Better Bounds for Online Load Balancing on Unrelated Machines

Ioannis Caragiannis*

Abstract

We study the problem of scheduling permanent jobs on unrelated machines when the objective is to minimize the L_p norm of the machine loads. The problem is known as load balancing under the L_p norm. We present an improved upper bound for the greedy algorithm through simple analysis; this bound is also shown to be best possible within the class of deterministic online algorithms for the problem. We also address the question whether randomization helps online load balancing under L_p norms on unrelated machines; this is a challenging question which is open for more than a decade even for the L_2 norm. We provide a positive answer to this question by presenting the first randomized online algorithms which outperform deterministic ones under any (integral) L_p norm for $p = 2, ..., 137$. Our algorithms essentially compute in an online manner a fractional solution to the problem and use the fractional values to make random choices. The local optimization criterion used at each step is novel and rather counterintuitive: the values of the fractional variables for each job correspond to flows at an approximate Wardrop equilibrium for an appropriately defined non-atomic congestion game. As corollaries of our analysis and by exploiting the relation between the L_p norm and the makespan of machine loads, we obtain new competitive algorithms for online makespan minimization, making progress in another longstanding open problem.

1 Introduction

We study the following classical scheduling problem. We have m parallel machines and n independent jobs, where job i induces a (possibly infinite) positive integral load w_{ij} when processed by machine j. The load of a machine is the total load of the jobs assigned to it. The cost of an assignment (or schedule) of a set of jobs is defined as the L_p norm of the load vector, i.e., $|\ell|_p = \left(\sum_{1\leq j\leq m} |\ell_j|^p\right)^{1/p}$ for finite p and $|\ell|_\infty = \max_{1\leq j\leq m}\{|\ell_j|\}$ for the load vector $\ell = (\ell_1, ..., \ell_m)$. The goal of a load balancing algorithm is to assign all jobs to machines so that the cost is minimized.

This is the most general version of the problem known as load balancing (or scheduling) on unrelated (parallel) machines. Important special cases are those with related machines and restricted assignments. In the related machines model, each machine j has a positive speed s_j and each job i has a weight w_i and the load of job i when it is assigned to machine j is w_i/s_j. The restricted assignment model is similar; the main difference is that each job has a subset of permissible machines where it can be scheduled. The load vector of job i in the restricted assignment model can be thought of as w_i/s_j for those machines j which are permissible for job i, and ∞ for all other machines. Special cases where all machines have the same speed (identical machines) or all jobs have the same weight are also interesting.

Almost all versions of the problem are known to be NP-hard [22]; hence, most of the work has focused on efficient approximation algorithms. A scenario which better reflects practical situations is the one when the information about the jobs is not known in advance and is only revealed gradually. Computation is then performed in steps; when a job appears (together with its load vector), an online algorithm has to make an irrevocable decision and assign it to a machine. A natural online algorithm is the greedy algorithm for the L_p norm which assigns each job to that machine that minimizes the increase of the p-th power of the cost. The performance of online algorithms is assessed through the notion of competitive ratio defined as the maximum ratio over all sequences of jobs of the expected cost of the algorithm over the optimal cost. This definition is quite general to capture randomized algorithms and assesses the performance of the online algorithm against oblivious adversaries, i.e., adversaries which may have access to the probability distribution used by the algorithm to make random choices but not to the random choices themselves.

Scheduling to minimize the L_∞ norm of the load vector (also called the makespan) has received much attention. Lenstra et al. [29] and Shmoys and Tardos [34] provided 2-approximation algorithms for unrelated machines while the problem has been proved to be inapproximable within a ratio better than 3/2 [29].

*Research Academic Computer Technology Institute & Department of Computer Engineering and Informatics, University of Patras, 26500 Rio, Greece. E–mail: caragian@ceid.upatras.gr. This work was partially supported by the European Union under IST FET Integrated Project FP6-015964 AEOLUS and by a "Caratheodory" grant from the University of Patras.

Horowitz and Sahni [26] presented a polynomial-time approximation scheme (PTAS) when the number of machines is constant. For related machines, Hochbaum and Shmoys [25] (see also [24]) presented a PTAS. In the online case, Azar et al. [9] showed that no randomized online algorithm can be better than $\ln m$-competitive against oblivious adversaries even on the restricted assignments model, where m is the number of machines. Their lower bound for deterministic algorithms is $\log m$. They also showed that the greedy algorithm that assigns each job to that machine so that the increase in the makespan is minimized has optimal competitiveness in the same model. Aspnes et al. [3] observed that this greedy algorithm is only $\Omega(m)$-competitive on m unrelated machines. They obtained a deterministic $4 \log m$-competitive algorithm by combining several sophisticated techniques such as exponential weighting and doubling of estimates of the optimal makespan. A randomized $e \log m$-competitive algorithm was obtained by introducing randomized doubling to the algorithm of Aspnes et al.; this result is attributed to Indyk in [12]. The interested reader may look in standard textbooks on online computation such as [13] for a coverage of these results and the related techniques. Constant competitiveness is possible for the related machines model [3, 12]. An important special case is scheduling to minimize the makespan on identical machines. Graham [23] showed that the greedy algorithm is $2 - \frac{1}{m}$ competitive in this case. The greedy algorithm is optimal only for $m \leq 3$; for any $m > 3$, better algorithms exist [21, 33]. Bartal et al. [10] were the first to show an algorithm whose competitive ratio is below $2 - \delta$ for some constant $\delta > 0$ and arbitrary m. See [1] for the best known bounds for the problem.

The L_p norm objective has been mainly introduced since, in many applications, the makespan is not a suitable way to measure how well the jobs are balanced. Chandra and Wong [15] and Cody and Coffman [18] were the first to consider the objective of minimizing the sum of squares of the machine loads (i.e., the square of the L_2 norm). For unrelated machines, Awerbuch et al. [6] showed that the greedy algorithm has competitiveness at most $1 + \sqrt{2}$ under the L_2 norm and at most $cp + O(\log p)$ for higher norms, where $c \approx 1.7632$ is the root of the equation $c \ln c = 1$. Furthermore, they proved a lower bound of approximately $0.5307p$ for any deterministic online algorithm. Among other results, Caragiannis et al. [14] showed that the upper bound of $1 + \sqrt{2}$ for the greedy algorithm under the L_2 norm is tight even for restricted assignments. Better competitiveness bounds under the L_2 norm are known for restricted assignments and jobs with equal weights [14, 17, 35]. For the identical machines case, Avidor et al. [4] proved competitiveness bounds of $\sqrt{4/3}$ under the L_2 norm and $2 - O\left(\frac{\ln p}{p}\right)$ for higher norms.

We point out that the greedy algorithm provided the best known approximation guarantee for the offline case as well until the recent work of Azar and Epstein [7] who used convex programming to compute fractional solutions to the problem and rounding schemes to obtain efficient integral solutions. These techniques yielded a randomized $\sqrt{2}$-approximation algorithm for the L_2 norm and deterministic 2-approximation algorithms for higher norms. Better approximation guarantees (especially for small norms) were presented in [28]. Prior to [7] and [28], 2-approximation algorithms for all norms simultaneously had been presented in [8] for restricted assignments while, in the same paper, the problem was proved to be APX-hard for any L_p norm with $p \geq 2$. Alon et al. [2] proved that PTASes are possible for restricted assignments on identical machines or for a constant number of unrelated machines. Epstein and Sgall [19] presented a PTAS for related machines.

Before presenting our results, we open a parenthesis to mention that the recently emerging field of Algorithmic Game Theory [31] has extensively considered load balancing from a different perspective. An assumption made there is that the assignment of jobs is not centrally controlled but, instead, each job is owned by a selfish agent. Each agent favors assignments of her job to a machine so that the latency she experiences is minimum given the assignment of the jobs controlled by the other agents. Following the seminal work of Koutsoupias and Papadimitriou [27], a vast amount of the recent related literature studies how much the overall performance of load balancing (under several objectives) deteriorates at Nash equilibria, i.e., at assignments where no agent has an incentive to change the assignment of her job. Load balancing games are just special cases of atomic congestion games [5, 16] which model selfish behavior in network routing. Actually, load balancing can be thought of as routing demands between two nodes through m parallel links. A slightly different model which has received much attention recently and whose study has started at least half a century ago in the Economics and Transportation literature (see [32] and the references therein) is that of non-atomic congestion games in networks. We describe the simplest case of such games here; these are the only game-theoretic definitions which we actually use in the current paper. In a non-atomic congestion game on m parallel links connecting a source node to a destination node, there are infinitely many agents, each controlling a negligible amount of traffic flow from the source to the destination so that the total amount of traffic flow is unity. Each link j has a non-negative latency function f_j which indi-

cates that the latency experienced by all agents routing their traffic flows through j is $f_j(x_j)$, where x_j is the total amount of flow routed through j. Again, agents favor assignments of their flow so that the latency experienced is minimum given the assignment of the other agents. Assignments from where no agent has an incentive to deviate are called Wardrop equilibria [36] and satisfy the following condition: a flow vector x is at a Wardrop equilibrium if for any two links j, j' with $x_j > 0$, it holds that $f_j(x_j) \leq f_{j'}(x_{j'})$. Beckmann et al. [11] have presented a convex potential function whose local minima correspond to Wardrop equilibria; hence, Wardrop equilibria for non-atomic congestion games in networks can be arbitrarily approximated using convex programming techniques (e.g., [30]). See also [20] (and the references therein) for recent distributed approximations of such equilibria. Much easier and faster approximations of Wardrop equilibria are possible for games on parallel links provided that the latency functions have some reasonable form (e.g., they are polynomials).

In this paper, we study the online optimization version of load balancing in a coordinated way. Although several recent papers have considered important special cases of online load balancing, no progress on the general problem with the L_p norm objective has been made since the work of Awerbuch et al. [6] in 1995. We make such a progress and improve most of the results of [6]. We strengthen and significantly simplify the analysis of the greedy algorithm and show that it is at most $\frac{1}{2^{1/p}-1}$-competitive under the L_p norm. We also show that this bound is tight for any $p \geq 1$, i.e., we show that no deterministic online algorithm can have better competitiveness than greedy. Our bound approaches $\frac{p}{\ln 2} \approx 1.4427p$ from below as p increases. As a corollary of our analysis and by exploiting the relation of the L_p norm and the makespan of machine loads, we obtain that the greedy algorithm for the $L_{\ln m}$ norm is $e \log m$-competitive for makespan minimization on m unrelated machines. This is a rather surprising result since it means that the same competitiveness with the randomized version of the algorithm of Aspnes et al. [3] can be obtained by a much simpler deterministic greedy-like algorithm. These results are presented in Section 3.

In Section 4, we present the most interesting and technically involved results of the paper. We demonstrate that we can beat the lower bounds on the competitiveness of deterministic algorithms for load balancing under L_p norms by using randomization. We present a general algorithm (called Balance) which is parameterized by p and a parameter vector α. The algorithm is watching the moments of the random variables denoting the load on each machine and, at each step, it uses this information together with the job loads in order to compute the probability distribution according to which it will make the decision for the job considered. This can be thought of as computing a fractional solution to the problem which is rounded in a randomized way in order to obtain an integral assignment. At each step, the criterion used in order to compute the probability distribution of the random choice suggests a nice interplay with games. The algorithm considers a non-atomic congestion game on parallel links (corresponding to the machines) with appropriately defined latency functions, computes an approximate Wardrop equilibrium, and sets the probabilities of the random choice equal to the flows in this equilibrium. Note that, unlike the approach in Algorithmic Game Theory mentioned above, the game is used as a tool by our algorithm in order to coordinate the assignment of jobs efficiently. Our analysis leads to sufficient conditions for the selection of the parameter vector α. Using these conditions, we are able to compute appropriate parameters so that algorithm Balance is at most $1.222p$-competitive against oblivious adversaries for integral values of p up to 137. For the L_2 norm, our algorithm is $\sqrt{5}$-competitive. By exploiting the relation of L_p norms to the makespan, we also obtain the first competitiveness upper bounds for makespan minimization which beat the result of Aspnes et al. [3] when the number of machines is up to an astronomically large constant (i.e., e^{137}).

We begin with some preliminary definitions in Section 2 and conclude with extensions and open problems in Section 5. Due to lack of space, some proofs have been omitted.

2 Preliminaries

We briefly present the notation used in the analysis of our algorithms for the L_p norm. For an integer k, we usually use $[k]$ to denote the set $\{1, 2, ..., k\}$. When we consider an instance of the problem, we denote by ℓ_j^* the load of machine j in the optimal solution. We use the binary variables $y_{ij} \in \{0, 1\}$ to denote whether job i is assigned to machine j in the optimal solution ($y_{ij} = 1$) or not ($y_{ij} = 0$). Clearly, $\sum_i y_{ij} w_{ij} = \ell_j^*$ and $\sum_i y_{ij} w_{ij}^t \leq \left(\sum_i y_{ij} w_{ij}\right)^t = \ell_j^{*t}$ for any $t \geq 1$. We also denote by Λ_{ij} the load of machine j after the assignment of job i. Hence, in order to prove upper bounds on the competitive ratio, it suffices to bound the ratio of $\mathbb{E}\left[\left(\sum_j \Lambda_{nj}^p\right)^{1/p}\right]$ (or simply $\left(\sum_j \Lambda_{nj}^p\right)^{1/p}$ when the greedy algorithm is concerned) over $\left(\sum_j \ell_j^{*p}\right)^{1/p}$.

In the analysis we use the Minkowski inequality (or the triangle inequality for the L_p norm) stating that $\left(\sum_{t=1}^k (a_t + b_t)^p\right)^{1/p} \leq \left(\sum_{t=1}^k a_t^p\right)^{1/p} + \left(\sum_{t=1}^k b_t^p\right)^{1/p}$

for any $p \geq 1$ and $a_t, b_t \geq 0$, as well as Hölder inequality stating that $\mathbb{E}[Z] \leq (\mathbb{E}[Z^t])^{1/t}$ for any non-negative random variable Z and $t \geq 1$ (see `wikipedia.org` for a detailed presentation of these inequalities).

Our results for makespan minimization follow by the analysis of the algorithms for the L_p norm and the following observation.

LEMMA 2.1. *Let A_p be a c-competitive online algorithm for load balancing on unrelated machines under the L_p norm. Then, algorithm A_p is $cm^{1/p}$-competitive for makespan minimization on m unrelated machines.*

3 Deterministic online algorithms

In this section we compute the exact competitiveness bound of the greedy algorithm for the L_p norm and we show that this is the best possible among deterministic online algorithms.

THEOREM 3.1. *For any $p \geq 1$, the greedy algorithm has competitive ratio at most $\frac{1}{2^{1/p}-1}$ for load balancing on unrelated machines under the L_p norm.*

Proof. At the step associated with job i, we have

$$\begin{aligned} & \sum_j \left(\Lambda_{ij}^p - \Lambda_{i-1,j}^p\right) \\ \leq & \sum_j \left((\Lambda_{i-1,j} + y_{ij}w_{ij})^p - \Lambda_{i-1,j}^p\right) \\ \leq & \sum_j \left((\Lambda_{nj} + y_{ij}w_{ij})^p - \Lambda_{nj}^p\right) \end{aligned} \tag{3.1}$$

where the first inequality follows by the greedy nature of the algorithm and the second inequality follows since $\Lambda_{nj} \geq \Lambda_{ij}$ for any i, j and the function $f(z) = (z+a)^p - z^p$ is non-decreasing in $[0, \infty)$ for $a \geq 0$ and $p \geq 1$.

Next we will need the following technical lemma.

LEMMA 3.1. *Let $p \geq 1$, $t \geq 0$ and $a_i \geq 0$, for $i = 1, ..., k$. Then,*

$$\sum_{i=1}^{k} ((t + a_i)^p - t^p) \leq \left(t + \sum_{i=1}^{k} a_i\right)^p - t^p.$$

Using (3.1) and summing over all steps of the algorithm, we obtain that

$$\begin{aligned} & \sum_j \Lambda_{nj}^p \\ = & \sum_i \sum_j \left(\Lambda_{ij}^p - \Lambda_{i-1,j}^p\right) \\ \leq & \sum_i \sum_j \left((\Lambda_{nj} + y_{ij}w_{ij})^p - \Lambda_{nj}^p\right) \\ = & \sum_j \sum_i \left((\Lambda_{nj} + y_{ij}w_{ij})^p - \Lambda_{nj}^p\right) \\ \leq & \sum_j \left((\Lambda_{nj} + \sum_i y_{ij}w_{ij})^p - \Lambda_{nj}^p\right) \\ = & \sum_j (\Lambda_{nj} + \ell_j^*)^p - \sum_j \Lambda_{nj}^p \\ \leq & \left(\left(\sum_j \Lambda_{nj}^p\right)^{\frac{1}{p}} + \left(\sum_j \ell_j^{*p}\right)^{\frac{1}{p}}\right)^p - \sum_j \Lambda_{nj}^p \end{aligned} \tag{3.2}$$

The second inequality follows by Lemma 3.1 while the last inequality is derived by applying Minkowski inequality. We divide all sides of (3.2) by $\sum_j \ell_j^{*p}$ and set $c = \left(\sum_j \Lambda_{nj}^p\right)^{\frac{1}{p}} / \left(\sum_j \ell_j^{*p}\right)^{\frac{1}{p}}$. Now (3.2) becomes $2c^p \leq (c+1)^p$ which yields the desired result $c \leq \frac{1}{2^{1/p}-1}$. ■

Using the inequality $e^z \geq 1 + z$, we have that $\frac{1}{2^{1/p}-1} \leq \frac{p}{\ln 2} \approx 1.4427p$. This improves the previous bound of $1.7632p + O(\log p)$ of [6]. We also show that greedy is optimal within the class of deterministic online algorithms for any L_p norm; this result also improves the previously best known lower bound of $0.5307p$ of [6].

THEOREM 3.2. *For any $\epsilon > 0$ and any $p \geq 1$, no deterministic online load balancing algorithm on unrelated machines can be better than $\frac{1}{2^{1/p}-1} - \epsilon$-competitive under the L_p norm.*

Proof. We use a similar construction with [6] (i.e., restricted assignments on identical machines) but we also allow jobs have different loads. This also generalizes a construction used in [14] to show a lower bound of $1 + \sqrt{2}$ on the competitiveness of the greedy algorithm under the L_2 norm.

Let $\delta > 0$ and integer k be parameters (to be defined later) and let T be such that $T \geq \frac{1}{(1+\delta)^{1/p}-1}$. Consider the execution of a deterministic online algorithm on $m = 2^k$ machines and an adversary that reveals jobs in k phases. Initially, all 2^k machines are active. In each phase, the adversary matches the active machines into pairs. In the phase i ($i = 0, ..., k-1$), for each pair of machines (a, b), the adversary presents one job with load $\lceil 2^{i/p}T \rceil$ on machines a and b, and infinite load on any other machine. The machines that are assigned jobs by the algorithm remain active for the next phase; machines that are not assigned jobs become inactive.

Denote by *opt* and *det* the p-th power of the cost of the optimal assignment and the cost of the assignment computed by the deterministic algorithm, respectively. We observe that a machine that becomes inactive immediately after phase i (for $i = 0, ..., k-1$) is not assigned any job at phase i and later phases while it is assigned one job in each phase before phase i (if

any). The machine that is assigned the job at phase $k-1$ is also assigned one job in each phase before phase $k-1$. Since 2^{k-1-i} machines become inactive immediately after phase i, we have that

$$
\begin{aligned}
(3.3)\quad & det \\
= & \left(\sum_{j=0}^{k-1} \lceil 2^{j/p} T \rceil\right)^p + \sum_{i=1}^{k-1} 2^{k-1-i} \left(\sum_{j=0}^{i-1} \lceil 2^{j/p} T \rceil\right)^p \\
\geq & \sum_{i=1}^{k} 2^{k-1-i} \left(\sum_{j=0}^{i-1} 2^{j/p} T\right)^p \\
= & \left(\frac{T}{2^{1/p}-1}\right)^p \sum_{i=1}^{k} 2^{k-1-i} \left(2^{i/p}-1\right)^p \\
= & \left(\frac{T}{2^{1/p}-1}\right)^p 2^{k-1} \sum_{i=1}^{k} \left(1-2^{-i/p}\right)^p \\
\geq & \left(\frac{T}{2^{1/p}-1}\right)^p 2^{k-1} \sum_{i=1}^{k} \left(1-p2^{-i/p}\right) \\
\geq & \left(\frac{T}{2^{1/p}-1}\right)^p 2^{k-1} \left(k - p\sum_{i=1}^{\infty} 2^{-i/p}\right) \\
= & \left(\frac{T}{2^{1/p}-1}\right)^p 2^{k-1} \left(k+p-p\frac{1}{1-2^{-1/p}}\right) \\
\geq & \left(\frac{T}{2^{1/p}-1}\right)^p 2^{k-1} \left(k+p-2p^2\right).
\end{aligned}
$$

The second inequality follows since $(1-a)^p \geq 1-ap$ for any $a \in [0,1]$ and $p \geq 1$ while the last inequality follows since $p(1-2^{-1/p})$ is increasing for $p \geq 1$ and, hence, $p(1-2^{-1/p}) \geq 1/2$ which yields $\frac{1}{1-2^{-1/p}} \leq 2p$.

In order to bound *opt* from above, consider the assignment in which each job is assigned in opposition to the algorithm. Given a job with finite load on a pair of machines (a,b), the job is assigned to a if the algorithm assigns it to b, and vice versa. Then, each machine has at most one job and using the relation between T and δ, we obtain

$$
\begin{aligned}
(3.4)\qquad opt & \leq \sum_{i=0}^{k-1} 2^{k-1-i} \lceil T 2^{i/p} \rceil^p \\
& \leq \sum_{i=0}^{k-1} 2^{k-1-i} \left(T2^{i/p}+1\right)^p \\
& \leq T^p(1+\delta)k2^{k-1}.
\end{aligned}
$$

By comparing (3.3) and (3.4), we obtain that for any $\epsilon > 0$ there are sufficiently large k and sufficiently small δ so that the ratio of the cost of the assignment computed by the deterministic algorithm over the optimal cost is $\left(\frac{det}{opt}\right)^{1/p} \geq \frac{1}{2^{1/p}-1} - \epsilon$. ∎

We remark that no restriction on the number of machines appears in the statement of Theorem 3.2. Our lower bound construction essentially shows that when $p = o\left(\sqrt{\log m}\right)$ there exists an instance with m machines on which any deterministic online algorithm has competitive ratio arbitrarily close to $\frac{1}{2^{1/p}-1}$. By tightening the inequalities used in order to obtain (3.3), we can show that the same holds when $p = o\left(\frac{\log m}{\log\log m}\right)$. The lower bound of Theorem 3.2 definitely does not hold for $p = \omega(\log m)$ since, in this case, the $O(\log m)$-competitive algorithms for makespan minimization can be easily proved to be $O(\log m)$-competitive under the L_p norm.

We conclude this section by presenting a result for makespan minimization. It follows as a corollary of our analysis of the greedy algorithm (Theorem 3.1) and Lemma 2.1.

THEOREM 3.3. *The greedy algorithm for the $L_{\ln m}$ norm is $e \log m$-competitive for makespan minimization, where m is the number of unrelated machines.*

The greedy algorithm for the $L_{\ln m}$ norm has several advantages over the $e \log m$-competitive randomized algorithm obtained by the techniques of Aspnes et al. [3] extended with randomized doubling. First, it is deterministic. Second, it does not use exponential weighting and, hence, polynomial space is always sufficient in order to perform computations. Actually, the greedy algorithm for the $L_{\ln m}$ norm can be thought of as using subexponential weighting. Third, it does not use estimates of the optimal makespan or doubling.

4 Randomized online algorithms

In this section we present the first randomized online algorithms for load balancing on unrelated machines that beat the lower bound on the competitiveness of deterministic ones under L_p norms. Our algorithm is called Balance, uses a vector α of p positive values, and works as follows. When a new job i arrives, Balance computes a probability distribution on the machines, i.e., probabilities x_{ij} that job i is assigned to machine j. Then, it simply casts a die that has one face for each machine j with $x_{ij} > 0$ (with x_{ij} being the probability that the face corresponding to machine j is the outcome of the die casting) and assigns job i to the machine corresponding to the outcome of die casting.

Motivated by the greedy algorithm, a naive way to compute the probabilities in each step could be to minimize the increase in $\mathbb{E}\left[\sum_j \Lambda_{nj}^p\right] = \sum_j \mathbb{E}[\Lambda_{nj}^p]$ due to the decision at the step. Unfortunately, it is not hard to slightly modify the proof of Theorem 3.2 and construct instances where such an algorithm mimics the greedy algorithm and cannot achieve a

better competitive ratio. Instead, algorithm Balance is watching all the integral moments of the machine loads and, at each step, it makes its decision trying to balance the increase in each of them. In order to do so, it defines appropriate games at each step, computes equilibria for them, and makes its decisions according to these equilibria.

At each step i, the algorithm considers a particular non-atomic congestion game with a unit amount of flow that has to be routed from a source node to a destination node which are connected with m parallel links. Each link j corresponds to machine j of the load balancing instance and has an appropriately defined non-decreasing latency function f_{ij}. Algorithm Balance computes an ϵ-approximate Wardrop equilibrium with $\epsilon = \frac{\alpha_p(2^p-1)}{m}$, i.e., a flow vector x_i such that for any two links $j, j' \in [m]$ with $x_{ij} > 0$, it holds that $f_{ij}(x_{ij}) - f_{ij'}(x_{ij'}) \leq \epsilon$. This also yields

$$\sum_j x_{ij} f_{ij}(x_{ij}) - \epsilon \leq f_{ij'}(x_{ij'}) \tag{4.5}$$

for any link $j' \in [m]$.

In order to define the latency functions on the links, the algorithm keeps track of auxiliary functions g_{ijt} for $i \in [n]$, $\in [m]$, and $t = 0, 1, ..., p$ defined as follows. At the first step, the algorithm sets $g_{1jt}(z) = zw_{ij}^t$ if $t > 0$, and $g_{1j0}(z) = 1$ for any $j \in [m]$ and $z \geq 0$. For $i > 1$, during step i, the algorithm defines the auxiliary functions g_{ijt} for $j \in [m]$ and $t = 0, ..., p$ using the auxiliary functions defined in the previous step, the vector of probabilities x_{i-1} computed at the previous step, and the load of job i on the machines, as follows:

$$\begin{aligned} g_{ijt}(z) &= z\sum_{s=0}^{t-1} \binom{t}{s} g_{i-1,js}(x_{i-1,j}) w_{ij}^{t-s} \\ &\quad + g_{i-1,jt}(x_{i-1,j}) \end{aligned}$$

Although complicated at first glance, the auxiliary functions have been defined in such a way that $g_{ijt}(x_{ij}) = \mathbb{E}[\Lambda_{ij}^t]$. This is stated in the following lemma together with other properties of the auxiliary functions g_{ijt} that will be useful in our analysis.

LEMMA 4.1. *For any $i > 0$, $j \in [m]$, and $t = 0, ..., p$, the following properties hold:*

1. $g_{ijt}(z) = z(g_{ijt}(1) - g_{ijt}(0)) + g_{ijt}(0)$, *for any* $z \geq 0$.
2. $g_{ijt}(1) = \sum_{s=0}^t \binom{t}{s} g_{ijs}(0) w_{ij}^{t-s}$.
3. $g_{ijt}(x_{ij}) = \mathbb{E}[\Lambda_{ij}^t] \leq \mathbb{E}[\Lambda_{nj}^t]$.
4. $(g_{ijt}(z))^u - (g_{ijt}(0))^u \leq uz\,(g_{ijt}(z))^{u-1}\,(g_{ijt}(1) - g_{ijt}(0))$, *for any* $u \geq 1$ *and* $z \geq 0$.

Proof. Properties 1 and 2 follow trivially by the definition of g_{ijt}.

The inequality in Property 3 is obvious. In order to prove the equality in Property 3, we will use induction on i. For $i = 1$, if $t = 0$, we have $\mathbb{E}[\Lambda_{1j}^0] = 1 = g_{1j0}(x_{1j})$ by definition. Also, if $t > 0$, then Λ_{1j}^t is w_{1j}^t with probability x_{1j} and 0 with probability $1 - x_{ij}$, i.e., $\mathbb{E}[\Lambda_{1j}^t] = x_{1j} w_{1j}^t = g_{1jt}(x_{1j})$ by definition. Now assume that $\mathbb{E}[\Lambda_{i'j}^t] = g_{i'jt}(x_{i'j})$ for any $i' < i$ and $t = 0, ..., p$. Then, the random variable Λ_{ij}^t equals $(\Lambda_{i-1,j} + w_{ij})^t$ with probability x_{ij} and $\Lambda_{i-1,j}^t$ with probability $1 - x_{ij}$. Hence, using linearity of expectation, the inductive hypothesis, and the definition of auxiliary function g_{ijt}, we have

$$\begin{aligned} \mathbb{E}[\Lambda_{ij}^t] &= \mathbb{E}\left[x_{ij}\,(\Lambda_{i-1,j} + w_{ij})^t + (1 - x_{ij})\Lambda_{i-1,j}^t\right] \\ &= \mathbb{E}\left[x_{ij} \sum_{s=0}^{t-1} \binom{t}{s} \Lambda_{i-1,j}^s w_{ij}^{t-s} + \Lambda_{i-1,j}^t\right] \\ &= x_{ij} \sum_{s=0}^{t-1} \binom{t}{s} \mathbb{E}[\Lambda_{i-1,j}^s] w_{ij}^{t-s} + \mathbb{E}[\Lambda_{i-1,j}^t] \\ &= x_{ij} \sum_{s=0}^{t-1} \binom{t}{s} g_{i-1,js}(x_{i-1,j}) w_{ij}^{t-s} \\ &\quad + g_{i-1,jt}(x_{i-1,j}) \\ &= g_{ijt}(x_{ij}). \end{aligned}$$

Property 4 clear holds if $z = 0$ or if g_{ijt} is a constant function. In order to prove it for $z > 0$ and when it is not constant, consider the function $h(z_1) = (z_1 + c)^u$ for some non-negative constant c. Since h is convex in $[0, +\infty)$ for $u \geq 1$, its derivative at point $z_2 > 0$ is not smaller than the slope of the line crossing points $(0, h(0))$ and $(z_2, h(z_2))$, i.e., $\frac{h(z_2) - h(0)}{z_2} \leq u(z_2 + c)^{u-1}$ which yields $h(z_2) - h(0) \leq u(z_2 + c)^{u-1} z_2$. Property 4 then follows by setting $z_2 = z(g_{ijt}(1) - g_{ijt}(0))$ and $c = g_{ijt}(0)$ since (by Property 1) $h(z_2) = z(g_{ijt}(1) - g_{ijt}(0)) + g_{ijt}(0) = g_{ijt}(z)$. ∎

The latency function associated with link j is defined as

$$f_{ij}(z) = \sum_{t=1}^p \frac{p\alpha_t}{t} (g_{ijt}(z))^{p/t-1} \sum_{s=0}^{t-1} \binom{t}{s} g_{ijs}(z) w_{ij}^{t-s}.$$

This completes the description of the algorithm. So far, it should have become clear that the algorithm does not use the outcome of previous random choices in order to make decisions at any step. Instead, it uses the moments of the random variables denoting the load on each machine which in turn depend only on the probability distribution of previous random choices. This nice property makes the analysis of the algorithm

tractable. We are now ready to prove the following statement which characterizes the competitiveness of algorithm Balance in terms of p and the parameter vector α.

LEMMA 4.2. *Let $p \geq 2$ be an integer. If there exist positive numbers $\xi_t > 0$ for $t = 0, ..., p$ such that the values of vector α satisfy*

$$\alpha_t \frac{p-t}{t} \xi_t^{-t} \sum_{s=0}^{t-1} \binom{t}{s} \xi_s^s + \sum_{\kappa=t+1}^{p} \alpha_\kappa \binom{\kappa-1}{t-1} \xi_\kappa^{p-\kappa} \xi_t^{t-p} \leq \alpha_t \tag{4.6}$$

for $t = 1, ..., p-1$, then algorithm (α, p)-Balance is $(\beta/\alpha_p)^{1/p}$-competitive against oblivious adversaries for load balancing on unrelated machines under the L_p norm, where

$$\beta = \sum_{t=1}^{p} \alpha_t \xi_t^{p-t} \sum_{s=0}^{t-1} \binom{t-1}{s} \xi_s^s.$$

Proof. We wish to show that the expectation of the random variable $X = \left(\sum_j \Lambda_{nj}^p\right)^{1/p}$ is at most $\left(\frac{\beta}{\alpha_p} \sum_j \ell_j^{*p}\right)^{1/p}$. By Hölder inequality on expectations of non-negative random variables, we have that $\mathbb{E}[X] \leq \mathbb{E}[X^p]^{1/p}$ and, hence, it suffices to show that $\mathbb{E}\left[\sum_j \Lambda_{nj}^p\right]^{1/p} \leq \left(\frac{\beta}{\alpha_p} \sum_j \ell_j^{*p}\right)^{1/p}$.

By linearity of expectation, we have

$$\begin{aligned} \alpha_p \mathbb{E}\left[\sum_j \Lambda_{nj}^p\right] &= \alpha_p \sum_j \mathbb{E}\left[\Lambda_{nj}^p\right] \\ &= \sum_j \sum_{t=1}^{p} \alpha_t \mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t} \\ &\quad - \sum_j \sum_{t=1}^{p-1} \alpha_t \mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t}. \end{aligned} \tag{4.7}$$

We will work on the first sum at the right-hand side of (4.7). Using the properties of functions g_{ijt}, we obtain

$$\begin{aligned} &\sum_j \sum_{t=1}^{p} \alpha_t \mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t} \\ &= \sum_{i=1}^{n} \sum_j \sum_{t=1}^{p} \alpha_t \left(\mathbb{E}\left[\Lambda_{ij}^t\right]^{p/t} - \mathbb{E}\left[\Lambda_{i-1,j}^t\right]^{p/t}\right) \\ &= \sum_{i=1}^{n} \sum_j \sum_{t=1}^{p} \alpha_t \left((g_{ijt}(x_{ij}))^{p/t} - (g_{ijt}(0))^{p/t}\right). \end{aligned} \tag{4.8}$$

Next we use a technical lemma which follows by the definition of function f_{ij} and the properties of functions g_{ijt}.

LEMMA 4.3. *At each step $i \in [n]$ and for each machine $j \in [m]$, it holds that*

$$\begin{aligned} &\sum_{t=1}^{p} \alpha_t \left((g_{ijt}(x_{ij}))^{p/t} - (g_{ijt}(0))^{p/t}\right) \\ &\leq x_{ij} f_{ij}(x_{ij}) - x_{ij}^2 \alpha_p (2^p - 1). \end{aligned}$$

Proof. We use the properties 1, 2, and 4 of functions g_{ijt} from Lemma 4.1 to obtain

$$\begin{aligned} &\sum_{t=1}^{p} \alpha_t \left((g_{ijt}(x_{ij}))^{p/t} - (g_{ijt}(0))^{p/t}\right) \\ &\leq x_{ij} \sum_{t=1}^{p} \alpha_t \frac{p}{t} \left(g_{ijt}(x_{ij})\right)^{p/t-1} \left(g_{ijt}(1) - g_{ijt}(0)\right) \\ &= x_{ij} \sum_{t=1}^{p} \alpha_t \frac{p}{t} \left(g_{ijt}(x_{ij})\right)^{p/t-1} \sum_{s=0}^{t-1} \binom{t}{s} g_{ijs}(0) w_{ij}^{t-s} \\ &= x_{ij} \sum_{t=1}^{p} \alpha_t \frac{p}{t} \left(g_{ijt}(x_{ij})\right)^{p/t-1} \sum_{s=0}^{t-1} \binom{t}{s} g_{ijs}(x_{ij}) w_{ij}^{t-s} \\ &\quad - x_{ij} \sum_{t=1}^{p} \alpha_t \frac{p}{t} \left(g_{ijt}(x_{ij})\right)^{p/t-1} \\ &\quad \cdot \sum_{s=0}^{t-1} \binom{t}{s} \left(g_{ijs}(x_{ij}) - g_{ijs}(0)\right) w_{ij}^{t-s} \\ &= x_{ij} f_{ij}(x_{ij}) - x_{ij}^2 \sum_{t=1}^{p} \alpha_t \frac{p}{t} \left(g_{ijt}(x_{ij})\right)^{p/t-1} \\ &\quad \cdot \sum_{s=0}^{t-1} \binom{t}{s} w_{ij}^{t-s} \left(g_{ijs}(1) - g_{ijs}(0)\right) \\ &= x_{ij} f_{ij}(x_{ij}) - x_{ij}^2 \sum_{t=1}^{p} \alpha_t \frac{p}{t} \left(g_{ijt}(x_{ij})\right)^{p/t-1} \\ &\quad \cdot \sum_{s=0}^{t-1} \binom{t}{s} \sum_{u=0}^{s-1} \binom{s}{u} g_{iju}(0) w_{ij}^{t-u} \\ &\leq x_{ij} f_{ij}(x_{ij}) - x_{ij}^2 \alpha_p \sum_{s=0}^{p-1} \binom{t}{s} g_{ij0}(0) w_{ij}^t \\ &\leq x_{ij} f_{ij}(x_{ij}) - x_{ij}^2 \alpha_p (2^p - 1). \end{aligned}$$

The last inequality follows since $g_{ij0}(0) = 1$ and $w_{ij} \geq 1$. ■

Now, working with (4.8) and using Lemma 4.3 and the inequality $\sum_j x_{ij}^2 \geq 1/m$ since $\sum_j x_{ij} = 1$, we obtain

$$\sum_j \sum_{t=1}^{p} \alpha_t \mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t} \tag{4.9}$$

$$\leq \sum_{i=1}^{n}\left(\sum_j x_{ij}f_{ij}(x_{ij}) - \alpha_p(2^p-1)\sum_j x_{ij}^2\right)$$
$$\leq \sum_{i=1}^{n}\left(\sum_j x_{ij}f_{ij}(x_{ij}) - \frac{\alpha_p(2^p-1)}{m}\right).$$

Since the flow vector x_i is an $\frac{\alpha_p(2^p-1)}{m}$-approximate Wardrop equilibrium for the congestion game considered at step i, we use (4.5) and (4.9) to obtain a relation of the left-hand side of (4.9) to the decisions in the optimal assignment, i.e.,

$$\sum_j\sum_{t=1}^{p}\alpha_t\mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t} \leq \sum_{i=1}^{n}\sum_j y_{ij}f_{ij}(x_{ij}).$$

By substituting f_{ij}, using the property $g_{ijt}(x_{ij}) \leq \mathbb{E}[\Lambda_{nj}^t]$ for any $i \in [n], j \in [m]$, and $t = 0, ..., p$, and since $\sum_{i=1}^n y_{ij}w_{ij}^t \leq \ell_j^{*t}$ for any $j \in [m]$ and $t \in [p]$, we obtain a more clear relation to the optimal assignment:

$$\begin{aligned}
(4.10)\quad & \sum_j\sum_{t=1}^{p}\alpha_t\mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t} \\
\leq\ & \sum_{i=1}^{n}\sum_j\sum_{t=1}^{p}\sum_{s=0}^{t-1}\frac{p\alpha_t}{t}\binom{t}{s}(g_{ijt}(x_{ij}))^{p/t-1} \\
& \cdot g_{ijt}(x_{ij})y_{ij}w_{ij}^{t-s} \\
\leq\ & \sum_{i=1}^{n}\sum_j\sum_{t=1}^{p}\sum_{s=0}^{t-1}\frac{p\alpha_t}{t}\binom{t}{s}\mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t-1}\mathbb{E}\left[\Lambda_{nj}^s\right] \\
& \cdot y_{ij}w_{ij}^{t-s} \\
=\ & \sum_j\sum_{t=1}^{p}\sum_{s=0}^{t-1}\frac{p\alpha_t}{t}\binom{t}{s}\mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t-1}\mathbb{E}\left[\Lambda_{nj}^s\right] \\
& \cdot\sum_{i=1}^{n}y_{ij}w_{ij}^{t-s} \\
\leq\ & \sum_j\sum_{t=1}^{p}\sum_{s=0}^{t-1}\frac{p\alpha_t}{t}\binom{t}{s}\mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t-1}\mathbb{E}\left[\Lambda_{nj}^s\right]\ell_j^{*t-s}.
\end{aligned}$$

In order to upper-bound the right-hand side of (4.10), we will use the following technical lemma.

LEMMA 4.4. *Let γ, δ be non-negative integers such that $\gamma + \delta \leq p$ and $\zeta_1, \zeta_2 > 0$. Then,*

$$(4.11)\quad z_1^\gamma z_2^\delta z_3^{p-\gamma-\delta} \leq \frac{\gamma}{p}\zeta_1^{\gamma-p}\zeta_2^\delta z_1^p + \frac{\delta}{p}\zeta_1^\gamma\zeta_2^{\delta-p}z_2^p + \left(1-\frac{\gamma+\delta}{p}\right)\zeta_1^\gamma\zeta_2^\delta z_3^p$$

for any $z_1, z_2, z_3 \geq 0$.

Given ζ_1 and ζ_2, what Lemma 4.4 essentially does is to bound the left-hand side of (4.11) with a polynomial of the form $c_1z_1^p + c_2z_2^p + c_3z_3^p$ so that the bound is tight when $z_1 = \zeta_1$ and $z_2 = \zeta_2$. We apply Lemma 4.4 to each term at the right-hand side of (4.10). In particular, applying Lemma 4.4 with $z_1 = \mathbb{E}\left[\Lambda_{nj}^t\right]^{1/t}$, $z_2 = \mathbb{E}\left[\Lambda_{nj}^s\right]^{1/s}$, $z_3 = \ell_j^*$, $\zeta_1 = \xi_t$, $\zeta_2 = \xi_s$, $\gamma = p - t$, and $\delta = s$, we obtain

$$\begin{aligned}
\mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t-1}\mathbb{E}\left[\Lambda_{nj}^s\right]\ell_j^{*t-s} \leq\ & \frac{p-t}{p}\xi_t^{-t}\xi_s^s\mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t} \\
& +\frac{s}{p}\xi_t^{p-t}\xi_s^{s-p}\mathbb{E}\left[\Lambda_{nj}^s\right]^{p/s} \\
& +\frac{t-s}{p}\xi_t^{p-t}\xi_s^s\ell_j^{*p}
\end{aligned}$$

and, hence, (4.10) yields

$$\begin{aligned}
& \sum_j\sum_{t=1}^{p}\alpha_t\mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t} \\
\leq\ & \sum_j\sum_{t=1}^{p}\sum_{s=0}^{t-1}\frac{\alpha_t}{t}\binom{t}{s}\Big((p-t)\xi_t^{-t}\xi_s^s\mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t} \\
& +s\xi_t^{p-t}\xi_s^{s-p}\mathbb{E}\left[\Lambda_{nj}^s\right]^{p/s} + (t-s)\xi_t^{p-t}\xi_s^s\ell_j^{*p}\Big) \\
=\ & \sum_j\sum_{t=1}^{p-1}\Bigg(\alpha_t\frac{p-t}{t}\xi_t^{-t}\sum_{s=0}^{t-1}\binom{t}{s}\xi_s^s \\
& +\sum_{\kappa=t+1}^{p}\alpha_\kappa\binom{\kappa-1}{t-1}\xi_\kappa^{p-\kappa}\xi_t^{t-p}\Bigg)\mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t} \\
& +\sum_j\sum_{t=1}^{p}\alpha_t\xi_t^{p-t}\sum_{s=0}^{t-1}\binom{t-1}{s}\xi_s^s\ell_j^{*p} \\
\leq\ & \sum_j\sum_{t=1}^{p-1}\alpha_t\mathbb{E}\left[\Lambda_{nj}^t\right]^{p/t} + \beta\sum_j\ell_j^{*p}
\end{aligned}$$

where the last inequality follows by (4.6) and by the definition of β. Hence, (4.7) yields $\mathbb{E}\left[\sum_j\Lambda_{nj}^p\right] \leq \frac{\beta}{\alpha_p}\sum_j\ell_j^{*p}$ and Lemma 4.2 follows. ∎

By applying Lemma 4.2, we can easily obtain the following result for the L_2 norm.

COROLLARY 4.1. *Algorithm $(\alpha, 2)$-Balance with $\alpha_1 = \alpha_2 = 1$ is $\sqrt{5}$-competitive against oblivious adversaries for load balancing on unrelated machines under the L_2 norm.*

Proof. Applying Lemma 4.2 with $\xi_0 = \xi_1 = \xi_2 = 2$, we have that inequality (4.6) is satisfied and $\beta = 5$. ∎

For larger norms, computing the best possible values for vector α so that there exist positive ξ_t's that

satisfy the conditions (4.6) of Lemma 4.2 and simultaneously minimize β is not easy since these conditions are quite complicated. So, in order to compute good values for the parameters of algorithm Balance, we reason as follows. We will denote by $\hat{\alpha}$ the corresponding parameter vector. First, we normalize vector $\hat{\alpha}$ by setting $\hat{\alpha}_p = 1$. We guess appropriate values $\hat{\xi}_t$ for ξ_t's and require that the conditions (4.6) of Lemma 4.2 are satisfied with equality. This yields a system of $p-1$ linear equations in which the $\hat{\alpha}_t$'s (for $t = 1, ..., p-1$) are the unknowns and which can be represented as $A\hat{\alpha} = b$ where A is an upper triangular $(p-1) \times (p-1)$ matrix with

$$A_{t,t} = 1 - \frac{p-t}{t}\hat{\xi}_t^{-t} \sum_{s=0}^{t-1} \left(\begin{array}{c} t \\ s \end{array} \right) \hat{\xi}_s^s$$

for $t = 1, ..., p-1$, $A_{t,\kappa} = 0$ for $t = 1, ..., p-1$ and $\kappa = 1, ..., t-1$, and

$$A_{t,\kappa} = -\left(\begin{array}{c} \kappa - 1 \\ t - 1 \end{array} \right) \hat{\xi}_\kappa^{p-\kappa} \hat{\xi}_t^{t-p}$$

for $t = 1, ..., p-1$ and $\kappa = t+1, ..., p-1$, and $b_t = (p-1)\hat{\xi}_t^{t-p}$ for $t = 1, ..., p-1$. Of course, we must guess the $\hat{\xi}_t$'s so that this system of linear equations yields positive values for the parameters $\hat{\alpha}_t$. The following lemma (its proof is omitted) provides a sufficient condition for this.

LEMMA 4.5. *If $\hat{\xi}_t$ is increasing in t, and $\hat{\xi}_1 \geq p$, then the system of linear equations $A\hat{\alpha} = b$ has a unique positive solution.*

We have used $\hat{\xi}_t = p + 0.229(t-1)$ and have solved the corresponding systems of linear equations for integral values of p up to 137 by implementing a simple back substitution routine in C. The limitation on p comes from the range of numbers the `double` data type of C can represent. Our results suggest that, for the particular selection of the parameter vector $\hat{\alpha}$, the competitive ratio of algorithm $(\hat{\alpha}, p)$-Balance is always better than the lower bound for deterministic online algorithms and at most $1.222p$. As a corollary, using Lemma 2.1, we obtain improved competitiveness for online makespan minimization as well when the number of machines is up to an astronomically large constant. The next statement summarizes the discussion of this section.

THEOREM 4.1. *Algorithm $(\hat{\alpha}, p)$-Balance has competitive ratio at most $1.222p$ against oblivious adversaries for load balancing under the L_p norm for integral $p \leq 137$. Algorithm $(\hat{\alpha}, \lceil \ln m \rceil)$-Balance has competitive ratio at most $1.222e\lceil \ln m \rceil \leq 2.304 \log m + 3.324$ against oblivious adversaries for makespan minimization on $m \leq e^{137}$ unrelated machines.*

5 Extensions and open problems

Our load balancing algorithms for L_p norms can be adapted to work with similar competitiveness (by making the analysis slightly more complicated) in network routing when the objective is to minimize the total latency or the maximum latency per link and the delay functions on the network links are polynomial; the adaptations include approximating the Wardrop equilibrium of more general non-atomic congestion games in networks.

Concerning open problems, there are still many. First, it would be interesting to theoretically prove upper bounds on the competitiveness of algorithm Balance for any value of p. We conjecture that these bounds will be only marginally larger than those computed in Section 4. Furthermore, the question about the limits of randomization for online load balancing under the L_p norm is very interesting even for the L_2 norm. Finally, although we have made some progress on online makespan minimization on unrelated machines, the gaps between our upper bounds and the lower bounds of [9] remain. For this problem, it is even widely open whether randomization is really necessary in order to obtain the best possible competitiveness.

Our results in Section 4 suggest that game equilibria can be useful in online combinatorial optimization. We plan to investigate this relation more systematically in the future by considering different online problems.

References

[1] S. Albers. On randomized online scheduling. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC '02)*, pp. 134-143, 2002.

[2] N. Alon, Y. Azar, G. J. Woeginger and T. Yadid. Approximation schemes for scheduling. In *Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '97)*, pp. 493-500, 1997.

[3] J. Aspnes, Y. Azar, A. Fiat, S. Plotkin, and O. Waarts. On-line routing of virtual circuits with applications to load balancing and machine scheduling. *Journal of the ACM*, 44(3), pp. 486-504, 1997.

[4] A. Avidor, Y. Azar, and J. Sgall. Ancient and new algorithms for load balancing in the L_p norm. *Algorithmica*, 29, pp. 422-441, 2001.

[5] B. Awerbuch, Y. Azar, and A. Epstein. The price of routing unsplittable flow. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC '05)*, pp. 57-66, 2005.

[6] B. Awerbuch, Y. Azar, E. F. Grove, M.-Y. Kao, P. Krishnan, and J. S. Vitter. Load balancing in the L_p norm. In *Proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS '95)*, pp. 383-391, 1995.

[7] Y. Azar and A. Epstein. Convex programming for scheduling unrelated parallel machines. In *Proceedings*

of the 37th Annual ACM Symposium on Theory of Computing (STOC '05), pp. 331-337, 2005.

[8] Y. Azar, L. Epstein, Y. Richter, and G. Woeginger. All-norm approximation algorithms. *Journal of Algorithms*, 52(2), pp. 120-133, 2004.

[9] Y. Azar, J. Naor, and R. Rom. The competitiveness of on-line assignments. *Journal of Algorithms*, 18(2), pp. 221-237, 1995.

[10] Y. Bartal, A. Fiat, H. J. Karloff and R. Vohra. New algorithms for an ancient scheduling problem. *Journal of Computer and System Sciences*, 51(3), pp. 359-366, 1995.

[11] M. Beckmann, C. B. McGuire, and C. B. Winsten. *Studies in the Economics of Transportation*, Yale University Press, 1956.

[12] P. Berman, M. Charikar, and M. Karpinski. On-line load balancing for related machines. *Journal of Algorithms*, 35(1), pp. 108-121, 2000.

[13] A. Borodin and R. El-Yaniv. Online computation and competitive analysis. *Cambridge University Press*, 1998.

[14] I. Caragiannis, M. Flammini, C. Kaklamanis, P. Kanellopoulos, and L. Moscardelli. Tight bounds for selfish and greedy load balancing. In *Proceedings of the 33rd International Colloquium on Automata, Languages, and Programming (ICALP '06)*, LNCS 4051, Springer, Part I, pp. 311-322, 2006.

[15] A. K. Chandra and C. K. Wong. Worst-case analysis of a placement algorithm related to storage allocation. *SIAM Journal on Computing*, 4(3), pp. 249-263, 1975.

[16] G. Christodoulou and E. Koutsoupias. The price of anarchy of finite congestion games. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC '05)*, pp. 67-73, 2005.

[17] G. Christodoulou, V. S. Mirrokni, and A. Sidiropoulos. Convergence and approximation in potential games. In *Proceedings of the 23rd Annual Symposium on Theoretical Aspects of Computer Science (STACS '06)*, LNCS 3884, Springer, pp. 349-360, 2006.

[18] R. A. Cody and E. G. Coffman, Jr. Record allocation for minimizing expected retrieval costs on crumlike storage devices. *Journal of the ACM*, 23(1), pp. 103-115, 1976.

[19] L. Epstein and J. Sgall. Approximation schemes for scheduling on uniformly related and identical parallel machines. *Algorithmica*, 39(1), pp. 43-57, 2004.

[20] S. Fischer, H. Räcke, and B. Vöcking. Fast convergence to Wardrop equilibria by adaptive sampling methods. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC '06)*, pp. 653-662, 2006.

[21] G. Galambos and G. Woeginger. An online scheduling heuristic with better worst case ratio than graham's list scheduling. *SIAM Journal on Computing*, 22(2), pp. 349-355, 1993.

[22] M. R. Garey and D. S. Johnson. *Computers and Intractability*, W. H. Freeman and Company, 1979.

[23] R. L. Graham. Bounds for certain multiprocessor anomalies. *Bell System Technical Journal*, 45, pp. 1563-1581, 1966.

[24] D. S. Hochbaum and D. B. Shmoys. Using dual approximation algorithms for scheduling problems theoretical and practical results. *Journal of the ACM*, 34(1), pp. 144-162, 1987.

[25] D. S. Hochbaum and D. B. Shmoys. A polynomial approximation scheme for scheduling on uniform processors: using the dual approximation approach. *SIAM Journal on Computing*, 17(3), pp. 539-551, 1988.

[26] E. Horowitz and S. Sahni. Exact and approximate algorithms for scheduling non-identical processors. *Journal of the ACM*, 23, pp. 317-327, 1976.

[27] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In *Proceedings of the 16th International Symposium on Theoretical Aspects of Computer Science (STACS '99)*, LNCS 1563, Springer, pp. 404-413, 1999.

[28] V. S. Anil Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan. Approximation algorithms for scheduling on multiple machines. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS '05)*, pp. 254-263, 2005.

[29] J. K. Lenstra, D. B. Shmoys, and E. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, 46, pp. 259-271, 1990.

[30] Y. Nesterov and A. Nemiroskii. Interior-point polynomial algorithms in convex programming. *SIAM Studies in Applied Mathematics*, SIAM, 1994.

[31] N. Nissan, T. Roughgarden, E. Tardos, and V. V. Vazirani. Algorithmic game theory, *Cambridge University Press*, 2007, to appear.

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

[33] S. S. Seiden. Online randomized multiprocessor scheduling. *Algorithmica*, 28(2), 173-216, 2000.

[34] D. B. Shmoys and E. Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming*, 62, pp. 461-474, 1993.

[35] S. Suri, C. Tóth and Y. Zhou. Selfish load balancing and atomic congestion games. *Algorithmica*, 47(1), pp. 79-96, 2007.

[36] J. G. Wardrop. Some theoretical aspects of road traffic research. In *Proceedings of the Institute of Civil Engineers*, Pt. II, pp. 325-378, 1952.