Selfish task allocation

Elias Koutsoupias
University of Athens and UCLA
elias@di.uoa.gr

September 1, 2003

1 Introduction

Recently there has been a lot of interest in problems at the intersection of Game Theory, Economics, and Computer Science. For example, there are interesting developments concerning algorithms for equilibria and cost sharing, algorithmic mechanism design, and the efficiency of systems with selfish users [17]. In this note, I will focus on the last area and in particular on the price of anarchy of scheduling or task allocation. I will discuss the issues of this area, mention the central results, and suggest some open problems.

This is definitely not a review paper: My aim is to stimulate the reader, not to provide complete coverage of the area. The presentation favors simplicity to preciseness, examples to formal presentation, and intuition to rigor.

2 The price of anarchy

Consider a set of users that share the resources of a system. In an ideal situation the users behave in a way that optimizes the objective of the system. However, if the users are selfish, they will act in a way that optimizes their own individual and usually conflicting objectives. As an example, consider a set of users that compete for bandwidth in a network. A similar situation arises when users submit tasks to be executed on machines. This is the case that will concern us in this note. Ideally the tasks are allocated to machines so that the makespan (or some other objective) is minimized. Selfish users however will be interested in minimizing their own completion time. This behavior may result in suboptimal allocation. The price of anarchy, introduced in [8], tries to address in a simple way how much is lost due to selfish behavior.

The first issue that arises in this approach is to characterize selfish behavior [15]. The classical approach assumes that the strategies of the users form a Nash equilibrium: At a Nash equilibrium no user can improve unilaterally its objective by selecting another strategy. This is the most robust concept of equilibrium and it has some nice properties; most notably that, in finite games, there always exists a mixed Nash equilibrium [12]. It has however some serious drawbacks. For example, it is not clear how the users are going to end up at a Nash equilibrium. This issue can be split into two questions: *How long* will it take for users to reach or converge to a Nash equilibrium? And, *which Nash equilibrium* will they play when a game has more than one equilibria? An obvious lower bound to convergence time is the computational complexity of finding a Nash equilibrium. This appears to be rather high but it is the most outstanding open problem at the intersection of Game Theory and Computational Complexity [16, 19]. It is not known whether the problem is in **P**; my belief is that for 2 players it is indeed in **P** but for 3 or more players it is not. The second issue of how the players will agree on a particular Nash equilibrium is more relevant to the concept of the price of anarchy: The most natural approach is to assume nothing so that the users may end up at any Nash equilibrium. Therefore to bound the inefficiency due to selfish behavior, we consider the worst-case Nash equilibrium.

Another issue that has yet to be explored algorithmically is how much information is available to the players. Here we assume that the players have *complete information*.

3 Task allocation

Perhaps the easiest way to introduce the issues related to the price of anarchy of task allocation is by an example: Consider three tasks of length 1, 2, and 3 to be executed on two identical machines. Each task is controlled by a selfish user who wants to select a machine to minimize the completion time of its own task. The completion time for a task depends on the tasks allocated to its machine as well as on the scheduling policy of that machine. For now we assume that the scheduling policy executes the tasks in random order, but we will return to this issue later. The situation faced by the users is essentially captured by a 3-player game which happens to have 5 Nash equilibria. In one of the Nash equilibria the first two tasks select the first machine and the third task selects the second machine. In another, which is a mixed (randomized) equilibrium, the first task goes to the first machine, the second task with probability 1/4 goes to the first machine and with the remaining probability goes to the second machine, and the third task goes to the first or second machine with probabilities 1/3 and 2/3, respectively. It is easy to check that this is indeed a Nash equilibrium. It should also be clear that the first of the two equilibria is better for the system —it has optimal makespan 3, while the *expected* makespan of the second equilibrium is $9/2$ $(= \frac{1}{4}\frac{1}{3}6 + \cdots + \frac{3}{4}\frac{2}{3}5)$. The price of anarchy in this case is (at least) $(9/2)/3 = 3/2$.

It is straightforward to generalize this example to the general case of n players/tasks with lengths $w_1, \ldots, w_n$ and m machines. The price of anarchy for m machines is defined as the worst-case ratio of the makespan of a Nash equilibrium over the optimal makespan $\text{opt}(w_1, \ldots, w_n)$:

$$\text{PA}_m = \max_{w_1,\ldots,w_n} \max_{\text{Nash eq. } E} \frac{\text{makespan}(E)}{\text{opt}(w_1, \ldots, w_n)}.$$

What is the price of anarchy of the general case?

Theorem 1 *The price of anarchy PA_m for m identical machines which execute their tasks in random order is $\Theta(\log m / \log\log m)$. In particular, for $m = 2$ the price of anarchy is $3/2$.*

Let me give a rough sketch of the proof [8, 1, 6]. The lower bound is easy: Consider m tasks of size 1. The optimal allocation is to assign each task on a separate machine. On the other hand, there is a Nash equilibrium at which each user selects randomly (and uniformly) among the m machines. The expected makespan is equal to the maximum number of tasks on a machine. This is the classical bins-and-balls problem [10] and the expected maximum turns out to be $O(\log m / \log\log m)$ and the lower bound follows.

To show the upper bound, we need to bound the expected makespan of the Nash equilibria and the optimal makespan. An obvious lower bound for the latter is $\max\{w_i, \sum_i w_i/m\}$ (the maximum task and the average load). The first quantity, the expected makespan, which is the expected maximum load, can be bounded indirectly. First we bound the maximum expected load: Intuitively, the expected load of a given machine cannot be much greater than the optimum, otherwise some player will have incentive to switch machines. By making this precise, we get that for each machine, its expected load at a Nash equilibrium is at most twice the optimum [8]. This is the only property of Nash equilibria that we need. To summarize: At a Nash equilibrium each player selects with some probability distribution a machine so that the expected load on each machine is at most $2\max\{w_i, \sum_i w_i/m\}$ and the question is what is the expected maximum load. This is a bins-and-balls situation with balls of arbitrary sizes and arbitrary probability distributions. But since the expected load on each machine is low we can use a Hoeffding bound [5, 1] to get that the expected maximum is at most $O(\log m/\log\log m)$ times the maximum expectation. The latter as we mentioned is in turn at most twice the optimum and the upper bound follows.

Czumaj and Vöcking [1] extended Theorem 1 to machines of different speeds: The price of anarchy for this case is $O(\log m/\log\log\log m)$.

It is disheartening that this proof makes so little use of the properties of Nash equilibria. Mavronikolas and Spirakis [11], proposed an interesting conjecture which strengthens the result of Theorem 1 and has potentially more game-theoretic nature. To describe the conjecture we need the notion of fully-mixed Nash equilibrium: A Nash equilibrium is *fully-mixed* when it assigns nonzero probability to every strategy.

Open Problem 1 (Fully-mixed Nash equilibrium conjecture) *The conjecture states that when the fully-mixed equilibrium exists then it has the maximum makespan among all Nash equilibria. Is it true?*

If the conjecture is true, then we can combine it with a result in [11] that bounds the price of anarchy of the fully-mixed Nash equilibrium to obtain Theorem 1. There is some progress on settling the conjecture [3, 9].

An interesting and unexplored extension of Theorem 1 is to consider games with many rounds. There are more than one variants of this extension and I will mention perhaps the cleanest one: At each round player i has to schedule a task of length w_i (the same for every round) knowing the allocations of the previous rounds.

Open Problem 2 *What is the price of anarchy for k rounds? It clearly tends to 1 as k tends to infinity, but at what rate?*

4 Coordination mechanisms

At this point the reader may wonder whether this topic is appropriate for an algorithmic column since I didn't mention any algorithmic issues. But such issues exist. To introduce them, let's recall the assumption that each machine executes its tasks in random order, and let's ask the question: *Are there scheduling policies that result in improved price of anarchy?* It is natural to consider local scheduling policies in which the schedule on each link depends *only on the loads of the link*. Otherwise, an obvious solution would be to force an optimal allocation to each link. It is also natural to allow each link to give priorities to the loads and perhaps introduce delays. A set of scheduling policies will be called a *coordination mechanism* [7].

I now define the problem more precisely: There is a finite set of players $N = \{1, \dots, n\}$ and m identical machines. Machine j has a scheduling policy c^j which receives tasks from a subset of the players and decides how to execute them. The input is the vector $(w_1, \dots, w_n)$ of the length of tasks that allocated to machine j. Naturally $w_i = 0$ when task i is not allocated to machine j. Notice that the input is a vector, not a set of tasks. This is equivalent to saying that the machines can order the tasks consistently; this is definitely true for tasks of distinct lengths but for tasks of equal length, the machines need some id to lexicographically order them. Without this assumption, when machines cannot distinguish between players of equal-length tasks, the bins-and-balls argument shows that the price of anarchy is still $O(\log m / \log \log m)$.

The scheduling policy of a machine is essentially determined by the completion times of its tasks. Let $c_i^j(w_1, \dots, w_n)$ denote the completion time of w_i which should satisfy the following natural constraints:

- When w_i is 0, i.e., the i-th task is not allocated to machine j, c_i^j is 0.
- For every subset S of players, the maximum completion time of the players in S must be at least equal to the total length of the tasks in S: $\max_{i \in S} c_i^j(w_1, \dots, w_n) \geq \sum_{i \in S} w_i$. As an example, a machine could schedule two tasks w_1 and w_2 so that the first task finishes at time $w_1 + w_2/2$ and the second task at time $2w_1 + w_2$.

Fix a coordination mechanism $c = (c^1, \dots, c^m)$, a set of tasks $w = (w_1, \dots, w_n)$ —some of them may be 0 indicating that the associated player does not participate. This defines a game between the tasks (players). Let

E be a Nash equilibrium of this game and let makespan$(w; c; E)$ denote its makespan. We define the price of anarchy of the coordination mechanism c as the maximum over all sets of tasks w and all Nash equilibria E of its makespan over the optimum makespan.

$$\mathrm{PA}(c) = \max_{w} \max_{\text{Nash eq. } E} \frac{\text{makespan}(w; c; E)}{opt(w)}$$

To illustrate the issues, we discuss first a simple coordination mechanism for two machines:

> The tasks are ordered by length. If two or more tasks have the same length, their order is the lexicographic order of the associated players. The first machine schedules its tasks in order of *increasing* length while the second facility schedules its tasks in order of *decreasing* length.

The mechanism aims to break the symmetry of tasks. With this mechanism, it is easy to see that the player with the minimum task goes always to the first machine. Similarly, the agent with the maximum task goes to the second machine.

The following is not hard to show:

Proposition 1 *The above increasing-decreasing coordination mechanism has price of anarchy* $4/3$. *In particular, for* $n = 3$ *players, it has price of anarchy 1.*

To show for example that the price of anarchy of the mechanism is no better than $4/3$, consider 4 tasks with lengths $1, 1, 2, 2$. Then there is a Nash equilibrium in which the first two tasks go to the first machine while the other two tasks go to the second machine (this happens to be a pure Nash equilibrium). Its price of anarchy is $4/3$.

Is there a coordination mechanism with smaller price of anarchy? Notice that the situation resembles the framework of competitive analysis of online algorithms:

> We, the designers, select a coordination mechanism $c = (c^1, \ldots, c^m)$, essentially a distributed scheduling algorithm. Then the adversary selects tasks $w = (w_1, \ldots, w_n)$ (some of them 0 indicating that the associated player does not participate). We then compute the worst-case expected makespan among Nash equilibria and divide by the optimum to get the price of anarchy.

Surprisingly, there are coordination mechanisms that have price of anarchy less than 4/3. Compare this to $\Theta(\log m/\log\log m)$ —the price of anarchy of the mechanism that executes the tasks in random order. In fact, any mechanism that has the same policy on each machine has price of anarchy $\Theta(\log m/\log\log m)$ (the balls-and-bins lower bound applies in this case too). The following mechanism breaks the symmetry in a simple way:

- Each machine schedules the tasks in order of decreasing length (using the lexicographic order to break any potential ties).
- Every task of machine j is delayed enough so that it finishes only at times t with $t = j \pmod{m}$.

For tasks of very large size, the delay introduced by the second rule is insignificant. But for tasks of small size, the delay may be significant; fortunately, there are simple ways to rectify the rule to make the delay arbitrarily small.

The above coordination mechanism has the nice property that there exists exactly one Nash equilibrium: The largest task knows that independently of the choices of the other players, it will be first on every machine. Furthermore, the completion times on the m machines are distinct and therefore there exists a unique optimal choice. This optimal choice is also known to the second largest task which with similar considerations selects a particular machine and so forth. This leads to greedy scheduling of the tasks in order of decreasing size [7]:

Theorem 2 *The above coordination mechanism for n players and m facilities has price of anarchy* $4/3 - 1/3m$.

Is there a coordination mechanism with better price of anarchy? It is an interesting open problem to determine the best price of anarchy achievable by coordination mechanisms. No lower bound better than 1 is known. The following intuitive non-rigorous argument shows how to establish non-trivial lower bounds: Consider $m = 2$ machines and $n = 5$ players with tasks of lengths $(3, 3, 2, 2, 2)$. After we fix the coordination mechanism the adversary selects either this input or an input of which one of the tasks of length 2 is missing, i.e., one of $(3, 3, 0, 2, 2)$, $(3, 3, 2, 0, 2)$, and $(3, 3, 2, 2, 0)$. Notice that the optimal allocations assign the 3's to the same or different machines depending on whether there are 3 2's or not. The local policies of a coordination mechanism cannot distinguish between the two cases and therefore cannot always achieve optimal allocation. Turning this intuitive argument into a concrete lower bound doesn't appear to be easy and it remains an open problem. To summarize, the best known upper bound is given by Theorem 2 and the best known lower bound is 1.

Open Problem 3 *Does the coordination mechanism of Theorem 2 have optimal price of anarchy among all coordination mechanisms? If not, show better upper and lower bounds.*

As I mentioned above, Czumaj and Vöcking [1] showed that the price of anarchy when the machines have different speeds is $O(\log m/\log\log\log m)$ (when each machine schedules its tasks in random order).

Open Problem 4 *How much can coordination mechanisms improve the price of anarchy for machines of different speeds? A coordination mechanism similar to the one of Theorem 2 can reduce the price of anarchy to a constant. Is there a better one?*

4.1 Truthful coordination mechanisms

In the traditional Game Theory there is a parallel of coordination mechanisms, Mechanism Design [13, 14]. A central concept in Mechanism Design is the notion of truthfulness. Similar issues arise for coordination mechanisms. In particular, the coordination mechanism of Theorem 2 has the property that it favors (schedules first) large tasks. This is undesirable since it gives incentive to players to lie and pretend to have larger tasks. For example, a selfish agent will pad its task to increase its length if this will guarantee a better completion time. Are there coordination mechanisms that avoid this problem? More precisely, let's call a coordination mechanism *truthful* when no player can unilaterally improve its completion time by increasing the length of its task.

As an example of a truthful coordination mechanism consider the mechanism of Theorem 2, but change the first rule so that the tasks in each machine are scheduled in order of increasing length. Now a similar argument establishes that starting from the task of minimum length, each task selects a unique machine. One can show that this coordination mechanism has price of anarchy $2 - 1/m$ [4]. Although this is greater than $4/3 - 1/3m$, the mechanism is very robust in that the players have no incentive to lie.

Open Problem 5 *Are there truthful coordination mechanisms with better price of anarchy? If there are, prove better upper and lower bounds.*

5 Conclusions

The study of selfish task allocation has motivated the new area of price anarchy. The initial questions have been successfully answered but many more problems remain open. I mentioned some of them above but there are many more. For example, the price of anarchy and more generally coordination mechanisms for objectives other than the makespan, such as (weighted) average completion time, have not been studied yet.

Finally, the notion of coordination mechanism can be extended to selfish routing and other generalizations of congestion games [7]. In particular, coordination mechanisms raise intriguing new questions for the selfish routing model of Roughgarden and Tardos [18].

For a more expanded treatment of the issues of this note please see the original publications. Most of these are surveyed by Czumaj in [2], except for coordination mechanisms which are discussed in [7].

References

[1] A. Czumaj and B. Vöcking. Tight Bounds for Worst-case Equilibria. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms,* pp. 413–420, January 2002.

[2] A. Czumaj Selfish routing on the Internet. http://www.cis.njit.edu/~czumaj/PUBLICATIONS/Selfish-Routing-Survey-2003.html

[3] D. Fotakis, S. Kontogiannis, E. Koutsoupias, M. Mavronicolas, and P. Spirakis. The structure and complexity of Nash equilibria for a selfish routing game. In *ICALP*, pp. 123–134, Malaga, Spain, 2002

[4] R. L. Graham. Bounds for certain multiprocessing anomalies, *Bell System Technical Journal*, 45, pages 1563-1581, 1966.

[5] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, March 1963.

[6] E. Koutsoupias, M. Mavronicolas and P. Spirakis. Approximate Equilibria and Ball Fusion. In *Proceedings of the 9th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, 2002

[7] E. Koutsoupias and A. Nanavati. Coordination mechanisms. http://www.di.uoa.gr/~elias/publications/paper-kn03.pdf

[8] E. Koutsoupias and C. H. Papadimitriou. Worst-case equilibria. In *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science*, pages 404-413, 1999.

[9] T. Luecking, M. Mavronicolas, B. Monien, M. Rode, P. Spirakis, I. Vrto. Which is the Worst-case Nash Equilibrium? In *MFCS 2003*.

[10] R. Motwani and P. Raghavan, *Randomized Algorithms,* Cambridge University Press, 1995.

[11] M. Mavronicolas and P. Spirakis. The price of selfish routing. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 510–519, Hersonissos, Crete, Greece, July 6–8, 2001.

[12] J. F. Nash, "Non-cooperative Games," *Annals of Mathematics,* Vol. 54, No. 2, pp. 286–295, 1951.

[13] N. Nisan. Algorithms for selfish agents: Mechanism design for distributed computation. In *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science*, pages 1–15, 1999.

[14] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35:166–196, 2001.

[15] M. J. Osborne and A. Rubinstein. *A Course in Game Theory.* The MIT Press, 1994.

[16] C. H. Papadimitriou. Algorithms, games, and the Internet. In *Proceedings of the 33rd Annual ACM Symposium on the Theory of Computing*, pages 749-753, 2001.

[17] C. Papadimitriou. Game theory and the Internet (course notes). http://www.cs.berkeley.edu/˜christos/games03/cs294.html

[18] T. Roughgarden and E. Tardos. How bad is selfish routing? *Journal of the ACM*, 49(2):236-259, 2002.

[19] B. von Stengel. Computing equilibria for two-person games. *Handbook of Game Theory*, Vol. 3, eds. R. J. Aumann and S. Hart, North-Holland, Amsterdam.