

Selfish Load Balancing and Atomic Congestion Games*

Subhash Suri[†]

Csaba D. Tóth[‡]

Yunhong Zhou[§]

Abstract

We revisit a classical load balancing problem in the modern context of decentralized systems and self-interested clients. In particular, there is a set of clients, each of whom must choose a server from a permissible set. Each client has a unit-length job and selfishly wants to minimize its own latency (job completion time). A server’s latency is inversely proportional to its speed, but it grows linearly with (or, more generally, as the p -th power of) the number of clients matched to it. This interaction is naturally modeled as an atomic congestion game, which we call *selfish load balancing*. We analyze the Nash equilibria of this game and prove nearly tight bounds on the *price of anarchy* (worst-case ratio between a Nash solution and the social optimum). In particular, for linear latency functions, we show that if the server speeds are relatively bounded and the number of clients is large compared to the number of servers, then *every Nash assignment approaches social optimum*. Without any assumptions on the number of clients, servers, and server speeds, the price of anarchy is at most 2.5. If all servers have the same speed, then the price of anarchy further improves to $1 + 2/\sqrt{3} \approx 2.15$. We also exhibit a lower bound of 2.01. Our proof techniques can also be adapted for the *coordinated* load balancing problem under L_2 norm, where it slightly improves the best previously known upper bound on the competitive ratio of a simple greedy scheme.

1 Introduction

Consider a set U of n selfish clients, each of whom must choose a server from a set V , in the absence of a coordinating authority. There is a bipartite graph G between U and V and a server j is *permissible* for client i only if (i, j) is an edge in G . Each client has a unit-length job and selfishly wants to minimize its latency (job completion time), and rationally prefers a fast server to a slower one. Servers can have different speeds, and a server’s latency is inversely proportional to its speed, but it is an increasing function of the server load (the number of clients served by it).

Each client independently trying to minimize its latency is essentially engaged in a game with other selfish clients. We call this the *selfish load balancing* game. Unlike traditional load balancing, however, the clients are not interested in optimizing the social welfare (e.g., *total* system-wide latency). Instead, each client has its own private objective. The stable outcomes of these interactions are the *Nash Equilibria*—outcomes in which no single client can improve its latency by switching unilaterally. Centralized optimal solutions, in general, are not stable—one or more clients may improve their latency by switching, while worsening the latency for others. On the other hand, the cost of Nash equilibrium solutions can be much worse than that of centralized outcomes, and Papadimitriou [17] has coined the term “price of anarchy” to denote the *worst-case ratio between a Nash outcome and the social optimum*. In this paper, we give nearly tight bounds for the price of anarchy in the selfish load balancing game.

*Research by the first two authors was partially supported by NSF grants CCR-9901958 and ANI-9813723. A preliminary version of this work has appeared in the *Proceedings of the 16th ACM Symposium on Parallelism in Algorithms and Architectures* (Barcelona, 2004), pp. 188–195.

[†]Department of Computer Science, University of California at Santa Barbara, CA 93106, USA. suri@cs.ucsb.edu

[‡]Department of Mathematics, Room 2-336, MIT, Cambridge, MA 02139, USA. toth@math.mit.edu

[§]Hewlett-Packard Laboratories, 1501 Page Mill Rd, Palo Alto, CA 94304, USA. yunhong.zhou@hp.com

Nash equilibrium is a compelling solution concept for decentralized systems with self-interested players. Unfortunately, the concept is descriptive, not prescriptive: it does not suggest algorithms for computing an equilibrium and computing Nash equilibria remains a topic of current research. We, therefore, also investigate the following obvious *greedy* strategy: clients arrive in the system online in an arbitrary order; upon arrival, each client selects a permissible server with the least current latency, and *this selection is irrevocable*. The greedy is a myopic strategy—each client makes the best choice available to it at the moment, although future choices by *other clients* may make it regret that selection. While greedy does not generally lead to Nash solutions, it does have the advantage of computational simplicity. Thus, a natural question to ask is: *how bad is the greedy assignment in the worst case?*

The greedy strategy has been analyzed before in the context of *centralized* L_2 norm load balancing [3]—the goal there is to assign clients so as to minimize the L_2 norm of the server loads. Because the total latency of all the clients is intimately related to the *squared sum of the server loads*, our techniques also lead to improved bounds for the competitive ratio of the greedy scheme of [3]. Interestingly, the Nash solutions are strictly better, suggesting the following conclusion: *despite lack of central coordination, selfish players find solutions that are better than greedy, which assumes centralized control but non-selfish players.*

1.1 Motivation and Model

Our load balancing game is inspired by the emerging class of Internet-centric applications like the peer to peer (P2P) networks, but it has broader implications for any *uncoordinated* distributed system. In a P2P system, for instance, data are often replicated to enable a high level of availability and fault tolerance. Thus, users typically have choice of many hosts from whom to download their data; each user wants to minimize its own latency (time to download); and there is no central authority to dictate a user’s choice.

An instance of the load balancing game is modeled as a bipartite graph G between a set U of n clients and a set V of m servers. A client i can be assigned to server j only if (i, j) is an edge in G . An outcome of the game is an assignment where each client is assigned to one of its permissible servers. Suppose server j has speed σ_j and is matched to ℓ_j clients, then we assume that the *response time*, or *latency*, to each client i connected to this server is $\lambda_i = f(\ell_j)/\sigma_j$, where f is an increasing function of the load ℓ_j , and σ_j is speed of server j . In general, there are two distinct contributors to a client’s latency: the server load, and the network congestion. In this paper, we focus on the latency at the server, and treat the network latency to be a constant. Because the network topology as well as the IP route change continuously, the network latency is both unpredictable and difficult to model. One can incorporate a simplified network latency in our model by folding it in the server speed—then a server with a fast connection to the network can be distinguished from a similar server with a slow link. A more accurate modeling of the combined server and network latency seems challenging and is left for future work.

The cost of an assignment M is the total latency of all the clients:

$$\text{cost}(M) = \sum_{i=1}^n \lambda_i = \sum_{j=1}^m \frac{\ell_j f(\ell_j)}{\sigma_j}.$$

Under the *linear model*, the latency of server j is simply ℓ_j/σ_j . More generally, we consider latency functions of the form ℓ_j^{p-1}/σ_j , for any $p \geq 1$. In that case, the cost of the matching is $\text{cost}(M) = \sum_{j=1}^m \ell_j^p/\sigma_j$, which corresponds to the p -th power of the weighted L_p norm of the server loads.

A *Nash equilibrium* assignment is one in which no client can improve its latency by unilaterally switching to another server. Let M_{nash} be a Nash solution, and let M_{opt} be the (coordinated) social optimum. The *price of anarchy* is the worst-case bound on the ratio between the costs of M_{nash} and M_{opt} .

1.2 Results

Our first result concerns the price of anarchy with *linear* latency functions. We show that if the server speeds are relatively bounded and the number of clients is large compared to the number of servers, then *every Nash assignment approaches social optimum*. Without any assumptions on the number of clients, servers, and server speeds, the price of anarchy is at most 2.5. If all servers have the same speed, then we can improve the upper bound to $(1 + 2/\sqrt{3}) \approx 2.15$. We also give a lower bound construction showing that the price of anarchy can be at least 2.01, even with equal speed servers and linear latency functions.

We next consider higher order monomial latency functions. In this case, we measure the L_p norm of the server loads, and show that the price of anarchy is $(p/\log p)(1 + o(1))$.

We then apply our technique to reanalyze a simple but centralized greedy scheme for load balancing. The best result known for this problem is due to Awerbuch et al. [3], who show that the greedy achieves the competitive ratio $(1 + \sqrt{2})^2 \approx 5.83$ for the *squared L_2 norm of the server loads* even for the unrelated machine model. We show that if all jobs have the same size and the servers have arbitrary speeds in the related machine model, then the competitive ratio of the greedy is at most $17/3 \approx 5.67$; more significantly, if all the servers have the same speed, then we can improve the competitive ratio to $2 + \sqrt{5} \approx 4.24$.

1.3 Related Work

Load balancing. The problem of assigning clients (jobs) to servers (machines) dates back to the earliest days of distributed computing or scheduling, and there is an enormous literature on it. A small sample of these results includes the following: [12, 18, 20] investigate the online assignment of unit length jobs under the L_∞ norm; [1, 14] consider offline assignments of unit length jobs; [2, 6, 8] consider greedy assignment of weighted jobs under the L_p norm, where the client-server graph is complete bipartite; [16] considers dynamic load balancing under the L_p norm. The work most relevant to us is the L_2 norm load balancing with an arbitrary client-server graph [3]. Because the total latency of the clients is related to the squared L_2 norm of the server loads, the setting of Awerbuch et al. [3] can be viewed as the *coordinated* or *centralized* version of our problem.

Congestion games. The load balancing game belongs to the general class of *congestion games* introduced by Rosenthal [19] in game theory. In these games, a set of players compete for a set of resources, and the cost of each resource depends *only* on the number of players using it. A key game-theoretic property of these games is that they always have at least one *pure strategy* Nash equilibrium. Thus, in our work, we focus on pure strategy Nash equilibria.

Selfish routing. In network routing games, selfish agents route their traffic between a source-destination pair in a network, where the network delay (latency) on each link is determined by a monotone increasing function of the load through the link. The models considered in the literature differ in many aspects.

The price of anarchy for the *maximal latency* was considered by Koutsoupias and Papadimitriou [13]. In their model, the network consists of m parallel edges between source and destination, and every agent can use any of the links. In [9], Czumaj and Vöcking were able to prove tight bounds for this problem. Recently, Awerbuch et al. [5] and Gairing et al. [11] have extended these results to a model where each agent can use only a subset of the (parallel) links, namely, a *permissible* set, and Fotakis et al. [10] have showed that the same bound holds in layered networks as well.

The price of anarchy for the *total flow latency* was considered by Roughgarden and Tardos [23] and Roughgarden [21] in *general networks*. They obtained tight bounds assuming *non-atomic* agents, where actions of an individual agent has negligible impact on others. Roughgarden [22] extends these results to an atomic splittable model, where agents control a positive fraction of the total traffic but each user's task (flow) can be split arbitrarily across multiple paths. Our load balancing game, by contrast, is characterized

by *atomic* players and *unsplittable* jobs: There are finitely many clients, each of whom is wholly assigned to a single server.

Similarly to our model, Lücking et al. [15] studied the price of anarchy for total flow latency with unsplittable flows. However, they consider only the special case where all servers are permissible for every user. In their model, they show that the price of anarchy is less than 2, while we present a lower bound of 2.01 in the general case with arbitrary permissible sets.

Since the conference publication of our work, Awerbuch et al. [4] and, independently, Christodoulou and Koutsoupias [7] have generalized our results to broader classes of congestion games. Awerbuch et al. [4] obtained a tight bound of 2.5 on the price of anarchy for routing games in general networks with linear latency functions. Christodoulou and Koutsoupias [7] discovered that the price of anarchy under linear latency functions is at most 2.5 for certain congestion games where each player’s strategy is a subset of the resources instead of a singleton set.

1.4 Organization

Our paper is organized as following. In Section 2, we establish two key results (Nash Condition and Nash Inequality), which are central to our analysis. In Section 3, we prove upper bounds on the price of anarchy with linear latency functions, and also show a lower bound construction. In Section 4, we extend our analysis to the latency functions under the L_p norm. In Section 5, we present our improved analysis of the greedy assignment scheme. Finally, we offer some conclusions and open problems in Section 6.

2 Preliminaries

Our primary model is the *linear latency model*: if a server has load ℓ and speed σ then *each* of its ℓ clients experiences latency $\lambda = \ell/\sigma$. If server j has load ℓ_j and speed σ_j in an assignment M , then the assignment has cost $\sum_{j=1}^m \ell_j^2/\sigma_j$, which is the weighted sum of the squares of server loads. We will consider higher order monomial latency functions in Section 4.

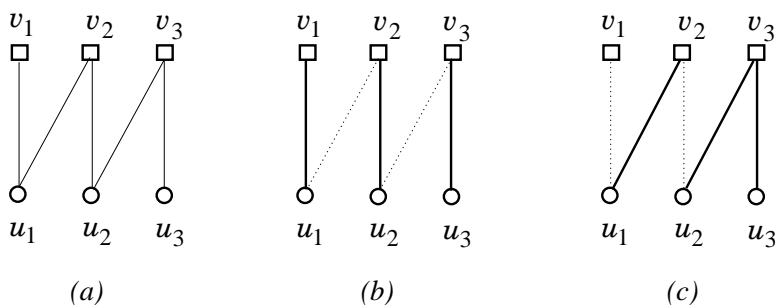


Figure 1: An instance (a), an optimal matching of cost 3 (b), and a Nash but not optimal matching of cost 5 (c), assuming unit speed machines.

An assignment is a *Nash equilibrium* if no single client can improve its latency by unilaterally switching to another (permissible) server. Given an instance of the client-server problem, let M_{opt} denote an assignment realizing the social optimum, and let M_{nash} denote a Nash assignment. (See a small example in Figure 1.) For server j , let O_j and N_j denote the *set of clients* assigned to j in M_{opt} and M_{nash} , respectively. We use the shorthand notation $o_j = |O_j|$ and $n_j = |N_j|$ for the cardinalities of these sets. The following lemma notes a simple but crucial condition imposed by a Nash equilibrium.

Lemma 2.1 (Nash Condition) *Given an optimal assignment M_{opt} and a Nash assignment M_{nash} , the following inequality holds for any two servers j, k , where σ_j, σ_k are the speeds of these servers:*

$$\frac{n_j}{\sigma_j} \leq \frac{n_k + 1}{\sigma_k} \quad \text{if} \quad N_j \cap O_k \neq \emptyset. \quad (1)$$

PROOF. If $k = j$ or $N_j \cap O_k = \emptyset$, then the argument is trivially true, so assume that $k \neq j$ and $N_j \cap O_k \neq \emptyset$. Pick an arbitrary client $i \in N_j \cap O_k$. This client has latency n_j/σ_j in the Nash assignment M_{nash} . The server k is also permissible for i because $i \in O_k$. By switching to k , the client i can achieve latency $(n_k + 1)/\sigma_k$. By the equilibrium property, this latency is the same or worse than its latency in M_{nash} . Thus, $n_j/\sigma_j \leq (n_k + 1)/\sigma_k$. \square

This simple Nash Condition leads to the following important inequality, which is the main basis for our upper bound analysis. (Later, we will prove a similar inequality for the greedy assignments.)

Lemma 2.2 (Nash Inequality) *Given an optimal assignment M_{opt} and a Nash assignment M_{nash} , the following inequality holds:*

$$\sum_{j=1}^m \frac{n_j^2}{\sigma_j} \leq \sum_{j=1}^m \frac{(n_j + 1)o_j}{\sigma_j}. \quad (2)$$

PROOF. Since the sets O_k , $k = 1, 2, \dots, m$, partition the set of clients, we have that $n_j = |N_j| = \sum_{k=1}^m |N_j \cap O_k|$, for every j . Similarly, we have $o_k = |O_k| = \sum_{j=1}^m |N_j \cap O_k|$, for every k . Using these equalities, we can rewrite the total cost of a Nash assignment as follows:

$$\begin{aligned} \sum_{j=1}^m \frac{n_j^2}{\sigma_j} &= \sum_{j=1}^m \frac{n_j}{\sigma_j} \sum_{k=1}^m |N_j \cap O_k| = \sum_{k=1}^m \sum_{j=1}^m \frac{n_j}{\sigma_j} |N_j \cap O_k| \\ &\leq \sum_{k=1}^m \sum_{j=1}^m \frac{n_k + 1}{\sigma_k} |N_j \cap O_k| = \sum_{k=1}^m \frac{n_k + 1}{\sigma_k} \sum_{j=1}^m |N_j \cap O_k| \\ &= \sum_{k=1}^m \frac{(n_k + 1)o_k}{\sigma_k}. \end{aligned}$$

In this chain of inequalities, we used the fact $(n_j/\sigma_j)|N_j \cap O_k| \leq ((n_k + 1)/\sigma_k)|N_j \cap O_k|$ for all j, k . This is trivially true if $|N_j \cap O_k| = 0$; otherwise, it follows from Inequality (1). \square

Finally we present a technical lemma which will be used later in Section 3 and Section 5.

Lemma 2.3 *Let n, m be positive integers and $x_j \geq 0, \sigma_j > 0$ for $j = 1, \dots, m$ be arbitrary real values with $\sum_{j=1}^m x_j = n$. Then the following inequality holds:*

$$\frac{\sum_{j=1}^m \frac{x_j}{\sigma_j}}{\sum_{j=1}^m \frac{x_j^2}{\sigma_j}} \leq \frac{m + \sqrt{\sum_{j=1}^m \sigma_j \sum_{j=1}^m \frac{1}{\sigma_j}}}{2n} \leq \left(1 + \sqrt{\max_{1 \leq j, k \leq m} \frac{\sigma_j}{\sigma_k}}\right) \frac{m}{2n}. \quad (3)$$

PROOF. Let

$$f(x_1, \dots, x_m) \equiv \frac{\sum_{j=1}^m \frac{x_j}{\sigma_j}}{\sum_{j=1}^m \frac{x_j^2}{\sigma_j}} \quad \text{and} \quad c(n, m, \sigma) \equiv \frac{m + \sqrt{\sum_{j=1}^m \sigma_j \sum_{j=1}^m \frac{1}{\sigma_j}}}{2n}.$$

It is sufficient to prove $f(x_1, \dots, x_m) \leq c(n, m, \sigma)$ because it immediately implies (3). In the following we use the Lagrange multiplier method to prove that the maximum value of $f(x_1, \dots, x_m)$ is exactly $c(n, m, \sigma)$ under the condition that $\sum_{j=1}^m x_j = n$ and $x_j \geq 0, j = 1, 2, \dots, m$.

Suppose that the maximum of $f(x_1, \dots, x_m)$ is attained at $x^* = (x_1^*, \dots, x_m^*)$ where there are m' coordinates $x_i^* > 0$. In the following we will focus on the case where $m' = m$ as $c(n, m', \sigma)$ is maximized when $m' = m$. In other words, we can assume that $x_j^* > 0$ for $j = 1, \dots, m$. By the Lagrange multiplier condition, there exists a multiplier λ such that

$$\frac{\partial f(x_1^*, \dots, x_m^*)}{\partial x_j} = \lambda, \quad \forall j = 1, \dots, m.$$

Let $A(x^*) \equiv \sum_{j=1}^m (x_j^*)^2 / \sigma_j$ and $B(x^*) \equiv \sum_{j=1}^m x_j^* / \sigma_j$. The above equation can be simplified to

$$A(x^*) - 2B(x^*)x_j^* = \lambda A^2(x^*)\sigma_j, \quad \forall j = 1, \dots, m. \quad (4)$$

summing up these equations for $j = 1, 2, \dots, m$, we obtain

$$\lambda = \frac{mA(x^*) - 2nB(x^*)}{A^2(x^*) \sum_j \sigma_j}.$$

Substituting the value of λ into each equation (4), we have

$$x_j^* = \frac{A(x^*)}{2B(x^*)} - \sigma_j \left(\frac{mA(x^*)}{2B(x^*) \sum_j \sigma_j} - \frac{n}{\sum_j \sigma_j} \right), \quad \forall j = 1, \dots, m.$$

By plugging in the values of x_j^* for all j into $A(x^*)$ and $B(x^*)$, we get

$$\begin{aligned} A(x^*) &= \sum_{j=1}^m \frac{(x_j^*)^2}{\sigma_j} = \frac{A^2(x^*)}{4B^2(x^*)} \left(\sum_j \frac{1}{\sigma_j} - \frac{m^2}{\sum_j \sigma_j} \right) + \frac{n^2}{\sum_j \sigma_j}, \\ B(x^*) &= \sum_{j=1}^m \frac{x_j^*}{\sigma_j} = \frac{A(x^*)}{2B(x^*)} \left(\sum_j \frac{1}{\sigma_j} - \frac{m^2}{\sum_j \sigma_j} \right) + \frac{nm}{\sum_j \sigma_j}. \end{aligned}$$

Let $z \equiv B(x^*)/A(x^*)$, and we get

$$z = \frac{\frac{1}{2z} \left(\sum_j \frac{1}{\sigma_j} \sum_j \sigma_j - m^2 \right) + nm}{\frac{1}{4z^2} \left(\sum_j \frac{1}{\sigma_j} \sum_j \sigma_j - m^2 \right) + n^2}, \quad \text{i.e.,} \quad n^2 z^2 - nmz - \frac{1}{4} \left(\sum_j \frac{1}{\sigma_j} \sum_j \sigma_j - m^2 \right) = 0.$$

We solve z from the above equation and obtain $z = c(n, m, \sigma)$. This completes the proof of Lemma 2.3. \square

3 Bounds on the Price of Anarchy

In this section, we prove upper and lower bounds on the price of anarchy. We prove three different upper bounds. The first upper bound (Theorem 3.1) is in terms of n (number of clients), m (number of servers), and the server speeds. This bound is the sharpest in the limit when $m/n \rightarrow 0$ and the ratio between the maximum and the minimum server speeds is bounded. For instance, if all servers have equal speed and m/n approaches 0, then Theorem 3.1 says that *every Nash approaches the social optimum*. For arbitrary

values of m, n and server speeds, our second bound (Theorem 3.2) is the best. It shows that the price of anarchy is at most 2.5 for any choice of n, m and server speeds. Finally, if all servers have the same speed, then Theorem 3.3 shows that the price of anarchy is at most $(1 + 2/\sqrt{3}) \approx 2.15$. We also give a lower bound construction showing that the price of anarchy is at least 2.01, even with equal speed servers. (This lower bound dashed our original hope that the true price of anarchy was 2.)

Theorem 3.1 *With linear latency functions, the price of anarchy is $1 + o(1)$ given that $n \gg m$ and server speeds are relatively bounded. Formally, the following is true:*

$$\frac{\text{cost}(M_{\text{nash}})}{\text{cost}(M_{\text{opt}})} \leq 1 + \frac{m + \sqrt{\sum_{j=1}^m \sigma_j \sum_{j=1}^m \frac{1}{\sigma_j}}}{n} \leq 1 + \left(1 + \sqrt{\max_{1 \leq j, k \leq m} \frac{\sigma_j}{\sigma_k}}\right) \frac{m}{n}. \quad (5)$$

PROOF. By using Nash inequality (2) and the fact that $n_j o_j \leq (n_j^2 + o_j^2)/2$, we get

$$\begin{aligned} \sum_{j=1}^m \frac{n_j^2}{\sigma_j} &\leq \sum_{j=1}^m \frac{(n_j + 1)o_j}{\sigma_j} \leq \sum_{j=1}^m \frac{1}{\sigma_j} \left(\frac{n_j^2 + o_j^2}{2} + o_j \right) \\ \Rightarrow \sum_{j=1}^m \frac{n_j^2}{\sigma_j} &\leq \sum_{j=1}^m \frac{o_j^2 + 2o_j}{\sigma_j} \\ \Rightarrow \frac{\text{cost}(M_{\text{nash}})}{\text{cost}(M_{\text{opt}})} &= \frac{\sum_{j=1}^m \frac{n_j^2}{\sigma_j}}{\sum_{j=1}^m \frac{o_j^2}{\sigma_j}} \leq 1 + 2 \frac{\sum_{j=1}^m \frac{o_j}{\sigma_j}}{\sum_{j=1}^m \frac{o_j^2}{\sigma_j}}. \end{aligned}$$

By Lemma 2.3, we get the desired result immediately. \square

Next, we give an upper bound of 2.5 independent of n, m and server speeds.

Theorem 3.2 *With linear latency functions and arbitrary values of n, m and server speeds, the price of anarchy is $\text{cost}(M_{\text{nash}})/\text{cost}(M_{\text{opt}}) \leq 2.5$.*

PROOF. Suppose that $M_{\text{opt}} = \{O_j \mid 1 \leq j \leq m\}$ is an optimal assignment and $M_{\text{nash}} = \{N_j \mid 1 \leq j \leq m\}$ is a Nash assignment. Fix an index j , it is straightforward to verify the following equality:

$$o_j n_j = \frac{1}{3} n_j^2 + \frac{3}{4} o_j^2 - \frac{1}{3} \left(n_j - \frac{3}{2} o_j \right)^2.$$

The Nash Inequality (2) together with the above inequality implies the following:

$$\begin{aligned} \sum_{j=1}^m \frac{n_j^2}{\sigma_j} &\leq \sum_{j=1}^m \frac{n_j o_j + o_j}{\sigma_j} = \sum_{j=1}^m \frac{1}{\sigma_j} \left(\frac{1}{3} n_j^2 + \frac{3}{4} o_j^2 - \frac{1}{3} (n_j - \frac{3}{2} o_j)^2 + o_j \right) \\ \Rightarrow \sum_{j=1}^m \frac{n_j^2}{\sigma_j} &\leq \sum_{j=1}^m \frac{1}{\sigma_j} \left(\frac{9}{8} o_j^2 + \frac{3}{2} o_j - \frac{1}{2} (n_j - \frac{3}{2} o_j)^2 \right). \end{aligned}$$

Thus, in order to show

$$\sum_{j=1}^m \frac{n_j^2}{\sigma_j} \leq \frac{5}{2} \sum_{j=1}^m \frac{o_j^2}{\sigma_j},$$

it suffices to prove that

$$\frac{9}{8}o_j^2 + \frac{3}{2}o_j - \frac{1}{2}\left(n_j - \frac{3}{2}o_j\right)^2 \leq \frac{5}{2}o_j^2, \quad \forall 1 \leq j \leq m.$$

The preceding inequality is equivalent to the following simplified form

$$o_j \left(3 - \frac{11}{4}o_j\right) \leq \left(n_j - \frac{3}{2}o_j\right)^2.$$

It holds trivially if either $o_j = 0$ or $o_j \geq 2$. Because o_j is an integer, thus the only remaining case is $o_j = 1$, and in this case the above inequality is equivalent to $1/4 \leq (n_j - 3/2)^2$. Because n_j is an integer, thus this holds, and the whole proof is complete. \square

3.1 An Improved Upper Bound for Equal Speed Servers

In this section, we show a further improvement in the upper bound when all servers have the same speed. The upper bound in this case turns out to be $1 + 2/\sqrt{3} \approx 2.15$, which is getting quite close to the lower bound of 2.01, shown in the next subsection. Without loss of generality, we assume that all servers have the unit speed. Thus, $\text{cost}(M_{\text{nash}}) = \sum_{j=1}^m n_j^2$ and $\text{cost}(M_{\text{opt}}) = \sum_{j=1}^m o_j^2$.

Theorem 3.3 *If all servers have equal speed and the latency function is linear, then the price of anarchy is at most $\text{cost}(M_{\text{nash}})/\text{cost}(M_{\text{opt}}) \leq 2/\sqrt{3} + 1 \approx 2.15$.*

PROOF. In order to prove the upper bound $\text{cost}(M_{\text{nash}}) \leq (2/\sqrt{3} + 1) \text{cost}(M_{\text{opt}})$, it is enough to prove the following:

$$\frac{2}{\sqrt{3}} \text{cost}(M_{\text{nash}}) \leq \left(\frac{2}{\sqrt{3}} - 1\right) \text{cost}(M_{\text{nash}}) + \left(\frac{2}{\sqrt{3}} + 1\right) \text{cost}(M_{\text{opt}}) \quad (6)$$

By Lemma 2.2 together with $\sigma_j = 1$ and $\sum_j o_j = n$, we obtain that $\text{cost}(M_{\text{nash}}) \leq n + \sum_{j=1}^m n_j o_j$. To prove Eq. (6), it is thus sufficient to show that

$$\frac{2}{\sqrt{3}} \left(n + \sum_{j=1}^m n_j o_j\right) \leq \sum_{j=1}^m \left(\left(\frac{2}{\sqrt{3}} + 1\right) o_j^2 + \left(\frac{2}{\sqrt{3}} - 1\right) n_j^2\right),$$

and it is equivalent to

$$n \leq \sum_{j=1}^m \left(\left(1 + \frac{\sqrt{3}}{2}\right) o_j^2 + \left(1 - \frac{\sqrt{3}}{2}\right) n_j^2 - o_j n_j\right) = \sum_{j=1}^m \left(\frac{\sqrt{3}+1}{2} o_j - \frac{\sqrt{3}-1}{2} n_j\right)^2. \quad (7)$$

Let $x_j = \frac{\sqrt{3}+1}{2} o_j - \frac{\sqrt{3}-1}{2} n_j$, then (7) is equivalent to the following:

$$n \leq \sum_{j=1}^m x_j^2, \quad \text{with} \quad \sum_{j=1}^m x_j = \frac{\sqrt{3}+1}{2} \sum_{j=1}^m o_j - \frac{\sqrt{3}-1}{2} \sum_{j=1}^m n_j = n. \quad (8)$$

We prove Eq. (8) by induction on m , which is also the number of pairs (o_j, n_j) . The base case, $m = 1$ is trivially true as $x_1 = n$ and $n^2 \geq n$ because n is an integer. It is also easy to verify for the case where $m = 2$ and there are two pairs $(a, 0)$ and $(0, a)$. Now assume that $m > 1$ and Eq. (8) holds for any $m' < m$.

If there is no j such that $o_j = 0$, then $n = \sum_{j=1}^m o_j \geq m$, and thus

$$\sum_{j=1}^m x_j^2 \geq \frac{(\sum_{j=1}^m x_j)^2}{m} = \frac{n^2}{m} \geq n.$$

Similarly the above inequality holds if there is no j such that $n_j = 0$. Now suppose that there is one pair $(o_j, n_j) = (a, 0)$ and another pair $(o_k, n_k) = (0, b)$. Without loss of generality, we assume that $a \geq b \geq 1$. If $a = b$, then we break the original problem into two subproblems, where the smaller one contains only two pairs $(a, 0)$ and $(0, a)$. The larger subproblem has $m - 2$ pairs. By induction, the inequality holds for both subproblems. If $a > b$, then $\sum_{j=1}^m x_j^2$ decreases if we break $(a, 0)$ into two pairs $(a - b, 0)$ and $(b, 0)$. Now we remove two pairs $(b, 0)$ and $(0, b)$ to get a subproblem with $m - 1$ pairs. So that by induction Eq. (8) holds. This also completes the whole proof. \square

The proof of Theorem 3.3 uses solely the Nash Inequality (2) and $\sum_{j=1}^m o_j = \sum_{j=1}^m n_j = n$. For assignments satisfying these constraints (but not necessarily Nash), the bound of Theorem 3.3 is the best possible. The following set of pairs (o_j, n_j) , for $j = 1, 2, \dots, m$, attains this bound (the values n_j , however, do not correspond to a Nash solution): Let $n = a + b + k$ and $m = a + k + 1$ and consider a pairs of $(1, 0)$, k pairs of $(1, 1)$, and one pair of $(b, a + b)$. Set $k = (a - 1)(a + b)$, then $n = a(a + b)$ and $\sum_j n_j^2 = \sum_j o_j n_j + n$. Now

$$\frac{\sum_j n_j^2}{\sum_j o_j^2} = \frac{k + (a + b)^2}{a + k + b^2} = \frac{2a^2 + 3ab + b^2 - a - b}{a^2 + ab + b^2 - b}.$$

Let b/a approximate $(\sqrt{3} - 1)/2$. As a goes to infinity, the above ratio approaches $1 + 2/\sqrt{3}$.

3.2 A Lower Bound

We now describe a construction showing that the worst-case price of anarchy is at least 2.01. Our lower bound holds even if all servers have equal speed.

Theorem 3.4 *In the worst case, the following lower bound holds for the price of anarchy:*

$$\frac{\text{cost}(M_{\text{nash}})}{\text{cost}(M_{\text{opt}})} > 2.01.$$

PROOF. We first describe a general family of instances of the load balancing game with a parameter $k \in \mathbb{N}$, $k \geq 2$. We then present one specific instance in this family for $k = 7$ which provides a lower bound of 2.01.

Consider an integer $k \in \mathbb{N}$, $k \geq 2$, and let S_k be the set of all monotone increasing sequences $s = (o_0, o_1, \dots, o_k) \in \mathbb{N}^{k+1}$, where $1 \leq o_0$, $o_k < k$, and $o_{i-1} \leq o_i \leq o_{i-1} + 1$, for $i = 1, 2, \dots, k$. We construct a bipartite graph $G = (U, V)$ for every sequence $s \in S_k$. A simple example with $k = 3$ and $(o_0, o_1, o_2, o_3) = (1, 1, 1, 1)$ is depicted in Figure 2. Both V and U are partitioned into $k + 1$ groups $V = V_0 \cup V_1 \cup \dots \cup V_k$ and $U = U_0 \cup U_1 \cup \dots \cup U_k$. The permissible set of every $u \in U_i$ is $V_i \cup V_{i+1}$ for $i = 0, 1, \dots, k - 1$, and V_k for $i = k$. We choose the cardinalities of U_i and V_i , $i = 0, 1, \dots, k - 1$, as follows. Let $x_s \in \mathbb{N}$ be an integer specified below. We set $|U_k| = x_s \cdot o_k$ and $|U_{k-1}| = x_s(k - o_k)$. The remaining cardinalities are determined by two recursion formulas: Let $|V_i| = |U_i|/o_i$ for $i = k, \dots, 1, 0$; and let $|U_i| = (i + 1) \cdot |V_{i+1}| = \frac{i+1}{o_{i+1}} \cdot |U_{i+1}|$ for $i = k - 2, \dots, 1, 0$. The solution to this recursion gives $|V_k| = x_s$ and $|V_i| = x_s \frac{k - o_k}{o_{k-1}} \prod_{j=i}^{k-2} \frac{j+1}{o_{j+1}}$, for $i = 0, 1, \dots, k - 1$. We choose x_s to be the minimum positive real number such that every $|V_i|$ is an integer.

Since the sequence (o_0, o_1, \dots, o_k) is monotone increasing, an optimal solution M_{opt} assigns every client of U_i to a server in V_i , and every server of V_i has load o_i , for $i = 0, 1, \dots, k$. We next describe a Nash

solution M_{nash} : It assigns every job in U_i to a server in V_{i+1} for $i = 0, 1, \dots, k-1$; and it assigns every job in U_k to a server in V_k . Every server in V_i has load $n_i = i$ for $i = 0, 1, \dots, k$.*

For $k = 7$ and $(o_0, o_1, \dots, o_7) = (1, 1, 1, 1, 2, 2, 2, 2)$, we apply the above recursion formulas with the minimum integer $x = 4$. We obtain $(|V_0|, |V_1|, \dots, |V_7|) = (1800, 1800, 900, 300, 75, 30, 10, 4)$, a total of 4919 servers. The social costs in the two solutions are $\text{cost}(M_{\text{opt}}) = \sum_{i=0}^7 |V_i| \cdot o_i^2 = 5276$ and $\text{cost}(M_{\text{nash}}) = \sum_{i=0}^7 |V_i| \cdot i^2 = 10606$; so the price of anarchy is $2 + 27/2638 > 2.01$. \square

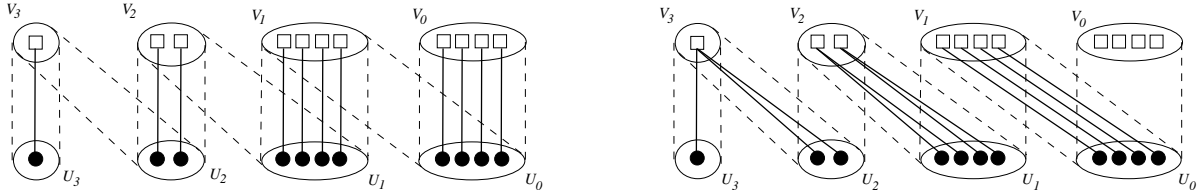


Figure 2: Our construction for $k = 3$ and $(o_0, o_1, o_2, o_3) = (1, 1, 1, 1)$: an optimal assignment (left), a worst case Nash assignment (right). Complete bipartite sub-graphs of G between groups are indicated by dashed lines.

We have presented a lower bound construction with $k = 7$ from a general family of instances. A natural question is how far can this construction be pushed? We have systematically tested all instances in this family up to $k = 29$. The construction first yields a lower bound strictly larger than 2 for $k = 6$. The price of anarchy then increases slowly and monotonically with k , reaching 2.01206694843168 when $k = 29$. With such a slow rate of growth, it seems unlikely to us that this construction will yield a significant improvement over 2.01. In fact, we conjecture that an upper bound of the ratio using this construction is 2.0121.

4 Price of Anarchy with the L_p Norm

In this section, we consider higher order monomial latency functions, and use L_p norm to measure the cost of an assignment. For any constant $p \geq 1$, we assume that a server with load ℓ and speed σ has latency ℓ^{p-1}/σ . Thus, each of the ℓ_j clients matched with server j incurs latency $\lambda_j = \ell_j^{p-1}/\sigma_j$. The L_p norm measure of the total latency is $(\sum_{i=1}^n \lambda_i)^{1/p} = (\sum_{j=1}^m \ell_j^p/\sigma_j)^{1/p}$. The case $p = 1$ is the extreme case where a server's latency is *independent* of its load—in such a case, all Nash equilibria are optimal. Thus, the interesting cases are only when $p > 1$. Our main result in this subsection shows that the price of anarchy with this latency measure is $(p/\log p)(1 + o(1))$. (By comparison, the greedy scheme of Awerbuch et al. [3], discussed in the next section, has competitive ratio $cp(1 + o(1))$ where $c \approx 1.77$.)

Theorem 4.1 *With the L_p norm latency measure and arbitrary server speeds, the price of anarchy is bounded by $\text{cost}(M_{\text{nash}})/\text{cost}(M_{\text{opt}}) \leq (p/\log p)(1 + o(1))$.*

PROOF. Let M_{nash} be a Nash assignment and M_{opt} be an optimal assignment, then $\text{cost}(M_{\text{nash}}) = (\sum_j n_j^p/\sigma_j)^{1/p}$ and $\text{cost}(M_{\text{opt}}) = (\sum_j o_j^p/\sigma_j)^{1/p}$. Suppose that there exists a constant c_p , for fixed $p > 1$, such that $\text{cost}(M_{\text{nash}}) \leq c_p \text{cost}(M_{\text{opt}})$, or equivalently,

$$\sum_{j=1}^m \frac{n_j^p}{\sigma_j} \leq (c_p)^p \sum_{j=1}^m \frac{o_j^p}{\sigma_j} \quad \Leftrightarrow \quad p \sum_{j=1}^m \frac{n_j^p}{\sigma_j} \leq (p-1) \sum_{j=1}^m \frac{n_j^p}{\sigma_j} + (c_p)^p \sum_{j=1}^m \frac{o_j^p}{\sigma_j}. \quad (9)$$

*In the simple case where $o_i = 1$ for all i 's, we have $x = 1$, and it is easy to verify that $\text{cost}(M_{\text{opt}}) = n$, and $\text{cost}(M_{\text{nash}}) = 2n - 1$. This relatively simple construction gives a lower bound of $2 - 1/n$, which is slightly below 2.

We have the following chain of inequalities:

$$\begin{aligned} \sum_{j=1}^m \frac{n_j^p}{\sigma_j} &= \sum_{j=1}^m \frac{n_j^{p-1}}{\sigma_j} \sum_{k=1}^m |N_j \cap O_k| = \sum_{k=1}^m \sum_{j=1}^m \frac{n_j^{p-1}}{\sigma_j} |N_j \cap O_k| \\ &\leq \sum_{k=1}^m \sum_{j=1}^m \frac{(n_k + 1)^{p-1}}{\sigma_k} |N_j \cap O_k| = \sum_{k=1}^m \frac{(n_k + 1)^{p-1} o_k}{\sigma_k}. \end{aligned} \quad (10)$$

In the above chain of inequalities, we used the fact that $(n_j^{p-1}/\sigma_j)|N_j \cap O_k| \leq ((n_k + 1)^{p-1}/\sigma_k)|N_j \cap O_k|$ for all j, k . This is trivial if $N_j \cap O_k = \emptyset$; otherwise, the derivation is similar to Nash Condition (1). Due to Eq (10), in order for Eq (9) to hold, it is sufficient to prove the following:

$$p \sum_{j=1}^m \frac{(n_j + 1)^{p-1} o_j}{\sigma_j} \leq (p-1) \sum_{j=1}^m \frac{n_j^p}{\sigma_j} + (c_p)^p \sum_{j=1}^m \frac{o_j^p}{\sigma_j}.$$

In fact, we prove the following much stronger inequality:

$$p(n_j + 1)^{p-1} o_j \leq (p-1)n_j^p + (c_p)^p o_j^p, \quad \forall 1 \leq j \leq m. \quad (11)$$

If $o_j = 0$, Eq. (11) is trivially true. Thus we assume that $o_j \geq 1$. We rewrite Eq. (11) as the following:

$$p \left(\frac{n_j}{o_j} + \frac{1}{o_j} \right)^{p-1} - (p-1) \left(\frac{n_j}{o_j} \right)^p \leq (c_p)^p. \quad (12)$$

Let $f(x) \equiv p(x+1)^{p-1} - (p-1)x^p$, $x \in [0, \infty)$. Since $1/o_j \leq 1$, it follows that the left side of Eq. (12) is bounded by $f(n_j/o_j)$. It is easy to see that $f(x)$ is bounded above. Let the maximum value of $f(x)$ be $f(x_0)$ where $x_0 \geq 0$, and set $(c_p)^p = f(x_0)$. Since x_0 is an extreme point, it follows that $f'(x_0) = 0$, which means that

$$p(p-1)(x_0+1)^{p-2} - p(p-1)(x_0)^{p-1} = 0, \quad \text{i.e.,} \quad (x_0+1)^{p-2} = (x_0)^{p-1}. \quad (13)$$

It is easy to derive from Eq. (13) the following bound on x_0 : $x_0 = \frac{p}{\log p}(1 + o(1))$. Now we can bound the value of c_p as follows:

$$\begin{aligned} c_p &= (p(x_0+1)^{p-1} - (p-1)(x_0)^p)^{1/p} = (p(x_0+1)(x_0)^{p-1} - (p-1)(x_0)^p)^{1/p} \\ &= x_0 \left(p \left(1 + \frac{1}{x_0} \right) - (p-1) \right)^{1/p} = x_0 \left(1 + \frac{p}{x_0} \right)^{1/p} \\ &\leq (1+p)^{1/p} x_0 = \frac{p}{\log p} (1 + o(1)). \end{aligned}$$

This completes the proof of Theorem 4.1. □

5 Analysis of Greedy

The cost of an assignment is related to the squared sum of the server loads. An elegant result of Awerbuch et al. [3] shows that a simple online greedy scheme achieves competitive ratio $(1 + \sqrt{2})^2 \approx 5.83$ for this measure of *centralized* load balancing. The greedy scheme assigns each client to a permissible server so as to *minimize the increase in the total objective*. Specifically, a client is assigned to server j that minimizes the quantity $\frac{(\ell_j+1)^2}{\sigma_j} - \frac{\ell_j^2}{\sigma_j} = \frac{2\ell_j+1}{\sigma_j}$.

The greedy scheme does not, in general, lead to equilibrium assignments. However, it does have a computational advantage—it is easy to implement. While the greedy policy above is designed for optimizing the social welfare, it is also a natural *selfish* strategy. Each client is essentially choosing the best possible server at the time it makes its selection. When all servers have equal speed, each client is simply choosing the server with minimum load. With arbitrary speeds, the greedy asks each client to choose the server j that minimizes $\frac{\ell_j+0.5}{\sigma_j}$; a true selfish strategy for the client would minimize $\frac{\ell_j+1}{\sigma_j}$. This minor change in the priority has a minuscule effect on our bounds.

In this section, we reanalyze the greedy scheme and present improved bounds on its competitive ratio. The basis for our analysis is the following Greedy Inequality:

Lemma 5.1 (Greedy Inequality) *If servers have arbitrary speeds and the latency function is linear, then the following holds for an optimal assignment M_{opt} and any greedy assignment M_{greedy} , where $o_j = |O_j|, g_j = |G_j|$, and O_j (resp. G_j) is the set of clients assigned to server j in M_{opt} (resp. M_{greedy}):*

$$\sum_{j=1}^m \frac{g_j^2}{\sigma_j} \leq \sum_{j=1}^m \frac{2g_j o_j + o_j}{\sigma_j}. \quad (14)$$

PROOF. Suppose the clients arrive in the order $1, 2, \dots, n$. Let $X_i = (x_{i1}, \dots, x_{im})$ denote the assignment vector for the client i in M_{greedy} , and let $Y_i = (y_{i1}, \dots, y_{im})$ denote the assignment vector for i in M_{opt} , where $x_{ij}, y_{ij} \in \{0, 1\}$ and $\sum_{j=1}^m x_{ij} = \sum_{j=1}^m y_{ij} = 1$. We use $L_i = (\ell_{i1}, \dots, \ell_{im})$ to denote the load vector for the greedy scheme after the first i clients have been assigned, for $i = 0, \dots, n$. Notice that $L_0 = (0, \dots, 0)$ and $L_n = (g_1, \dots, g_m)$. Since greedy minimizes the total increase in the latency at each step, we have

$$\sum_{j=1}^m \frac{\ell_{ij}^2}{\sigma_j} - \sum_{j=1}^m \frac{\ell_{i-1,j}^2}{\sigma_j} \leq \sum_{j=1}^m \frac{(\ell_{i-1,j} + y_{ij})^2}{\sigma_j} - \sum_{j=1}^m \frac{\ell_{i-1,j}^2}{\sigma_j} = \sum_{j=1}^m \frac{y_{ij}^2 + 2y_{ij}\ell_{i-1,j}}{\sigma_j}.$$

Summing up these increments for i from 1 to n , we get

$$\begin{aligned} \sum_{j=1}^m \frac{g_j^2}{\sigma_j} &\leq \sum_{i=1}^n \sum_{j=1}^m \frac{y_{ij}^2 + 2y_{ij}\ell_{i-1,j}}{\sigma_j} = \sum_{j=1}^m \sum_{i=1}^n \frac{y_{ij}(y_{ij} + 2\ell_{i-1,j})}{\sigma_j} \\ &\leq \sum_{j=1}^m \sum_{i=1}^n \frac{y_{ij}(1 + 2g_j)}{\sigma_j} = \sum_{j=1}^m \frac{(2g_j + 1)o_j}{\sigma_j}. \end{aligned}$$

In the above chain of inequalities, we used the fact that $\ell_{i-1,j} \leq g_j$ and $y_{ij} \leq 1$ for all i, j , $\sum_i y_{ij} = o_j$ for all j . The proof is complete. \square

With the Greedy Inequality, we can prove the following theorem.

Theorem 5.2 *With linear latency functions, the ratio of a greedy solution to the optimal solution is at most $4 + o(1)$ assuming that $n \gg m$ and the server speeds are relatively bounded. Formally, we have the following:*

$$\frac{\text{cost}(M_{\text{greedy}})}{\text{cost}(M_{\text{opt}})} \leq 4 + \frac{m + \sqrt{\sum_{j=1}^m \sigma_j \sum_{j=1}^m \frac{1}{\sigma_j}}}{n} \leq 4 + \left(1 + \sqrt{\max_{1 \leq j, k \leq m} \frac{\sigma_j}{\sigma_k}}\right) \frac{m}{n}. \quad (15)$$

PROOF. By using the Greedy Inequality (14) and the fact that $2g_j o_j \leq (1/2)g_j^2 + 2o_j^2$, we get

$$\begin{aligned} \sum_{j=1}^m \frac{g_j^2}{\sigma_j} &\leq \sum_{j=1}^m \frac{2g_j o_j + o_j}{\sigma_j} \leq \sum_{j=1}^m \frac{1}{\sigma_j} \left(\frac{g_j^2}{2} + 2o_j^2 + o_j \right) \\ \Rightarrow \sum_{j=1}^m \frac{g_j^2}{\sigma_j} &\leq \sum_{j=1}^m \frac{4o_j^2 + 2o_j}{\sigma_j} \\ \Rightarrow \frac{\text{cost}(M_{\text{greedy}})}{\text{cost}(M_{\text{opt}})} &= \frac{\sum_{j=1}^m \frac{g_j^2}{\sigma_j}}{\sum_{j=1}^m \frac{o_j^2}{\sigma_j}} \leq 4 + 2 \frac{\sum_{j=1}^m \frac{o_j}{\sigma_j}}{\sum_{j=1}^m \frac{o_j^2}{\sigma_j}}. \end{aligned}$$

By Lemma 2.3, we get the desired result immediately. \square

For arbitrary values of n, m and arbitrary server speeds, our second theorem gives an upper bound of $17/3 \approx 5.67$, which is a slight improvement over the $(\sqrt{2} + 1)^2 \approx 5.83$ bound proved in [3].

Theorem 5.3 *If servers have arbitrary speeds and the latency function is linear, then the following bound holds: $\text{cost}(M_{\text{greedy}})/\text{cost}(M_{\text{opt}}) \leq 17/3$. Formally, we have*

$$\sum_{j=1}^m \frac{g_j^2}{\sigma_j} \leq \frac{17}{3} \sum_{j=1}^m \frac{o_j^2}{\sigma_j}.$$

PROOF. It is easy to verify the following equality:

$$2g_j o_j = \frac{2}{5}g_j^2 + \frac{5}{2}o_j^2 - \frac{2}{5} \left(g_j - \frac{5}{2}o_j \right)^2.$$

From Lemma 5.1, we know that

$$\begin{aligned} \sum_{j=1}^m \frac{g_j^2}{\sigma_j} &\leq \sum_{j=1}^m \frac{2g_j o_j + o_j}{\sigma_j} = \sum_{j=1}^m \frac{1}{\sigma_j} \left(\frac{2}{5}g_j^2 + \frac{5}{2}o_j^2 - \frac{2}{5} \left(g_j - \frac{5}{2}o_j \right)^2 + o_j \right) \\ \Rightarrow \sum_{j=1}^m \frac{g_j^2}{\sigma_j} &\leq \sum_{j=1}^m \frac{1}{\sigma_j} \left(\frac{25}{6}o_j^2 + \frac{5}{3}o_j - \frac{2}{3} \left(g_j - \frac{5}{2}o_j \right)^2 \right). \end{aligned}$$

In order to prove $\sum_{j=1}^m g_j^2/\sigma_j \leq (17/3) \sum_{j=1}^m o_j^2/\sigma_j$, it is sufficient to show that

$$\frac{25}{6}o_j^2 + \frac{5}{3}o_j - \frac{2}{3} \left(g_j - \frac{5}{2}o_j \right)^2 \leq \frac{17}{3}o_j^2, \text{ for all } j = 1, \dots, m.$$

The above inequality can be simplified to the following:

$$\frac{9}{4}o_j \left(\frac{10}{9} - o_j \right) \leq \left(g_j - \frac{5}{2}o_j \right)^2.$$

This inequality is obviously true if $o_j = 0$ or $o_j \geq 2$. Since o_j is an integer, there is only one remaining case $o_j = 1$, where the inequality simplifies to $1/4 \leq (g_j - 5/2)^2$. This holds because g_j is an integer. This completes the proof. \square

5.1 An Improved Bound for Servers of Equal Speed

If all servers have the same speeds then we can further improve the upper bound for the greedy. First, theorem 5.2 implies that $\text{cost}(M_{\text{greedy}})/\text{cost}(M_{\text{opt}}) \leq 4 + 2m/n$. Thus, the competitive ratio of the greedy approaches 4 as $m/n \rightarrow 0$; in practice, this is not a bad assumption since the number of clients often far exceeds the number of servers. A more complicated analysis gives an upper bound of $2 + \sqrt{5} \approx 4.24$ for arbitrary m, n .

Theorem 5.4 *If all servers have the same speed and the latency function is linear, then the ratio $\text{cost}(M_{\text{greedy}})/\text{cost}(M_{\text{opt}})$ is at most $2 + \sqrt{5}$.*

PROOF. In order to prove the inequality $\text{cost}(M_{\text{greedy}}) \leq (2 + \sqrt{5})\text{cost}(M_{\text{opt}})$, it is enough to prove the following:

$$\frac{\sqrt{5} + 1}{2} \text{cost}(M_{\text{greedy}}) \leq \frac{\sqrt{5} - 1}{2} \text{cost}(M_{\text{greedy}}) + (2 + \sqrt{5})\text{cost}(M_{\text{opt}}). \quad (16)$$

Lemma 5.1, together with the fact $\sigma_j = 1$ and $\sum_j o_j = n$, implies that $\text{cost}(M_{\text{greedy}}) \leq n + 2 \sum_{j=1}^m o_j g_j$. To prove Eq. (16), it suffices to show the following:

$$\frac{\sqrt{5} + 1}{2} \left(n + 2 \sum_{j=1}^m o_j g_j \right) \leq \sum_{j=1}^m \left((2 + \sqrt{5})o_j^2 + \frac{\sqrt{5} - 1}{2}g_j^2 \right), \quad (17)$$

and it is equivalent to

$$n \leq \sum_{j=1}^m \left(\frac{3 + \sqrt{5}}{2}o_j^2 + \frac{3 - \sqrt{5}}{2}g_j^2 - 2o_j g_j \right) = \sum_{j=1}^m \left(\frac{\sqrt{5} + 1}{2}o_j - \frac{\sqrt{5} - 1}{2}g_j \right)^2. \quad (18)$$

Let $x_j = \frac{\sqrt{5} + 1}{2}o_j - \frac{\sqrt{5} - 1}{2}g_j$, then Eq. (18) is equivalent to the following:

$$n \leq \sum_{j=1}^m x_j^2, \quad \text{with} \quad \sum_{j=1}^m x_j = \frac{\sqrt{5} + 1}{2} \sum_{j=1}^m o_j - \frac{\sqrt{5} - 1}{2} \sum_{j=1}^m g_j = n. \quad (19)$$

We prove Eq. (19) by induction on m . The inductive proof is similar to the proof of Eq. (8) in Theorem 3.3 and we omit the details. This completes the proof. \square

5.2 Some lower bounds

The bound of Theorem 5.4 is the best possible using only the Greedy Inequality. Consider the following example: There are a pairs of $(1, 0)$, k pairs of $(1, 1)$ and one pair of $(b, a + b)$. Let $k = (a - b - 1)(a + b)/2$, then $\sum_{j=1}^m g_j^2 = 2 \sum_{j=1}^m o_j g_j + n$ with $n = a + b + k$. Now

$$\frac{\sum_{j=1}^m g_j^2}{\sum_{j=1}^m o_j^2} = \frac{k + (a + b)^2}{a + b^2 + k} = \frac{3a^2 + b^2 + 4ab - (a + b)}{a^2 + b^2 + a - b}.$$

Let b/a approximate $(\sqrt{5} - 1)/2$. As a goes to infinity, the above ratio approaches $2 + \sqrt{5}$.

The following theorem establishes that the cost of greedy is strictly more than 3 times the optimal in the worst case.

Theorem 5.5 *In the worst-case, $\text{cost}(M_{\text{greedy}})/\text{cost}(M_{\text{opt}}) \geq 3 + \frac{1}{12} \approx 3.08$, even with equal speed servers and linear latency functions.*

PROOF. Consider the bipartite graph $G = (U, V)$ with $n = |U| = 66$ and $m = |V| = 64$. The edge set is defined as $E = \{(u_i, v_j) \mid j \geq \min\{i, 64\}\}$.

The cost of an optimal assignment, which assigns job i to server $\min\{i, 64\}$, is $\text{cost}(M_{\text{opt}}) = 63 + 9 = 72$. One worst-case greedy assignment maps each client $1, 2, \dots, 64$ to a permissible server minimizing the increase in L_2 norm, it breaks ties by choosing the server with higher index. Its cost amounts to $\text{cost}(M_{\text{greedy}}) = 16 + 8 \cdot 2^2 + 4 \cdot 3^2 + 2 \cdot 4^2 + 1 \cdot 5^2 + 1 \cdot 9^2 = 222$. Thus, the ratio $\text{cost}(M_{\text{greedy}})/\text{cost}(M_{\text{opt}}) = 3 + 1/12 \approx 3.08 > 3$. \square

6 Closing Remarks

The users of a decentralized systems like the Internet are sometimes best modeled as selfish and strategic players, who want to optimize their own private utility. Our selfish load balancing game models one such fundamental situation, where a set of clients must each choose a server. We showed that the worst-case Nash solution of this game is within a small constant factor of the social optimum.

The sum of the clients' latency is related to the squared sum of the server loads. In that respect, our problem can also be viewed as the *uncoordinated* version of the classical L_2 norm load balancing [3]. We reanalyzed the simple online greedy scheme and gave improved bounds on its competitive ratio.

The uncoordinated greedy scheme can also be viewed as a myopic strategy for the clients: they choose the best server available when they arrive and are not allowed to switch afterwards. By contrast, a Nash solution requires that clients reach a stable point, where no client has an incentive to switch. Our analysis shows that, despite lack of central coordination, selfish players find solutions that are *better* than greedy, which assumes centralized control but non-selfish players.

Interestingly, the Nash and the greedy can be viewed as two extremes of a *server switching cost* model—in greedy, the cost to switch is infinite; in Nash, it is zero. An intriguing open question is to investigate the tradeoffs of a finite switching cost. Another open question relates to the effect of server speeds. Our upper bounds are better for equal speed servers, but we know of no lower bound construction that gives a worse solution for arbitrary speeds than equal speeds. Do arbitrary speeds help or hurt the price of anarchy? Finally, there remains a small gap between our upper and lower bounds for the equal speed servers, and it would be interesting to determine where the truth lies.

References

- [1] N. Alon, Y. Azar, G. J. Woeginger, and T. Yadid. Approximation schemes for scheduling. In *Proc. 8th ACM-SIAM Sympos. on Discrete Algorithms*, ACM Press, 1997, pp. 493–500.
- [2] A. Avidor, Y. Azar, and J. Sgall. Ancient and new algorithms for load balancing in the L_p norm. *Algorithmica* **29** (3) (2001), 422–441.
- [3] B. Awerbuch, Y. Azar, E. F. Grove, M. Y. Kao, P. Krishnan, and J. S. Vitter. Load balancing in the L_p norm. *Proc. 36th Sympos. Foundations of Comp. Sci.*, IEEE Comp. Soc. Press, 1995, pp. 383–391.
- [4] B. Awerbuch, Y. Azar, and A. Epstein. The price of routing unsplittable flow. *Proc. 37th ACM Symposium on Theory of Computing*, ACM Press, 2005, pp. 57–66.
- [5] B. Awerbuch, Y. Azar, Y. Richter, and D. Tsur. Tradeoffs in worst-case equilibria. In *Proc. 1st Workshop on Approximation and Online Algorithms*, LNCS 2909, Springer-Verlag, 2003, pp. 41–52.
- [6] A. K. Chandra and C. K. Wong. Worst-case analysis of a placement algorithm related to storage allocation. *SIAM J. Computing* **4** (3) (1975), 249–263.

- [7] G. Christodoulou and E. Koutsoupias. The price of anarchy of finite congestion games. *Proc. 37th ACM Symposium on Theory of Computing*, ACM Press, 2005, pp. 67–73.
- [8] R.A. Cody and E. G. Coffman. Record allocation for minimizing expected retrieval costs on drum-like storage devices. *Journal of the ACM* **23** (1) (1976), 103–115.
- [9] A. Czumaj and B. Vöcking. Tight bounds for worst-case equilibria. *Proc. 13th ACM-SIAM Sympos. on Discrete Algorithms*, ACM Press, 2002, pp. 413–420; (to appear at *J. Algorithms*).
- [10] D. Fotakis, S. Kontogiannis, and P. Spirakis. Selfish unsplittable flows. In *Proc. 31st ICALP*, vol. 3142 of LNCS, Springer-Verlag, 2004, pp. 593 - 605
- [11] M. Gairing, T. Lücking, M. Mavronicolas, and B. Monien. Computing Nash Equilibria for scheduling on restricted parallel links. in *Proc. 36th ACM Sympos.on Theory of Computing*, ACM Press, 2004, pp. 613-622.
- [12] R. L. Graham. Bounds for certain multiprocessor anomalies. *Bell System Technical Journal* **45** (1966), 1563–1581.
- [13] E. Koutsoupias and C. Papadimitriou. Worst-case Equilibria. *Proc. 16th Sympos. Theoretical Aspects of Comp. Sci.*, LNCS, Vol. 1563, 1999. Springer-Verlag, pp. 404–413.
- [14] J. K. Lenstra, D. B. Shmoys, and E. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming: Ser. A* **46** (3) (1990), 259–271.
- [15] T. Lücking, M. Mavronicolas, B. Monien, M. Rode. A new model for selfish routing. *Proc. of 21st STACS*, LNCS 2996, Springer, 2004, pp. 547–558.
- [16] S. Muthukrishnan and R. Rajaraman. An adversarial model for distributed dynamic load balancing. *J. Interconnection Networks* **3** (1-2) (2002), 35-47.
- [17] C. Papadimitriou. Algorithms, games, and the internet. *Proc. 33rd ACM Symposium on Theory of Computing*, ACM Press, 2001, pp. 749-753.
- [18] S. Phillips and J. Westbrook. On-line load balancing and network flow. *Algorithmica* **21** (3) (1998), 245-261.
- [19] R. W. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory* **2** 1973, 65–67.
- [20] D. B. Shmoys, J. Wein, and D. P. Williamson. Scheduling parallel machines on-line. *SIAM J. Computing* **24** (6) (1995), 1313–1331.
- [21] T. Roughgarden. The Price of Anarchy is Independent of the Network Topology. *Journal of Computer and System Sciences*, 67(2):341–364, 2003.
- [22] T. Roughgarden. Selfish routing with atomic players. *Proc. 16th ACM-SIAM Sympos. on Discrete Algorithms*, ACM Press, 2005.
- [23] T. Roughgarden and É. Tardos. How bad is selfish routing? *Journal of ACM* **49** (2002), 235–259.