Single Batch Machine Scheduling with Deliveries

B.-Y. Cheng,1,2 J.Y.-T. Leung,1,2,3 K. Li,1,2 S.-L. Yang1,2

1School of Management, Hefei University of Technology, Hefei 230009, People’s Republic of China
2Key Laboratory of Process Optimization and Intelligent Decision-Making, Ministry of Education, Hefei 230009, People’s Republic of China
3Department of Computer Science, New Jersey Institute of Technology, Newark, New Jersey 07012

Received 4 August 2014; revised 11 August 2015; accepted 12 August 2015
DOI 10.1002/nav.21641
Published online 26 August 2015 in Wiley Online Library (wileyonlinelibrary.com).

Abstract: We consider the problem of scheduling a set of $n$ jobs on a single batch machine, where several jobs can be processed simultaneously. Each job $j$ has a processing time $p_j$ and a size $s_j$. All jobs are available for processing at time 0. The batch machine has a capacity $D$. Several jobs can be batched together and processed simultaneously, provided that the total size of the jobs in the batch does not exceed $D$. The processing time of a batch is the largest processing time among all jobs in the batch. There is a single vehicle available for delivery of the finished products to the customer, and the vehicle has capacity $K$. We assume that $K = rD$, where $r \geq 2$ and $r$ is an integer. The travel time of the vehicle is $T$; that is, $T$ is the time from the manufacturer to the customer. Our goal is to find a schedule of the jobs and a delivery plan so that the service span is minimized, where the service span is the time that the last job is delivered to the customer. We show that if the jobs have identical sizes, then we can find a schedule and delivery plan in $O(n \log n)$ time such that the service span is minimum. If the jobs have identical processing times, then we can find a schedule and delivery plan in $O(n \log n)$ time such that the service span is asymptotically at most $11/9$ times the optimal service span. When the jobs have arbitrary processing times and arbitrary sizes, then we can find a schedule and delivery plan in $O(n \log n)$ time such that the service span is asymptotically at most twice the optimal service span. We also derive upper bounds of the absolute worst-case ratios in both cases. © 2015 Wiley Periodicals, Inc. Naval Research Logistics 62: 470–482, 2015

Keywords: batch machine; scheduling; delivery; polynomial time algorithms; NP-hard; approximation algorithms

1. INTRODUCTION

Production and distribution are two key operational functions for manufacturers. To provide a fast response to the customers, the two functions should be coordinated in an effective way. This is particularly important for manufacturers with batch-processing machines and arbitrary-size jobs, for example, food-processing and semiconductor-manufacturers. For a typical manufacturer of this kind, the machines have a given size capacity and the jobs are processed in batches. A batch is processed without preemption until all the jobs in the batch are completed. The total size of the jobs in a batch cannot exceed the machine capacity. After production, completed products are delivered to the customers by the available vehicles. The vehicles also have fixed capacities. When all the products have been delivered to the customers, the service of the company is completed. It is clear that the production schedule influences the distribution requirements directly since only completed batches can be delivered. From the customer’s point of view, the key performance measure is the time period lasting from the release of the order to delivery of all the products. We define this time span as the service span.

Take food processing companies which make bread and biscuits for example. The food-processing company receives orders from a retailer. It then produces food and delivers the food to retailers. In the production part, the products are made after semi-finished products are processed in the baking oven. The semifinished products are doughs in different shapes which are required by the retailers. Different doughs can be processed in the baking oven at the same time. Since the baking oven has a fixed capacity, semifinished products can be processed at the same time only when the total size is not larger than the capacity of the oven. After production, bread and biscuits need to be delivered as soon as possible to keep them fresh. Since the vehicles have fixed capacities,
completed products also need to be assigned to the deliveries. The number of deliveries needs to be minimized to make the distribution done in a shorter time.

In this article, we will study the problem of scheduling a set of \( n \) jobs on a single batch-processing machine where completed batches are delivered by a single vehicle so as to minimize the service span for the manufacturer.

### 1.1. Model and Notations

There are \( n \) jobs to be processed and delivered. The job set is \( J = \{1, 2, \ldots, n\} \). Each job \( j \) has a size \( s_j \) and a processing time \( p_j \). The manufacturer has a single batch-processing machine with a capacity \( D \) to process the jobs in batches. The total size of the jobs in a batch cannot exceed \( D \). Suppose we form \( z \) batches, \( B_1, B_2, \ldots, B_z \). Let \( B = \{B_1, B_2, \ldots, B_z\} \) denote the batch set. The processing time of batch \( B_i \) is the largest processing time among all the jobs in \( B_i \). Letting \( P_i \) denote the processing time of \( B_i \), we have \( P_i = \max \{p_j | j \in B_i\} \). Thus, the makespan of the schedule is \( \sum_{i=1}^{z} P_i \). When a batch is being processed, no job can be added into the batch nor can it be removed from the batch.

In the distribution, the manufacturer has a single vehicle with a capacity of \( K \). We assume that \( K = rD \), where \( r \geq 2 \) and \( r \) is an integer. Thus, the vehicle can deliver at most \( r \) batches at a time, although it can deliver less. We assume that the vehicle will always deliver an integer number of batches at a time, since the batches are packaged in boxes or pallets of uniform size. Moreover, this assumption is necessary to make our study tractable. Also, \( r \) is usually greater than or equal to 2, since the vehicle usually has larger capacity than the machine.

The manufacturer serves a single retailer. The travel time from the manufacturer to the retailer is denoted by \( T \). After one delivery, the vehicle has to return to the manufacturer to make another delivery. Suppose the manufacturer makes \( x \) deliveries. The delivery set is denoted by \( \{d_1, d_2, \ldots, d_x\} \), and the departure time of delivery \( d_l \) \((1 \leq l \leq x) \) is denoted by \( \alpha_l \). Clearly, we have \( \alpha_{l+1} - \alpha_1 \geq 2T \) for \( 1 \leq l < x \). Also, when the vehicle starts to make the delivery \( d_l \), all the batches in \( d_l \) must have completed their processing. The objective is to minimize the service span, which lasts from time 0 until all the products have been delivered to the customer. We use the symbol SS to denote the service span.

It will be more convenient to use the five-field notation of Chen [3] to denote the problem under investigation: \( \gamma_1|\gamma_2|\gamma_3|\gamma_4|\gamma_5 \). In this notation, \( \gamma_1 \) represents the machine configuration and \( \gamma_2 \) represents the constraints in production. \( \gamma_3 \) represents the vehicle configuration; it is usually in the form of \( (\eta_1, \eta_2) \), where \( \eta_1 \) represents the number of vehicles and \( \eta_2 \) denotes the vehicle capacity. \( \gamma_4 \) represents the number of customers and \( \gamma_5 \) represents the objective function.

Using the five-field notation, our problem can be denoted by \( 1|p - \text{batch, } D, s_j, p_j|1, K|1|\text{SS} \). In the \( \gamma_1 \) field, we use “1” to denote a single machine. In \( \gamma_2 \), we use “p-batch” to denote that the machine is a batch-processing machine with capacity \( D \). Also, we use \( s_j \) and \( p_j \) to denote the size and processing time of the job, respectively. \( \gamma_4 \) signifies that there is a single vehicle with capacity \( K \). \( \gamma_5 \) indicates one customer and \( \gamma_5 \) indicates that the objective function is service span.

Before we study the general problem, we shall be studying two special cases. The first special case is when the jobs have identical sizes. We denote this special case by \( 1|p - \text{batch, } D, s_j = 1, p_j|1, K|1|\text{SS} \). The second special case is when the jobs have identical processing times. We denote this special case by \( 1|p - \text{batch, } D, s_j, p_j = p|1, K|1|\text{SS} \). We will call these special cases \( \Psi_1 \) and \( \Psi_2 \), while the general problem will be called \( \Psi_3 \). The problems are shown as follows:

\[
\Psi_1 : 1|p - \text{batch, } D, s_j = 1, p_j | 1, K | 1 | \text{SS}
\]

\[
\Psi_2 : 1|p - \text{batch, } D, s_j, p_j = p | 1, K | 1 | \text{SS}
\]

\[
\Psi_3 : 1|p - \text{batch, } D, s_j, p_j | 1, K | 1 | \text{SS}
\]

In this article, we will give an \( O(n \log n) \)-time algorithm, \( A_1 \), to solve \( \Psi_1 \); this result will be shown in Section 3. If we let \( T \) to be very small relative to \( p \), then \( \Psi_2 \) is exactly the bin-packing problem which is known to be strongly NP-hard [11]. Thus, we give an \( O(n \log n) \)-time approximation algorithm, \( A_2 \), to solve the problem, and we provide upper bounds for its absolute and asymptotic worst-case ratios. These results will be shown in Section 4. Finally, we give an \( O(n \log n) \)-time approximation algorithm, \( A_3 \), to solve the general problem, and we derive upper bounds for its absolute and asymptotic worst-case ratios. These results will be shown in Section 5. In Section 6, we will give some concluding remarks.

Before we leave this section, we will give the definitions of absolute worst-case ratio and asymptotic worst-case ratio. For a given instance \( I \) of the problem and an approximation algorithm \( A \), let \( A(I) \) and \( \text{OPT}(I) \) denote the service span obtained by algorithm \( A \) and an optimization algorithm, respectively, when applied to \( I \). Let \( R_A(I) \equiv A(I)/\text{OPT}(I) \). The absolute worst-case ratio \( R_A \) for algorithm \( A \) is defined as

\[
R_A \equiv \inf \{q \geq 1 : R_A(I) \leq q \text{ for all } I\}.
\]

The asymptotic worst-case ratio \( R^\infty_A \) is defined as

\[
R^\infty_A \equiv \inf \{q \geq 1 : \text{for some } N > 0, R_A(I) \leq q \text{ for all } I \text{ with } \text{OPT}(I) \geq N\}.
\]

### 1.2. An Example

In this section, we give an example to illustrate the integrated scheduling problem. There are \( 20 \) jobs to be processed;
that is, \( n = 20 \). The processing times and sizes of the jobs are shown in Table 1. The first line represents the job indexes, the second line shows the processing times and the last line shows the job sizes. The machine capacity is \( K = 2D = 20 \). The travel time from the manufacturer to the retailer is \( T = 35 \).

A feasible batching result is shown in Table 2. In Table 2, column 1 shows the batches; we see that all jobs are assigned to seven batches. The batches are processed one by one on the machine; that is, from \( B_1 \) to \( B_7 \). Column 2 lists the jobs in each batch. Columns 3 and 4 show the total size of the jobs in the batch and the processing time of the batch, respectively. Columns 5 and 6 show the start time and the completion time of the batch, respectively.

In the distribution, a delivery can be made only when all the batches in the delivery are completed and the vehicle is available. If the vehicle is not available, then we have to wait until the vehicle returns to the manufacturer to start the next delivery. Table 3 shows the distribution schedule. In Table 3, column 1 represents the deliveries; there are four deliveries in this schedule. Column 2 shows the delivery time of each delivery. For example, \( B_2 \) and \( B_3 \) are in the second delivery. Column 3 shows the completion time of all the batches in the delivery. For example, the latest completion time of \( B_2 \) and \( B_3 \) in \( d_2 \) is 77, which can be obtained from Table 2. Column 4 shows the available time of the vehicle for each delivery. For example, the vehicle is available at time 0 for \( d_1 \). For \( d_2 \), the first available time is 95 = 25 + 70, since the departure time of \( d_1 \) is 25 and the round-trip time of the vehicle is 70. Columns 5 and 6 show the departure time and the arrival time of each delivery, respectively. For example, the departure time of \( d_2 \) is 95 and its arrival time is 95 + 35 = 130. Comparing columns 3 and 5, we find that only the first delivery can be made as soon as the batches in the delivery have completed processing, while the remaining three deliveries have to wait even though the batches in the delivery have completed processing. The last delivery is completed at time 270. Therefore, the service span is \( SS = 270 \).

### 2. LITERATURE REVIEW

Integrated scheduling has aroused a lot of interests among researchers in recent years. Hall and Potts [12] studied the scheduling of supply chain that includes the supplier, the manufacturer and the customer. This kind of scheduling is concerned with the supply of materials, production and distribution. Selvarajah and Steiner [26] proposed a 3/2 approximation algorithm to minimize delivery and inventory holding costs. Sawik [25] extended the problem to a long-term product case. Yeung et al. [31] and Osman and Demirli [21] considered the problem with time windows and synchronized replenishment, respectively. Yimer and Demirli [32] proposed a division technique for the problem. The problem was divided into two phases, that is, the manufacturing and delivery phases, and a genetic algorithm was used to solve it.

Other researchers studied integrated scheduling problems between the manufacturer and the customers. Chen and Vairaktarakis [5] showed that most of the integrated scheduling problems of production and distribution are NP-hard. Lee and Chen [16] studied scheduling problems where jobs are first processed on a single machine, parallel machines, or a flow shop, and then delivered by one or more vehicles to the customer. They studied both the objectives of service span and total completion time. Polynomial-time algorithms, pseudopolynomial-time algorithms, ordinary NP-hardness and strong NP-hardness are derived for various special cases. As we shall see in Section 3, we will be utilizing some of their characterizations for a single machine in developing our optimal algorithm for the special case of identical sizes.

Amorim et al. [2] proposed models and integrated schemes for manufacturers making perishable products. Agnetis...
et al. [1] studied the integrated scheduling with interstage batch deliveries in the production. Chen and Pundoor [4] studied manufacturers making time-sensitive products. In the distribution, there are different models in practice. Leung and Chen [17] considered the integrated scheduling problem with fixed distribution time in the distribution. Li and Ou [18] studied the integrated problem with pickup of materials to minimize makespan. Wang and Cheng [29] studied the problem with machine availability constraint. Pundoor and Chen [24] considered the make-to-order production-distribution system and the objective was to minimize the maximum delivery tardiness and total distribution cost. Since most of the integrated scheduling problems are NP-hard, the algorithms applied include approximation algorithms and intelligent algorithms.

Current research on integrated scheduling focuses on the classical production model (Pinedo [23]), in which a machine can only process one job at a time (All of the references mentioned above deal with the classical production model.) Little research has been done on the production model with batch-processing machines and arbitrary-size jobs. In contrast to the classical production model, this type of production is more complex to solve. Uzsoy [28] introduced several heuristics for a single batch-processing machine. Zhang et al. [34] analyzed the worst-case ratios of several heuristics proposed by Uzsoy. Approximation algorithms with better performances were proposed to solve single-machine problems ([15, 19]) and multi-machine problems ([7, 8]). Jula and Leaehman [14] provided a greedy heuristic method and Parsa et al. [22] provided a branch and price algorithm. Intelligent algorithms were also applied to solve the problem including genetic algorithms ([9, 27, 33]), simulated annealing [20], and ant colony optimization [6].

3. IDENTICAL SIZES

In this section, we will study $\Psi_1$. Since $s_j = 1$ for all $1 \leq j \leq n$, we can put $D$ jobs together in a batch. For the rest of the article, we will use $X$ to represent variables of an optimal solution. For example, $z^*$ and $x^*$ represent the numbers of batches and deliveries in an optimal solution, respectively; $\pi^*$ represents an optimal solution. Similarly, $z$ and $x$ represent the numbers of batches and deliveries in the solution obtained by our algorithm, respectively; $\pi$ represents the solution obtained by our algorithm.

**PROPOSITION 1:** The minimum number of batches is $z_{\text{min}} = \lceil n/D \rceil$ and the minimum number of deliveries is $x_{\text{min}} = \lceil n/K \rceil = \lceil n/(rD) \rceil$, where $[a]$ represents the smallest integer greater than or equal to $a$. If there are $z$ batches in $\pi$, then $x \geq [z/r]$.

We propose the following algorithm, to be called Algorithm A1, to solve $\Psi_1$.

**Algorithm A1.**

**Step 1.** Order the jobs in nonincreasing order of their processing times.

**Step 2.** Assign the jobs to batches using the FBLPT rule, which is defined as follows. Put the first $D$ jobs into the first batch $B_1$. Put the next $D$ jobs into the second batch $B_2$. Continue the assignment until the number of remaining jobs is less than $D$. Then put the remaining jobs in the last batch $B_r$.

**Step 3.** Order the batches in nondecreasing order of their processing times. Then process the batches, one by one, starting at time 0.

**Step 4.** Let $z = kr + y$, where $k \geq 0$ and $0 \leq y \leq r - 1$; $k$ and $y$ are both integers. If $y = 0$ (i.e., jobs can be delivered in $k$ deliveries), then deliver $r$ batches at a time from the beginning until all the batches are delivered. If $1 \leq y \leq r - 1$ (i.e., jobs can be delivered in $k + 1$ deliveries), then assign the first $y$ batches as the first delivery $d_1$. For the remaining $kr$ batches, deliver $r$ batches at a time as soon as the vehicle is back and $r$ batches have completed processing. Repeat this process until all the batches are delivered. □

The following lemma is adapted from Property 2 of Lee and Chen (page 12 of [16]).

**LEMMA 1 (Lee and Chen [16]):** There exists an optimal schedule for the problem $\Psi_1$ that satisfies the following conditions.

1. Batches are processed in nondecreasing order of processing times on the machine.
2. Each delivery contains consecutively processed batches.
3. Earlier processed batches are delivered no later than later processed batches.
4. If $y > 0$, then there are $k + 1$ deliveries. The first delivery consists of the first $y$ batches and each of the remaining deliveries consists of $r$ batches.
5. If $y = 0$, then there are $k$ deliveries and each delivery contains $r$ batches.

By Lemma 1, it is easy to see that the delivery plan in Steps 3 and 4 of Algorithm A1 is an optimal delivery plan. Thus, we have the following lemma.

**LEMMA 2:** Given a fixed set of batches, the delivery plan in Steps 3 and 4 of Algorithm A1 is an optimal delivery plan.

The remaining question is how to assign jobs to batches. In Algorithm A1, we use the FBLPT rule to create batches. We will show that this is an optimal way to batch the jobs in problem $\Psi_1$. 

Naval Research Logistics DOI 10.1002/nav
THEOREM 1: Algorithm A1 solves the problem $\Psi_1$ in $O(n \log n)$ time.

PROOF: Let the jobs be sorted in nonincreasing order of their processing times; that is, $p_1 \geq p_2 \geq \cdots \geq p_n$. Without loss of generality, let job 1 be assigned to $B^*_1$ in the optimal solution $\pi^*$. Since the processing time of a batch is the longest processing time among all jobs in the batch, we have $p_1^* = p_1$. Since $s_1 = 1$, we can assign $D - 1$ more jobs into $B^*_1$. Clearly, assigning $D - 1$ more jobs with the longest processing times will not affect the processing time of $B^*_1$ but it will reduce the processing times of subsequent batches. Thus, $B^*_1$ should have jobs 1, 2, $\ldots$, $D$. Repeating the same argument, we see that $B^*_2$ should have jobs $D + 1, D + 2, \ldots, 2D$, and so on. This is exactly the batches produced by the FBLPT rule. By Lemma 2, the delivery plan of Algorithm A1 is optimal. Thus, Algorithm A1 solves the problem $\Psi_1$.

Steps 1 and 3 of Algorithm A1 take $O(n \log n)$ time, since they involve sorting. Step 2 takes $O(n)$ time. Step 4 takes $O(z) = O(n/D)$ times. Thus, the overall running time of Algorithm A1 is $O(n \log n)$. □

4. IDENTICAL PROCESSING TIMES

In this section, we will study $\Psi_2$. Since the jobs have the same processing time $p$, the batches also have the same processing time $p$. Consider the special case of $\Psi_2$ where $T$ is smaller than $p/2$. In this case, the vehicle will always wait for the completion of $r$ batches. Thus, minimizing the service span is equivalent to minimizing the makespan of the schedule. If we treat each batch as a bin with capacity $D$, the problem is equivalent to the one-dimensional bin-packing problem which is known to be strongly NP-hard [11]. Thus, we have the following proposition.

PROPOSITION 2: $\Psi_2$ is NP-hard in the strong sense.

Since $\Psi_2$ is strongly NP-hard, we consider an approximation algorithm, A2, to solve it. It is well known that the First Fit Decreasing (FFD) algorithm for bin packing is an effective algorithm to minimize the number of bins. If we use the FFD algorithm to generate batches, then $z \leq \frac{11}{9} z^* + 4$ [13]. Here, $z$ is the number of batches generated by the FFD algorithm and $z^*$ is the optimum number of batches. Recently, Dosa et al. [10] have shown a tight absolute bound for the FFD algorithm: $z \leq \frac{11}{9} z^* + \frac{6}{9}$. We will be using this new result in our proofs.

Algorithm A2 uses the FFD algorithm to generate the batches and then delivers the batches using Step 4 of Algorithm A1.

Algorithm A2.

Step 1. Order the jobs in nonincreasing order of their sizes.

Step 2. Generate the batches using the FFD algorithm. Assign the first job to the first batch $B_1$. Subsequent jobs will be assigned one at a time. The next job will be assigned to the first batch for which the job can fit. If none of the existing batch can accommodate the job, then assign the job to the next empty batch. Repeat this process until all jobs have been assigned. Let there be $z$ batches generated, $B_1, B_2, \ldots, B_z$.

Step 3. Sort the batches in nondecreasing order of their processing times; that is, $p_1 \geq p_2 \geq \cdots \geq p_z$.


Suppose that the optimal solution generates $z^*$ batches. Let $z^* = k^* r + y^*$. Since the batches are generated by the FFD algorithm in Algorithm A2, we have the following proposition on $z$.

PROPOSITION 3 (Dosa et al. [10]): The number of batches generated by A2 satisfies $z \leq \frac{11}{9} z^* + \frac{6}{9}$.

In $\Psi_2$, the starting time of a delivery depends on two factors: the completion time of the batches in the current delivery and the available time of the vehicle. As shown in Lemma 1, the optimal starting time is

$$a_l = \begin{cases} yp & \text{if } l = 1 \\ a_{l-1} + \max \{2T, rp\} & \text{if } l \geq 2. \end{cases}$$  \hspace{1cm} (1)$$

If $rp \leq 2T$, with the exception of the first delivery $d_1$, all other deliveries have to wait for the vehicle to return before it can make a delivery. Therefore, for $z = kr + y$, the optimal service span is

$$SS = \begin{cases} yp + 2kT + T & \text{if } 1 \leq y \leq r - 1 \\ rp + (2k - 2)T + T & \text{if } y = 0. \end{cases}$$  \hspace{1cm} (2)$$

In Eq. (2), the last delivery only takes $T$ time units and the distribution will be completed. If $rp > 2T$, any delivery can be started as soon as the batches have completed processing. Therefore, the service span is

$$SS = yp + kr + T, \quad 0 \leq y \leq r - 1.$$  \hspace{1cm} (3)$$

If there is only one delivery in the optimal solution, there are at most two deliveries in the solution generated by Algorithm A2. This case is not interesting since in practice the optimal solution has more than one delivery. Thus, in the following, we assume that $x^* \geq 2$ where $x^*$ is the number of deliveries in the optimal solution.

THEOREM 2: If $rp \leq 2T$ and $x^* \geq 2$, then $SS(A2)/SS(OPT) \leq 5/3 = 1.66 \ldots$, where $SS(A2)$ and
we have

\[ SS(OPT) \text{ are the service spans of Algorithm A2 and the } \]
\[ \text{optimization algorithm, respectively.} \]

PROOF: Let the optimal solution has \( z^* = k^*r + y^* \)
\[ \text{batches, where } k^* \geq 1. \text{ We consider two cases: } k^* \leq 8 \]
\[ \text{and } k^* \geq 9. \]

CASE 1: \( k^* \leq 8. \)

We first consider the case where \( k^* = 1. \) Now \( z^* \)
\[ = k^*r + y^* = r + y^*. \text{ Since } 0 \leq y^* \leq r - 1, \text{ by Proposition 3,} \]
\[ \text{we have} \]
\[ z \leq \frac{11}{9} z^* + \frac{6}{9} = \frac{11}{9} r + \frac{11}{9} y^* + \frac{6}{9} \]
\[ \leq \frac{11}{9} r + \frac{11}{9} (r - 1) + \frac{6}{9} = \frac{22}{9} r - \frac{5}{9} < 3r \quad (4) \]

This implies that there are at most three deliveries in the solution obtained by Algorithm A2. We now consider two separate cases: \( \frac{11}{9} r + \frac{11}{9} y^* + \frac{6}{9} > 2r \) and \( \frac{11}{9} r + \frac{11}{9} y^* + \frac{6}{9} \leq 2r. \) If \( \frac{11}{9} r + \frac{11}{9} y^* + \frac{6}{9} > 2r, \) then three deliveries are needed in Algorithm A2. In the first delivery, the number of batches is \( \frac{11}{9} r + \frac{11}{9} y^* + \frac{6}{9} - 2r = \frac{11}{9} y^* + \frac{6}{9} - \frac{7}{9} r, \) and the remaining two deliveries each has \( r \) batches. By (2), we have
\[ SS(A2) = \left( \frac{11}{9} y^* + \frac{6}{9} - \frac{7}{9} r \right) p + 4T + T. \quad (5) \]

In the optimal solution, two deliveries are needed. Therefore, we have
\[ SS(OPT) = y^* p + 2T + T \quad (6) \]

Since \( \frac{6}{9} - \frac{7}{9} r < 0, \) we have
\[ \frac{11}{9} y^* + \frac{6}{9} - \frac{7}{9} r) p + 4T + T < \frac{11}{9} y^* p + 5T \]
\[ \leq \frac{5}{9} y^* p + 5T = \frac{5}{3}. \quad (7) \]

If \( \frac{11}{9} r + \frac{11}{9} y^* + \frac{6}{9} \leq 2r, \) then only two deliveries are needed. The number of batches in \( d_1 \) is \( \frac{11}{9} r + \frac{11}{9} y^* + \frac{6}{9} - r = \frac{5}{9} r + \frac{11}{9} y^* + \frac{6}{9}, \) and the number of batches in \( d_2 \) is \( r. \) So, we have
\[ SS(A2) = \left( \frac{2}{9} r + \frac{11}{9} y^* + \frac{6}{9} \right) p + 2T + T \quad (8) \]

The number of batches in the first delivery is not more than \( r; \) that is, \( \frac{5}{9} r + \frac{11}{9} y^* + \frac{6}{9} \leq r. \) Therefore, we have
\[ SS(A2) = \frac{\left( \frac{5}{9} r + \frac{11}{9} y^* + \frac{6}{9} \right) p + 3T}{\frac{y^* p + 3T}{y^* p + 3T}} \leq \frac{r p + 3T}{y^* p + 3T} \]
\[ \leq \frac{2T + 3T}{3T} = \frac{5}{3}. \quad (9) \]

We now consider the cases \( k^* = 2, 3, \ldots, 8. \) The number of batches in \( A2 \) is
\[ z \leq \frac{11}{9} (k^* r + y^*) \leq \frac{11}{9} \left( k^* r + r - 1 \right) + \frac{6}{9} \]
\[ < \frac{11}{9} (k^* + 1) r. \quad (10) \]

Therefore, we have \( \frac{2}{r} < \frac{11}{9} (k^* + 1) \) and hence
\[ SS(A2) \leq yp + \left( \frac{11}{9} (k^* + 1) - 1 \right) \times 2T + T \]
\[ \leq \frac{r p + (22k^* + 4) T}{9} \leq \frac{(22k^* + 31) T}{9}. \quad (11) \]

Hence, we have
\[ \frac{SS(A2)}{SS(OPT)} = \frac{\frac{(22k^* + 31) T}{9}}{\frac{y^* p + 2k^* T + T}{9}} \leq \frac{22k^* + 31}{18k^* + 9} \]
\[ = \frac{11}{9} + \frac{20}{18k^* + 9} \leq \frac{11}{9} + \frac{20}{45} = \frac{5}{3}. \quad (12) \]

By (9) and (12), we see that when \( k^* \leq 8, \) the ratio is no more than \( \frac{5}{3}. \)

CASE 2: \( k^* \geq 9. \)

Since \( k^* \geq 9, \) we can find \( m \) and \( i \) to make \( k^* = 9m + i, \)
where \( m \geq 1 \) and \( 0 \leq i \leq 8. \) In general, we can express the number of deliveries, \( x, \) in Algorithm A2 as
\[ \frac{z}{r} \leq k^* + \frac{2}{9} k^* + \frac{11}{9} y^* + \frac{6}{9} \leq k^* + \frac{2}{9} k^* + \frac{11}{9} (r - 1) + \frac{6}{9} \]
\[ = k^* + \frac{2}{9} k^* + 11 \frac{6}{9} r - \frac{5}{9} r \leq k^* + \frac{2}{9} k^* + 2. \quad (13) \]

We now analyze the following two cases: \( 0 \leq i \leq 4 \) and \( 5 \leq i \leq 8. \)

CASE 2.1: \( i = 0, 1, 2, 3, 4. \)

By (13), we have
\[ \frac{z}{r} < \frac{(9m + i) + 2}{9} (9m + i + 2) = 9m + i + 2m + \frac{2}{9} i + 2 \]
\[ < 11m + i + 3, \quad (14) \]
since \( i \leq 4 \). Therefore, the number of deliveries in \( A2 \) is \( x < 11m + i + 3 \). By (2), we have
\[
SS(A2) = yp + 2(11m + i + 2)T + T \leq (22m + 2i + 7)T, \tag{15}
\]
since \( yp < rp \leq 2T \). In the optimal solution, the number of batches is \( k^* + 1 \), and so
\[
SS(OPT) = y^* p + 2(9m + i)T + T \geq (18m + 2i + 1)T. \tag{16}
\]
By (15) and (16), we have
\[
\frac{SS(A2)}{SS(OPT)} \leq \frac{22m + 2i + 7}{18m + 2i + 1} = \frac{11}{9} + \frac{52 - 4i}{9(18 + 2i + 1)}. \tag{17}
\]
Since \( \frac{52 - 4i}{9(18 + 2i + 1)} \) is a decreasing function of \( i \) and since \( i \geq 0 \), we have
\[
\frac{SS(A2)}{SS(OPT)} \leq \frac{11}{9} + \frac{52}{171} \leq \frac{261}{171} < 1.53. \tag{18}
\]
CASE 2.2: \( i = 5, 6, 7, 8 \).

Since \( 1 < \frac{2}{9} i < 2 \), the number of deliveries in this case is
\[
\frac{z}{r} \leq 9m + i + 2 + 2m + \frac{2}{9} i < 11m + i + 4. \tag{19}
\]
Therefore, we have
\[
SS(A2) = yp + 2(11m + i + 3)T + T \leq (22m + 2i + 9)T, \tag{20}
\]
since \( yp \leq rp \leq 2T \). Hence, we have
\[
\frac{SS(A2)}{SS(OPT)} \leq \frac{22m + 2i + 9}{18m + 2i + 1} \leq \frac{31 + 2i}{19 + 2i}. \tag{21}
\]
The last inequality in (21) is due to the fact that \( \frac{22m + 2i + 9}{18m + 2i + 1} \) is a decreasing function of \( m \) and \( m \geq 1 \). Since \( \frac{31 + 2i}{19 + 2i} \) is a decreasing function of \( i \) and since \( i \geq 5 \), we have
\[
\frac{SS(A2)}{SS(OPT)} \leq \frac{31 + 2i}{19 + 2i} \leq \frac{41}{29} < 1.42. \tag{22}
\]
In both cases, the theorem follows. \( \square \)

Now we consider the case when \( rp > 2T \).

**THEOREM 3:** If \( rp > 2T \) and \( x^* \geq 2 \), then
\[
SS(A2)/SS(OPT) \leq 13/9 = 1.44 \ldots, \text{ where } SS(A2) \text{ and } SS(OPT) \text{ are the service spans of Algorithm 2 and the optimization algorithm, respectively.}
\]
PROOF: Since \( rp > 2T \), the vehicle needs to wait after returning from the customer before it makes another delivery. By (3), we have
\[
SS(OPT) = (y^* + k^*r)p + T = z^* p + T, \tag{23}
\]
and
\[
SS(A2) = (y + kr)p + T = zp + T \leq \left(\frac{11}{9} z^* + \frac{6}{9}\right) p + T. \tag{24}
\]
If \( z^* = 1 \) or \( 2 \), then there is only one delivery in the optimal solution, since \( r \geq 2 \). Again, this case is not interesting. So we assume that \( z^* \geq 3 \). Then, we have
\[
\frac{11}{9} z^* + \frac{6}{9} = \frac{11}{9} \left( z^* + \frac{2}{9} \right). \tag{25}
\]
Therefore, the ratio is
\[
\frac{SS(A2)}{SS(OPT)} \leq \frac{(\frac{11}{9} z^* + \frac{2}{9})p + T}{z^* p + T} \leq \frac{13 z^* p + T}{z^* p + T} < \frac{13}{9} = 1.44 \ldots. \tag{26}
\]
Theorem 3 follows. \( \square \)

We are now ready to present the main theorem of this section.

**THEOREM 4:** The running time of \( A2 \) is \( O(n \log n) \). Moreover, \( R_{2} = 5/3 \) and \( R_{A2}^* = 11/9 \).

PROOF: Step 1 of Algorithm 2 takes \( O(n \log n) \) time. By maintaining the level of the nonempty batches in a balanced binary tree, we can implement Step 2 of \( A2 \) in \( O(n \log z) \) time. Since \( z \leq n \), Step 2 can be implemented in \( O(n \log n) \) time. Step 3 takes \( O(n \log n) \) time and Step 4 takes linear time. Thus, the overall running time of Algorithm 2 is \( O(n \log n) \).

The absolute worst-case ratio follows from Theorems 2 and 3. We now examine the asymptotic worst-case ratio. In Theorem 2, we only need to consider Case 2, since \( n \) approaches infinity. From Eqs. (17) and (21), we see that
\[
\frac{SS(A2)}{SS(OPT)} \leq \frac{22m + 2i + 9}{18m + 2i + 1}
\]
and
\[
\frac{SS(A2)}{SS(OPT)} \leq \frac{22m + 2i + 9}{18m + 2i + 1}.
\]
When \( n \) approaches \( \infty \), \( m \) approaches \( \infty \). Therefore, \( R_{\infty \infty}^\infty = 22/18 = 11/9 \) in both cases. In Theorem 3, from Eq. (26), we see that

\[
\frac{\text{SS}(A2)}{\text{SS}(\text{OPT})} \leq \left( \frac{11}{9} \frac{z^* + 6}{7} \right) p + T \frac{z^* p + T}{}. \]

When \( n \) approaches \( \infty \), \( z^* \) approaches \( \infty \). Therefore, \( R_{\infty \infty}^\infty = 11/9 \). The examples achieving the ratios \( 22/18 \) and \( 11/9 \) will be shown in the following paragraphs. This completes the proof of the theorem. \( \square \)

REMARK: The example achieving 11/9 can be obtained from the example given in Johnson [13] or Dosa et al. [10], by letting \( p > 2T \). Since \( p > 2T \), the service span objective reduces to the makespan objective. The example achieving 5/3 can be obtained from the following example. There are 60 jobs to be processed. The processing time of each job is \( p \). The job sizes are shown in Table 4, where \( \epsilon \) is a sufficiently small positive real number, that is, \( \epsilon < \frac{1}{17} D \). 

<table>
<thead>
<tr>
<th>Job types</th>
<th>( JT_1 )</th>
<th>( JT_2 )</th>
<th>( JT_3 )</th>
<th>( JT_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job size</td>
<td>( \frac{1}{3} D + \epsilon )</td>
<td>( \frac{1}{3} D + 2\epsilon )</td>
<td>( \frac{1}{3} D + \epsilon )</td>
<td>( \frac{1}{3} D - 2\epsilon )</td>
</tr>
<tr>
<td>Number of jobs</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>24</td>
</tr>
</tbody>
</table>

An optimal solution is shown in Table 5. Line 1 shows the types of batches, that is, \( BT_1^* \) and \( BT_2^* \). Line 2 shows the jobs in both types and we see there are respectively three and four jobs in \( BT_1^* \) and \( BT_2^* \). Line 3 shows the total size of jobs in each batch and we find in \( \pi^* \), all batches are full. Line 4 shows the number of batches in each type. So \( z^* = 18 \) and all the completed jobs can be delivered in two deliveries. Since \( T \) is larger than \( p \), we have \( \text{SS}(\text{OPT}) = 8p + 3T \).

The solution obtained by A2 is shown in Table 6. There are three types of batches and all of them are not full. The number of batches is \( z = 22 \) and three deliveries are needed. The service span is \( \text{SS}(A2) = 2p + 5T \).

### Table 5. An optimal solution

<table>
<thead>
<tr>
<th>Batch type</th>
<th>( BT_1^* )</th>
<th>( BT_2^* )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jobs in the batch</td>
<td>( JT_1, JT_3, JT_4 )</td>
<td>( JT_2, JT_2, JT_4, JT_4 )</td>
</tr>
<tr>
<td>Toal size</td>
<td>( D )</td>
<td>( D )</td>
</tr>
<tr>
<td>Number of batches</td>
<td>12</td>
<td>6</td>
</tr>
</tbody>
</table>

### Table 6. Solution obtained by A2

<table>
<thead>
<tr>
<th>Batch type</th>
<th>( BT_1 )</th>
<th>( BT_2 )</th>
<th>( BT_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jobs in the batch</td>
<td>( JT_1, JT_2 )</td>
<td>( JT_3, JT_3, JT_3 )</td>
<td>( JT_4, JT_4, JT_4, JT_4 )</td>
</tr>
<tr>
<td>Toal size</td>
<td>( \frac{1}{3} D + 3\epsilon )</td>
<td>( \frac{1}{3} D + 3\epsilon )</td>
<td>( 1 - 8\epsilon )</td>
</tr>
<tr>
<td>Number of batches</td>
<td>12</td>
<td>4</td>
<td>6</td>
</tr>
</tbody>
</table>

By the above results, we have

\[
\frac{\text{SS}(A2)}{\text{SS}(\text{OPT})} = \frac{2p + 5T}{8p + 3T}. \]

Since \( T > p \), \( \frac{\text{SS}(A2)}{\text{SS}(\text{OPT})} \) approaches 5/3, which is the desired bound.

### 5. THE GENERAL CASE

In this section, we consider \( \Psi_3 \), where the jobs have arbitrary sizes and arbitrary processing times. Since \( \Psi_3 \) is a special case of \( \Psi_3 \) and since \( \Psi_2 \) is NP-hard in the strong sense, \( \Psi_3 \) is also strongly NP-hard. We propose Algorithm A3 (to be shown later) to solve this problem. Algorithm A3 first sorts the jobs in nonincreasing order of their processing times. Then it assigns the jobs into the batches by the First Fit method. After the batches are generated, it sorts the batches in nondecreasing order of their processing times, and processes the batches in that order. Finally, it delivers the batches by Step 4 of Algorithm A1. The description of Algorithm A3 is as shown.

**Algorithm A3.**

**Step 1.** Sort the jobs in nonincreasing order of their processing times.

**Step 2.** Assign the first job to the batch \( B_1 \). Subsequent jobs will be assigned by the First Fit method. Let there be \( z \) batches generated.

**Step 3.** Order the batches in nondecreasing order of their processing times for processing.

**Step 4.** Deliver the batches by Step 4 of Algorithm A1. \( \square \)

Since our batching algorithm is the same as the First Fit method used in bin packing, it is tempting to conjecture that the asymptotic worst-case ratio of A3 is the same as that of the First Fit method in bin packing. Xia and Tan [30] have given tighter bounds for the First Fit algorithm used in bin packing. They showed that for any list \( L \), \( C^{FF}(L) \leq \frac{\sqrt{15}}{10} C^*(L) + \frac{2}{5} \), where \( C^{FF}(L) \) and \( C^*(L) \) are the numbers of bins used by the First Fit algorithm and the optimization algorithm, respectively. Moreover, they showed that the absolute performance ratio of First Fit is at most \( \frac{12}{5} \). Unfortunately, while \( \Psi_3 \) shares some similarity with bin packing, it is quite different in that...
the width of the bin (which corresponds to the processing time of the batch) is not identical for each bin. Thus, we cannot even argue that the makespan of the schedule produced by A3 is asymptotically at most \( \frac{12}{7} \) times that of the optimal solution. To further support our argument, Zhang et al. [34] have given a complicated algorithm for a single machine and showed that the makespan of the algorithm is at most \( \frac{17}{10} \) times that of the optimal. To the best of our knowledge, this is the best worst-case ratio for makespan minimization on a single machine. The service span objective not only involves makespan but also the delivery time. Thus, it is unlikely that we can show that the asymptotic worst-case ratio of A3 is \( \frac{17}{10} \). However, we will be using the absolute performance ratio of First Fit given in [30] to bound the number of batches used by A3.

Before we analyze Algorithm A3, we will illustrate the algorithm by means of an example. The example is given in Section 1.2, where the set of jobs is shown in Table 1. The jobs in Table 1 have been ordered in nonincreasing order of their processing times. By Steps 2 and 3 of Algorithm A3, we obtain the batches as shown in Table 2. By Step 4 of Algorithm A3, we obtain the distribution schedule as shown in Table 3. In this case, the service span is 270.

From the results given by Xia and Tan [30], we have the following proposition.

**PROPOSITION 4 (Xia and Tan [30]):** \( z \leq \frac{12}{7} z^* \).

For convenience of the proof, we add dummy batches to the end of the batch list generated by Algorithm A3 so that \( z = 2z^* \). (Note that we will always be able to add dummy batches since \( z \leq \frac{12}{7} z^* < 2z^* \).) The dummy batches have zero processing time and are only added for the convenience of the proof. Since the dummy batches have zero processing times, they have no effect on the service span.

After the execution of Steps 1 and 2 of Algorithm A3, the batches are in nonincreasing order of their processing times. We let \( E_i \) and \( Q_i \) denote the batches and their processing times, respectively, \( 1 \leq i \leq z \). However, after the execution of Step 3, the order of the batches is reversed; that is, they are in nondecreasing order. We let \( B_i \) and \( P_i \) denote the batches and their processing times, respectively, after Step 3. Clearly, \( B_i = E_{z-i+1} \) and \( P_i = Q_{z-i+1} \). Now we consider the properties of \( E_i \) and \( Q_i \).

**LEMMA 3:** After the execution of Step 1 of Algorithm A3, the jobs are in nonincreasing order of their processing times. Let the job list be denoted as \( 1, 2, \ldots, j, \ldots, n \). In Step 2, if job \( e \) satisfies \( \sum_{j=1}^{i-1} s_j \leq fD \) and \( \sum_{j=1}^{i} s_j > fD \), then all the jobs in \( \{1, 2, \ldots, e\} \) are assigned to the batches in \( \{E_1, \ldots, E_{2f}\} \).

**PROOF:** We prove the lemma by contradiction. Suppose job \( e \) is assigned to the batch \( E_{2f+1} \). Clearly, the sum of the sizes of the jobs in \( E_{2i-1} \) and \( E_{2i} \) is larger than \( D \), \( 1 \leq i \leq f \). Otherwise, a job in \( E_{2i} \) can be put in \( E_{2i-1} \). When job \( e \) is assigned, jobs \( 1, 2, \ldots, e-1 \) have been assigned and they are assigned to \( E_j \), \( 1 \leq j \leq 2f \). Thus, \( \sum_{j=1}^{e-1} s_j > fD \), contradicting our assumption that \( \sum_{j=1}^{e-1} s_j \leq fD \).

After Step 2 of Algorithm A3, the batches are in nonincreasing order of their processing times; that is, \( Q_1 \geq \cdots \geq Q_{2z-1} \geq Q_{2z} \). We also order the batches in the optimal solution in nonincreasing order of their processing times; that is, \( Q^*_1 \geq \cdots \geq Q^*_z \geq Q^*_{z+1} \).

**LEMMA 4:** \( Q_{2i} \leq Q_{2i-1} \leq Q^*_i \) for \( 1 \leq i \leq z^* \).

**PROOF:** By Lemma 3, we see that job \( e \) is assigned to a batch in \( \{E_1, \ldots, E_{2f}\} \). So any job \( j \) with \( j < e \) is also assigned to a batch in \( \{E_1, \ldots, E_{2f}\} \). Therefore, any job \( j \) assigned to the following batches, that is, \( E_i (i = 2f+1, 2f+2, \ldots) \), will satisfy \( p_j \leq p_e \). Because the processing time of a batch is no more than the longest processing time of all the jobs in the batch, we have

\[
Q_{2f+2} \leq Q_{2f+1} \leq p_e. \tag{27}
\]

In the optimal solution, since \( \sum_{j=1}^{e} s_j > fD \), the total size of the \( e \) jobs is larger than \( fD \). Therefore, the jobs in \( \{1, 2, \ldots, e\} \) cannot all be assigned to the first \( f \) batches. Therefore, we have

\[
Q_{f+1}^* \geq p_e. \tag{28}
\]

Combining (27) and (28), we have \( Q_{2f} \leq Q_{2f-1} \leq Q^*_i \) for all \( f \geq 1 \). If \( f = 1 \), all jobs can be put into one batch in the optimal solution. Clearly, Algorithm A3 also generates one batch. So, \( Q_1 = Q^*_1 \) and \( Q_{2i} \leq Q_{2i-1} \leq Q^*_i \) also holds.

In Step 3 of Algorithm A3, the batches are reordered and now they are in nondecreasing order of their processing times. Denote the batches and their processing times as \( B_i \) and \( P_i \), respectively. By Lemma 1, we may assume that the batches are also ordered in nondecreasing order of their processing times in an optimal solution. By Lemma 4, we have the following lemma.

**LEMMA 5:** \( P_{2i-1} \leq P_{2i} \leq P^*_i \) for \( i = 1, 2, \ldots, z^* \) and \( \sum_{i=1}^{z^*} P_i \leq 2 \sum_{i=1}^{z^*} P^*_i \).

Let \( z^* = k^* r + y^* \). If \( k^* = 0 \), then there is only one delivery in the optimal solution and there are at most two deliveries in the solution generated by Algorithm A3. This case is not interesting since it is rare to have only one delivery. Therefore, we investigate the cases where there are more than one delivery in the optimal solution; that is, \( z^* \geq 2 \).

Naval Research Logistics DOI 10.1002/nav
Before we give a general conclusion on the performance of A3, we investigate the cases where \( x^* = 2 \) and 3.

**Lemma 6:** When \( x^* = 2 \), \( \frac{SS(A3)}{SS(OPT)} < \frac{7}{3} \).

**Proof:** Let \( z^* = r + y^* \) where \( y^* \) is an integer and \( 1 \leq y^* \leq r \). By Lemma 1, we may assume that in \( \pi^* \), the first \( y^* \) batches are delivered in the first delivery and the next \( r \) batches are delivered in the second delivery. By Proposition 4, we have \( z < \frac{2}{3} z^* = \frac{2}{3}(r + y^*) \leq \frac{2}{3}r < 4r \). This implies that there are at most four deliveries in \( \pi \). Now we consider the worst case when there are four deliveries and analyze two cases: \( 2T \geq \sum_{i=y^*+1}^{y^*+r} P_i^* \) and \( 2T < \sum_{i=y^*+1}^{y^*+r} P_i^* \). By Step 4 of A3, we see that the batches are in nondecreasing order of their processing times. The number of batches in the first delivery is \( 2y^* - r \) and the number of batches in the next three batches is \( r \).

**Case 1:** \( 2T \geq \sum_{i=y^*+1}^{y^*+r} P_i^* \).

The total processing time of the batches in the second delivery is \( \sum_{i=d_2}^{d_3} P_i^* = \sum_{i=y^*+1}^{y^*+r} P_i^* \). By Lemma 5, we have \( P_{2y^*+1} \leq P_{2y^*+2} \leq P_{2y^*+1} \leq \cdots \leq P_{2y^*+2r} \leq P_{2y^*+r} \). Hence,

\[
\sum_{B_i \in d_2} P_i \leq \sum_{B_i \in d_3} P_i \leq \sum_{B_i \in d_2^*} P_i^*.
\]

Since the batches are in nondecreasing order of their processing times, we have \( 2T \geq \sum_{i=y^*+1}^{y^*+r} P_i^* \). Now compare \( d_2 \) and \( d_2^* \) in a similar way. We have \( \sum_{B_i \in d_3} P_i \leq \sum_{B_i \in d_2^*} P_i^* \). Therefore, we have

\[
\frac{SS(A3)}{SS(OPT)} = \frac{\sum_{B_i \in d_3} P_i + 7T}{\sum_{B_i \in d_2^*} P_i^* + 3T} < \frac{7}{3}.
\]

**Case 2:** \( 2T < \sum_{i=y^*+1}^{y^*+r} P_i^* \).

In this case, the service span of the optimal solution is \( SS(OPT) = \sum_{B_i \in d_2^*} P_i^* + T \). For \( SS(A3) \), we consider the following four cases.

**Case 2.1:** \( 2T \geq \sum_{B_i \in (d_2 \cup d_3 \cup d_4)} P_i \). Here we have

\[
\frac{SS(A3)}{SS(OPT)} = \frac{\sum_{B_i \in d_3} P_i + 7T}{\sum_{B_i \in d_2^*} P_i^* + 3T} < \frac{7}{3}.
\]

**Case 2.2:** \( \sum_{B_i \in (d_2 \cup d_3 \cup d_4)} P_i \leq 2T \leq \sum_{B_i \in (d_2 \cup d_3 \cup d_4)} P_i \). If \( 4T < \sum_{B_i \in (d_2 \cup d_3 \cup d_4)} P_i \), we have

\[
\frac{SS(A3)}{SS(OPT)} = \frac{\sum_{B_i \in (d_2 \cup d_3 \cup d_4)} P_i + T}{\sum_{B_i \in d_2^*} P_i^* + T} < \frac{7}{3}.
\]

If \( 4T \geq \sum_{B_i \in (d_2 \cup d_3 \cup d_4)} P_i \), we have

\[
\frac{SS(A3)}{SS(OPT)} = \frac{\sum_{B_i \in (d_2 \cup d_3 \cup d_4)} P_i + 7T}{\sum_{B_i \in d_2^*} P_i^* + 7T} < \frac{7}{3}.
\]

**Case 2.3:** \( \sum_{B_i \in d_3} P_i \leq 2T < \sum_{B_i \in (d_2 \cup d_3 \cup d_4)} P_i \). The result is the same as Case 2.2.

**Case 2.4:** \( 2T < \sum_{B_i \in d_3} P_i \). In this case, we have

\[
\frac{SS(A3)}{SS(OPT)} = \frac{\sum_{B_i \in d_3} P_i + T}{\sum_{B_i \in d_2^*} P_i^* + T} < \frac{7}{3}.
\]

By the above discussions, we have \( \frac{SS(A3)}{SS(OPT)} < \frac{7}{3} \) when \( x^* = 2 \).

Now we consider the case when \( x^* = 3 \).

**Lemma 7:** When \( x^* = 3 \), \( \frac{SS(A3)}{SS(OPT)} < \frac{11}{5} \).

**Proof:** When \( x^* = 3 \), there are three deliveries in \( \pi^* \). Let \( z^* = 2r + y^* \), where \( 1 \leq y^* \leq r \). By Proposition 4, we have \( z < \frac{2}{3} z^* = \frac{2}{3}(2r + y^*) \leq \frac{2}{3}r < 6r \). So, there are at most six deliveries in \( \pi \). Now we consider the worst case when there are six deliveries, that is, \( x = 6 \).

**Case 1:** \( 4T \geq \sum_{B_i \in (d_2 \cup d_3 \cup d_4)} P_i^* \).

In this case, all the deliveries in \( \pi^* \) needs to wait before distribution. So

\[
SS(OPT) = \sum_{B_i \in d_3} P_i^* + 5T \geq \sum_{i=1}^{3} \sum_{B_i \in d_3} P_i^* + T = \sum_{i=1}^{3} P_i^* + T.
\]

In considering the service span of \( \pi \), there are two cases to consider.

1. If \( 10T \leq \sum_{i=2}^{6} \sum_{B_i \in d_3} P_i \), then \( SS(A3) = \sum_{i=1}^{6} \sum_{B_i \in d_3} P_i + T \). By Lemma 5, we have

\[
SS(OPT) = \sum_{i=1}^{6} \sum_{B_i \in d_3} P_i + T.
\]
SS(A3) \over SS(OPT)} = \frac{\sum_{i=1}^{6} \sum_{B_i \in d_i} P_i + T}{\sum_{i=1}^{6} \sum_{B_i \in d_i} P_i + T} \leq 2.

(36)

2. If $10T > \sum_{i=1}^{6} \sum_{B_i \in d_i} P_i$, then we have

$$\frac{SS(A3)}{SS(OPT)} = \frac{\sum_{i=1}^{6} \sum_{B_i \in d_i} P_i + 11T}{\sum_{i=1}^{6} \sum_{B_i \in d_i} P_i + 11T} \leq 2.\ (37)$$

CASE 2: $4T < \sum_{B_i \in (d' \cup d)} P_i$.

In this case, the service span of $\pi^*$ is

$$SS(OPT) = \sum_{i=1}^{2} \sum_{B_i \in d_i} P_i + T > \sum_{B_i \in d_i} P_i + 5T.\ (38)$$

For SS(A3), we consider the following two cases.

1. If $10T \leq \sum_{i=1}^{6} \sum_{B_i \in d_i} P_i$, then we have

$$SS(A3) \over SS(OPT)} = \frac{\sum_{i=1}^{6} \sum_{B_i \in d_i} P_i + T}{\sum_{i=1}^{6} \sum_{B_i \in d_i} P_i + T} \leq 2.\ (39)$$

2. If $10T > \sum_{i=1}^{6} \sum_{B_i \in d_i} P_i$, then we have

$$SS(A3) \over SS(OPT)} = \frac{\sum_{i=1}^{6} \sum_{B_i \in d_i} P_i + 11T}{\sum_{i=1}^{6} \sum_{B_i \in d_i} P_i + 11T} \leq 2.\ (40)$$

By the above discussions, Lemma 7 follows.

We now investigate the worst case ratio of A3 for the general case.

THEOREM 5: If $x^* \geq 2$, then $SS(A3)/SS(OPT) \leq \frac{7}{2}$, where $SS(A3)$ and $SS(OPT)$ are the service spans of Algorithm A3 and an optimization algorithm, respectively.

PROOF: In Lemmas 6 and 7, we have obtained the worst case ratios for $x^* = 2$ and $x^* = 3$. Here we discuss the result for $x^* \geq 4$. Let $z^* = k^* + y^*$, where $k^* \geq 3$ and $1 \leq y^* \leq r$. By Proposition 4, the number of batches in $\pi$ satisfies the following relation: $z \leq \frac{12}{7}z^* = \frac{12}{7}(k^* + y^*) \leq \frac{12}{7}(k^* + 1)r$.

Since $(2k^*+1) - \frac{12}{7}(k^*+1) = \frac{2}{7}k^* - \frac{5}{7} > 0$ (because $k^* \geq 3$), there are at most $2k^* + 1$ deliveries. Let us consider a different algorithm $A3'$ which is the same as Algorithm A3, except that in Step 2, it add dummy batches with zero processing times so that $z' = (2k^* + 1)r$. Then the batches will be delivered by Step 4 in exactly $2k^* + 1$ deliveries. It is clear that $SS(A3) \leq SS(A3')$. Thus, if we can obtain an upper bound of the ratio $SS(A3')/SS(OPT)$, then $SS(A3)/SS(OPT)$ will also be bounded above by this upper bound. In the following, we will derive an upper bound for $SS(A3')/SS(OPT)$.

Let $\pi'$ be the solution obtained by $A3'$. There are $z' = (2k^* + 1)r$ batches in $A3'$. Since there are $2k^* + 1$ deliveries in $A3'$, there are $2y^*$ batches assigned in the first delivery. Let $P_i' \leq P_i \leq \cdots \leq P_z'$ be the processing time of the batches. Then we have

$$SS(A3') \leq \sum_{i=1}^{2y^*} P_i' + \max \left\{ 2T, \sum_{i=2y^*+1}^{2y^*+r} P_i' \right\} + \cdots + \max \left\{ 2T, \sum_{i=2y^*+(2k^*-r+1)+1}^{2y^*+(2k^*+1)r} P_i' \right\} + T.\ (41)$$

In the optimal solution, we have

$$SS(OPT) = \sum_{i=1}^{y^*} P_i + \max \left\{ 2T, \sum_{i=y^*+1}^{y^*+r} P_i' \right\} + \cdots + \max \left\{ 2T, \sum_{i=y^*+(k^*-r)+1}^{y^*+(k^*+1)r} P_i' \right\} + T.\ (42)$$

By Lemma 5, $P_i' \leq P_i \leq P_i^*$. Therefore, we have

$$\sum_{i=2y^*+(2k^*-r)+1}^{2y^*+(2k^*+1)r} P_i' \leq \sum_{i=2y^*+(2k^*-r)+1}^{2y^*+(2k^*+1)r} P_i \leq \sum_{i=y^*+(k^*-r)+1}^{y^*+(k^*+1)r} P_i^*,\ (43)$$

and hence

$$\max \left\{ 2T, \sum_{i=2y^*+1}^{2y^*+r} P_i' \right\} + \max \left\{ 2T, \sum_{i=2y^*+r+1}^{2y^*+(2k^*+1)r} P_i' \right\} + \cdots$$

Naval Research Logistics DO 10.1002/nav
\[
\begin{align*}
+ \max \left\{ 2T, \sum_{i=2y^*+(2k^*+2)r+1}^{2y^*+(2k^*+1)r} P_i' \right\} \\
+ \max \left\{ 2T, \sum_{i=2y^*+(2k^*-2)r+1}^{2y^*+2k^*r} P_i' \right\} \\
\leq 2 \left( \max \left\{ 2T, \sum_{i=y^*+1}^{y^*+r} P_i^* \right\} + \ldots \right) \\
+ \max \left\{ 2T, \sum_{i=y^*+(k^*-1)r+1}^{2y^*+k^*r} P_i^* \right\} . \quad (44)
\end{align*}
\]

For simplicity, denote (44) as \( \beta(A^3) \leq 2\beta^* \). Then, we have
\[
\frac{SS(A^3)}{SS(OPT)} = \frac{\sum_{i=1}^{2y^*} P_i' + \beta(A^3) + T}{\sum_{i=1}^{\infty} P_i^* + \beta^* + T}. \quad (45)
\]

By Lemma 5, we have
\[
\sum_{i=1}^{2y^*} P_i' \leq 2 \sum_{i=1}^{\infty} P_i^*. \quad (46)
\]

Therefore,
\[
\frac{SS(A^3)}{SS(OPT)} < 2. \quad (47)
\]

Therefore, \( SS(A^3)/SS(OPT) < 2 \) when \( x^* \geq 4 \). By Lemmas 6 and 7, we have \( SS(A^3)/SS(OPT) < 7/3 \) when \( x^* = 2 \) and \( x^* = 3 \). So the theorem holds. \( \square \)

**THEOREM 6:** The running time of Algorithm A3 is \( O(n \log n) \). Moreover, \( R_{A3} \leq 7/3 = 2.33 \ldots \) and \( R_{A3}^\infty \leq 2 \).

**PROOF:** Steps 1 and 2 of Algorithm A3 can be done in \( O(n \log n) \) time. Steps 3 and 4 can be done in linear time. Thus, the overall running time of Algorithm A3 is \( O(n \log n) \).

The absolute worst-case ratio follows from Theorem 5. We now show the asymptotic worst-case ratio.

If \( 2T > \sum_{i=2y^*-r+1}^{2y^*} P_i' \), then by Eq. (45), we have
\[
\frac{SS(A^3)}{SS(OPT)} \leq \frac{\sum_{i=1}^{2y^*} P_i' + \beta(A^3) + T}{\sum_{i=1}^{\infty} P_i^* + \beta^* + T}.
\]

When \( n \) approaches \( \infty \), \( SS(A^3)/SS(OPT) \) approaches \( \frac{\beta(A^3)}{\beta} \). Therefore, \( R_{A3}^\infty \leq 2 \).

Conversely, if \( 2T \leq \sum_{i=2y^*-r+1}^{2y^*} P_i' \), then we have
\[
\frac{SS(A^3)}{SS(OPT)} = \frac{\sum_{i=1}^{\infty} P_i' + T}{\sum_{i=1}^{\infty} P_i^* + T} < 2.
\]

Therefore, \( R_{A3}^\infty \leq 2 \). In both cases, \( R_{A3}^\infty \leq 2 \). Therefore, \( R_{A3}^\infty \leq 2 \).

**6. CONCLUSIONS**

In this article, we have studied a class of integrated scheduling problem of production and distribution. The objective is to minimize the service span, which lasts from the beginning of the production until the completion of the last delivery. In the production, a single batch-processing machine is used to process jobs with arbitrary sizes and processing times. In the distribution, a single vehicle with fixed capacity is used to deliver products. We first propose an optimal algorithm for the special case where the jobs have identical sizes. Then we propose an approximation algorithm for the special case where the jobs have identical processing times, since this special case is NP-hard in the strong sense. Finally, we propose an approximation algorithm for the general problem. For the approximation algorithms we show the absolute worst-case ratios and asymptotic worst-case ratios. All of the proposed algorithms run in \( O(n \log n) \) time.

From the computational complexity point of view, we have delineated a sharp boundary between problems that are solvable in polynomial time and problems that are strongly NP-hard. When the sizes of the jobs are identical, then the problem is solvable in polynomial time even when the jobs have different processing times. Conversely, when the sizes of the jobs are different, then the problem is strongly NP-hard even when the jobs have identical processing times. Thus, the single parameter, size, is the deciding factor of the complexity of the problem.

While our problem share some similarity with the bin packing problem, the two problems are in fact quite different. First, the width of each bin in the bin packing problem is identical while the width of each bin in our problem is different. Second, our problem involves delivery time which is absent in the bin packing problem. Conversely, we can use the result of bin packing to get an upper bound on the number of batches. This is helpful in our analysis of the absolute and asymptotic worst-case ratios.

There are some interesting directions for future work. In the production side, machine configurations may be more complex in practice. For example, parallel machines and flow shops are often encountered in food processing industries. Since the single-machine integrated scheduling problem is NP-hard, the problems with parallel machines and flow shops are also NP-hard. Approximation algorithms for these problems deserve more research. Other objective functions can also be studied. For example, the total cost of production and distribution is an interesting objective. Multiobjective problems are also interesting directions for future work. For example, integrated scheduling problems of production, inventory and distribution have not been considered in the literature. This will involve several objectives such as service span and total cost.
ACKNOWLEDGMENTS

This work is partly supported by the National Natural Science Foundation of China under Grants 71202048, 71131002, 71071045, and 71471052. This work is also partly supported by the Specialized Research Fund for the Doctoral Program of Higher Education of China under Grant 20110111120015. We would like to thank the two reviewers whose suggestions have greatly improved the article.

REFERENCES

[26] E. Selvarajah and G. Steiner, Approximation algorithms for the supplier’s supply chain scheduling problem to minimize delivery and inventory holding costs, Oper Res 57 (2009), 426–438.