Coordination mechanisms

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

October 29, 2004

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 [25]. In this note, I will focus on the last area and in particular on the price of anarchy of task allocation and selfish routing. I will discuss the issues of this area, mention the central results, and suggest some open problems[1].

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

1.1 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. Typical examples include the selfish task allocation problem—in which users have tasks to schedule on machimes and compete for the execution time—and the selfish routing problem—in which selfish traffic competes for the bandwidth of a network. Both are variants of congestion games [26, 18].

The task allocation problem will concern us in the first part of this note. Ideally the tasks are allocated to machines so that the makespan (or some other objective) is minimized. Selfish users however are only interested in minimizing their own completion time. This behavior may result in suboptimal allocation. The price of anarchy, introduced in [15], tries to address in a simple way how much is lost due to selfish behavior.

[1] This is an updated and expanded version of a note that I wrote for the Bulletin of EATCS [12].

The first issue that arises in this approach is to characterize selfish behavior [23]. 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 [20]. 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 the most outstanding open problem at the intersection of Game Theory and Computational Complexity [24, 30]. It is not known whether the problem is in **P**; my guess is that for 2 players it is indeed in **P** but for 3 or more players it is not. There is a well-known natural pivot algorithm, the Lemke-Howson algorithm, to compute a Nash equilibrium which is similar to Simplex. In the worst case, it takes exponentially many steps (see, for example, a recent work [29] which employs in a clever way cyclic polytopes).

The second issue is about how the players agree on a particular Nash equilibrium and it 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 in depth with respect to the price of anarchy is how much information is available to the players. Here we assume that the players have *complete information*.

2 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 $\mathrm{opt}(w_1, \ldots, w_n)$:

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

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 [15, 4, 13]. 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 [17] 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 [15]. 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 [10, 4] 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 [4] extended Theorem 1 to machines of different speeds: The price of anarchy for this case is at most $O(\log m/\log\log\log m)$.

It is disheartening that this proof makes so little use of the properties of Nash equilibria. Mavronikolas and Spirakis [19], 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. The fully-mixed equilibrium for identical machines, is when each player the selects randomly and uniformly a machine. For non-identical machines the fully-mixed Nash equilibrium may not exist.

Open Problem 1 (Fully-mixed Nash equilibrium conjecture) *The fully-mixed equilibrium, when it exists, has the maximum makespan among all Nash equilibria.*

If the conjecture is true, then we can combine it with a result in [19] that bounds the price of anarchy of the fully-mixed Nash equilibrium to obtain Theorem 1. There is some progress on settling the conjecture [7, 16, **?**].

2.1 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* [3].

I now define the problem more precisely: There is a finite set of players $N = \{1, \ldots, n\}$ 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 $(w_1, \ldots, w_n)$ 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 $c_i^j(w_1, \ldots, w_n)$ 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: $\max_{i\in S} c_i^j(w_1, \ldots, w_n) \geq \sum_{i\in S} w_i$. 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 $c = (c^1, \ldots, c^m)$, a set of tasks $w = (w_1, \ldots, w_n)$ —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 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 (with dominant strategies): 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 [3]:

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 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 always 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—especially for mechanisms that may have no pure Nash equilibrium—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 2 *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 [4] 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 3 *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* $2-2/(m+1)$. *Is there a better one?*

2.2 Truthful coordination mechanisms

In the traditional Game Theory there is a parallel of coordination mechanisms, Mechanism Design [21, 22]. 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 has a dominant strategy and ends up at a unique machine. One can show that this coordination mechanism has price of anarchy $2-1/m$ [9]. 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 4 *Are there truthful coordination mechanisms with better price of anarchy? If there are, prove better upper and lower bounds.*

A related issue is the notion of randomized coordination mechanisms. Are they better than their deterministic counterparts?

3 Coordination mechanisms for selfish routing

The above ideas can be naturally extended to the selfish routing problem whose price of anarchy study was initiated by Roughgarden and Tardos [28]. This problem together with the task allocation problem are the two major problems studied so far with respect to the price of anarchy.

An instance (G, ℓ, r) of the (single-commodity) selfish routing problem, is defined by a network G with latency functions ℓ_e on each edge e, an origin s, a destination t, and rate r. In the more general multi-commodity selfish routing problem there are more than one pairs of source and destination, but in this note we will deal mainly with single-commodity networks. Intuitively, a flow can be thought of consisting of (infinitely) many players, each one controlling an infinitesimal amount of flow, in the same way that a driver in a large city drives a single car and his decisions do not really affect the traffic experienced by the others.

The two typical examples of selfish routing are shown in Figure 1. Consider for example, the network on the right, the well-known Braess' paradox network, where a flow of rate 1 must travel from A to D and experiences delay on each edge given by the latency functions on the edges. At the unique Nash equilib-

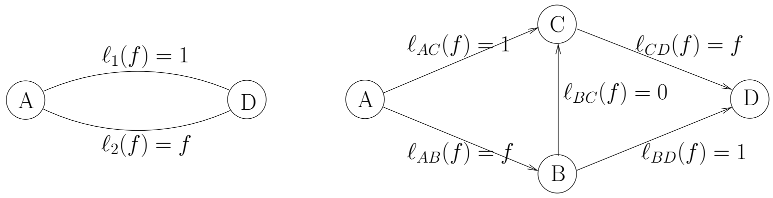

Figure 1: Two parallel links and the Braess' network

rium all the flow follows the path $ABCD$ with latency 3/2. On the other hand, the optimal solution (which minimizes the average or the maximum latency) is to split the flow evenly on the paths ABD and ACD with latency 3/2. The price of anarchy in this case is 4/3. A similar situation occurs on the example with the two parallel links. A celebrated theorem by Roughgarden and Tardos [28] asserts that 4/3 is the maximum price of anarchy for linear networks, that is, networks with linear latency functions.

The natural question is what can we do to improve the price of anarchy in this situation. One proposal [27], inspired by the Braess' paradox, is to remove some edges. For example, in the Braess' network, when we remove the edge BC, the price of anarchy drops to 1 (hence the paradox). But this approach is very restricted; it does not even improve the price of anarchy on the two parallel links network.

3.1 Taxes

A more general proposal is to try to improve the price of anarchy by adding taxes (tolls) to edges. More precisely, a tax τ_e on an edge e raises the cost from $\ell_e(f)$ to $\ell_e(f) + \tau_e$. Note that the tax on an edge is a constant, independent of the flow on the edge.

Can taxes improve the price of anarchy? The answer depends on whether we are interested on $\ell_e(f)$ or $\ell_e(f) + \tau_e$. From now on we will refer to the first as latency and to the later as cost (cost is equal to latency plus taxes). If we are interested on the latency only, then there is a nice characterization of taxes that improve the price of anarchy to 1. This is achieved by the marginal cost tax which is based on the principle that a user pays on each edge a tax equal to the additional delay his presence causes to the other users of the edge. If the derivatives ℓ'_e of the latency functions exist, then the marginal cost tax is $\tau_e = f^*_e \cdot \ell'_e(f^*_e)$, where f^*_e is the flow on edge e in the optimal solution. As an example, the marginal cost tax on the Braess' network (when the rate is 1) changes the cost on the edges AB and CD from f to $f + 1/2$ and leaves everything else unchanged. An old theorem [**?**] asserts that:

Proposition 2 *The marginal cost tax achieves optimal latency for every network (even non-linear ones).*

Even for heterogeneous networks in which players may differ in their sensitivity to taxes, there is always a set of taxes that achieve optimal latency. For single-commodity networks this was shown in [1] and was recently extended to multi-commodity networks in [6, 11].

If we however are interested in minimizing the cost, which is latency plus tax, the situation is more complicated. This is the case that we will consider from now on. When we compute the price of anarchy, taxes affect only the cost of the Nash equilibrium, so we still divide by the original optimal cost. Note first that the marginal cost tax fails to improve the situation. In particular, Cole, Dodis, and Roughgarden [2] showed that for every linear network, the marginal cost tax can only increase the cost. But for other tax policies, the situation may be much better. For example, in the Braess' network, when we add a large tax to the edge BC, the price of anarchy drops to 1.

A natural algorithmic problem is to compute, for a given network and rate, optimal taxes—taxes for which the worst-case Nash equilibrium has minimum cost. For networks with linear delays, a theorem in [2] asserts that there are optimal taxes that are either 0 or ∞ on each edge. Essentially this shows that for linear networks, taxes are not really more powerful than edge removals. It is known that the problem of computing the optimal edge removals is an NP-hard problem [27]. Another immediate implication is that taxes fail to improve the worst-case price of anarchy for linear networks; in particular, for the two parallel links network of Figure 1, taxes cannot improve the price of anarchy below 4/3. To summarize:

Proposition 3 *With constant taxes the worst-case price of anarchy for linear networks remains* $4/3$.

3.2 Beyond taxes and fixed rates

In the framework of the price of anarchy, the restriction in which the rates are fixed seems in some cases unnatural. Are we, for example, interested in the price of anarchy of a transportation network at the rush hour or in the middle of the night? What if we want to find the worst-case price of anarchy of a given network when an adversary selects the rates? It should be clear however that in the framework of Roughgarden and Tardos nothing changes essentially; the reason is that in this framework, we simply observe or measure the inefficiency of the system. The situation however changes drastically when we act in order to improve the efficiency of a network. By employing the competitive analysis setting, we first design our mechanism and then an adversary selects the rate that maximizes the price of anarchy.

In another direction, taxes were so far assumed to be constant, independent of the flow. For practical reasons, in transportation networks for example, it may be hard to implement taxes that depend on the traffic. Furthermore, for fixed rates this makes no difference mathematically. But when we allow an adversary to select the rate, constant and varying taxes are completely different. The way the marginal cost tax is defined (as the value of the function $f \cdot \ell'(f)$ for a particular value f) also shows that it may be more natural to consider taxes that depend on the flow. The tax on edge e can depend on the rates (a global policy) or the flow of the edge e (a local, distributed policy). The natural approach, especially in vast networks, is to consider only the later case.

So, from now on we will consider variable taxes $\tau_e(f_e)$ which are functions of the flow that crosses an edge. The following therefore is a natural question:

> Given a network, find taxes $\tau_e(f_e)$ that minimize the worst case price of anarchy over all rates.
>
> This problem is close in spirit to online algorithms: we select the taxes, the adversary selects the rates, we compute the cost (latency plus taxes) of the worst Nash equilibrium, and divide by the optimal latency.

For example, recall that constant taxes fail to improve the situation on the network with two parallel edges of Figure 1. But with varying taxes, the price of anarchy drops to 1, as follows: add a tax to the lower edge

$$\tau_2(f) = \begin{cases} 0 & \text{if } f < 1/2 \\ 1/2 & \text{otherwise} \end{cases}$$

Notice that this is not continuous function. This is unavoidable, as it is not hard to see that every continuous tax for this network fails to lower the cost.

The example of the Braess' network with rate 1 is more illuminating. Recall that when we remove the edge BC (or equivalently add tax $\tau_{BC} = \infty$), the price of anarchy drops to 1. For other rates however, the removal of the edge has devastating effects. In particular for small rates $r \leq 1/2$, for which the optimal uses the edge BC, it is easy to calculate that the price of anarchy becomes unbounded (when r tends to 0, each player has cost $1 + r/2$, while the optimum uses the edge BC and has cost at most $2r^2$). So, while the removal of the edge is an optimal solution for fixed rate 1, it is a bad solution for other rates. Can we find a better solution with varying taxes? Yes, albeit there is no way to improve the price of anarchy to 1. For example, with the tax

$$\tau_{BC}(f) = \begin{cases} 0 & \text{if } f < 2/3 \\ 1 & \text{otherwise} \end{cases}$$

the price of anarchy becomes 16/15. Notice again that we employed a non-continuous function, although in this case, there exists continuous taxes that improve the price of anarchy below 4/3. The above examples suggest the following problem:

Open Problem 5 *Which is the worst-case price of anarchy for linear networks when we employ varying taxes* $\tau_e(f_e)$.

Since optimal taxes may turn out to be complicated functions, a natural restriction is to consider piecewise linear taxes.

4 Congestion games and coordination mechanisms

The task allocation problem and the selfish routing problem share common characteristics. This is because they are both variants of congestion games [26, 18]. A *congestion game* is a tuple $(N, M, (\Sigma_i)_{i\in N}, (c^j)_{j\in M})$ where $N = \{1, \ldots, n\}$ is the set of players, M is a set of facilities (machines or edges), Σ_i is a collection of strategies for player i: a strategy $A_i \in \Sigma_i$ is a subset of facilities, and finally c^j is the cost functions of facility j: when k players use facility j, the cost for each player is $c^j(k)$ which depends only on the number of players, not their identities. An important special case is when the cost functions c^j are linear.

There are essentially two differences between congestion games and the selfish task allocation problem. First, congestion games are more general than the task allocation problem in that each job wants service from a subset of machines instead from a single machine. Second, congestion games is more restricted, in that the jobs (players) are identical. In other words, the task allocation problem is the special case of *weighted* congestion games on a network of parallel edges. In a weighted congestion game, each player i has a weight w_i and the cost of a facility is a function of the total weight assigned to this facility. The selfish routing problem, on the other hand, is a variant of (unweighted) congestion games with infinitely many players (it is called non-atomic).

As an example of an (unweighted) congestion game, consider the network of Figure 2 and two players. Player 1 wants to use a path from node A to node E and player 2 wants to use a path from node B to node E. The cost on each edge is equal to the number of players using it. The strategies ABDE and BACE are at a Nash equilibrium with price of anarchy 2. A nice theorem by Rosenthal [26], who first proposed congestion games, asserts that every unweighted congestion game has at least one pure Nash equilibrium. This is not true however for weighted congestion games (not even network congestion games [8]).

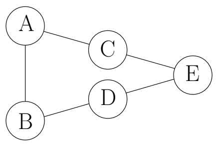

Figure 2: A simple congestion game.

Coordination mechanisms can be naturally extended to weighted congestion games:

> We, the designers, select delays (or taxes) and priorities on the facilities, the adversary selects the weights of the players, and then we compute the price of anarchy of the worst case Nash equilibrium.

There are symmetric (when we are allowed to select only delays) and asymmetric mechanisms (when we can also select priorities). In the discussion above, we considered asymmetric mechanisms for the task allocation and symmetric ones for the selfish routing problem.

Another issue that we have to deal with is the notion of social (system) cost. The price of anarchy depends not only on the game itself but also on the social cost. For example, in the task allocation problem above, we were interested in the makespan, and in the selfish routing problem in the average latency. From the system designers point of view there are two natural notions of social cost: The maximum or the average cost, among the players; suitably, the first was used for the task allocation problem and the second for the selfish routing problem.

5 Conclusions

The study of selfish task allocation has motivated the new area of price anarchy. Another central problem in the area turned out to be the selfish routing problem. The initial questions concerning the price of anarchy have been successfully answered but many more problems remain open. I mentioned some of them above but there are many more.

With many interesting cases of congestion games (weighted or unweighted, single-commodity or multi-commodity or general, atomic or non-atomic, splittable or unsplittable flow, etc), with more than one notions of social cost, and with different types of coordination mechanisms (symmetric or asymmetric), it is clear that there a lot of open questions about coordination mechanisms for congestion games. Which ones of them are tractable or even interesting remains to be seen.

For a more expanded treatment of the issues of this note please see the original publications. Most of the early results are surveyed by Czumaj in [5]. For taxes a good source is [1, 2]. Finally, coordination mechanisms for the task allocation problem are discussed in [3].

References

[1] R. Cole, Y. Dodis, and T. Roughgarden. Pricing network edges for heterogeneous selfish users. In *STOC 2003*, pages 521–530, 2003.

[2] R. ole, Y. Dodis, and T. Roughgarden. How much can taxes help selfish routing? In *ACM EC 2003*, pages 98–107, 2003.

[3] G. Christodoulou, E. Koutsoupias, and A. Nanavati. Coordination Mechanisms. In *Proceedings of the 31st International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 345–357, Turku, Finland, July 2004.

[4] 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.

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

[6] L. Fleischer, K. Jain, and M. Mahdian. Tolls for heterogeneous selfish users in multicommodity networks and generalized congestion games. In FOCS 2004, pages 277–285, 2004.

[7] 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

[8] D. Fotakis, S. Kontogiannis, and P. Spirakis. Selfish unsplittable flows. In *ICALP 2004*, pages 593–605, 2004.

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

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

[11] G. Karakostas and S. Kolliopoulos. Edge pricing of multicommodity networks for heterogeneous selfish users. In *FOCS 2004*, pages 268–276, 2004.

[12] E. Koutsoupias. Selfish task allocation. *Bulletin of EATCS*, Number 81, pages 79–88, October 2003.

[13] 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

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

[15] 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.

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

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

[18] D. Monderer and L. S. Shapley. Potential Games. *Games and and Economic Behavior* 14, pages 124-143, 1996.

[19] 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.

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

[21] 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.

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

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

[24] 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.

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

[26] R. W. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2:65-67, 1973.

[27] T. Roughgarden. Designing networks for selfish users is hard. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, pages 472–481, 2001.

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

[29] R. Savani and B. von Stengel. Exponentially many steps for finding a Nash equilibrium in a bimatrix game. In *FOCS 2004*, pages 258–267, 2004.

[30] 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.