RAIRO-Oper. Res. 53 (2019) 1261–1266
https://doi.org/10.1051/ro/2018097

RAIRO Operations Research
www.rairo-ro.org

A PTAS FOR SINGLE-MACHINE SCHEDULING WITH RELEASE DATES AND JOB DELIVERY TO MINIMIZE MAKESPAN

Lingfa Lu[1,*] and Liqi Zhang[2]

Abstract. We consider the single-machine scheduling problem with release dates and job delivery to minimize makespan. Preemption is not allowed in the processing of the jobs. All jobs are first processed on a single machine and then delivered by a capacitated vehicle to a single customer. The vehicle can deliver at most $c \geq 1$ jobs in each shipment. The round-trip transportation time between the machine and customer is a constant $T > 0$. The problem was proved to be strongly NP-hard and a $\frac{3}{2}$-approximation algorithm was presented in the literature. In this paper we provide a polynomial-time approximation scheme (PTAS) for the problem.

Mathematics Subject Classification. 90B35.

Received December 29, 2016. Accepted October 28, 2018.

1. Introduction

Machine scheduling has been one of the most important and active topics in operations research over the last half century. However, most of the classical scheduling literature considers only job processing without taking into account job delivery. To achieve overall optimality, we may need to consider coordination of the job processing stage and the job delivery stage on different occasions.

Scheduling with job delivery was first studied by Potts [14]. In this model, it was assumed the number of the vehicles is infinite. That is, the vehicles always send the finished jobs individually and immediately to their customers. For this problem, he presented a heuristic algorithm with the worst-case performance ratio $\frac{3}{2}$. After that, machine scheduling problems with job delivery have been widely discussed in the context of manufacturing. Herrmann and Lee [7], Yuan [17] and Chen [3] considered several batch scheduling problems where each delivery batch incurs a delivery cost. Mastrolilli [13] studied a more general scheduling problem than that in [14] on parallel machines with release dates and delivery times. For this problem, he proposed some efficient approximation schemes. Hall and Potts [6] considered a variety of scheduling, batching and delivery problems in the context of supply chain scheduling to minimize the overall scheduling and delivery cost. They provided efficient dynamic programming algorithms for these problems. Chen and Vairaktarakis [5] studied the integrated scheduling problem of minimizing a convex combination of completion times and total delivery cost. They also provided exact or approximation algorithms for the studied problems. Kacem and Kellerer [9]

Keywords. Scheduling, job delivery, polynomial-time approximation scheme.

[1] School of Mathematics and Statistics, Zhengzhou University, 450001 Zhengzhou, Henan, PR China.

[2] College of Information and Management Science, Henan Agricultural University, 450003 Zhengzhou, Henan, PR China.

[*]Corresponding author: lulingfa@zzu.edu.cn

introduced the concept of “no idle time” into the scheduling problem studied by Potts [14]. For the studied problem, they provided a faster $\frac{3}{2}$-approximation algorithm and a polynomial time approximation scheme with efficient time complexity.

Lee and Chen [10] considered several scheduling problems with batch delivery to minimize the makespan. In their models, both the number and the capacity of the vehicles are limited. Two types of transportation situations were considered in their models. The first type, Type-1, involves intermediate transportation of jobs from one machine to another for further processing. The second type, Type-2, involves the transportation of jobs from the machine to their customers. The Type-2 model was extended by Chang and Lee [1] by considering the situation where each job might occupy a different amount of physical space in a vehicle. For the single-machine problem, they provided a heuristic algorithm with the worst-case ratio $\frac{5}{3}$. Wang and Lee [16] considered logistics scheduling with two transport mode choices, where the mode with a shorter delivery time will incur a higher delivery cost. They provided a branch and bound algorithm with two different lower bounds. Chen and Lee [2] considered logistics scheduling with multiple transportation mode choices to minimize the sum of weighted completion times and total delivery cost. For the above problem, they also provided two approximation algorithms.

Recently, Chen [4] and Wang *et al.* [15] provided two comprehensive reviews for production-distribution problems that integrate scheduling and delivery aspects. Thus, for more results on this topic, we refer the reader to Chen [4] and Wang *et al.* [15].

2. Problem formulation

The single-machine scheduling problem with release dates and job delivery to minimize makespan can be described as follows. There are n jobs $J_1, \ldots, J_n$ to be first processed on a single machine and then delivered by a capacitated vehicle to a single customer. Each job J_j has a processing time p_j and a release date r_j. Only one vehicle is employed to deliver all the jobs. The vehicle can deliver at most $c \geq 1$ jobs in each shipment. The set of all the jobs delivered together in one shipment forms a delivery batch. The round-trip transportation time between the machine and customer is a constant $T > 0$. The delivery completion time of J_j is defined as the time at which the delivery batch containing J_j is delivered to the customer and the vehicle returns to the machine. The objective is to minimize the makespan, *i.e.*, the maximum delivery completion time of the jobs. Using the general notation for schedule problems, this problem is denoted by $1 \to D|r_j, c \geq 1|C_{\max}$. If preemption is allowed, the corresponding scheduling problem is denoted by $1 \to D|r_j, \text{pmtn}, c \geq 1|C_{\max}$. Without loss of generality, we only consider regular schedules in which no job can start earlier without affecting the processing of other jobs.

If all the jobs are released at time zero, the corresponding problem is denoted by $1 \to D|c \geq 1|C_{\max}$. Lee and Chen [10] presented a polynomial-time algorithm for the problem. The problems $1 \to D|r_j, c \geq 1|C_{\max}$ and $1 \to D|r_j, \text{pmtn}, c \geq 1|C_{\max}$ were first studied by Lu *et al.* [11]. For $1 \to D|r_j, \text{pmtn}, c \geq 1|C_{\max}$, they presented a polynomial-time algorithm. For $1 \to D|r_j, c \geq 1|C_{\max}$, they showed that the problem is strongly NP-hard and presented a $\frac{5}{3}$-approximation algorithm. Their approximation algorithm is based on the online and greedy strategy. For the latter problem, Liu and Lu [12] further proposed an improved $\frac{3}{2}$-approximation algorithm. However, their method cannot lead to a polynomial-time approximation scheme (PTAS).

In this paper we consider further the scheduling problem $1 \to D|r_j, c \geq 1|C_{\max}$. By using more information on the jobs, we provide a polynomial-time approximation scheme (PTAS).

3. Preliminaries

For a given instance I and a feasible schedule π, we will use the following notation:

- $r_{\max}$, the maximum release date of the jobs, *i.e.*, $r_{\max} = \max\{r_j : 1 \leq j \leq n\}$.
- P, the total processing time of the jobs, *i.e.*, $P = \sum_{1 \leq j \leq n} p_j$.
- ρ_j, the ready time of J_j, which represents the completion time of J_j on the machine.
- δ_j, the departure time of the vehicle from the machine to deliver J_j.

Let $C^*(I)$ and $C^A(I)$ be the makespan of an instance I given by an optimal schedule π^* and an approximation algorithm A, respectively. When no ambiguity occurs, we simplify $C^*(I)$ and $C^A(I)$ by C^* and C^A, respectively.

For problem $1 \to D|r_j, \text{pmtn}, c \geq 1|C_{\max}$, Lu *et al.* [11] presented a polynomial-time algorithm, called A_1 in the following, which will be used in Section 4.

Polynomial-time algorithm A_1

Step 1: The jobs are processed on the machine by the smallest remaining processing time (SRPT) rule, *i.e.*, at any decision time t, the available job with the smallest remaining processing time is selected for processing.

Step 2: Assign the jobs into $\lceil \frac{n}{c} \rceil$ delivery batches and transport all the delivery batches to the customer such that (a) the job with a smaller ready time is delivered no later than that with a larger ready time, and (b) each delivery batch, except the first delivery batch, contains exactly c jobs.

Step 3: Whenever the vehicle and a delivery batch are available, the vehicle always transports the delivery batch with the smallest ready time. When all the jobs are delivered to the customer and the vehicle returns to the machine, stop.

Notice that the jobs can be optimally delivered to the customer by the above strategy (Steps 2 and 3) when a processing schedule of the jobs is given. Thus, the strategy is denoted by Optimal Delivery Strategy. Clearly, Steps 2(a) and 3 can be proved by a pair-wise interchange in two delivery batches. Furthermore, if a delivery batch, with the exception of the first delivery batch, contains less than c jobs, we can always fill the delivery batch with more jobs from earlier delivery batches without increasing the objective value. Thus, it follows that Step 2(b) is also reasonable. In [10], the authors used the same strategy to transport all the jobs to the customer. Indeed, once the processing schedule is given, the remaining job delivery problem is equivalent to the single-machine batch processing problem with release dates and identical processing times. Ikura and Gimple [8] also used the same strategy to solve the latter problem.

4. A polynomial-time approximation scheme

To propose a polynomial-time approximation scheme, we make a series of transformations of the input instance I such that its structure is simplified. Given $\epsilon = 1/E$ for some positive integer E, each transformation may increase the objective value by ϵC^*. When we describe such a transformation, we say that it produces a $1 + \epsilon$ loss.

Let $\varDelta = \max\{P, r_{\max}\}$ and $\delta = \epsilon\varDelta$. Obviously, we have $\delta \leq \epsilon C^*$. We say that a job J_j is small if $p_j \leq \epsilon\delta$, and large otherwise. It is easy to see that there are at most $E^2 - 1$ large jobs in any instance.

Property 4.1. With a $1 + \epsilon$ loss, we can assume that all the release dates in the instance are multiples of δ, and there are at most $E + 1$ distinct release dates in the rounded instance.

Proof. We round every release date up to the nearest multiple of δ. Then, each release date is increased by a value of at most δ. Let π^* be an optimal schedule for the original instance. By increasing the starting time of each job and the departure time of each delivery batch in π^* by δ, we can obtain a feasible schedule of the rounded instance with makespan increased by $\delta \leq \epsilon C^*$. Furthermore, there are at most $E + 1$ distinct release dates in the rounded instance since $E\delta \geq r_{\max}$. □

We partition $[0, +\infty)$ into $E + 1$ disjoint intervals in the form $[R_i, R_{i+1})$, where $R_i = (i - 1)\delta$ for each $1 \leq i \leq E + 1$ and $R_{E+2} = +\infty$. Note that each of the first E intervals has a length δ. Thus, the first E intervals is denoted by regular intervals. By Property 4.1, we can assume that each job is released at some R_i, $1 \leq i \leq E + 1$.

Property 4.2. There exists an optimal schedule such that the jobs starting at the same interval are processed in the order of non-decreasing processing times; furthermore, there is no idle time between the jobs starting at the same interval.

For each feasible schedule π, the machine configuration of the large jobs is defined by the vector $\Gamma = (A_1, \ldots, A_E; S_1, \ldots, S_E)$, where A_i is exactly the set of the large jobs starting in interval $[R_i, R_{i+1})$ and S_i is the starting time of the first large job in A_i, $1 \le i \le E$.

Property 4.3. With a $1+\epsilon$ loss, we can assume that each S_i is a multiple of $\epsilon\delta$, $1 \le i \le E$.

Proof. Consider an optimal schedule with the properties in Properities 4.1 and 4.2. We delay the starting time of the first large job in each regular interval up to the nearest multiple of $\epsilon\delta$ and shift the other jobs accordingly. The new schedule does not change the processing order of the large jobs. Recall that there are exactly E regular intervals. Thus, the ready time of each job is delayed at most $E\epsilon\delta = \delta$. Therefore, by increasing the departure time of each delivery batch by δ, we can obtain a feasible schedule with the objective value increased by at most $\delta \le \epsilon C^*$. □

Property 4.4. There are at most $(E+1)^{E^2+E-1}$ distinct machine configurations of the large jobs.

Proof. Note that there are at most E^2-1 large jobs and each large job has at most $E+1$ choices in different intervals. Thus, all the partitions of the large jobs have at most $(E+1)^{E^2-1}$ possibilities. By Property 4.3, if there are some large jobs processed in $[R_i, R_{i+1})$, we have $R_i \le S_i < R_{i+1}$ and S_i is a multiple of $\epsilon\delta$. Thus, each S_i has at most $E+1$ distinct choices. Consequently, there are at most $(E+1)^{E^2+E-1}$ choices for vector $\Gamma = (A_1, \ldots, A_E; S_1, \ldots, S_E)$. □

Approximation scheme A_2

Step 1: Round each release date up to the nearest multiple of δ.

Step 2: Construct all the machine configurations of the large jobs. Then, for each of them, do the following:

- Step 2.1: Assign the large jobs to the machine according to the machine configuration (if feasible). This leaves several idle-time intervals on the machine. If some R_i with $1 \le i \le E+1$ is an interior point of some idle-time interval H, we partition H into two idle-time intervals H' and H'' at R_i. The resulting idle-time intervals are denoted by $H_1, H_2, ..., H_m, H_{m+1}$ from left to right in this order, where H_{m+1} is the unique unbounded idle-time interval.
- Step 2.2: For i from 1 to m, assign as many as possible of the available unscheduled small jobs to the time interval H_i in the order of non-decreasing processing times; if there is not enough space for some small job, we stretch the length of this time interval by at most $\epsilon\delta$ such that the small job can be exactly assigned.
- Step 2.3: The remaining jobs (which may include both small jobs and large jobs) are assigned to H_{m+1} by the rule that the available unscheduled jobs are scheduled in the order of non-decreasing processing times.
- Step 2.4: Transport all the jobs by the Optimal Delivery Strategy.

Step 3: Among all the schedules obtained above, select the one with the minimum makespan.

Clearly, we can re-index all the jobs in $O(n \log n)$ time such that $p_1 \le p_2 \le \ldots \le p_n$. For each machine configuration of the large jobs, constructing the corresponding schedule can be done in $O(n)$ time. Thus, the time complexity of A_2 is $O(n \log n + n(E+1)^{E^2+E-1})$, which is polynomial in n when ϵ is a constant.

Theorem 4.5. *Approximation scheme A_2 is a polynomial-time approximation scheme for the problem $1 \to D|r_j, c \ge 1|C_{\max}$.*

Proof. Let I be an arbitrary original instance. Let I_1 be the instance obtained from I by Step 1. By Property 4.1, we have $C_{\max}(I_1) \leq (1+\epsilon)C^*$. Note that after Step 2(2.1), there are $m+1 \leq 2E+1$ idle-time intervals. Since H_{m+1} is unbounded, the total length of the time space caused by stretching the time slots in Step 2(2.2) is at most $m\epsilon\delta \leq 2\epsilon C^*$. Let I_2 be the instance obtained from I_1 by Step 2. Thus, we have $C_{\max}(I_2) \leq C_{\max}(I_1) + 2\epsilon C^* \leq (1+3\epsilon)C^*$. Let π^* be a restricted optimal schedule of I_2 satisfying Property 4.3. Thus, we have $C_{\max}(\pi^*) \leq C_{\max}(I_2) + \epsilon C^* \leq (1+4\epsilon)C^*$. By Property 4.2, we can assume that the jobs starting in the last interval are processed in the order of non-decreasing processing times under π^*.

The processing of the large jobs in π^* represents a machine configuration Γ in Step 2. For the machine configuration Γ, the algorithm produces a feasible schedule π of I_2. Note that each job is released at some R_i, $1 \leq i \leq E+1$. Thus, π schedules all the small jobs by the SRPT rule.

Now we consider the feasible schedules of I_2 under the restriction that the processing of the large jobs is the same as that in π^*. Such schedules are called π^*-restricted schedules of I_2.

Claim 1: π is an optimal π^*-restricted schedule of I_2.

Indeed, we will prove a stronger statement by contradiction that π is an optimal preemptive π^*-restricted schedule of I_2. Let Π be the set which consists of all optimal preemptive π^*-restricted schedules of I_2. We assume to the contrary that $\pi \notin \Pi$. Let σ with $\sigma \in \Pi$ be an arbitrary optimal preemptive π^*-restricted schedule of I_2. We define $t(\pi,\sigma)$ to be the maximum time t such that the sub-schedules of π and σ are identical in the time interval $[0,t)$. Let π' with $\pi' \in \Pi$ be a schedule such that $t(\pi,\pi') \geq t(\pi,\sigma)$ holds for any schedule $\sigma \in \Pi$. Set $t' = t(\pi,\pi')$. Clearly, t' is the minimum time such that π' does not coincide with π. Suppose that π processes job J_j at time t' and π' processes job J_i at time t'. Note that both π and π' are π^*-restricted schedule of I_2. Thus, the processing of the large jobs is identical in π and π'. That is, both J_j and J_j are small jobs. Let $p_j(t')$ and $p_i(t')$ be the (remaining) processing times of J_j and J_i, respectively. By the implementation of A_2, both J_i and J_j are available at time t'. Thus, we have $p_j(t') \leq p_i(t')$.

Let S be the interval set consisting of all the time slots starting after or at t' in which J_i and J_j are processed in π'. Define a new schedule π'' from π' by rescheduling the remaining parts of J_i and J_j in such a way that, in the time slots in S, the remaining parts of J_j are processed before J_i. Note that $p_j(t') \leq p_i(t')$ and π' and π'' process J_i and J_j at time t', respectively. One can easily see that $\rho_j(\pi'') < \min\{\rho_i(\pi'), \rho_j(\pi')\}$, $\rho_i(\pi'') = \max\{\rho_i(\pi'), \rho_j(\pi')\}$ and $\rho_k(\pi'') = \rho_k(\pi')$ for all $k \neq i,j$. The early finished jobs are delivered no later than the later finished jobs and the number of jobs in each delivery batch is the same as that in π'. Then we obtain a feasible preemptive π^*-restricted schedule π'' with $C_{\max}(\pi'') \leq C_{\max}(\pi')$. This implies that π'' is also an optimal preemptive π^*-restricted schedule of I_2. That is, $\pi'' \in \Pi$. By the definition of π'', the subschedules of π'' and π are identical in the time interval $[0,t')$. Note further that J_j is processed at time t' in both π'' and π. Thus, we have $t(\pi,\pi'') > t(\pi,\pi')$. This contradicts the choice of π'. Therefore, we can conclude that $\pi \in \Pi$. This follows that π is an optimal preemptive π^*-restricted schedule of I_2. The claim follows.

By Claim 1, we have $C^{A_2} \leq C_{\max}(\pi) = C_{\max}(\pi^*) \leq (1+4\epsilon)C^*$. The result follows. □

Acknowledgements. This research was supported in part by NSFCs (11771406, U1504103 and 11571321).

References

[1] Y.-C. Chang and C.-Y. Lee, Machine scheduling with job delivery coordination. *Eur. J. Oper. Res.* **158** (2004) 470–487.

[2] B. Chen and C.-Y. Lee, Logistics scheduling with batching and transportation. *Eur. J. Oper. Res.* **189** (2008) 871–876.

[3] Z.-L. Chen, Scheduling and common due date assignment with earliness-tardiness penalties and batch delivery costs. *Eur. J. Oper. Res.* **93** (1996) 49–60.

[4] Z.-L. Chen, Integrated production and outbound distribution scheduling: review and extensions. *Oper. Res.* **58** (2010) 130–148.

[5] Z.-L. Chen and G.L. Vairaktarakis, Integrated scheduling of production and distribution operations. *Manage. Sci.* **51** (2005) 614–628.

[6] N.G. Hall and C.N. Potts, Supply chain scheduling: batching and delivery. *Oper. Res.* **51** (2003) 566–584.

[7] J.W. Herrmann and C.-Y. Lee, On scheduling to minimize earliness and batch delivery costs with a common due date. *Eur. J. Oper. Res.* **70** (1993) 272–288.

[8] Y. Ikura and M. Gimple, Efficient scheduling algorithms for a single batch processing machine. *Oper. Res. Lett.* **2** (1986) 61–65.

[9] I. Kacem and H. Kellerer, Approximation algorithms for no idle time scheduling on a single machine with release times and delivery times. *Discrete Appl. Math.* **164** (2014) 154–160.

[10] C.-Y. Lee and Z.-L. Chen, Machine scheduling with transportation considerations. *J. Sched.* **4** (2001) 3–24.

[11] L.F. Lu, J.J. Yuan and L.Q. Zhang, The single machine scheduling with release dates and job delivery to minimize makespan. *Theor. Comput. Sci.* **393** (2008) 102–108.

[12] P.H. Liu and X.W. Lu, An improved approximation algorithm for single machine scheduling with job delivery. *Theor. Comput. Sci.* **412** (2011) 270–274.

[13] M. Mastrolilli, Efficient approximation schemes for scheduling problems with release dates and delivery times. *J. Sched.* **6** (2003) 521–531.

[14] C.N. Potts, Analysis of a heuristic for one machine sequencing with release dates and delivery times. *Oper. Res.* **28** (1980) 1436–1441.

[15] D.Y. Wang, O. Grunder and A.E. Moudni, Integrated scheduling of production and distribution operations: a review. *Int. J. Ind. Syst. Eng.* **19** (2015) 94–122.

[16] H. Wang and C.-Y. Lee, Production and transport logistics scheduling with two transport mode choices. *Nav. Res. Logist.* **52** (2005) 796–809.

[17] J.J. Yuan, A note on the complexity of single-machine scheduling with a common due date, earliness-tardiness, and batch delivery costs. *Eur. J. Oper. Res.* **94** (1996) 203–205.