induces NE on jobs. There are four types of policies:

- Shortest-first policy. We sequence the jobs in non-decreasing order.
- Longest-first policy. We sequence the jobs in non-increasing order.
- Random order policy. We processes the jobs in a ransom order.
- Makespan Policy. We process all jobs on the same machine in parallel.

2.3 Machine Scheduling Models

There are also different assumptions regarding the relationship between processing times yield different scheduling problems.

1. Identical machines $P||C_{max}$. $p_{ij} = p_{ik} = p_i$ for each job i and k.
2. Related machies $Q||C_{max}$. $p_{ij} = \frac{p_i}{s_j}$, where $s_j \leq 1$ is the speed of machine j.
3. Restricted assignment $B||C_{max}$. Each job i can be scheduled on a subset S_i of machines, i.e., p_{ij} is equal to p_i if $j \in S_i$ and is equal to ∞ otherwise.
4. Unrelated machines $R||C_{max}$. The processing times p_{ij} are arbitrary positive numbers.

PoA of $R||C_{max}$ using the shortest first policy can be derived as following:

$$p_{ij} \rightarrow p_i = \min_j p_{ij}$$

$$\text{OPT} \geq \frac{\sum p_i}{m}$$

$$\text{NE A: } M_i = \text{completion of the job } i$$

$$M_i \leq M_{i-1} + p_i$$

$$\text{Makespan of A} = M_n = (M_n - M_{n-1}) + ... + (M_1 - M_0) \leq p_n + p_{n-1} + ... + p_1 \leq m\text{OPT}$$

PoA results

	Makespan	Shortest-first	Longest-first	Randomized
$P\|\|C_{max}$	$2 - 2/(m+1)$	$2 - 2/(m+1)$	$4/3 - 1/3m$	$2 - 2/m$
$Q\|\|C_{max}$	O($\log m$)	O($\log m$)	2-2/m	O($\log m$)
$B\|\|C_{max}$	O($\log m$)	O($\log m$)	O($\log m$)	O($\log m$)
$R\|\|C_{max}$	Unbounded	O(m)	Unbounded	O(m)

Pure NE resutls

	Makespan	Shortest-first	Longest-first	Randomized
$P\|\|C_{max}$	Exists	Exists	Exists	Exists
$Q\|\|C_{max}$	Exists	Exists	Exists	Exists
$B\|\|C_{max}$	Exists	Exists	Exists	Exists
$R\|\|C_{max}$	Exists	Exists	???	Open

2.4 Lower Bound for Strongly Local Policy

Policies can be further categorized into:

- Local policy - depends on jobs assigned to machine.
- Strongly local policy - depends only on processing time of jobs on that machine.
- Ordering policy - independence of irrelevent alternative (IIA)

Here we want to prove the lower bound for strongly local policy starting with shortest-first. If there are m types of jobs, type j can be scheduled on machines j and $j+1$, processing time of type j on machine j is low and on machine $j+1$ is high with ratio j, and all jobs on machine j have almost the same processing time. The OPT assignment is all jobs of type j to machine j. Therefore the number of jobs is chosen such that OPT has the same completion time for all machines.

There exists an NE that about half jobs of type j are on machine j and half on machine $j+1$, so the completion time of NE grows linearly in m.

2.5 Efficiency Based Algorithm

In efficiency based algorithm, we order jobs on each machine by their efficiency defined as the ratio between job's best processing time to its processing time on this machine. The PoA of algorithm is $O(\log m)$. However, pure NE may not exist.

The algorithm can be modified to the following. Each machine simulate $\log m$ sub-machines by round robin scheme. Submachine k of machine$j$ handles jobs of efficiency between 2^{-k} and 2^{-k+1}. Finally, jobs are ordered on sub-machines by shortest-first resulting in PoA of $O(\log^2 m)$.

3 Further reading

Recommended reading:

George Christodoulou, Elias Koutsoupias, Akash Nanavati: Coordination Mechanisms. ICALP 2004: 345-357

Nicole Immorlica, Li Li, Vahab S. Mirrokni, Andreas Schulz: Coordination Mechanisms for Selfish Scheduling. WINE 2005: 55-69