Operations Research Letters 49 (2021) 842–843

Contents lists available at ScienceDirect

Operations Research Letters

www.elsevier.com/locate/orl

A simple proof of the Moore-Hodgson Algorithm for minimizing the number of late jobs

Joseph Cheriyan [a,1], R. Ravi [b,*,2], Martin Skutella [c,3]

[a] C&O Dept., University of Waterloo, Waterloo, ON N2L 3G1, Canada
[b] Tepper School of Business, Carnegie Mellon University, Pittsburgh, USA
[c] Institute of Mathematics, Technische Universität Berlin, Germany

ARTICLE INFO

Article history:
Received 12 April 2021
Received in revised form 3 September 2021
Accepted 20 September 2021
Available online 23 September 2021

Keywords:
Scheduling theory
Moore-Hodgson Algorithm
Number of late jobs

ABSTRACT

The Moore-Hodgson Algorithm minimizes the number of late jobs on a single machine. That is, it finds an optimal schedule for the classical problem $1\mid\mid\sum U_j$. Several proofs of the correctness of this algorithm have been published. We present a new short proof.

1. Introduction

In 1968, J. M. Moore [5] presented an algorithm and analysis for minimizing the number of late jobs on a single machine. Moore stated "The algorithm developed in this paper, however, consists of only two sorting operations performed on the total set of jobs, ... Consequently, this method will be computationally feasible for very large problems and can be performed manually on many smaller problems." At the end of the paper, Moore presented a version of his algorithm that he attributed to T. E. Hodgson; we follow that version. In hindsight, the algorithm is "just right" for the problem, and it is a popular topic in courses on Scheduling. Several proofs of correctness have been published in the literature, see, e.g., [5,4,2,1,6]. But, in our opinion, none of these proofs matches the simplicity of the algorithm. We present a proof that, hopefully, remedies this discrepancy.

Our notation usually follows the notation of Pinedo [6]. For a positive integer ℓ, we use $[\ell]$ to denote the set $\{1, 2, \ldots, \ell\}$.

An instance I of the scheduling problem $1\mid\mid\sum U_j$ consists of one machine and n jobs; the jobs are denoted $1, \ldots, n$ (we identify a job with its index). Each job j has a non-negative processing time p_j and a non-negative due date d_j.

A schedule for this problem is a permutation of the n jobs. For a given schedule S, the completion time of job j, denoted C_j, is the sum of the processing times of job j and the processing times of the jobs that precede j in S. A job j is called *late* (in the schedule S) if $C_j > d_j$. The goal is to find a schedule such that the number of late jobs is minimum. We use $\textsc{opt}(I)$ or $\textsc{opt}$ to denote the minimum number of late jobs of the instance I (over all possible schedules).

A key feature of our proof is that we do not use induction on a particular instance (which is the plan of Moore's proof), and instead, we use induction on $\textsc{opt}$ over all instances.

2. The algorithm and analysis

The *EDD rule* (earliest due date rule) orders the jobs in non-decreasing order of their due dates; this results in an *EDD sequence*. From here on, we assume that the jobs are indexed according to the EDD rule; that is, $d_1 \le d_2 \le \cdots \le d_n$.

Proposition 1. *If the EDD sequence has a late job, then* $\textsc{opt}$ *is* ≥ 1.

Proof. Let k be the first late job in the EDD sequence. Thus, $C_k = \sum_{i\in[k]} p_i > d_k = \max_{i\in[k]} d_i$. Consider any schedule S. Let ℓ be the last of the jobs in $[k]$ in S. Then, the completion time of ℓ in S is at least $\sum_{i\in[k]} p_i > d_\ell$. Thus ℓ is a late job of S. □

* Corresponding author.
E-mail addresses: jcheriyan@uwaterloo.ca (J. Cheriyan), ravi@cmu.edu (R. Ravi), martin.skutella@tu-berlin.de (M. Skutella).

[1] This author acknowledges support from the Natural Sciences & Engineering Research Council of Canada (NSERC), No. RGPIN-2019-04197.
[2] This material is based upon work supported in part by the U.S. Office of Naval Research under award number N00014-21-1-2243 and the Air Force Office of Scientific Research under award number FA9550-20-1-0080.
[3] Supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – The Berlin Mathematics Research Center MATH+ (EXC-2046/1).

https://doi.org/10.1016/j.orl.2021.09.005
0167-6377/

Clearly, if there is a sub-instance I' that consists of a subset of the jobs such that $\text{OPT}(I') = 0$, then, the EDD sequence of I' has no late jobs.

The Moore-Hodgson Algorithm applies a number of iterations. Each iteration maintains an EDD sequence σ of a subset of the jobs. Initially, $\sigma = 1, 2, \ldots, n$. Each iteration either rejects one job from the sequence σ, or terminates with the guarantee that σ has no late jobs. The algorithm finishes by outputting the concatenated schedule σ, ζ, where σ is the sequence from the last iteration (that has no late jobs), and ζ is an arbitrary permutation of all the rejected (i.e., late) jobs.

At the start of each iteration, σ is an EDD sequence of the non-rejected jobs. An iteration of the algorithm examines the sequence of jobs $\sigma_1, \sigma_2, \ldots, \sigma_\ell$ (where $\ell \le n$), and finds the smallest index k such that the job σ_k is late (thus, $C_{\sigma_k} > d_{\sigma_k}$ and $C_{\sigma_j} \le d_{\sigma_j}, \ \forall j \in [k-1]$). The iteration terminates if there are no late jobs; otherwise, it examines the "prefix" subsequence $\sigma_1, \ldots, \sigma_k$, picks an index m such that p_{σ_m} is maximum among $p_{\sigma_1}, \ldots, p_{\sigma_k}$, and rejects the job σ_m.

The following example illustrates the working of the algorithm; the example is from Moore's paper [5]. There is one "Completion time" row for each iteration. Whenever a job is rejected, its index is noted in the right-most column, and its completion time in subsequent iterations is indicated by an asterisk.

EDD sequence:	1	2	3	4	5	6	7	8	Rejected
Due date d_j:	6	8	9	11	20	25	28	35	Jobs
Processing time p_j:	4	1	6	3	6	8	7	10	
Completion time C_j:	4	5	11						
C_j:	4	5	$*$						3
C_j:	4	5	$*$	8	14	22	29		3
C_j:	4	5	$*$	8	14	$*$	21		3, 6
C_j:	4	5	$*$	8	14	$*$	21	31	3, 6

Theorem 2. *The Moore-Hodgson Algorithm outputs an optimal schedule for the problem* $1 \,||\, \sum U_j$.

Our proof of Theorem 2 is based on the following result.

Lemma 3. *Assume that there are late jobs in the EDD sequence* $\sigma = 1, 2, \ldots, n$. *Let* k *be the first late job, and let* $m \in [k]$ *be the job rejected by the Moore-Hodgson Algorithm, i.e.,* $p_m = \max_{i\in[k]} p_i$. *There is an optimal schedule* π *that rejects job* m.

Proof. Consider an optimal schedule π. Let $R_\pi \subseteq [n]$ denote its subset of rejected (i.e., late) jobs, and let $A_\pi := [n] \setminus R_\pi$ denote its subset of on-time (i.e., non-late) jobs. By Proposition 1, we may assume that π schedules the jobs in A_π in EDD order first, followed by the jobs in R_π in arbitrary order.

If $m \in R_\pi$, we are done.

Otherwise, by Proposition 1, there is a job $r \in [k]$ other than m that has been rejected. Consider the schedule π' that sequences the jobs in $A_{\pi'} := (A_\pi \setminus \{m\}) \cup \{r\}$ in EDD order first, followed by the jobs in $[n] \setminus A_{\pi'} = (R_\pi \setminus \{r\}) \cup \{m\}$ in arbitrary order. We will prove that π' schedules all jobs in $A_{\pi'}$ on time. It is thus optimal since $|R_{\pi'}| = |R_\pi|$; moreover, by construction, π' rejects job m.

First, the jobs in $A_{\pi'} \cap [k-1]$ are completed on time since the EDD rule completes all jobs in $[k-1]$ on time. Second, if $k \in A_{\pi'}$, then its completion time is at most $\sum_{i\in[k]\setminus\{m\}} p_i \le \sum_{i\in[k-1]} p_i \le d_{k-1} \le d_k$. The first inequality follows from the choice of the job m by the Moore-Hodgson Algorithm that ensures that $p_m = max_{i\in[k]} p_i \ge p_k$. Third, compared to the former schedule π, the completion times of jobs in $A_{\pi'} \setminus [k] = A_\pi \setminus [k]$ have been changed in the new schedule π' by $p_r - p_m \le 0$, hence, these jobs also remain on time. □

Proof of Theorem 2. We use induction on OPT over all instances of the problem.
Induction basis: If an instance has $\text{OPT} = 0$, then by Proposition 1, the algorithm outputs the EDD sequence with no late jobs.
Induction step: Let I be an instance with $\text{OPT}(I) \ge 1$ late jobs, and let $I^\ominus$ be obtained from I by deleting the job m rejected in the first iteration of the algorithm. By Lemma 3, $\text{OPT}(I^\ominus) = \text{OPT}(I) - 1$. Thus, by the induction hypothesis, the algorithm finds an optimal schedule $S^\ominus$ for $I^\ominus$. The algorithm for I outputs the schedule S such that S is the same as $S^\ominus$ except that job m is added at the end, as a rejected job. Clearly, S has at most $\text{OPT}(I^\ominus) + 1 = \text{OPT}(I)$ rejected jobs and is thus optimal. □

Remark. Some of the previous proofs ([5], [7], [2], [6, 2nd edition]) use an induction-type argument on a particular instance; then, the argument has to track the parameters of the sequence of rejected jobs throughout; this is not difficult, but, a rigorous presentation takes more than one page.

Remark. Our proof generalizes to the problem of $1 \,||\, \sum w_j U_j$ where jobs j are provided with a weight w_j in addition to their processing time and the minimization objective is now the weighted number of late jobs, in the special case when the processing times and job weights are oppositely ordered: i.e., $p_i \le p_j$ implies $w_i \ge w_j$ (see [3]). The key observation is that in the proof of Lemma 3 when we replace the alternate choice r that was rejected in R_π by the correct choice m in the first bad prefix $[k]$, the opposite ordering relation implies that since $p_r \le p_m$, then $w_r \ge w_m$. Since we replace w_r in the weighted objective with the potentially smaller w_m, this change also makes the weighted objective no worse. The remaining elements of the proof are unchanged.

Acknowledgements

We thank several colleagues for insightful discussions over the years. The idea of the presented proof stems from a discussion between the authors at the Tenth Cargèse Workshop on Combinatorial Optimization at the Institut d'Études Scientifiques de Cargèse, Corsica (France). We are much indebted to the organizers of the workshop, in particular for putting the focus of the 2019 edition on "Proofs from the Book in Combinatorial Optimization."

References

[1] M. v.d. Akker, H. Hoogeveen, Minimizing the Number of Tardy Jobs, Chapter 12 Handbook of Scheduling: Algorithms, Models, and Performance Analysis, CRC Press, 2004.
[2] S. French, Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop, Ellis Horwood Limited, John Wiley & Sons, 1982.
[3] Jan Karel Lenstra, David B. Shmoys, Elements of scheduling, https://elementsofscheduling.nl/. (Accessed April 2021).
[4] W.L. Maxwell, On sequencing n jobs on one machine to minimize the number of late jobs, Manag. Sci. 16 (5) (1970) 295–297.
[5] J.M. Moore, An n job, one machine sequencing algorithm for minimizing the number of late jobs, Manag. Sci. 15 (1) (1968) 102–109.
[6] M.L. Pinedo, Scheduling: Theory, Algorithms, and Systems, Springer Science+Business Media, LLC, 2016.
[7] L.B.J.M. Sturm, A simple optimality proof of Moore's sequencing algorithm, Manag. Sci. 17 (1) (1970) 116–118.