APPROXIMATION RATIO OF LD ALGORITHM FOR MULTI-PROCESSOR SCHEDULING AND THE COFFMAN–SETHI CONJECTURE

PERUVEMBA SUNDARAM RAVI, LEVENT TUNÇEL

Abstract. Coffman and Sethi proposed a heuristic algorithm, called LD (Longest Decreasing), for multi-processor scheduling, to minimize makespan over flowtime-optimal schedules. The LD algorithm is an extension of a very well-known list scheduling algorithm, Longest Processing Time (LPT) list scheduling, to this bicriteria scheduling problem. Coffman and Sethi conjectured (in 1976) that the LD algorithm has the following precise worst-case performance bound: $\frac{5}{4} - \frac{3}{4(4m-1)}$, where m is the number of machines. In this paper, utilizing some recent work by the authors and Huang (2016), which exposed some very strong combinatorial properties of various presumed minimal counterexamples to the conjecture, we provide a proof of this conjecture. The problem and the LD algorithm have connections to some other fundamental problems (such as the assembly line-balancing problem) and algorithms.

1. Introduction

The most fundamental machine environment in multiprocessor scheduling problems is a *parallel identical machine model*. In this basic set-up, we have m parallel identical machines and n independent jobs, all simultaneously available at time zero, indexed by $1, 2, \ldots, n$ with given processing times $p_1, p_2, \ldots, p_n$. No pre-emption is allowed, and the machines are assumed to be completely reliable. For a scheduling problem environment described above, with data $m, p_1, p_2, \ldots, p_n$, there are two performance criteria that immediately come to mind:

- minimize the completion time of the last job (i.e., *makespan*),
- minimize the total (or equivalently the average) time that the jobs spend in the system (i.e., total or average *flowtime*).

Given a feasible schedule, let C_j denote the completion time of job j in that schedule. By denoting $C_{\max} := \max_{j\in\{1,2,\ldots,n\}} \{C_j\}$, our two criteria are: minimize $C_{\max}$, minimize $\sum_{j=1}^{n} C_j$. Both of these objective functions are easily justifiable. Indeed, the minimization of makespan may ensure optimal utilization of resources (i.e., machines) as well as ensuring the earliest possible start times for other tasks that require the completion of all the jobs $1, 2, \ldots, n$ to be started. Minimization of total flow time $F := \sum_{j=1}^{n} C_j$, minimizes the amount of time the jobs spend in the system (in our setting this is C_j for each job j). Thus, minimizing F

Date: December 9, 2019.

Key words and phrases. parallel identical machines, makespan, total completion time, total flowtime, approximation algorithms, multi-processor scheduling, bicriteria scheduling problems.

Peruvemba Sundaram Ravi: Operations and Decision Sciences, School of Business and Economics, Wilfrid Laurier University, Waterloo, Ontario N2L 3C5, Canada (e-mail: pravi@wlu.ca)

Levent Tunçel: Department of Combinatorics and Optimization, Faculty of Mathematics, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada (e-mail: ltuncel@uwaterloo.ca).

equivalently minimizes, in many applications, work-in-process inventory. A feasible schedule is called *flowtime-optimal* if it minimizes F. In this paper, we consider the bicriteria optimization problem of minimizing makespan among all flowtime-optimal schedules. In scheduling theory notation, let F^* denote the optimal objective function value of $P\ /\ \ /\sum C_j$. Then, our bicriteria optimization problem is: $P\ /\ \ /\ (C_{\max}; \sum C_j = F^*)$, which we call *Flowtime-Makespan (FM)* problem.

There are two single objective function scheduling problems that make up our bicriteria optimization problem: $P\ /\ \ /\ C_{\max}$, and $P\ /\ \ /\sum C_j$. The second problem is as easy as sorting and, as a result, admits algorithms with $O(n\log(n))$ complexity. Moreover, we have a complete characterization of all optimal solutions of $P\ /\ \ /\sum C_j$. The mathematical foundations of such characterizations go back to an inequality (and characterization of when it is tight) due to Hardy, Littlewood and Pólya [12]. Related to this fundamental result, Conway, Maxwell and Miller [4], in their seminal book, develop the notion of *rank* for the FM problem.

Without loss of generality, we may assume that m divides n (if m does not divide n, we can add $(m\lceil n/m\rceil - n)$ dummy jobs with zero processing times). We may further assume that the jobs are indexed in nonincreasing order of processing times.

Definition 1. The number of ranks *for an FM instance is defined by* $k := n/m$*, and the set of jobs belonging to* rank r *are the following:* $(r-1)m+1, (r-1)m+2, \ldots, (r-1)m+m$.

A feasible schedule in which all rank $(r+1)$ jobs are started before all rank r jobs (where $r \in \{1, 2, \ldots, (n/m)-1\}$) is said to satisfy the *rank restriction* or *rank constraint*. A feasible schedule without idle time and satisfying the rank constraint, and in which all rank n/m jobs start at time zero, is a *flowtime-optimal schedule*. Since within each rank the assignment of jobs to machines can be arbitrary, it immediately follows that there are at least $(m!)^{(n/m)}$ flowtime-optimal schedules. From at least a mathematical viewpoint, it makes sense to consider a secondary criterion to choose a "best" flowtime-optimal schedule among these (a huge number of) schedules each of which minimizes total flowtime. Also, this is reasonable from a practical viewpoint.

The first problem, $P\ /\ \ /\ C_{\max}$, is $\mathcal{NP}$-hard even for $m = 2$ (trivial reduction from PARTITION). Graham's ground-breaking work on the subject in the 1960's tackled the problem $P\ /\ \ /\ C_{\max}$. This work was ground-breaking not only in approximation algorithms for scheduling, but in approximation algorithms in general. Graham first proved:

Theorem 1. *(Graham* [8]*) The* List Scheduling *algorithm has a worst-case approximation ratio of* $\left(2 - \frac{1}{m}\right)$*. Moreover, this bound is tight for every* $m \geq 2$.

Then, Graham analyzed the List Scheduling algorithm when the list is given in LPT order and provided a very elegant proof of the following result:

Theorem 2. *(Graham* [9]*) The LPT-List Scheduling algorithm has a worst-case approximation ratio of* $\left(\frac{4}{3} - \frac{1}{3m}\right)$*. Moreover, this bound is tight for every* $m \geq 2$.

Just like the scheduling problem $P\ /\ \ /\ C_{\max}$, the FM problem is also $\mathcal{NP}$-hard (a result of Bruno, Coffman and Sethi [1]). In 1976, Coffman and Sethi [2] proposed some approximation algorithms for the FM problem. Among these algorithms, the LD algorithm is the closest extension of LPT list scheduling to the FM problem. In short, LD is an implementation of the LPT idea which respects the rank constraint. Detailed description follows:

LD Algorithm

Input: positive integers $m \leq n$ such that m divides n, nonnegative integers $p_1, p_2, \ldots, p_n$ such that $p_1 \geq p_2 \geq \cdots \geq p_n$.

1: $k := \frac{n}{m}$,
2: **for** $i := 1$ to m **do**
3:     $C_i := 0$,
4:     $J_i := \emptyset$,
5:     $\sigma(i) := i$
6: **end for**
7: **for** $\ell := 1$ to k **do**
8:     **for** $i := 1$ to m **do**
9:         $C_{\sigma(i)} := C_{\sigma(i)} + p_{(\ell-1)m+i}$,
10:         $J_{\sigma(i)} := \left[J_{\sigma(i)}, (\ell-1)m+i\right]$
11:     **end for**
12:     $\sigma :=$ sorting permutation for C
13: **end for**
14: **for** $i := 1$ to m **do**
15:     **reverse**(J_i)
16: **end for**

Output: $J_1, J_2, \ldots, J_m$.

In the above, C_i represents the total amount of processing time assigned to machine i. When the algorithm terminates, C_i is the completion time on machine i for the LD schedule. J_i is an ordered string of integers, it represents the jobs assigned to machine i. Until Steps 14–16 of the algorithm are executed, jobs listed in J_i are in the LPT order. To generate a flowtime optimal schedule, in the Steps 14–16, the order of assigned jobs in each machine is reversed. σ denotes the permutation that sorts the integers $C_1, C_2, \ldots, C_m$ (the total processing time assigned to each machine so far) so that

$$C_{\sigma(1)} \leq C_{\sigma(2)} \leq \cdots \leq C_{\sigma(m)}.$$

The algorithm starts with all C_i equal to zero and σ being the identity permutation. However, after each run of the loop given by Steps 8–11, we update σ so that $\sigma(i)$ denotes the number of the machine which has the i^{th} smallest total processing time at that point. Note that the LD Algorithm also has very close ties to List Scheduling. For each ℓ from Step 7, the inside loop just implements List Scheduling. Coffman and Sethi conjectured the following worst-case bound for the LD algorithm.

<u>Coffman-Sethi conjecture</u> [2]: For every $m \in \mathbb{Z}_+$, the LD algorithm has a makespan ratio with a worst-case bound equal to
$$\frac{5m-2}{4m-1} = \frac{5}{4} - \frac{3}{4(4m-1)}.$$

The authors and Huang [16] constructed a family of instances, proving that the above conjectured ratio cannot be improved for any $m \in \mathbb{Z}_+$. (See, Appendix A.) The Coffman-Sethi conjecture has remained open for four decades. Some of the difficulties in dealing with this conjecture and similar ones are due to the lack of efficiently computable tight lower bounds on the makespan of flowtime-optimal schedules (see Lin and Liao [14] for a discussion of such lower bounds and related heuristics). This is in contrast to the situation in the $P\ /\ \ /\ C_{\max}$ problem and Theorem 2. In the $P\ /\ \ /\ C_{\max}$ problem setting, the simple lower bound on $C^*_{\max}$ given by $\max\left\{p_1, \sum_{j=1}^{n} p_j/m\right\}$ (maximum of the length of the longest job and the average work to be done per machine), when used with a fundamental notion of minimality (smallest n) in hypothesized counterexamples leads to a very elegant and short proof of the exact worst-case performance ratio of LPT. However, in the FM problem, the above lower bound and its variants and the above used basic notion of minimality seem to be too weak to lead to an exact worst-case analysis for the LD algorithm. The first major advances on the Coffman-Sethi conjecture were reported by the authors and Huang in [16] (this was a result of the work done in 2004 by Huang and Tunçel, and then continued by Ravi and Tunçel during 2006–2013). These efforts reduced the task of verifying the correctness of this long-standing conjecture to checking the cases when the number of ranks are either 4 or 5, but number of machines in the unsolved cases remained unbounded. At that time, to verify the remaining cases, the only viable tool seemed to be setting up and solving to optimality finitely many LP problems for each $m \geq 4$. In the next section, utilizing our recent work with Huang [16], we provide a proof for all the remaining cases of this conjecture which does not rely on extensive computations or extensive case analysis. Moreover, our proof here not only covers the remaining cases $m \geq 4$ with $k \in \{4m, 5m\}$, it works for all $m \geq 4$ and $k \geq 4$. See Figure 1 .

Eck and Pinedo [7] propose a new algorithm LPT* (which is closely related to the LD algorithm) for the FM problem and for the two machine case, prove the worst-case approximation ratio of 28/27 which is an exact performance bound in this particular case. Gupta and Ruiz-Torres [11] present their computational study of LPT* and its variants as well as some lower bounds for the FM problem (for other approaches to MULTIFIT, see Dósa [6] and the references therein). Gupta and Ho [10] propose a modified MULTIFIT algorithm for the FM problem when $m = 2$ and present computational results. A slight generalization of the FM problem can be formulated as the problem of permuting the elements within the columns of an m-by-n matrix with nonnegative entries, so as to minimize its maximum row sum. This problem, which models the assembly line balancing problem, was studied by Coffman and Yannakakis [3] as well as Hsu [13]. The LD algorithm is also closely related to some fundamental heuristics for such problems.

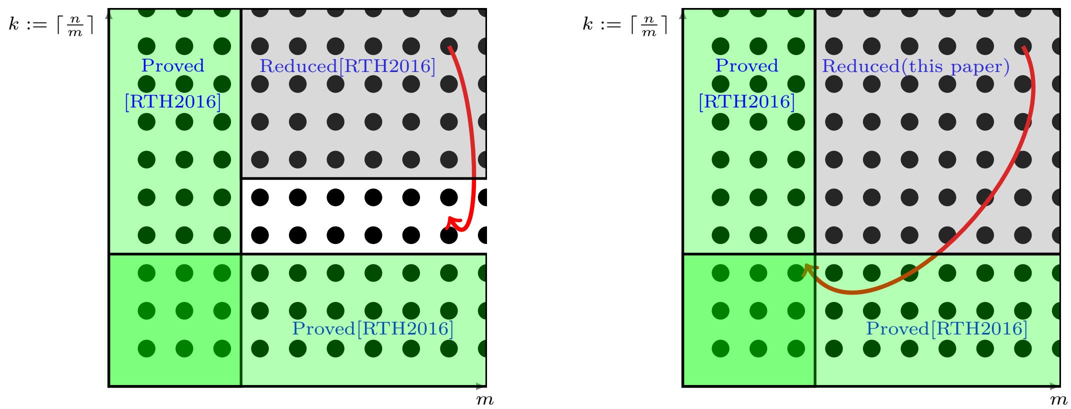

FIGURE 1. Coverage of proof techniques.

Another related problem is studied by Dokka, Crama and Spieksma [5]. In the next section, we present a proof of the Coffman–Sethi conjecture.

2. A proof of the Coffman–Sethi conjecture

2.1. **Notation and some properties.** In every rank r, we identify the largest and smallest processing times and denote them by λ_r and μ_r. Therefore, we have

$$\lambda_1 \geq \mu_1 \geq \lambda_2 \geq \mu_2 \geq \cdots \geq \lambda_{k-1} \geq \mu_{k-1} \geq \lambda_k \geq \mu_k \geq 0. \tag{1}$$

Given an instance of the FM problem, we denote by t_{LD} the makespan of the LD schedule(s). We use t^* to denote the makespan of the optimal schedule(s). We will restrict our attention to problem instances with integer data in this paper (as established in [15, 16] this is without loss of generality). It follows that the smallest nonzero processing time is bounded below by one. In our characterizations of minimal counterexamples to the Coffman–Sethi conjecture, minimality is defined, as in [16], based on the following five attributes, in hierarchical order (in decreasing order of priority):

(a) smallest k (number of ranks)
(b) smallest m (number of machines)
(c) smallest $|\{j : p_j \geq 1\}|$ (number of jobs with nonzero processing times)
(d) largest $\frac{t_{LD}}{t^*}$
(e) smallest $\sum_{j=1}^{n} p_j$.

We may assume that in a minimal counterexample, the following property holds (see [16]):

$$\text{(Property.2)} \qquad \mu_r = \lambda_{r+1}, \forall r \in \{1, 2, \ldots, k-1\} \text{ and } \mu_k = 0.$$

A *rectangular schedule* is a feasible schedule for FM, without any idle time between time zero and the makespan. Every rectangular schedule minimizes the makespan, since its objective value matches an obvious lower bound of $\sum_{j=1}^{n} p_j/m$ on the makespan of every feasible schedule for FM. Next, we describe two useful ways (procedures REDUCE(P1,r) and $\square$-REDUCE(P1,r)) of generating "smaller" FM instances from a given FM instance. Let P1 denote an FM problem instance. Let $r \in \{2, 3, \ldots, k\}$.

REDUCE(P1,r): Construct P2 from P1 by subtracting one time unit from the processing time of every job in rank $r-1$ and subtracting one time unit from the processing time of every job in rank r that has a processing time of λ_r. Leave the remaining processing times unchanged.

Figure 2 presents an LD schedule S for $m := 3$, $P1 := [9, 8, 7, 7, 6, 5, 5, 2, 1]$. Completion times are 15, 16, and 19 on the machines 1, 2, and 3 respectively.

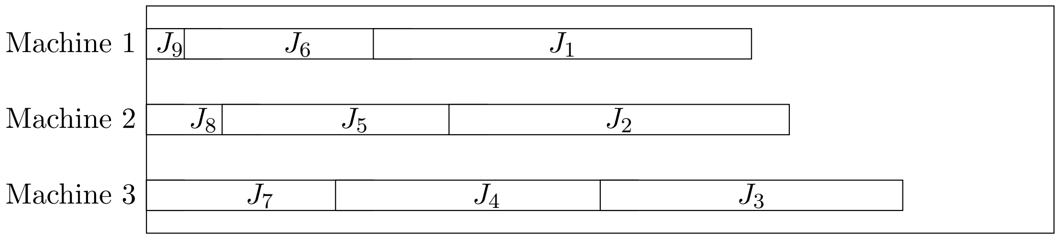

FIGURE 2. An LD schedule S for the instance given by P1

Figure 3 presents the result of the application of REDUCE(P1,2) on the original LD schedule S, yielding REDUCE(P1,2) $= [8, 7, 6, 6, 6, 5, 5, 2, 1]$, and the completion times become 14, 15, and 17 on the machines 1, 2, and 3 respectively.

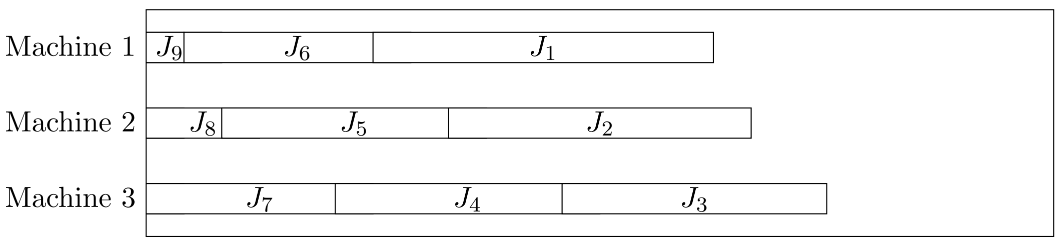

FIGURE 3. LD schedule S_1 for the instance REDUCE(P1,2)

$\square$-REDUCE(P1,r): Construct P2 from P1 by applying the procedure REDUCE(P1,r) to P1. Construct P2R from P2 as follows. For every job in rank 1 of the optimal schedule for P2 that is processed on a machine with a completion time after rank k that is less than the makespan, increase the processing time so that the completion time after rank k becomes equal to the makespan.

Every instance generated by $\square$-REDUCE(P1,$\cdot$) has, by construction, a rectangular optimal schedule. Figure 4 presents an optimal schedule for the instance given by
P2 := REDUCE(P1,2) $= [8, 7, 6, 6, 6, 5, 5, 2, 1]$. In this optimal schedule, the completion times are 15, 15, and 16 on the machines 1, 2, and 3 respectively.

Figure 5 presents an optimal schedule for the instance $\square$-REDUCE(P1,2). Here,
P2R := $\square$-REDUCE(P1,2) $= [9, 8, 6, 6, 6, 5, 5, 2, 1]$ whose optimal schedules yield a makespan of 16 on every machine.

If the Coffman–Sethi conjecture is false, a counterexample to the conjecture of *Type I2* is a counterexample that has an LD schedule with the following properties (see [16]):

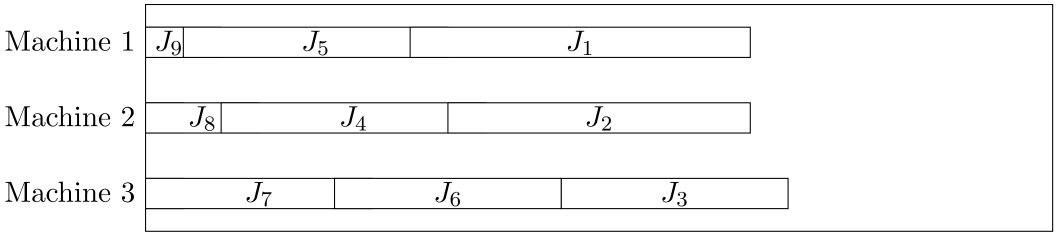

FIGURE 4. An optimal schedule for the instance given by P2

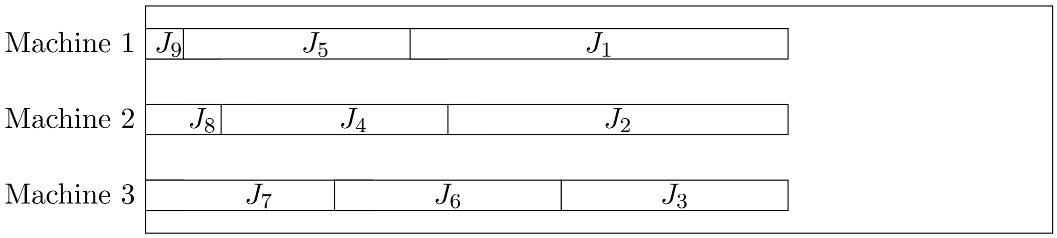

FIGURE 5. An optimal schedule for the instance ☐-REDUCE(P1,2)

(i) It has only one machine i' with a completion time after rank k equal to the makespan.
(ii) Machine i' has a processing time equal to λ_r in rank r for every $r \in \{2, 3, \ldots, k\}$.

The next lemma can be proved by utilizing procedures like REDUCE and ☐-REDUCE repeatedly.

Lemma 1. *(Ravi, Tunçel and Huang* [16]*) If the Coffman–Sethi conjecture is false, then there exists a minimal counterexample to the conjecture of Type I2. Moreover, in a minimal counterexample of Type I2, all of the following properties hold:*

- *the sole machine i' with a completion time after rank k equal to the makespan in the LD schedule has a processing time equal to μ_1 in rank 1;*
- *there exists at least one machine i'' with $i'' \neq i'$, such that the completion time after rank $(k-1)$ on machine i'' is greater than or equal to the completion time after rank $(k-1)$ on machine i';*
- *the smallest completion time after rank k on any machine is at least*

$$t_{LD} - \max_{r \in \{2,3,\ldots,k\}} \{\lambda_r - \mu_r\}.$$

Proof. See Lemmas 6, 7, 8, 9, and 10 of [16]. ☐

Lemma 1 exposes many, combinatorially very strong properties of a minimal counterexample of Type I2. These properties allow us to deduce very strong inequalities on the optimal makespan in terms of the makespan of an LD schedule for such minimal counterexamples. Another important ingredient in our proof is the fact that the Coffman–Sethi conjecture has been verified for all instances with either small m or small k:

Theorem 3. *(Ravi, Tunçel and Huang* [16]*) The Coffman–Sethi conjecture holds for all instances with either property given below:*

(i) $m \le 3$ *(FM instances with at most three machines),*
(ii) $k \le 3$ *(FM instances with at most three ranks, i.e., for all machine-job pairs* (m, n) *satisfying* $n \le 3m$*).*

Proof. See, respectively, Theorems 2 and 3 of [16]. □

Next, we prove that the conjecture holds for all of the remaining cases.

Theorem 4. *The Coffman–Sethi conjecture holds for all instances with both of the properties given below:*

(i) $m \ge 4$ *(FM instances with at least four machines),*
(ii) $k \ge 4$ *(FM instances with at least four ranks, i.e., for all machine-job pairs* (m, n) *satisfying* $n \ge 3m + 1$*).*

Proof. Suppose the above claim is false. Then, by Lemma 1, there exists a minimal counterexample of Type I2 to the Coffman–Sethi conjecture. Since the conjecture holds for all instances with $m \le 3$ as well as for all instances with $k \le 3$ (by Theorem 3), there must exist a minimal counterexample of Type I2 to the claim with k and m both at least equal to four. Let t denote the makespan for an LD schedule of the minimal counterexample of Type I2 with k ranks. Then, by Lemma 1, we have

$$mt^* \ge t + (t - \lambda_k) + (m - 2)\left(t - \max_{r \in \{2,\ldots,k\}}\{\lambda_r - \mu_r\}\right).$$

Note that each term on the right-hand side is obtained from one of the three properties listed in Lemma 1. The last relation is equivalent to

$$t^* \quad \ge \quad t - \frac{\lambda_k}{m} - \left(1 - \frac{2}{m}\right)\max_{r \in \{2,\ldots,k\}}\{\lambda_r - \mu_r\}. \tag{2}$$

Since we are working with a counterexample,

$$t \quad > \quad \left(\frac{5m-2}{4m-1}\right)t^*. \tag{3}$$

Inequalities (2) and (3) imply,

$$\left(\frac{m-1}{4m-1}\right)t^* \quad < \quad \frac{\lambda_k}{m} + \left(1 - \frac{2}{m}\right)\max_{r \in \{2,\ldots,k\}}\{\lambda_r - \mu_r\}. \tag{4}$$

Suppose that the maximum, $\max_{r \in \{2,\ldots,k\}}\{\lambda_r - \mu_r\}$, is attained by $r = k$. Then, (4) implies (since $\mu_k = 0$ due to (Property.2)): $t^* < \left(4 - \frac{1}{m}\right)\lambda_k$. However, $t^* \ge \lambda_k + \sum_{r=1}^{k-1}\mu_r > k\lambda_k$. Since $k \ge 4$, we reach a contradiction. Therefore, we may assume, there exists $s \in \{2, 3, \ldots, k-1\}$ such that $\max_{r \in \{2,\ldots,k\}}\{\lambda_r - \mu_r\} = \lambda_s - \mu_s$.

Using (2) and (3), as well as the facts $\mu_s = \lambda_{s+1}$ (due to (Property.2)) and $\lambda_k \le \lambda_{s+1}$, we obtain

$$\left(\frac{m-1}{5m-2}\right)t \quad < \quad \frac{\lambda_{s+1}}{m} + \left(1 - \frac{2}{m}\right)(\lambda_s - \lambda_{s+1}). \tag{5}$$

From the first property in Lemma 1, it follows that

$$t \quad = \quad 2\lambda_2 + \sum_{r=3}^{k}\lambda_r, \tag{6}$$

substituting this for t in (5), we derive:

$$\lambda_2 \quad < \quad -\frac{1}{2}\sum_{r=3}^{k}\lambda_r + \frac{1}{2(m-1)}\left(5-\frac{2}{m}\right)\lambda_{s+1} + \left(\frac{m-2}{2(m-1)}\right)\left(5-\frac{2}{m}\right)(\lambda_s-\lambda_{s+1}). \tag{7}$$

Next, we consider a lower bound on t^* based on λ_s:

$$t^* \quad \geq \quad \sum_{r=1}^{s-1}\mu_r + \lambda_s + \sum_{r=s+1}^{k}\mu_r = \sum_{r=2}^{s-1}\lambda_r + 2\lambda_s + \sum_{r=s+2}^{k}\lambda_r, \tag{8}$$

where we used (Property.2). Note that we are using the convention that an empty sum is zero. Since we are working with a counterexample, we have $\frac{t}{t^*} > \frac{5m-2}{4m-1}$. This, together with the relations (8) and (6) imply

$$\frac{2\lambda_2 + \sum_{r=3}^{k}\lambda_r}{\sum_{r=2}^{s-1}\lambda_r + 2\lambda_s + \sum_{r=s+2}^{k}\lambda_r} \quad > \quad \frac{5m-2}{4m-1}. \tag{9}$$

If $s \in \{3, 4, \ldots, k-1\}$, then the last inequality is equivalent to

$$\lambda_2 \quad > \quad \frac{1}{3}\left(1-\frac{1}{m}\right)\sum_{r=3}^{s-1}\lambda_r + \left(2-\frac{1}{m}\right)\lambda_s - \frac{1}{3}\left(4-\frac{1}{m}\right)\lambda_{s+1} + \frac{1}{3}\left(1-\frac{1}{m}\right)\sum_{r=s+2}^{k}\lambda_r. \tag{10}$$

Finally, relations (7) and (10) imply

$$\left(\frac{5}{6}-\frac{1}{3m}\right)\left(\sum_{r=3}^{s-1}\lambda_r + \sum_{r=s+2}^{k}\lambda_r\right) + \left(\frac{5}{2}-\frac{1}{m}-\frac{(5m-2)(m-2)}{2m(m-1)}\right)\lambda_s$$
$$+\left(\frac{5}{3}-\left(\frac{17m-8}{3m(m-1)}\right)\right)\lambda_{s+1} < 0.$$

For m and k at least four, sum of the first two terms on the left-hand-side is clearly positive. The last term is nonnegative for every $m \geq 4$. Hence, we reached a contradiction. Therefore, we may assume, $s = 2$.

Let us go back to relation (4) and use $s = 2$ and $\lambda_k \leq \lambda_4$ to obtain:

$$\left(\frac{m-1}{5m-2}\right)t \quad < \quad \frac{\lambda_4}{m} + \left(1-\frac{2}{m}\right)(\lambda_2-\lambda_3). \tag{11}$$

Substituting (6) into the above, we have

$$\lambda_2 \quad > \quad \left(\frac{6m^2-13m+4}{3m^2-10m+4}\right)\lambda_3 + \left(\frac{m^2-6m+2}{3m^2-10m+4}\right)\lambda_4. \tag{12}$$

Since $s = 2$, (9) becomes

$$\lambda_2 \quad < \quad \left(\frac{4m-1}{2(m-1)}\right)\lambda_3 - \frac{1}{2}\sum_{r=4}^{k}\lambda_r. \tag{13}$$

Now, using the fact that $m \geq 4$, relations (12) and (13) imply

$$\left(\frac{5m^2+8m-7}{m-1}\right)\lambda_3 \quad < \quad -\left(5m^2-22m+8\right)\lambda_4 - \left(3m^2-10m+4\right)\sum_{r=5}^{k}\lambda_r.$$

For m at least four, the coefficient of λ_3 in the left-hand-side above is positive, thus the left-hand-side is positive; however, for m at least four, the right-hand-side is always nonpositive.

Hence, we reached a contradiction. Therefore, our original claim that Coffman–Sethi conjecture holds for all instances with m and k at least four is true. □

Theorem 5. *The LD algorithm has a makespan ratio with a worst-case bound equal to* $\frac{5m-2}{4m-1}$. *Moreover, this bound is tight for every* $m \geq 2$.

Proof. Validity of the bound follows from Theorems 3 and 4. The second statement of the theorem is established by the family of worst-case instances presented in [16]. □

Acknowledgment: The work was supported in part by Discovery Grants from NSERC (Natural Sciences and Engineering Research Council of Canada). The second author's research was also supported in part by U.S. Office of Naval Research under award number: N00014-15-1-2171.

References

[1] Bruno, J., E. G. Coffman and R. Sethi, Algorithms for minimizing mean flow time, *INFORMATION PROCESSING '74: Processings of the IFIPS Conference*, 504–510, North-Holland Publishing Company, 1974.

[2] Coffman, E. G. and R. Sethi, Algorithms minimizing mean flowtime: schedule-length properties, *Acta Informatica* 6, 1–14, 1976.

[3] Coffman, E. G. and M. Yannakakis, Permuting elements within columns of a matrix in order to minimize maximum row sum, *Mathematics of Operations Research*, 9(3), 384–390, 1984.

[4] Conway, R. W., W. L. Maxwell and L. W. Miller, *Theory of Scheduling*, Addison Wesley, MA, USA, 1967.

[5] Dokka T., Y. Crama and F. C. R. Spieksma, Multi-dimensional vector assignment problems, *Discrete Optimization* 14, 111–125, 2014.

[6] Dósa, G. Generalized multifit-type methods for scheduling parallel identical machines, *Pure Mathematics and Applications*, 12(3), 287–296, 2001.

[7] Eck, B. T. and M. Pinedo, On the minimization of the makespan subject to flowtime optimality, *Operations Research*, 41(4), 797–801, 1993.

[8] Graham, R. L., Bounds for certain multiprocessing anomalies, *Bell Systems Technical Journal*, 45(9), 1563–1581, 1966.

[9] Graham, R. L., Bounds on multiprocessing timing anomalies, *SIAM Journal of Applied Mathematics*, 17(2), 416–429, 1969.

[10] Gupta, J. N. D. and J. C. Ho, Minimizing makespan subject to minimum flowtime on two identical parallel machines, *Computers and Operations Research* 28, 705–717, 2001.

[11] Gupta, J. N. D. and A. J. Ruiz-Torres, Minimizing makespan subject to minimum total flow-time on identical parallel machines, *European Journal of Operational Research* 58, 370–380, 2000.

[12] Hardy, G. H., J. E. Littlewood and G. Pólya, *Inequalities*, Second ed., Cambridge University Press, 1952.

[13] Hsu, W.-L., Approximation algorithms for the assembly line crew scheduling problem, *Mathematics of Operations Research*, 9, 367–383, 1984.

[14] Lin, C. H. and C. J. Liao, Makespan minimization subject to flowtime optimality on identical parallel machines, *Computers and Operations Research*, 31, 1655–1666, 2004.

[15] Ravi, P. S., *Techniques for Proving Approximation Ratios in Scheduling*, M.Math. Thesis, Dept. of Combinatorics and Optimization, Faculty of Mathematics, University of Waterloo, Canada, 2010.

[16] Ravi, P. S., L. Tunçel and M. Huang, Worst-case performance analysis of some approximation algorithms for minimizing makespan and flowtime, *Journal of Scheduling*, DOI:10.1007/s10951-015-0467-4, 2016.