Contents lists available at ScienceDirect

Operations Research Letters

www.elsevier.com/locate/orl

The efficiency-fairness balance of Round Robin scheduling

Benjamin Moseley [a], Shai Vardi [b,*]

[a] *Tepper School of Business, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213, USA*
[b] *Krannert School of Management, Purdue University, 610 Purdue Mall, West Lafayette, IN 47907, USA*

A R T I C L E I N F O

Article history:
Received 26 April 2021
Received in revised form 20 September 2021
Accepted 7 November 2021
Available online 16 November 2021

Keywords:
Scheduling
Completion time
Potential function
Lk-norms

A B S T R A C T

Round Robin is a widely used scheduling policy, used primarily because it is intuitively fair, splitting the resources evenly among the jobs. Little is known, however, of its fairness with respect to completion times for the jobs, which is typically measured using the ℓ_p-norm of the completion times of the jobs for small p.
This paper studies Round Robin's performance for the ℓ_p-norm of the completion times when scheduling n preemptive jobs on a single machine, for all integral $p \geq 1$. We show that if all jobs arrive at the same time Round Robin's approximation ratio is exactly $\sqrt[p]{p+1}$. When jobs arrive over time, we show that Round Robin's competitive ratio is at most 4 for any $p \geq 1$.

© 2021 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1. Introduction

Round Robin (aka Equipartition or Processor Sharing) is one of the most popular scheduling algorithms in environments where multiple agents or jobs share a common resource, such as processing power or bandwidth. It is used in a wide variety of environments, including operating systems [15], distributed storage systems [33], communication systems [34], and bandwidth allocation in LAN [2] and mobile networks [27]. Round Robin (RR) assigns short time slices to the agents in equal portions and in circular order. These time slices are typically very short; as such one can alternatively characterize Round Robin scheduling as splitting the resource equally among all active jobs. As such, RR is *instantaneously fair*: it assigns all active agents an equal amount of the resource at any given moment. Other than meeting this intuitive definition of fairness, RR has other advantages, such as being starvation-free and incentivizing jobs to reveal their true size to the scheduler [13,29,24,4].

While RR is instantaneously fair, scheduling systems are typically also interested in meeting some Quality of Service (QoS) goal. One of the most common QoS metrics considered in operations research and industry is minimizing the average (or total) completion times [3,18,11,31]. Unfortunately, schedules that optimize QoS are typically extremely 'unfair', leading to undesirable effects such as job starvation [32]. To illustrate this, consider a dynamic system using the shortest remaining processing time next (SRPT) algorithm, which is optimal for the average completion time metric. Initially two jobs arrive, one of size 1 (meaning it takes one time period to complete) and one of size 2. At every time period thereafter, a job of size 1 arrives. It is easy to see that the job of size 2 will never be served. Therefore, many systems aim for a compromise between QoS and (some notion of) fairness.

The most common way to compromise between QoS and fairness is to use ℓ_p-norm of the job completion, for small values of p; typically $p = 2$ or $p = 3$ [7,5,22]. WThis objective encourages minimizing the average completion time while also ensuring the completion times are balanced across the different jobs. When jobs arrive at the same time then it is folklore that the Shortest Job First (SJF) algorithm is optimal for the ℓ_p-norm of completion times for every p [19]. If jobs arrive over time, it is NP-Hard to minimize the ℓ_p-norm for any $2 \leq p < \infty$ [23]; Shortest Remaining Processing Time (SRPT) is optimal for the L_1-norm and First In First Out (FIFO) is optimal for the L_∞-norm [23]. In fact, it is not difficult to verify that any non-idle scheduler (i.e., one that processes jobs whenever at least one job is in the system) is optimal for the L_∞ norm.

While much is known about ℓ_p-norms (see the related literature), RR's performance for these objectives is poorly understood. Given that RR is one of the most widely used algorithms for ensuring fairness in practice, and that ℓ_p-norms are the most widely used criterion for guaranteeing a balance of fairness and QoS, it is important to understand how well the algorithm performs with respect to these objectives.

To our knowledge, the only prior work to study RR with respect to ℓ_p-norms is [20], who consider the resource augmentation

* Corresponding author.
E-mail addresses: moseleyb@andrew.cmu.edu (B. Moseley), svardi@purdue.edu (S. Vardi).

https://doi.org/10.1016/j.orl.2021.11.008
0167-6377/© 2021 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

model when jobs arrive over time, and focus on flow time. While they do not compute precise constants, their results imply a bound on the competitive ratio that is at least $40p^2$ for the ℓ_p-norm of completion times; i.e., that RR is roughly 160-competitive for the L_2-norm and 360-competitive for the L_3 norm.

1.1. Our results

This work studies Round Robin's performance when scheduling n preemptive jobs on a single machine with respect to the ℓ_p-norm of job completion times. When all jobs arrive concurrently, we show that RR is a $\sqrt[p]{p+1}$-approximation algorithm for the ℓ_p-norm of completion times for all integer $p \geq 1$, and that this is tight. This implies that the approximation ratio of RR is exactly $\sqrt{3} \approx 1.73$ for the L_2-norm and $\sqrt[3]{4} \approx 1.59$ for the L_3-norm.

When jobs arrive over time, we show that RR is 4-competitive for the ℓ_p-norms of completion times, for all $p \geq 1$, using a potential function argument. Our results show that RR's instantaneous fairness guarantees do not come at a large cost with respect to the ℓ_p-norms of the completion times both when jobs arrive concurrently and over time.

1.2. Overview of technical contributions

The proof for the case where all jobs arrive together uses an inductive argument on the base of the norm. To the best of our knowledge, this is a new proof technique; we are unaware of similar inductive arguments in the literature. As Shortest Job First (SJF) is optimal for the ℓ_p-norm of completion times, $p \geq 1$, it suffices to bound $\frac{\mathrm{RR}_p(x)}{\mathrm{SJF}_p(x)}$ over all possible x, where x is a vector denoting the set of jobs and their sizes and $\mathrm{SJF}_p(x)$ and $\mathrm{RR}_p(x)$ are the p-th power of the ℓ_p-norms of completion times for SJF and RR respectively for job set x. This ratio is non-convex, and difficult to optimize over. Instead, we consider the difference between $\mathrm{RR}_p(x)$ and $\mathrm{RR}_p(y)$, where y is x minus the smallest job. We show that this difference can be represented as sums of the lesser norms of RR, i.e., $\mathrm{RR}_{p'}(\cdot) : p' < p$. We then use induction on the base of the norm to show that differences in $\mathrm{SJF}_p(x)$ and $\mathrm{SJF}_p(y)$ can also be represented using the lesser norms of RR. Thus we can represent the ratio as a function of the lesser norms of RR. We then compute the value for which this function is minimized. Coupled with an example giving a matching lower bound, we obtain our tight result. An artifact of this proof technique is that the results only hold for integral values of p. We conjecture that they hold for all real-valued $p \geq 1$.

For the case where jobs arrive over time, we use a potential function proof that is a form of amortized analysis. The key of the proof is the design of the potential. Intuitively, we want an algebraic expression for the remaining cost of RR minus the cost of the optimal solution. To do so, we first write an expression that is the total remaining cost of RR's ℓ_p-norm cost, *including* jobs that have not yet arrived. This is in contrast to many previous potential analyses of scheduling algorithms, which do not include future jobs. By definition this potential has enough credit to pay for RR's remaining cost. But this potential is too large; we therefore replace each job size by the lag of the job. This is the remaining processing time of the job in the algorithm minus the remaining processing time of the job in the optimal solution. In this way, we can then relate RR to optimal algorithm though the potential function.

1.3. Related literature

The L_p-norm objective is well understood in multiple scheduling environments. When jobs arrive at the same time, SRPT (which reduces to SJF when all jobs arrive simultaneously) is known to be optimal for all L_p-norms of completion time [19]. When jobs arrive over time, the problem is NP-Hard for $2 \leq p < \infty$ and optimal algorithms exist for $p \in \{1, \infty\}$ [23]. For uniformly related multiple machines, [10] give a polynomial time approximation scheme for the L_1 norm of weighted completion times, building upon work on identical machines [1]. A long line of research [14,8,9,16] gives online algorithms that are constant competitive with a small amount of resource augmentation for minimizing the total flow time in multiple machine environments. Other metrics are popular in scheduling theory besides the norms of completion time and flow time; e.g., [26] and the references therein.

Due to its popularity as a scheduling algorithm, Round Robin's performance has been extensively studied, e.g., [13,25], including several works specifically dedicated to its performance with respect to the average completion time, e.g., [28,25,30]. We note that while it was previously known that RR is 2-competitive w.r.t. the L_1 norm (which is precisely the average completion time), we give a short proof for completeness, in the proof of Lemma 3.6.

As mentioned above, our work is most closely related to [20], which shows that RR is $O(p/\epsilon)$-competitive for $0 < \epsilon \leq 1/10$ using a processor $2p(1 + 10\epsilon)$ times faster than optimal when the objective is minimizing total flow time. It is easy to verify using standard techniques and a known connection of flow time and completion time, that their bound implies a competitive ratio of at least $40p^2$ for the L_p-norm of completion times. We note that our techniques completely differ from those of [20]; they use a dual fitting analysis, while we use a potential function to obtain our bound for jobs that arrive over time.

A complementary line of work focuses on bounding the unfairness of algorithms that perform well for total completion time, like SJF and SRPT [6,12]. These papers compare these algorithms to RR and demonstrate that these algorithms do not starve jobs much more than RR under certain conditions.

2. Preliminaries

There are n jobs, $1, \ldots, n$, to be scheduled preemptively on a single machine. Jobs arrive over time and each job i has a release date (arrival time) r_i. We denote the size of job i by x_i; job i is completed once it is processed for x_i (possibly nonconsecutive and infinitesimal) time units after its arrival. Let C_i be the completion time for job i. The objective we study is minimizing the ℓ_p-norm of the job completion times, $\left(\sum_i (C_i)^p\right)^{1/p}$, for $p \geq 1$. Note that minimizing this value is identical to minimizing $\sum_i (C_i)^p$; we will focus on the latter, ignoring the p-th root for the analyses. In Section 3, we only consider integer values of p; this is an artifact of our analysis. The results of Section 4 hold for all real-values $p \geq 1$.

Our goal is to bound the approximation ratio of Round Robin with respect to the ℓ_p-norm of the completion time, where the approximation ratio of an algorithm is the ratio between the result obtained by the algorithm and the optimal value. That is, if $OPT(x)$ is best possible result for input x and $ALG(x)$ is the value of the algorithm on input x, we say that ALG is an α-approximation algorithm if for all x, $ALG(x) \leq \alpha OPT(x)$. We say that the approximation ratio is *tight* if in addition there is an instance x such that $ALG(x) = \alpha OPT(x)$. We note that in Section 4, Round Robin is treated as an online algorithm, and the approximation ratio is given with respect to the optimal offline solution. Traditionally, the approximation ratio of online algorithms is referred to as the competitive ratio, and we adhere to this tradition.

We call jobs that have been released but are incomplete *live*. The Round Robin algorithm splits the processor evenly among all of live jobs in the system. Although in practice only one job is allocated to a (single core) processor at any time, we assume, as is standard in the literature on Round Robin scheduling, that all live jobs are allocated to the processor at the same time. More precisely, we assume that when there are m live jobs, each live job receives a $1/m$ fraction of the processing power.

3. Jobs arrive concurrently

In this section we prove the approximation ratio for the ℓ_p-norm of completion times of Round Robin when the jobs arrive concurrently. Our main result is the following.

Theorem 3.1. *When scheduling jobs that arrive concurrently on a single machine, the approximation ratio of Round Robin with respect to the ℓ_p-norm of completion times is $\sqrt[p]{p+1}$ for all integer $p \ge 1$, and this is tight.*

To prove Theorem 3.1 we first need some notation. Let $x = \{1, \dots, n\}$ be a set of jobs, sorted so that $x_1 \ge x_2 \ge \dots \ge x_n$. Denote the set of the ℓ largest jobs of x by $x^\ell = \{1, \dots, \ell\}$. The completion time of job j when SJF is run on x^ℓ is $C_{\text{SJF}}(x_j) = \sum_{i=j}^{\ell} x_i$. For any $p \ge 1$ and any $1 \le \ell \le n$, define $\text{SJF}_p\left(x^\ell\right)$ as follows:

$$\text{SJF}_p\left(x^\ell\right) = \sum_{j=1}^{\ell} C_{\text{SJF}}(x_j)^p = \sum_{j=1}^{\ell}\left(\sum_{i=j}^{\ell} x_i\right)^p. \tag{1}$$

For RR, job j waits $\min\{x_i, x_j\}$ as a result of job i being in the system. That is, job j waits $\min\{x_i, x_j\}$ time while job i is being processed. The completion time for job j is $C_{\text{RR}}(x_j) = \sum_{i=1}^{n}\min\{x_i, x_j\}$. Therefore $C_{\text{RR}}(x_j) = \sum_{i=j+1}^{\ell} x_i + jx_j$. Define $\text{RR}_p\left(x^\ell\right)$ as follows:

$$\text{RR}_p\left(x^\ell\right) = \sum_{j=1}^{\ell} C_{\text{RR}}(x_j)^p = \sum_{j=1}^{\ell}\left(\sum_{i=j+1}^{\ell} x_i + jx_j\right)^p. \tag{2}$$

Shortest Job First (SJF) is optimal for the ℓ_p-norm of completion times, $p \ge 1$. Note that $\text{RR}_p(x)$ and $\text{SJF}_p(x)$ are the p^{th} power of the objectives of RR and SJF respectively for input x. Our goal is to bound $\frac{\text{RR}_p(x)}{\text{SJF}_p(x)}$ over all possible x. We first show that $\text{SJF}_p\left(x^\ell\right) - \text{SJF}_p\left(x^{\ell-1}\right)$ and $\text{RR}_p\left(x^\ell\right) - \text{RR}_p\left(x^{\ell-1}\right)$ can be characterized precisely in terms of their lesser norms.

Lemma 3.2. *For any set of jobs $x = \{1, \dots, n\}$, any integer ℓ such that $1 \le \ell \le n$ and any integer $p \ge 1$,*

1. $\text{SJF}_p\left(x^\ell\right) - \text{SJF}_p\left(x^{\ell-1}\right) = {x_\ell}^p + \sum_{i=0}^{p-1}\binom{p}{i}{x_\ell}^{p-i}\text{SJF}_i\left(x^{\ell-1}\right)$,
2. $\text{RR}_p\left(x^\ell\right) - \text{RR}_p\left(x^{\ell-1}\right) = \ell^p {x_\ell}^p + \sum_{i=0}^{p-1}\binom{p}{i}{x_\ell}^{p-i}\text{RR}_i\left(x^{\ell-1}\right)$.

Proof. For job j, set $\psi_j = \sum_{i=j}^{\ell-1} x_i$. Note that

- $\text{SJF}_p\left(x^{\ell-1}\right) = \sum_{j=1}^{\ell-1}\left(\psi_j\right)^p$,
- $\text{SJF}_p\left(x^{\ell}\right) = \sum_{j=1}^{\ell-1}\left(\psi_j + x_\ell\right)^p + {x_\ell}^p$.

$$\text{SJF}_p\left(x^\ell\right) - \text{SJF}_p\left(x^{\ell-1}\right) = \sum_{j=1}^{\ell-1}\left(\psi_j + x_\ell\right)^p + {x_\ell}^p - \sum_{j=1}^{\ell-1}\left(\psi_j\right)^p \tag{3}$$

$$= {x_\ell}^p + \sum_{j=1}^{\ell-1}\sum_{i=0}^{p}\binom{p}{i}\left(\psi_j\right)^i {x_\ell}^{p-i} - \sum_{j=1}^{\ell-1}\left(\psi_j\right)^p \tag{4}$$

$$= {x_\ell}^p + \sum_{j=1}^{\ell-1}\sum_{i=0}^{p-1}\binom{p}{i}\left(\psi_j\right)^i {x_\ell}^{p-i}$$

$$= {x_\ell}^p + \sum_{i=0}^{p-1}\binom{p}{i}{x_\ell}^{p-i}\sum_{j=1}^{\ell-1}\left(\psi_j\right)^i \tag{5}$$

$$= {x_\ell}^p + \sum_{i=0}^{p-1}\binom{p}{i}{x_\ell}^{p-i}\text{SJF}_i\left(x^{\ell-1}\right), \tag{6}$$

where (3) and (6) are by the definition of SJF, (4) uses the binomial expansion of $\left(\psi_j + x_\ell\right)$ and (5) is by rearrangement of the summations.

Identical reasoning gives the result for RR, using $Z_j = \sum_{i:i>j} x_i + jx_j$, $\text{RR}_p\left(x^{\ell-1}\right) = \sum_{j=1}^{\ell-1}\left(Z_j\right)^p$ and $\text{RR}_p\left(x^{\ell}\right) = \sum_{j=1}^{\ell-1}\left(Z_j + x_\ell\right)^p + \left(\ell x_\ell\right)^p$. $\square$

The characterization given in Lemma 3.2 is used as a building block in the proof of Lemma 3.6, which provides the upper bound needed for Theorem 3.1. Before proving Lemma 3.6, we need several more technical results.

Lemma 3.3. *For any $\ell \ge 1$ and $k \ge 1$,*

$$\underset{x:|x|=\ell-1}{\arg\min}\ \frac{\ell^p + \sum_{i=0}^{p-1}\binom{p}{i}\text{RR}_i(x)}{1 + \sum_{i=0}^{p-1}\binom{p}{i}\frac{\text{RR}_i(x)}{i+1}} = (1, 1, \dots, 1).$$

Proof. Setting $x = \underbrace{(1, 1, \dots, 1)}_{\ell-1 \text{ times}}$, gives $\text{RR}_i(x) = (n-1)^{i+1}$. Once again, set $Z_j = \sum_{i:i>j} x_i + jx_j$. We show that

$$\frac{\ell^p + \sum_{i=0}^{p-1}\binom{p}{i}(n-1)^{i+1}}{1 + \sum_{i=0}^{p-1}\binom{p}{i}\frac{(n-1)^{i+1}}{i+1}} \ge \frac{\ell^p + \sum_{i=0}^{p-1}\binom{p}{i}\sum_{j=1}^{\ell-1}\left(Z_j\right)^i}{1 + \sum_{i=0}^{p-1}\binom{p}{i}\frac{\sum_{j=1}^{\ell-1}\left(Z_j\right)^i}{i+1}},$$

for any $x = (x_1, \dots, x_{\ell-1})$, where x is sorted non-increasing and $x_{\ell-1} \ge 1$. Note that

$$\sum_{j=1}^{\ell-1}\left(Z_j\right)^i \ge (n-1)^{i+1} \tag{7}$$

for any $i \ge 0$.

From (10), the above is equivalent to

$$\left(\ell^p + \sum_{i=0}^{p-1}\binom{p}{i}(n-1)^{i+1}\right)\left(1 + \frac{1}{p+1}\sum_{i=0}^{p-1}\binom{p+1}{i+1}\sum_{j=1}^{\ell-1}\left(Z_j\right)^i\right)$$
$$\ge \left(\ell^p + \sum_{i=0}^{p-1}\binom{p}{i}\sum_{j=1}^{\ell-1}\left(Z_j\right)^i\right)\left(1 + \frac{1}{p+1}\sum_{i=0}^{p-1}\binom{p+1}{i+1}(n-1)^{i+1}\right)$$

Multiplying by $p+1$, and simplifying using (9), the above holds iff

$$\ell^p\sum_{i=0}^{p-1}\binom{p+1}{i+1}\left(\sum_{j=1}^{\ell-1}\left(Z_j\right)^i - (n-1)^{i+1}\right)$$
$$-\quad (p+1)\sum_{i=0}^{p-1}\binom{p}{i}\left(\sum_{j=1}^{\ell-1}\left(Z_j\right)^i - (n-1)^{i+1}\right)$$
$$+\quad \sum_{i=0}^{p-1}\binom{p}{i}(n-1)^{i+1}\sum_{j=0}^{p-1}\binom{p}{j+1}\left(\sum_{j=1}^{\ell-1}\left(Z_j\right)^j\right)$$

$$- \sum_{i=0}^{p-1}\binom{p}{i+1}(n-1)^{i+1}\sum_{j=0}^{p-1}\binom{p}{j}\left(\sum_{j=1}^{\ell-1}(Z_j)^j\right) \geq 0$$

The left hand of the inequality above is equivalent to

$$\begin{aligned}
&= (\ell^{p+1}-(n-1)^{p+1})\sum_{j=0}^{p-1}\binom{p}{j+1}\left(\sum_{j=1}^{\ell-1}(Z_j)^j\right)\\
&\quad -k\sum_{j=0}^{p-1}\binom{p}{j}\left(\sum_{j=1}^{\ell-1}(Z_j)^j\right)\\
&\quad - \ell^p\left(\sum_{i=0}^{p-1}\binom{p}{i+1}(n-1)^{i+1}+\sum_{i=0}^{p-1}\binom{p}{i}(n-1)^{i+1}\right)\\
&\quad + (p+1)\sum_{i=0}^{p-1}\binom{p}{i}(n-1)^{i+1}\\
&\geq \sum_{i=0}^{p-1}\binom{p}{i}(n-1)^{i+1}\sum_{j=0}^{p-1}\binom{p}{j+1}\left(\sum_{j=1}^{\ell-1}(Z_j)^j\right)\\
&\quad -k\sum_{j=0}^{p-1}\binom{p}{j}\left(\sum_{j=1}^{\ell-1}(Z_j)^j\right)\\
&\quad - (\ell^p-k-1)\left(\sum_{i=0}^{p-1}\binom{p}{i}(n-1)^{i+1}\right) \qquad (8)\\
&= \sum_{i=0}^{p-1}\binom{p}{i}(n-1)^{i+1}\left(\sum_{j=0}^{p-1}\binom{p}{j+1}\left(\sum_{j=1}^{\ell-1}(Z_j)^j\right)-\ell^p+p+1\right)\\
&\quad - k\sum_{j=0}^{p-1}\binom{p}{j}\left(\sum_{j=1}^{\ell-1}(Z_j)^j\right)\\
&= \sum_{i=0}^{p-1}\binom{p}{i}(n-1)^{i+1}\left(\sum_{j=0}^{p-1}\binom{p}{j+1}\left(\sum_{j=1}^{\ell-1}(Z_j)^j\right)\right.\\
&\quad \left.-\sum_{i=0}^{p-1}\binom{p}{i+1}(n-1)^{i+1}\right)\\
&\quad - k\left(\sum_{j=0}^{p-1}\binom{p}{j}\left(\sum_{j=1}^{\ell-1}(Z_j)^j\right)-\sum_{i=0}^{p-1}\binom{p}{i}(n-1)^{i+1}\right)\\
&\geq 0,
\end{aligned}$$

where (8) is due to (7). □

Lemma 3.4. *The following hold for any $k, j, \ell \geq 0$:*

$$\binom{p+1}{j+1} = \binom{p}{j+1}+\binom{p}{j}, \qquad (9)$$

$$\frac{1}{j+1}\binom{p}{j} = \frac{1}{p+1}\binom{p+1}{j+1}, \qquad (10)$$

$$\ell^p + \sum_{i=0}^{p-1}\binom{p}{i}(\ell-1)^{i+1} = \ell^{p+1}-(\ell-1)^{p+1}, \qquad (11)$$

$$\sum_{i=1}^{p}\binom{p+1}{i}(\ell-1)^i = \ell^{p+1}-(\ell-1)^{p+1}-1. \qquad (12)$$

Proof. Equation (9) is the standard binomial coefficient equality; Equation (10) is straightforward to verify; Equations (11) and (12) follow from the equality $\ell^p = (\ell-1+1)^p = \sum_{i=0}^{p}\binom{p}{i}(\ell-1)^i$. For (11), note that $\sum_{i=0}^{p-1}\binom{p}{i}(\ell-1)^{i+1} = (\ell-1)\sum_{i=0}^{p-1}\binom{p}{i}(\ell-1)^i = (\ell-1)\left(\ell^p-(\ell-1)^p\right)$.

For (12), $\sum_{i=1}^{p}\binom{p+1}{i}(\ell-1)^i = \sum_{i=0}^{p+1}\binom{p+1}{i}(\ell-1)^i-(\ell-1)^{p+1}-(\ell-1)^0 = (\ell-1)^p-(\ell-1)^{p+1}-1$. □

Lemma 3.5. *For any $\ell \geq 1$, integral $p \geq 1$ and x such that $|x| = \ell-1$,*

$$\frac{\ell^p+\sum_{i=0}^{p-1}\binom{p}{i}\text{RR}_i(x)}{1+\sum_{i=0}^{p-1}\binom{p}{i}\frac{\text{RR}_i(x)}{i+1}} \leq p+1.$$

Proof. We have that

$$\begin{aligned}
\frac{\ell^p+\sum_{i=0}^{p-1}\binom{p}{i}\text{RR}_i\left(x^{\ell-1}\right)}{1+\sum_{i=0}^{p-1}\binom{p}{i}\frac{\text{RR}_i\left(x^{\ell-1}\right)}{i+1}} &\leq \frac{\ell^p+\sum_{i=0}^{p-1}\binom{p}{i}(n-1)^{i+1}}{1+\sum_{i=0}^{p-1}\binom{p}{i}\frac{(n-1)^{i+1}}{i+1}} \qquad (13)\\
&= \frac{\ell^p+\sum_{i=0}^{p-1}\binom{p}{i}(n-1)^{i+1}}{1+\frac{1}{p+1}\sum_{i=0}^{p-1}\binom{p+1}{i+1}(n-1)^{i+1}} \qquad (14)\\
&= \frac{\ell^{p+1}-(n-1)^{p+1}}{1+\frac{1}{p+1}\left(\ell^{p+1}-(n-1)^{p+1}-1\right)} \qquad (15)\\
&\leq p+1,
\end{aligned}$$

where (13) is from Lemma 3.3, (14) is due to (10) and (15) is due to (11) and (12) in Lemma 3.4. □

We are now ready to prove the upper bound required for Theorem 3.1.

Lemma 3.6. *For any x, and any integral $p \geq 1$,*

$$\text{RR}_p(x) \leq (p+1)\text{SJF}_p(x).$$

Proof. The proof is by induction on p.

Base case. Let $p = 1$ and let x be any set of ℓ jobs, with $x_1 \geq x_2 \geq \ldots \geq x_\ell$. From (1),

$$\text{SJF}_1(x) = \sum_{j=1}^{\ell}\sum_{i:i\geq j} x_i = \sum_{j=1}^{\ell} jx_j.$$

Therefore, using (2), $\text{RR}_1(x) = \sum_{j=1}^{\ell}\left(\sum_{i:i>j}x_i + jx_j\right) = \sum_{j=1}^{\ell}\sum_{i:i>j}x_i + \sum_{j=1}^{\ell} jx_j \leq 2\sum_{j=1}^{\ell} jx_j = 2\text{SJF}_1(x)$.

Inductive step. Let $x = \{1,\ldots,n\}$ be an arbitrary set of jobs and recall that the set of the ℓ largest jobs of x is denoted by $x^\ell = \{x_1,\ldots,x_\ell\}$. We bound $\text{SJF}_p\left(x^\ell\right)-\text{SJF}_p\left(x^{\ell-1}\right)$ as follows:

$$\begin{aligned}
\text{SJF}_p\left(x^\ell\right)-\text{SJF}_p\left(x^{\ell-1}\right) &= {x_\ell}^p + \sum_{i=0}^{p-1}\binom{p}{i}{x_\ell}^{p-i}\text{SJF}_i\left(x^{\ell-1}\right) \qquad (16)\\
&\geq {x_\ell}^p + \sum_{i=0}^{p-1}\binom{p}{i}{x_\ell}^{p-i}\frac{\text{RR}_i\left(x^{\ell-1}\right)}{i+1}, \qquad (17)
\end{aligned}$$

where (16) is due to Lemma 3.2 and (17) uses the inductive hypothesis. As both SJF and RR scale identically when the job sizes are multiplied by a constant for any p, we can set $x_\ell = 1$ without loss of generality. We now bound the following ratio.

$$\frac{\mathrm{RR}_p\left(x^{\ell}\right)-\mathrm{RR}_p\left(x^{\ell-1}\right)}{\mathrm{SJF}_p\left(x^{\ell}\right)-\mathrm{SJF}_p\left(x^{\ell-1}\right)} \le \frac{\ell^p+\sum_{i=0}^{p-1}\binom{p}{i}\mathrm{RR}_i\left(x^{\ell-1}\right)}{1+\sum_{i=0}^{p-1}\binom{p}{i}\frac{\mathrm{RR}_i\left(x^{\ell-1}\right)}{i+1}} \le p+1, \quad (18)$$

where the first inequality is due to the second part of Lemma 3.2 and (17) and the second inequality is due to Lemma 3.5.

We bound the ratio $\frac{\mathrm{RR}_p(x)}{\mathrm{SJF}_p(x)}$ as follows.

$$\begin{aligned}\frac{\mathrm{RR}_p\left(x\right)}{\mathrm{SJF}_p\left(x\right)} &= \frac{\mathrm{RR}_p\left(x^1\right)+\sum_{\ell=2}^{n}\left(\mathrm{RR}_p\left(x^{\ell}\right)-\mathrm{RR}_p\left(x^{\ell-1}\right)\right)}{\mathrm{SJF}_p\left(x^1\right)+\sum_{\ell=2}^{n}\left(\mathrm{SJF}_p\left(x^{\ell}\right)-\mathrm{SJF}_p\left(x^{\ell-1}\right)\right)}\\ &\le \max_{2\le\ell\le n}\left\{\frac{\mathrm{RR}_p\left(x^{\ell}\right)-\mathrm{RR}_p\left(x^{\ell-1}\right)}{\mathrm{SJF}_p\left(x^{\ell}\right)-\mathrm{SJF}_p\left(x^{\ell-1}\right)}\right\} \quad (19)\\ &\le p+1,\end{aligned}$$

where (19) is because $\frac{\mathrm{RR}_p(x^1)}{\mathrm{SJF}_p(x^1)}=1$, and $\frac{\mathrm{RR}_p(x)}{\mathrm{SJF}_p(x)}\ge 1$ for any x and any $p\ge 1$; the last inequality follows from (18). □

We now give a lower bound that shows that the result of Lemma 3.6 is tight.

Lemma 3.7. *For any integral $p \ge 1$ and any $\epsilon > 0$ there exists a set of jobs x such that $\mathrm{RR}_p(x) \ge (p+1)\mathrm{SJF}_p(x) - \epsilon$.*

Proof. The bound is obtained from the following simple example: let there be n identical jobs: $x = \{1,2,\ldots,n\}, x_1 = x_2 = \cdots = x_n = 1$. Then, $\mathrm{RR}_p(x) = \sum_{j=1}^{n} n^p = n^{p+1}$ and $\mathrm{SJF}_p(x) = \sum_{j=1}^{n} j^p$.

From Faulhaber's formula, $\sum_{j=1}^{n} j^p = \frac{n^{p+1}}{p+1} + O(n^p)$. Taking $n \to \infty$ completes the proof. □

Theorem 3.1 follows from Lemmas 3.6 and 3.7.

4. Jobs arrive over time

In this section, we bound the L_p-norms of the completion times when jobs arrive over time. We prove the following theorem.

Theorem 4.1. *Round Robin is 4-competitive for the L_p-norm of completion times when scheduling jobs that arrive over time on a single machine, for all integer $p \ge 1$.*

4.1. Slowing down the optimal scheduler

When bounding the cost of RR, we assume that the optimal scheduler's processor is slowed down by a factor of s for $s \ge 1$. In particular, we will assume that RR has a processor of speed s and the optimal scheduler is given a processor of speed one. (Note that this is the same as increasing the size of each job in the optimal scheduler by a factor s.) It is known that this adds a factor of at most s to the approximation ratio. For instance, this easily follows by extending a lemma in [17]. We state the result for completeness.

Lemma 4.2 *([17]). For any $p \ge 1$, the following holds. Let OPT_σ denote the value of the L_p-norm of the completion times of the optimal algorithm where the optimal algorithm can process σ jobs in each time step. Then for any $s \ge 1$, $\mathrm{OPT}_1 \le s\mathrm{OPT}_s$.*

4.2. The potential function framework

Fix $p \ge 1$. For any scheduling algorithm $\mathcal{A}$, we call the L_p-norm of the completion times $\mathcal{A}$'s *cost*. Let OPT denote both the optimal scheduler as well as its cost, where the meaning will be clear from context.

Throughout this section, we will be focusing on bounded the pth power of the algorithm's objective by the pth power of the optimal objective. Call this the pth power cost. Let OPT^p be the optimal objective cost to the pth power. We focus on bounding the derivative of RR's p-power cost at each point in time. Let $R(t)$ be jobs that are still incomplete at time t under RR including those that have yet to arrive. Let $\frac{dRR}{dt} = \sum_{i\in R(t)} p\cdot t^{p-1}$ be the derivative in RR's p-power cost at time t; each job that is incomplete contributes $p\cdot t^{p-1}$ to the objective at time t. Notice that $\int_{t=0}^{\infty}\frac{dRR}{dt}dt = \int_{t=0}^{\infty}\sum_{i\in R(t)} p\cdot t^{p-1}dt = \sum_{i\in[n]}(C_i)^p$ is RR's pth power cost. Here C_i is the completion time of i in the RR's schedule. We define an analogous quantity for OPT. Let $O(t)$ be the set of jobs that are incomplete in OPT at time t. Let $\frac{dO}{dt} = \sum_{i\in O(t)} p\cdot t^{p-1}$ be the derivative in OPT's pth power cost at time t.

We will define a potential function $\Phi(t)$ that has the following properties. For details on potential function analysis in scheduling see [21].

1. $\Phi(0) = \Phi(\infty) = 0$.
2. The function $\Phi(t)$ may be discontinuous, but at any such discontinuity the function does not increase. At all other times the function is continuous and differentiable.
3. There exists a $c \ge 1$ such that for all continuous differentiable changes in Φ it is the case that $\frac{dRR}{dt} + \frac{d\Phi(t)}{dt} \le c\frac{dO}{dt}$ for every time t.

Let I be a collection of disjoint intervals in $[0,\infty]$ such that the function is continuous and differentiable at all times in any interval $I_i = (b_i, e_i) \in I$. Moreover, let T be the collection of time steps where the potential function is non-differentiable. Together T and I are required to partition all time. Let D be the total summation of the changes in the potential at times T where the potential is non-differentiable.

This will allow us to bound RR's pth power cost as follows. Notice the first term is RR's pth power cost.

$$\int_{t=0}^{\infty}\frac{dRR}{dt}dt = \sum_{I_i\in I}\int_{t=b_i}^{e_i}\frac{dRR}{dt}dt \quad (20)$$

$$= D + \sum_{I_i\in I}\int_{t=b_i}^{e_i}\left(\frac{d\Phi(t)}{dt}+\frac{dRR}{dt}\right)dt \quad (21)$$

$$\le \sum_{I_i\in I}\int_{t=b_i}^{e_i} c\frac{dO}{dt}dt = \int_{t=0}^{\infty} c\frac{dO}{dt}dt = c\cdot\mathrm{OPT}^p. \quad (22)$$

Note that (21) follows because $\Phi(0) = \Phi(\infty) = 0$. Inequality (22) follows from the second and third properties of the potential function. In particular $D \le 0$ and $\frac{dRR}{dt} + \frac{\Phi(t)}{dt} \le c\frac{dO}{dt}$ for every time t. The integration is only over time steps where the function in continuous. The function maybe discontinuous, but property (2) ensures that this will not increase the potential. Thus, the above properties of the potential function therefore allow us to bound RR's pth power cost by OPT's.

4.3. Potential function definition and analysis

Let $x_i(t)$ and $x_i^O(t)$ be the remaining processing time of job i in RR and OPT at time t, respectively, let a_t and a_t^O be the number of live jobs (recall that a job is *live* if it has been released but is as yet incomplete), at time t in RR and OPT respectively, and let n_t and n_t^O be the number of live jobs at time t as well as those that have yet to arrive in RR and OPT respectively. Note that these are the jobs that contribute to the objective at time t. Let w_t be the number of jobs that have yet to arrive at time t.

Let $z_i(t) = \max\{x_i(t) - x_i^O(t), 0\}$ be the lag of job i. This is 'how far behind' RR is compared to OPT on job i. Let $R(t)$ and $O(t)$ be the set of jobs that have yet to be completed at time t under RR and OPT respectively. This includes jobs that have yet to arrive and by definition $|R(t)| = n_t$ and $|O(t)| = n_t^O$.

We now define the potential. Let $c > 0$ be a constant and $s \geq 1$ be the speed reduction of OPT's processor.

$$\Phi(t) := \sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^p - \sum_{i\in R(t)} t^p. \tag{23}$$

Intuition Behind the Potential: Consider the first term in the potential. This is an estimate of the remaining pth power cost of RR over OPT. Each $z_j(t)$ is how far RR has fallen behind the optimal on job j. The value of $c\sum_{j\in R(t)} z_j(t)$ is an estimate of the remaining work that RR will need to do before completing a job i, which the optimal does not have to do in the future. Intuitively, this is the extra time a job will wait in RR versus the optimal.

At time t, a job in $R(t)$ must have completion time greater than t. Thus, such a job has already contributed t^p to the objective. The potential seeks to capture its remaining pth power cost that OPT will not pay, but RR will. This is estimated as $\left(t + c\sum_{j\in R(t)} z_j(t)\right)^p - t^p$. We sum over all jobs to get an estimate on the total pth power cost that RR may need to pay that the optimal will not.

4.4. Bounding the change in the potential

We begin by considering non-continuous changes in the potential.

Lemma 4.3. *For non-continuous changes, which can only occur at the arrival and completion of jobs, the change in potential is non-positive.*

Proof. Every job has a corresponding term in all three summations of (23) at time 0 by definition of $R(t)$. The potential at time 0 is $\Phi(0) = 0$. This is because at $t = 0$, $z_i(t) = 0$ for all jobs i as no processing has been done by either algorithm.

When OPT completes a job it does not change the potential. Consider when RR completes job i at time t: $z_i(t) = 0$ at this time. Thus, in the summation in first term for job $j \neq i$, the inner summation does not change due to removing job i. When RR completes job i, terms are removed from the potential function for i both in the first outer summation and the second. The terms removed are $\left(t + c\sum_{j\in R(t)} z_j(t)\right)^p - t^p \geq 0$.

The inequality follows from the fact that $z_j(t) \geq 0$ for all jobs. As the terms removed are positive, the potential decreases. $\square$

We look at continuous changes in the potential at points where the potential is differentiable. We compute the partial derivatives with respect to the change of time, OPT processing a job and RR processing a job. OPT chooses one job and works on it with unit speed. In the worst case, this is a job j in $R(t)$ and $z_j(t)$ increases at the rate of a unit. This gives the following.

Lemma 4.4. *At any time t, the partial derivative in the potential due to OPT processing a job is at most $cp\sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^{p-1}$.*

We now bound the change in the potential due to the change in time. Time changes both terms of Φ and there is no c in front of the first term. The second term in the change is exactly the increase in the algorithm's objective at time t.

Lemma 4.5. *At any time t, the partial derivative in the potential due to the change in time is at most $p\sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^{p-1} - p\sum_{i\in R(t)} t^{p-1}$.*

Next we bound potential change due to RR processing jobs.

Lemma 4.6. *At any time t, the partial derivative in the potential due to RR processing jobs is at most $-cp\left(\sum_{i\in R(t)\setminus O(t)} \frac{s}{a_t}\right) \times \sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^{p-1}$.*

Proof. Each $z_j(t)$ decreases at a rate of $\frac{s}{a_t}$ if $j \in R(t)\setminus O(t)$. These are jobs that have arrived and OPT has completed so $z_i(t)$ is positive. RR processes each job at a rate of $\frac{s}{a_t}$. It maybe the case that $z_j(t)$ decreases for a $j \in R(t)\cap O(t)$ that RR processes, but this will only decrease the potential more; we can therefore ignore this change in the potential. Thus the change is upper bounded by the expression in the lemma statement. $\square$

Recall that $\frac{dRR}{dt}$ and $\frac{dO}{dt}$ are the increases in RR's and OPT's cost at time t respectively. We are now ready to bound the total change in the potential.

Lemma 4.7. *Assuming that $(s-1)c \geq 1$ the total derivative in the potential summing over all continuous changes is at most $-\frac{dRR}{dt} + (1+c)^{p-1} cs\frac{dO}{dt}$.*

Proof. Combining Lemmas 4.4, 4.5 and 4.6, we can bound the total change in potential as follows. Let $L(t)$ be the jobs that have yet to arrive at time t and $S(t) := (R(t)\cap O(t))\setminus L(t)$ be the jobs that have arrived and are incomplete for both RR and OPT at t.

$$\begin{aligned}
& cp\sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^{p-1} + p\sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^{p-1} \\
& - p\sum_{i\in R(t)} t^{p-1} - cp\left(\sum_{i\in R(t)\setminus O(t)} \frac{s}{a_t}\right) \sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^{p-1} \\
& = (c+1)p\sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^{p-1} - \frac{dRR}{dt} \\
& \quad - cp\left(\sum_{i\in R(t)\setminus O(t)} \frac{s}{a_t}\right) \sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^{p-1} \\
& = (c+1)p\sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^{p-1} - \frac{dRR}{dt} \\
& \quad - cp\left(\sum_{i\in R(t)\setminus L(t)} \frac{s}{a_t} - \sum_{i\in S(t)} \frac{s}{a_t}\right) \sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^{p-1} \\
& = (c+1)p\sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^{p-1} - \frac{dRR}{dt} \\
& \quad - cp\left(s - \sum_{i\in S(t)} \frac{s}{a_t}\right) \sum_{i\in R(t)} \left(t + c\sum_{j\in R(t)} z_j(t)\right)^{p-1}.
\end{aligned} \tag{24}$$

We now use the assumption that $(s-1)c \geq 1$, specifically that $cps \geq (c+1)p$. As $\sum_{i\in R(t)}\left(t+c\sum_{j\in R(t)} z_j(t)\right)^{p-1} > 0$, (24) is at most: $-\frac{dRR}{dt}+cp\left(\sum_{i\in S(t)}\frac{s}{a_t}\right)\sum_{i\in R(t)}\left(t+c\sum_{j\in R(t)} z_j(t)\right)^{p-1}$

$\sum_{j\in R(t)} z_j(t) \leq t$ because this is at most the total work that RR fell behind OPT and OPT has a unit processor. Thus it can process at most t work during $[0,t]$. Thus, the prior term is upper bounded by the following. Recall that w_t is the number of jobs yet to arrive.

$$\begin{aligned}
&-\frac{dRR}{dt}+cp\left(\sum_{i\in S(t)}\frac{s}{a_t}\right)\sum_{i\in R(t)}((1+c)t)^{p-1}\\
&= -\frac{dRR}{dt}+(1+c)^{p-1}cp\left(\sum_{i\in S(t)}\frac{s}{a_t}\right)\sum_{i\in R(t)}t^{p-1}\\
&= -\frac{dRR}{dt}+(1+c)^{p-1}cpt^{p-1}\left(\sum_{i\in S(t)}\frac{s}{a_t}\right)n_t\\
&= -\frac{dRR}{dt}+(1+c)^{p-1}cpt^{p-1}\left(\sum_{i\in S(t)}\frac{s}{a_t}\right)(w_t+a_t) \qquad (25)\\
&= -\frac{dRR}{dt}+(1+c)^{p-1}cpt^{p-1}s\left(\frac{a_t^{O\cap RR}}{a_t}\right)(w_t+a_t)\\
&= -\frac{dRR}{dt}+(1+c)^{p-1}cpt^{p-1}s\left(\frac{a_t^{O\cap RR}w_t}{a_t}+\frac{a_t^{O\cap RR}a_t}{a_t}\right)\\
&\leq -\frac{dRR}{dt}+(1+c)^{p-1}cpt^{p-1}s\left(w_t+a_t^{O\cap RR}\right)\\
&\leq -\frac{dRR}{dt}+(1+c)^{p-1}cpt^{p-1}sn_t^{O}\\
&= -\frac{dRR}{dt}+(1+c)^{p-1}cs\frac{dO}{dt},
\end{aligned}$$

where (25) is because $n_t = w_t + a_t$ by definition. □

We now have all the building blocks for the proof of Theorem 4.1.

Proof of Theorem 4.1. The potential function is clearly 0 at times 0 and ∞. Lemmas 4.3 and 4.7 complete the proof of the potential function framework. Overall this proves that RR with processing speed s times faster than OPT is $\sqrt[p]{cs(1+c)^{p-1}}$-competitive for the L_p-norm, $p\geq 1$. Finally, using Lemma 4.2, we conclude that RR is $s\sqrt[p]{cs(1+c)^{p-1}}$-competitive when given the same speed as OPT. Setting $s=2$ and $c=1$ gives that RR is $2\sqrt[p]{2\cdot(2)^{p-1}}=4$-competitive. □

References

[1] Foto N. Afrati, Evripidis Bampis, Chandra Chekuri, David R. Karger, Claire Kenyon, Sanjeev Khanna, Ioannis Milis, Maurice Queyranne, Martin Skutella, Clifford Stein, Maxim Sviridenko, Approximation schemes for minimizing average weighted completion time with release dates, in: 40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA, 1999, pp. 32–44.

[2] Eitan Altman, Nahum Shimkin, Individual equilibrium and learning in processor sharing systems, Oper. Res. 46 (6) (1998) 776–784.

[3] Edward J. Anderson, Chris N. Potts, Online scheduling of a single machine to minimize total weighted completion time, Math. Oper. Res. 29 (3) (2004) 686–697.

[4] Arash Asadi, Vincenzo Mancuso, A survey on opportunistic scheduling in wireless communications, IEEE Commun. Surv. Tutor. 15 (4) (2013) 1671–1688.

[5] Adi Avidor, Yossi Azar, Jirí Sgall, Ancient and new algorithms for load balancing in the l_p norm, in: Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, 25-27 January 1998, San Francisco, California, USA, 1998, pp. 426–435.

[6] Nikhil Bansal, Mor Harchol-Balter, Analysis of SRPT scheduling: investigating unfairness, in: Proceedings of the Joint International Conference on Measurements and Modeling of Computer Systems, SIGMETRICS/Performance 2001, June 16–20, 2001, Cambridge, MA, USA, 2001, pp. 279–290.

[7] Nikhil Bansal, Kirk Pruhs, Server scheduling in the weighted l_p norm, in: LATIN 2004: Theoretical Informatics, 6th Latin American Symposium, Buenos Aires, Argentina, April 5-8, 2004, in: Proceedings, 2004, pp. 434–443.

[8] Carl Bussema, Eric Torng, Greedy multiprocessor server scheduling, Oper. Res. Lett. 34 (4) (2006) 451–458.

[9] Chandra Chekuri, Ashish Goel, Sanjeev Khanna, Amit Kumar, Multi-processor scheduling to minimize flow time with epsilon resource augmentation, in: Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13–16, 2004, 2004, pp. 363–372.

[10] Chandra Chekuri, Sanjeev Khanna, A PTAS for minimizing weighted completion time on uniformly related machines, in: Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8–12, 2001, in: Proceedings, 2001, pp. 848–861.

[11] C. Chekuri, R. Motwani, B. Natarajan, C. Stein, Approximation techniques for average completion time scheduling, SIAM J. Comput. 31 (2001) 146–166.

[12] Mark Crovella, Robert Frangioso, Mor Harchol-Balter, Connection scheduling in web servers, in: 2nd USENIX Symposium on Internet Technologies and Systems, USITS'99, Boulder, Colorado, USA, October 11-14, 1999, 1999.

[13] A. Demers, S. Keshav, S. Shenker, Analysis and simulation of a fair queueing algorithm, SIGCOMM Comput. Commun. Rev. 19 (4) (August 1989) 1–12.

[14] Jeff Edmonds, Sungjin Im, Benjamin Moseley, Online scalable scheduling for the L_k-norms of flow time without conservation of work, in: Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011, 2011, pp. 109–119.

[15] H. Christian Gromoll, Philippe Robert, Bert Zwart, Fluid limits for processor-sharing queues with impatience, Math. Oper. Res. 33 (2) (2008) 375–402.

[16] Anupam Gupta, Ravishankar Krishnaswamy, Kirk Pruhs, Online primal-dual for non-linear optimization with applications to speed scaling, in: Approximation and Online Algorithms - 10th International Workshop, WAOA 2012, Ljubljana, Slovenia, September 13-14, 2012, in: Revised Selected Papers, 2012, pp. 173–186.

[17] Varun Gupta, Benjamin Moseley, Marc Uetz, Qiaomin Xie, Stochastic online scheduling on unrelated machines, in: Integer Programming and Combinatorial Optimization - 19th International Conference, IPCO 2017, Waterloo, oN, Canada, June 26-28, 2017, in: Proceedings, 2017, pp. 228–240.

[18] Leslie A. Hall, Andreas S. Schulz, David B. Shmoys, Joel Wein, Scheduling to minimize average completion time: off-line and on-line approximation algorithms, Math. Oper. Res. 22 (3) (1997) 513–544.

[19] Wiebke Höhn, Tobias Jacobs, On the performance of Smith's rule in single-machine scheduling with nonlinear cost, ACM Trans. Algorithms 11 (4) (April 2015) 25:1–25:30.

[20] Sungjin Im, Janardhan KuLkarni, Benjamin Moseley, Temporal fairness of round robin: competitive analysis for L_k-norms of flow time, in: Proceedings of the 27th ACM on Symposium on Parallelism in Algorithms and Architectures, SPAA 2015, Portland, OR, USA, June 13–15, 2015, 2015, pp. 155–160.

[21] Sungjin Im, Benjamin Moseley, Kirk Pruhs, A tutorial on amortized local competitiveness in online scheduling, SIGACT News 42 (2) (2011) 83–97.

[22] Michael M. Kostreva, Włodzimierz Ogryczak, Adam Wierzbicki, Equitable aggregations and multiple criteria analysis, Eur. J. Oper. Res. 158 (2) (2004) 362–377.

[23] Benjamin Moseley, Kirk Pruhs, Cliff Stein, The complexity of scheduling for p-norms of flow and stretch - (extended abstract), in: Integer Programming and Combinatorial Optimization - 16th International Conference, IPCO 2013, Valparaíso, Chile, March 18–20, 2013, in: Proceedings, 2013, pp. 278–289.

[24] Hervé Moulin, Richard Stong, Fair queuing and other probabilistic allocation methods, Math. Oper. Res. 27 (1) (2002) 1–30.

[25] T.M. O'Donovan, Direct solutions of M/G/1 processor-sharing models, Oper. Res. 22 (6) (1974) 1232–1235.

[26] Kirk Pruhs, Jirí Sgall, Eric Torng, Online scheduling, in: Joseph Y.-T. Leung (Ed.), Handbook of Scheduling - Algorithms, Models, and Performance Analysis, Chapman and Hall/CRC, 2004.

[27] Pablo Rodriguez, Rajiv Chakravorty, Julian Chesterfield, Ian Pratt, Suman Banerjee Mar, A commuter router infrastructure for the mobile internet, in: Proceedings of the Second International Conference on Mobile Systems, Applications, and Services (MobiSys 2004), 01 2004.

[28] M. Sakata, S. Noguchi, J. Oizumi, An analysis of the M/G/1 queue under round-robin scheduling, Oper. Res. 19 (2) (April 1971) 371–385.

[29] M. Shreedhar, George Varghese, Efficient fair queueing using deficit round-robin, IEEE/ACM Trans. Netw. 4 (3) (1996) 375–385.

[30] Abraham Silberschatz, Peter Galvin, Greg Gagne, Applied Operating System Concepts, 1st edition, John Wiley & Sons, Inc., New York, NY, USA, 2001.

[31] M. Skutella, G.J. Woeginger, A PTAS for minimizing the total weighted completion time on identical parallel machines, Math. Oper. Res. 25 (2000) 63–75.

[32] Andrew S. Tanenbaum, Operating Systems: Design and Implementation, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1987.

[33] Yin Wang, Arif Merchant, Proportional-share scheduling for distributed storage systems, in: Proceedings of the 5th USENIX Conference on File and Storage Technologies, FAST '07, USENIX Association, USA, 2007, p. 4.

[34] Jiheng Zhang, J.G. Dai, Bert Zwart, Law of large number limits of limited processor-sharing queues, Math. Oper. Res. 34 (4) (2009) 937–970.