JOURNAL OF SCHEDULING
J. Sched. 2001; **4**:3–24

Machine scheduling with transportation considerations

Chung-Yee Lee[1,*,†] and Zhi-Long Chen[2,‡]

[1]*Department of Industrial Engineering, Texas A&M University, College Station, TX 77843-3131, U.S.A.*
[2]*Department of Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104-6315, U.S.A.*

SUMMARY

In most manufacturing and distribution systems, semi-finished jobs are transferred from one processing facility to another by transporters such as automated guided vehicles (AGVs) and conveyors, and finished jobs are delivered to customers or warehouses by vehicles such as trucks. Most machine scheduling models assume either that there are an infinite number of transporters for delivering jobs or that jobs are delivered instantaneously from one location to another without transportation time involved. In this paper, we study machine scheduling problems with explicit transportation considerations. Models are considered for two types of transportation situations. The first situation involves transporting a semi-finished job from one machine to another for further processing. The second appears in the environment of delivering a finished job to the customer or warehouse. Both transportation capacity and transportation times are explicitly taken into account in our models. We study this class of scheduling problems by analysing their complexity. We show that many problems are computationally difficult and propose polynomial or pseudo-polynomial algorithms for some problems. Copyright © 2001 John Wiley & Sons, Ltd.

1. INTRODUCTION

In most manufacturing and distribution systems, semi-finished jobs are transferred from one facility to another for further processing through material handling systems such as automated guided vehicles (AGVs) and conveyors, and finished jobs are delivered to customers or warehouses by vehicles such as trucks. In the last four decades, many books and numerous papers have been published in the area of machine scheduling. However, most of the published literature explicitly or implicitly assumes that either there are an infinite number of transporters for delivering jobs or that jobs are transported instantaneously from one location to another without transportation time invloved.

Problems that address the optimal co-ordination of machine scheduling and job transporting are certainly more practical than those scheduling problems that do not take these factors into consideration. It is clear that seemingly different operations in manufacturing and distribution systems must be co-ordinated carefully in order to achieve ideal overall system performance. The

*Correspondence to: Chung-Yee Lee, Department of Industrial Engineering, Texas A&M University, College Station, TX 77843-3131, U.S.A.
†E-mail: cylee@acs.tamu.edu
‡E-mail: zlchen@seas.upenn.edu

Contract/grant sponsor: NSF; contract/grant numbers: DMI-9610229, DMI-9988427 and DMI-9908221

recent tremendous interest in supply chain management in both academic and industrial communities has demonstrated the importance of such coordination.

In this paper, we investigate machine scheduling models that explicitly consider constraints on both transportation capacity and transportation times. We consider two types of transportation. The first type is intermediate transportation in a flow shop where jobs are transported from one machine to another for further processing. The second type is the transportation necessary to deliver finished jobs to the customer. We consider various scheduling problems involving these two types of transportation where both transportation capacity and transportation times are explicitly taken into account.

The earliest scheduling paper that explicitly considers the transportation factor is probably the one by Maggu and Das [1]. They consider a two-machine flow shop makespan problem with unlimited buffer spaces on both machines in which there are a sufficient number of transporters so that whenever a job is completed on the first machine it can be transported, with a job-dependent transportation time, to the second machine immediately. They generalize the well-known Johnson's rule [2] to solve their problem. Maggu *et al.* [3] consider the same problem with the additional constraint that some jobs must be scheduled consecutively. They show that a similar rule solves the problem. Kise [4] studies a similar problem but with only one transporter with a capacity of one (i.e. it can transport only one job at a time). He shows that this problem is ordinarily NP-hard even with job-independent transportation times.

Following most of the flow shop scheduling literature, Maggu and Das [1], Maggu *et al.*, [3], and Kise [4] assume that intermediate buffer space is infinite. Several closely related problems, including buffer space constraints, have been studied in the literature. Stern and Vitner [5] consider a two-machine flow shop makespan problem where there is only one transporter with a capacity of one. They assume that transportation times are job-dependent and that there is no intermediate buffer space at either machine. They formulate the problem as an asymmetric traveling salesman problem and give a polynomial-time heuristic. As pointed out later by Ganesharajah *et al.* [6], this problem is strongly NP-hard. Panwalkar [7] considers the same problem as the one studied by Stern and Vitner except that the buffer space at the second machine is infinite. He provides an optimal polynomial-time algorithm. Stevens and Germill [8] provide heuristics to the problem studied by Panwalkar except that their objective is minimizing maximum lateness.

Other related problems have been studied by Mitten [9], Maggu *et al.* [10], Langston [11], Yu [12], and Ganesharajah *et al.* [6]. Mitten [9] presents a simple rule to solve a two-machine flow shop makespan problem in which each job has a starting time lag and a completion time lag. A starting (completion) time lag forces the starting (completion) time of a job on the second machine to be at least some time later than that on the first machine. Clearly, if each job has either a starting time lag or a completion time lag, but not both, the resulting problem is then equivalent to the problem considered by Maggu and Das [1]. Maggu *et al.* [10] show that a simple rule can solve a more general problem that has both the features of the problem of Maggu and Das [1] and those of the problem of Mitten [9]. Langston [11] analyses some heuristics for a k-station flow shop makespan problem where each station has a number of machines that can be used to process jobs, and there is only one transporter with a capacity of one to transport jobs with transportation times dependent on the physical locations of the origin and destination machines. Yu [12] considers various special cases of the problem studied by Maggu and Das [1]. Ganesharajah *et al.* [6] study problems of jointly scheduling machines and AGV in repetitive

manufacturing systems with unidirectional loops for AGV movements. Each AGV can transport only one job at a time. There is an input buffer and output buffer at each machine with a buffer size of one. The objective is the minimization of the steady-state cycle time required to finish jobs in a minimal job set under some prespecified AGV dispatching policies. They analyse the computational complexity of various cases.

Another line of research focuses on problems where the completion time of a job is defined as the time when the job arrives at its customer. Potts [13], and Hall and Shmoys [14] study a single-machine problem with unequal job arrival times and delivery times. They implicitly assume that there are a sufficient number of vehicles such that whenever a job is completed on a machine it is delivered immediately to its customer. They provide heuristics and error bound analysis. Woeginger [15] considers the same problem but with parallel machines and equal job arrival times. He gives some heuristics with a constant worst-case bound guarantee. Herrmann and Lee [16], Chen [17], Cheng *et al.* [18], and Yuan [19] also consider scheduling problems which include delivery. However, they do not consider transportation times, i.e. they assume that deliveries are made instantaneously. The completion time of a job is defined as the time when a job (probably with some other jobs together as a batch) is ready for delivery.

There are also papers that address scheduling decisions from the material handling systems design viewpoint. Recent reviews in the design and analysis of material handling systems can be found in References [6, 20]. Papers studying scheduling issues in the underlying material handling systems include, among others, Raman *et al.* [21], Jaikumar and Solomon [22], Kise *et al.* [23], Levner *et al.* [24], Bilge and Ulusoy [25] and Agnetis *et al.* [26].

Note that the first type of transportation problem we consider is the same as the one studied in References [1, 4]. The second type is the same as the one considered in Reference [13]. Our problems may be viewed as extensions of the few problems considered in the literature [1, 4, 13]. While most related literature has only addressed special problems, we give a systematic and unified treatment of these important problems by classifying their computational complexity. While we have been conducting this research, Hurink and Knust [27] have independently studied some cases of a problem with *the first type of transportation*, which are closely related to one of the problems we consider. They assume that there is only one transporter available which can carry only one job at a time and that the returning time of the transporter is zero. Motivated by problems in general logistics management, we do not assume zero returning time, and in general, we allow multiple transporters, each capable of carrying multiple jobs.

The remainder of this paper is organized as follows. In Section 2, we introduce our notation and describe the two types of transportation and their underlying scheduling problems. In Sections 3 and 4, we then study problems with the first and second type of transportation, respectively. Finally, we conclude the paper in Section 5.

2. PROBLEMS AND NOTATION

As discussed above, we consider two types of transportation. Usually, type-1 transportation happens inside a manufacturing facility, while type-2 happens between a manufacturing facility and the customers or warehouses. Since our focus in this paper is on scheduling with transportation constraints, we will not consider other types of constraints, such as buffer space.

2.1. Scheduling with type-1 transportation

The machine configuration inside a manufacturing facility can be flow shop, job shop, open shop, or other types. In this paper, we consider type-1 transportation mainly in a flow shop environment. Our problems with type-1 transportation can be described in general as follows. We are given a set of n jobs to be processed on m machines in a flow shop. Each job must first be processed on machine 1, then machine 2, etc., and finally machine m. The processing time of job j on machine k is p_{jk}. We assume that all of the jobs start at machine 1. After a job is processed on machine k, it is transported to machine $k+1$ by a transporter. There are a total of v identical transporters initially located at machine 1. Each transporter has a capacity of c, i.e. it can carry up to c jobs in one shipment. The transportation time from machine k to machine $k+1$ is denoted by $t_{k,k+1}$, which is assumed to be independent of the jobs being transported. We assume that loading and unloading times are included in processing times of jobs and are not considered separately.

Let C_j denote the completion time of job j, that is the time when job j is completed on the last machine m. We are mainly concerned about minimizing makespan $C_{\max}$ and total completion time $\sum C_j$. We follow the commonly used three-field notation $\alpha|\beta|\gamma$ for machine scheduling problems (see, e.g. Reference [28]). In the α field, we use notation 'TF' to denote a flow shop problem with transportation between machines. Hence, $\mathrm{TF}_2|v=x,\ c=y|C_{\max}$ represents the 2-machine flow shop makespan problem with x transporters, each with capacity y. We note that following this notation the problem considered by Maggu and Das [1] can be denoted as $\mathrm{TF}_2|v \geqslant n,\ c \geqslant 1|C_{\max}$.

2.2. Scheduling with type-2 transportation

A problem with type-2 transportation can be described as follows. A set of n jobs is to be processed at a manufacturing facility. Depending on specific problems characteristics, the facility can be a single machine, a set of parallel machines, or a series of flow shop machines. After the processing, jobs must be delivered to the corresponding customers or warehouses. For ease of exploration, we only consider the case where all customers are located in close proximity to each other. Hence, the transportation time is assumed to be the same for each job. As in the case of type-1 transportation, we assume that there are v identical vehicles available to deliver jobs. Each vehicle has a capacity of c and is initially located at the manufacturing facility. The transportation time from the manufacturing facility to the customer is t_1 and that from the customer to the manufacturing facility is t_2, where t_1 and t_2 are independent of the jobs being transported. The completion time of a job is defined as the time when it arrives at the customer.

Note that we may consider a problem with type-2 transportation as a special case of a problem with type-1 transportation by treating the customer as the last machine on which processing times of jobs are zero. However, for notational convenience and for future research purposes (we will consider routing decisions between the manufacturing facility and customers in our further research), we treat problems with type-2 transportation as a different problem class.

For a problem with type-2 transportation, by the three-field notation $\alpha|\beta|\gamma$, we use '$1 \to D$', '$\mathrm{Pm} \to D$', or '$\mathrm{Fm} \to D$' in the α field to represent problems where jobs are first processed on a single machine, m parallel machine, or m flow shop machines, respectively, and then delivered to the customer. For example, $P_2 \to D|v \geqslant 2,\ c=1|\sum C_j$ represents the total completion time problem with two identical parallel machines and 2 or more delivery vehicles, each with capacity 1. Note that the problem considered by Potts [13] can then be denoted as $1 \to D|r_j,\ v \geqslant n,\ c \geqslant 1|C_{\max}$.

2.3. Two frequently cited NP-hard problems

Throughout this paper, we will use the 3-*partition problem*, a well-known strongly NP-hard problem, and the *equal-size partition problem*, a well-known ordinarily NP-hard problem [29], to show the NP-hardness of our scheduling problems. For ease of presentation, we describe them here so that we can directly cite them later when necessary.

2.3.1. 3-Partition problem (*3-PP*). Given $3h$ items, $H = \{1, 2, \ldots, 3h\}$, each item $j \in H$ has a positive integer size a_j satisfying $b/4 < a_j < b/2$, and $\sum_{j=1}^{3h} a_j = hb$, for some integer b, the question asks whether there are h disjoint subsets $H_1, H_2, \ldots, H_h$ of H such that each subset contains exactly three items and its total size is equal to b.

2.3.2. Equal-size partition problem (*ESPP*). Given $2h$ items, $H = \{1, 2, \ldots, 2h\}$, each item $i \in H$ has a positive integer size a_i, such that $\sum_{i \in H} a_i = 2A$, for some integer A. The question asks if there is a subset $G \subseteq H$ such that $\sum_{i \in G} a_i = A$ and there are h items in G.

3. SCHEDULING PROBLEMS WITH TYPE-1 TRANSPORTATION

It is known (see, e.g. Reference [28]) that the classical 3-machine flowshop makespan problem without transportation $F_3||C_{\max}$ is strongly NP-hard, while the 2-machine problem $F_2||C_{\max}$ is polynomially solvable by Johnson's rule [2]. Thus, any 3-machine problem with type-1 transportation, except in special cases, must be strongly NP-hard. Hence, we focus on problems with two machines. We will clarify the complexity of 2-machine flow shop problems with type-1 transportation and identify some polynomially solvable cases. We use t_1 and t_2 to denote, respectively, the transportation time from machine 1 to machine 2 and that from machine 2 to machine 1.

We note that a transporter in a 2-machine flow shop problem with type-1 transportation may be viewed as a 'machine' (whose duty is to transport jobs) between the two real machines (whose duty is to process jobs). However, as the transporter is returning from machine 2 to machine 1, it is occupied but not carrying any job. Hence, a transporter is different from what a real machine is supposed to be in traditional scheduling problems. Thus, the problem $\mathrm{TF}_2|v = 1, c = 1|C_{\max}$ (to be studied in Section 3.1) is different from the traditional problem without transportation $F_3|p_{j2} \equiv p|C_{\max}$.

In a given schedule, we call all the jobs transported together in one shipment from machine 1 to machine 2 a *batch*. We use the notation B_k to denote the kth batch of jobs, and prescribe that a batch transported earlier has a smaller index. We let d_k denote the departure time of batch B_k from machine 1, and let C_{j1} and C_{j2} denote the completion times of job j on machine 1 and machine 2, respectively. Sometimes, we also use C_j to denote the completion time of job j on machine 2.

It is easy to prove that the following property holds for all 2-machine flow shop problems with type-1 transportation and a regular objective function, $\mathrm{TF}_2|v \geqslant 1, c \geqslant 1|f(C_1, \ldots, C_n)$, where the objective $f(C_1, \ldots, C_n)$ is a non-decreasing function of $C_1, \ldots, C_n$.

Property 1. There exists an optimal schedule for the problem $\mathrm{TF}_2|v \geqslant 1, c \geqslant 1|f(C_1, \ldots, C_n)$ that satisfies the following conditions.

(i) Jobs are processed on machine 1 without idle time.
(ii) Jobs transported in the same batch are processed consecutively without idle time on both machines.

(iii) Jobs finished earlier on machine 1 are delivered earlier to machine 2. Furthermore, the sequence of jobs on machine 1 is the same as that on machine 2. Namely, it is a permutation schedule.
(iv) Vehicle k delivers all of the batches with an index that can be written as $k + qv$ for some integer $q \geqslant 0$.
(v) The departure times of two consecutive batches delivered by the same vehicle k (i.e. batches B_{k+qv} and $B_{k+(q+1)v}$ for some integer $q \geqslant 0$) satisfy that either $d_{k+(q+1)v} = d_{k+qv} + t$, or $d_{k+(q+1)v}$ is the completion time of the last job in $B_{k+(q+1)v}$ on machine 1, where $t = t_1 + t_2$ is the transportation time of a round trip between machine 1 and machine 2.

Proof:

(i) If there exists idle time, we can always move the subsequent jobs earlier without increasing the objective value.
(ii) If they are not processed consecutively, then we can always move jobs later on a machine such that all of the jobs of a batch are processed consecutively and the completion time of the batch is not delayed. The objective value does not increase as a result of these moves. Furthermore, similar to (i), we can see that there should not be idle time between jobs in the same batch.
(iii) Since we assume that each job has the same size and that the objective function is regular, it is clear that jobs finished earlier on machine 1 are delivered earlier to machine 2. Furthermore, if the job sequence on machine 2 is not the same as that on machine 1, then we can re-sequence jobs on machine 2 to be in the same order as those on machine 1 without increasing the objective value.
(iv) Since we assume that the transportation time is independent of jobs and all the vehicles are identical, it is optimal to use all of the vehicles once before we use them again. Thus, we can index deliveries by vehicle k as $k + qv$.
(v) When vehicle k finishes its $(q + 1)$th delivery and returns to machine 1, there are two possible cases. Either this vehicle will immediately transport jobs at time $d_{k+(q+1)v} = d_{k+qv} + t$, or it will wait until more jobs are complete and hence $d_{k+(q+1)v}$ is the completion time of the last job in $B_{k+(q+1)v}$ on machine 1. □

Before we start investigating more complex problems, we first note that among all of the 2-machine flow shop problems with type-1 transportation, the simplest one is the problem $\mathrm{TF}_2|v = n, c \geqslant 1|C_{\max}$. In this problem, whenever a job is finished on machine 1, there is always a transporter available to transport it to machine 2. Hence, there is actually no constraint on transportation capacity associated with this problem although there is a transportation time between the two machines. This problem is a special case of the problem studied by Maggu and Das [1] in which transportation times are job dependent.

In the following subsections, we consider various problems with constraints on transportation capacity and on transportation times.

3.1. Problem $TF_2|v = 1, c = 1|C_{\max}$

Two cases of the problem $\mathrm{TF}_2|v = 1, c = 1|C_{\max}$ have been studied in the literature. Kise [4] proves the NP-hardness for the the case with general transportation times t_1 and t_2. Hurink and Knust [27] show that the case with a general t_1 and $t_2 = 0$ is strongly NP-hard. Note that the problem considered by Hurink and Knust is equivalent to the classical 3-machine flow shop

problem with identical processing times on the second machine, i.e. $F_3|p_{j2} \equiv p|C_{\max}$, and hence the latter problem is also strongly NP-hard.

We note that in practice, the two-way transportation times between two given machines are usually identical, i.e. $t_1 = t_2$. However, this problem can be shown easily to be equivalent to that considered by Hurink and Knust [27]. Hence, the problem $\mathrm{TF}_2|v = 1,\ c = 1|C_{\max}$ is strongly NP-hard even if $t_1 = t_2$.

Next, we consider the problem with one transporter but with greater capacity. Will the problem still be NP-hard?

3.2. Problem $TF_2|v = 1,\ c \geqslant 3|C_{\max}$

Theorem 1. The problem $\mathrm{TF}_2|v = 1,\ c \geqslant 3|C_{\max}$ is strongly NP-hard even if $t_1 = t_2$.

Proof. We show the NP-hardness of the problem by a reduction from the 3-PP. Given a 3-PP instance, we construct an instance for our problem as follows:

$$n = 3h + 1 \quad \text{jobs}, \quad N = H \cup \{3h + 1\}$$

$$\text{Processing times: } p_{j1} = 2a_j, \quad \text{for} \quad j \in H,\ p_{3h+1,1} = 1$$

$$p_{j2} = 2a_j, \quad \text{for} \quad j \in H,\ p_{3h+1,2} = 2b,$$

One-way transportation time $t_1 = t_2 = b$, capacity $c \geqslant 3$, makespan threshold value $y = 1 + (2h + 3)b$.

$\rightarrow$ If there is a solution to the 3-PP instance, we show that there is a schedule to our problem with a makespan of no more than y. Given a solution to the 3-PP instance, $H_1, H_2, \ldots, H_h$, we construct a schedule for our problem as shown in Figure 1.

In this schedule, the transporter departs from machine 1 at each time point $1 + 2kb$, for $k = 0, 1, \ldots, h$, and transports all of the jobs (3 jobs, except the first trip which carries only one job) finished on machine 1 by that time. It is easy to see that the above schedule is feasible and the makespan is y.

$\leftarrow$ Now suppose that there exists a schedule for our problem with a makespan of no greater than y. Since the total processing time of jobs on machine 2 equals $2(h + 1)b$, and the earliest possible starting time of processing jobs on machine 2 is $1 + t_1 = 1 + b = y - 2(h + 1)b$, we can see that (i) job $3h + 1$ is the first job scheduled; (ii) the first batch only contains job $3h + 1$ and is transported from machine 1 at time 1; and (iii) there is no idle time on machine 2 after it starts processing the first job at time $1 + b$.

Suppose that there are q batches in the schedule for some $q > 1$. Denote these batches by B_1, $B_2, \ldots, B_q$. Let the departure times of the transporter from machine 1 corresponding to these

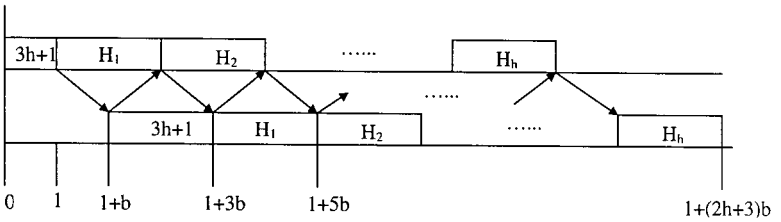

Figure 1. A schedule for the instance of the problem $\mathrm{TF}_2|v = 1,\ c \geqslant 3|C_{\max}$.

batches be denoted by $d_1, d_2, \ldots, d_q$, respectively. Similarly, denote the corresponding arrival times and completion times of these batches on machine 2 by $r_1, r_2, \ldots, r_q$, and $C_1, C_2, \ldots, C_q$, respectively. Clearly, $r_k = d_k + b$ for all k. From the discussion above, we know that B_1 contains job $3h + 1$ only, and $d_1 = 1$, $r_1 = 1 + b$, and $C_1 = 1 + 3b$. Also, as discussed above, there is no idle time between batches of jobs at machine 2. Thus, the second batch must arrive at machine 2 at a time no later than C_1, i.e. $r_2 \leqslant C_1$. Since the transporter takes $t_1 + t_2 = 2b$ units of time to finish one delivery and return to machine 1, then $d_{k+1} \geqslant d_k + 2b$, for all k. This means $d_2 \geqslant 1 + 2b$ and hence $r_2 \geqslant 1 + 3b = C_1$. This implies that $r_2 = C_1 = 1 + 3b$ and $d_2 = 1 + 2b$. Hence, we can see that $\sum_{i \in B_2} p_{i1} \leqslant 2b$. If $\sum_{i \in B_2} p_{i1} < 2b$, then $\sum_{i \in B_2} p_{i2} < 2b$, and hence $C_2 < 1 + 5b$. On the other hand, it can be seen that $r_3 \geqslant r_2 + 2b = 1 + 5b$. This implies that there will be some idle time between the batches B_2 and B_3 on machine 2. From the earlier discussion, we know this will not happen. Thus $\sum_{i \in B_2} p_{i1} = 2b$ must be true.

By the same argument, we can show that for each other batch B_k, for $k = 3, \ldots, q$, it must be true that $\sum_{i \in B_k} p_{i1} = 2b$. Then it is easy to see that $q = h + 1$ and the h batches $B_2, B_3, \ldots, B_{h+1}$ form a solution to the 3-PP instance. □

Note that the complexity of the problem $\mathrm{TF}_2|v = 1, c = 2|C_{\max}$ is still open. Since $\mathrm{TF}_2|v = 1, c = 1|C_{\max}$ and $\mathrm{TF}_2|v = 1, c \geqslant 3|C_{\max}$ are both strongly NP-hard, it will be interesting to see under what special cases the problem become polynomially solvable.

3.3. *Problem* $TF_2|p_{j1} \equiv p_1, v \geqslant 1, c \geqslant 1|C_{\max}$

In this section, we assume that the number of vehicles v is fixed, i.e. it is not part of the problem input. We show that the problem $\mathrm{TF}_2|p_{j1} \equiv p_1, v \geqslant 1, c \geqslant 1|C_{\max}$ is polynomially solvable by a dynamic programming algorithm. The following lemma can be proved easily, and hence we omit the proof.

Lemma 1.

(i) There exists an optimal solution for $\mathrm{TF}_2|p_{j1} \equiv p_1, v \geqslant 1, c \geqslant 1|C_{\max}$ such that jobs are sequenced in the non-increasing order of p_{j2} on both machines,

(ii) If $t = t_1 + t_2 \leqslant vp_1$, where p_1 is the identical processing time of jobs on machine 1, then there exists an optimal solution such that each job is transported from machine 1 to 2 immediately after it is completed on machine 1.

By Lemma 1(i), the sequencing problem is trivial. If $t = t_1 + t_2 \leqslant vp_1$, then by Lemma 1(ii), the decision of the starting time of each trip is also trivial. However, for the case with $t > vp_1$, the starting time of the trip of each vehicle needs to be decided, and this can be done by dynamic programming. Let the jobs be reindexed such that $p_{12} \geqslant p_{22} \geqslant \cdots \geqslant p_{n2}$. Schedule jobs in the order $(1, 2, \ldots, n)$ on machine 1 without idle time. Then the completion time of job j on machine 1 is, $C_{j1} = jp_1$, for $j = 1, \ldots, n$. For each transporter, there are only a finite number of possible time points for departure from machine 1. We can specify these time points following Property 1(iv) and (v). When a transporter returns from machine 2 to 1, it will either transport a batch of jobs immediately or wait until the completion time of a job before transporting the batch that contains that job. In the first case, the departure time is $x + t$ where x is the departure time of the last batch transported by the transporter. In the second case, the departure time is C_{j1}, for some job j. Note that in the first case, x can be traced back and the departure time can be expressed as $C_{j1} + qt$ for some $q \leqslant n - j$ and some job j.

In summary, the possible departure times of a transporter from machine 1 can be C_{j1}, $C_{j1}+t$, $C_{j1}+2t, \ldots, C_{j1}+qt$, for some $q \leqslant n-j$, and $j=1,\ldots,n$. Thus, there are at most n^2 candidate departure time points. For ease of presentation, let us assume that we have $T(T \leqslant n^2)$ candidate departure time points that are indexed as $1, 2, \ldots, T$ such that *earlier time points have smaller indices*. Let $t(k)$ denote the actual time corresponding to departure time point k, for $k=1,2,\ldots,T$. Also, let $t(0)=0$.

Define $F(k; j; s_1, s_2, \ldots, s_v)$ as the minimum completion time of a partial schedule containing the first k jobs $\{1,2,\ldots,k\}$, provided that the current last v batches contain jobs $j, j+1, \ldots, k$ and are transported at times $t(s_1)$, $t(s_2), \ldots, t(s_v)$ from machine 1 to 2, where $s_h \in \{0,1,2,\ldots,T\}$ for $h=1,2,\ldots,v$ and $0 \leqslant s_1 \leqslant s_2 \leqslant \cdots \leqslant s_v$. Note that if we know the available time of machine 2 before we deliver jobs $j, j+1, \ldots, k$ from machine 1 to 2, then the increase of makespan due to jobs $j, j+1, \ldots, k$ is actually fixed. Define $C(x; k; j; s_1, \ldots, s_v)$ as the minimum increase of the makespan due to jobs $j, j+1, \ldots, k$, given that machine 2 is ready at time x for processing these jobs which are transported in v batches, respectively, at times $t(s_1)$, $t(s_2), \ldots, t(s_v)$ from machine 1 to 2. Note that x depends on the previous v deliveries and the corresponding jobs contained in these deliveries. Namely, $x = F(j-1; i; b_1, b_2, \ldots, b_v)$ for some $(i; b_1, b_2, \ldots, b_v)$ that satisfy the following conditions.

(i) $0 \leqslant j-i \leqslant vc$,
(ii) $0 \leqslant b_1 \leqslant b_2 \leqslant \cdots \leqslant b_v$, and $t(s_h) - t(b_h) \geqslant t_1 + t_2$, for all $h=1,2,\ldots,v$.

For all $q=1,2,\ldots,v$, if $t(s_q)$ is not the completion time of some job on machine 1, then by Property 1(v), b_q is the unique time point such that $t(b_q) = t(s_q) - t$; otherwise, b_q can be any point with $0 \leqslant b_q \leqslant T$ which satisfies (i) and (ii).

First, let $C(0;0;0;,\ldots,0)=0$. Given $(x;k;j;s_1,\ldots,s_v)$ with $x \geqslant 0$, $\max\{0, k-vc+1\} \leqslant j \leqslant k \leqslant n$, and $0 \leqslant s_1 \leqslant s_2 \leqslant \cdots \leqslant s_v$, the value of $C(x; k; j; s_1, \ldots, s_v)$ can be calculated easily in $O(v)$ time by the following procedures:

Let g be the earliest time when machine 2 becomes available. Initially, let $g=x$. Let i be the first job to be delivered next. Initially, let $i=j$ and $u=1$.

(1) If $(i+c-1)p_1 > t(s_u)$, let $h = \min\{k, [(t(s_u)/p_1]\}$ where the notation $[Z]$ represents the largest integer no more than Z. Otherwise, $h = \min\{k, i+c-1\}$.
(2) Transport jobs $\{i,\ldots,h\}$ by transporter u at time $t(s_u)$.
(3) Let $g = \max\{g, t(s_u)+t_1\} + p(i,h)$, where $p(i,h) = p_{i2} + p_{i+1,2} + \cdots + p_{h2}$.
(4) Let $i=h+1$. If $i=k+1$ or $u=v$, then go to (5). Otherwise, let $u=u+1$ and go to (1).
(5) If $i=k+1$ and $u=v$, then set $C(x; k; j; s_1, \ldots, s_v) = g-x$ and terminate. Otherwise, let $C(x; k; j; s_1, \ldots, s_v) = \infty$ and terminate.

Note that we can calculate $p(1,j)$ for all j in advance. Then in Step (3), $p(i,h) = p(1,h) - p(1,i-1)$. Therefore, the above procedures take $O(v)$ time because the main loop runs v times.

3.3.1. DP algorithm for $TF_2|p_{j1} \equiv p_1, v \geqslant 1, c \geqslant 1|C_{\max}$

Initial conditions: $F(0;0;0,0,\ldots,0)=0$

Recursive equations: For $k=1,2,\ldots,n$; $j=\max\{1, k-vc\}, \max\{1, k-vc\}+1, \ldots, \max\{1, k-v\}$; and $0 \leqslant s_1 \leqslant s_2 \leqslant \cdots \leqslant s_v$,
$F(k; j, s_1, s_2, \ldots, s_v) = \min\{F(j-1; i; b_1, b_2, \ldots, b_v) + C(F(j-1; \quad i; b_1, b_2, \ldots, b_v); \quad k; \quad j; \quad s_1, s_2, \ldots, s_v) | i, b_1, \ldots, b_v$, satisfying the conditions (i) and (ii) described above$\}$.

Optimal solution: $\min\{F(n; j; s_1, s_2, \ldots, s_v) | \text{all possible states } (j; s_1, s_2, \ldots, s_v)\}$

Time complexity: $O(c^{v+2}v^{v+3}n^{2v+1})$. Given k and j, for each $q = 1, 2, \ldots, v$, if s_q is a time point corresponding to the completion time of some job (there are at most $O(cv)$ possible such s_q's because $k - j \leqslant cv$), then b_q can have up to $T + 1$ choices; otherwise, b_q has only one choice (i.e. $t(b_q) = t(s_q) - t$). Thus, there are a total of $O((cv + 1)n^2)$ possible combinations of s_q and b_q in the DP. Therefore, we have $O((cv)^v n^{2v})$ possible combinations of $(s_1, \ldots, s_v; b_1, \ldots, b_v)$. For each combination, there are a total of $O((cv)^2 n)$ possibilities of (k, j, i). Furthermore, as we indicated earlier, it takes $O(v)$ time to calculate the function C for each state. This verifies the overall complexity of the algorithm. Since the number of vehicles v is a fixed number, the problem is polynomially solvable. For the special case with $v = 1$, this algorithm has the complexity $O((cn)^3)$.

Remarks.

(i) When v is not fixed, whether there exists a polynomial time algorithm for $\mathrm{TF}_2|p_{j1} \equiv p_1$, $v \geqslant 1$, $c \geqslant 1|C_{\max}$ is left as an open question.
(ii) A similar DP algorithm with the same complexity can be easily constructed for the problem $\mathrm{TF}_2|p_{j2} \equiv p_2$, $v \geqslant 1$, $c \geqslant 1|C_{\max}$. We omit the details of the algorithm.

4. SCHEDULING PROBLEMS WITH TYPE-2 TRANSPORTATION

In this section, we consider scheduling problems with type-2 transportation where jobs are first processed on a single machine, parallel machines, or a flow shop, then delivered by one or more vehicles to the customer. We use the same notation as that in Section 3 and call jobs delivered together in one shipment a *batch*. Let B_k denote the kth batch of jobs delivered. Batches delivered earlier have smaller indices. For batch k, let d_k denote the departure time from the manufacturing facility. Hence, $d_k + t_1$ is the arrival time at the customer for all jobs in batch k.

4.1. *Problem* $1 \to D|v \geqslant 1, c \geqslant 1|C_{\max}$

Let $u = n - cq$, where q is the largest integer no more than n/c. Clearly, $u \geqslant 0$. For this problem, it is easy to see that the following property holds.

Property 2. There exists an optimal schedule for the problem $1 \to D|v \geqslant 1, c \geqslant 1|C_{\max}$ that satisfies the following conditions.

(i) Jobs are processed in nondecreasing order of processing times on the machine.
(ii) Each delivery batch contains consecutively processed jobs.
(iii) Earlier processed jobs are delivered no later than those processed later.
(iv) If $u > 0$, then there are $q + 1$ delivery batches. The first batch contains u jobs, and each of the other batches contains c jobs.
(v) If $u = 0$, then there are q delivery batches and each batch contains c jobs.

Proof:

(i) Can be proved by a pair-wise interchange argument.
(ii) If a batch contains non-consecutive jobs, then we can rebatch the jobs so that all the jobs in the same batch are processed consecutively without increasing the objective value.
(iii) By (ii) and a pair-wise interchange argument on the batches.

(iv) and (v) If a batch, with exception of the first batch, contains less than c jobs, we can always fill the batch with more jobs from earlier batches without increasing the objective value. □

Let jobs be indexed in the non-decreasing order of their processing times. By this property, we know exactly what jobs each batch contains and what time each batch is delivered. Let $P(j) = \sum_{k=1}^{j} p_k$. Let r_h denote the completion time of the last job in batch B_h on the machine. Then there are two cases.

Case 1: If $u > 0$, then $B_1 = \{1, 2, \ldots, u\}$, $r_1 = P(u)$; and $B_h = \{u + (h-2)c + 1, \ldots, u + (h-1)c\}$, $r_h = P(u + (h-1)c)$, for $h = 2, \ldots, q+1$.

Case 2: If $u = 0$, then $B_h = \{(h-1)c + 1, \ldots, hc\}$, $r_h = P(hc)$, for $h = 1, \ldots, q$.

Now we can see that the problem $1 \to D|v \geqslant 1, c \geqslant 1|C_{\max}$ reduces to the problem of scheduling vehicles to deliver these batches so that the time when the last batch gets to the customer is minimum. It is not difficult to see that, if we view a vehicle as a machine and delivering a batch as a job, then this vehicle dispatching problem is similar to the classical parallel machine makespan problem with different job arrival times and equal processing times, i.e. $Pv|r_j, p_j \equiv p|C_{\max}$, where there are q (or $q+1$ if $u > 0$) jobs and v identical parallel machines, the arrival time of job j is r_j, for $j = 1, \ldots, q+1$, and the processing times of jobs are all equal to $t = t_1 + t_2$.

It is easy to see that sorting jobs in non-decreasing order of their arrival times and then assigning available jobs in this order to the available machines is optimal for $Pv|r_j, p_j \equiv p|C_{\max}$. Therefore, our problem $1 \to D|v \geqslant 1, c \geqslant 1|C_{\max}$ can be solved by assigning undelivered batches to the available vehicles in the order of their completion times on the machine. It can be seen easily that the overall time complexity for solving this problem is bounded by $O(n \log n)$. Note that the makespan of the problem $1 \to D|v \geqslant 1, c \geqslant 1|C_{\max}$ is equal to the makespan of the corresponding problem $Pv|r_j, p_j \equiv p|C_{\max}$ minus t_2.

4.2. *Problem* $1 \to D|v \geqslant 1, c \geqslant 1|\sum C_j$

It is easy to see that there exists an optimal schedule for this problem that satisfies Property 2(i)–(iii). However, more than one partial batch may exist in an optimal schedule, i.e. Property 2(iv) and (v) may not hold for this problem.

In the following, we propose a DP algorithm similar to the one given in Section 3.3. Reindex jobs in non-decreasing order of p_j. Schedule jobs on the machine in the order $(1, 2, \ldots, n)$ without idle time. Denote the completion time of job j on the machine by C_j, for $j = 1, 2, \ldots, n$. Then use the same arguments as that in Section 3.3 to identify a total of no more than n^2 candidate departure time points of vehicles. These time points are $C_j + kt$, for $k = 0, 1, \ldots, n-1$, and $j = 1, 2, \ldots, n$. Index these time points by $1, 2, \ldots, T$, where $T \leqslant n^2$ such that an earlier time point has a smaller index. Let $t(h)$ denote the corresponding time of time point h, for $h = 1, 2, \ldots, T$ and let $t(0) = 0$. Following the same notation, we can construct a similar DP algorithm as the one described in Section 3.3 where that the recursive relation should be changed to the following

$$F(k; j; s_1, s_2, \ldots, s_v) = \min\{F(j-1; i; b_1, b_2, \ldots, b_v) + C(k; j; s_1, s_2, \ldots, s_v) | i; b_1, \ldots, b_v\}$$

where (i) $F(k; j; s_1, s_2, \ldots, s_v)$ is defined as the minimum total completion time of a partial schedule corresponding to the state $(k; j; s_1, s_2, \ldots, s_v)$; (ii) the ranges for i and $b_1, \ldots, b_v$ are the same respectively, as those in the algorithm of Section 3.3; and (iii) $C(k; j; s_1, s_2, \ldots, s_v)$ is *the minimum total completion time* of jobs $j, \ldots, k$ under the schedule corresponding to $(k; j; s_1, s_2, \ldots, s_v)$. Note

that similar to the discussion in Section 3.3, the value of $C(k; j; s_1, s_2, \ldots, s_v)$ can be easily calculated in $O(v)$ time. The resulting algorithm has the same time complexity $O(c^{v+2}v^{v+3}n^{2v+1})$. When $v = 1$, the complexity of this algorithm is $O((cn)^3)$ and it can be refined to $O(n^3)$.

We note that for the special case of this problem with only one vehicle $v = 1$, i.e. the problem $1 \to D|v = 1, c \geqslant 1|\sum C_j$ is equivalent to a problem studied in Reference [30].

4.3. Problem $Pm \to D|v = 1, c \geqslant 1|\sum C_j$

It is known that scheduling jobs in the non-decreasing order of their processing times is optimal for the corresponding classical problem without transportation $Pm||\sum C_j$. In this section, we prove that $P_2 \to D|v = 1, c \geqslant 1|\sum C_j$ is NP-hard. Hence, the general problem $Pm \to D|v = 1, c \geqslant 1|\sum C_j$ is NP-hard for $m \geqslant 2$.

Theorem 2. The problem $P_2 \to D|v = 1, c \geqslant 1|\sum C_j$ is NP-hard even if $t_1 = t_2$.

Proof. We prove this by a reduction from the ESPP. Given an instance of ESPP, we construct the following instance for our problem.

Number of jobs $n = 2h + 1$, $N = H \cup \{2h + 1\} = \{1, 2, \ldots, 2h + 1\}$.
One-way transportation time $t_1 = t_2 = (h^2A/2 + A/2h)/2$. (Let $t = t_1 + t_2$).
Processing times $p_j = h^2A + a_j$, for $j \in H$; $p_{2h+1} = t = h^2A/2 + A/2h$.
Threshold of total completion time $Y = (2h + 1)(2h + 3)(h^2A/2 + A/2h)/2$.

$\to$ If there is a solution to the ESPP instance, we show that there is a schedule for the above-constructed instance with a total completion time of no more than Y. Let G be a subset of H that solves the ESPP instance. We can construct the following schedule. Machine 1 first processes job $2h + 1$, which is followed by jobs from G in nondecreasing order of processing times (called SPT order), and machine 2 processes jobs from $H \backslash G$ in SPT order. Let $C_{[k],1}$ and $C_{[k],2}$ denote the completion times of the kth job on machine 1 and machine 2, respectively. Since $\sum_{j \in G} a_j = \sum_{j \in H \backslash G} a_j = A$, and all the jobs in $H \backslash G$ and in H are in respective SPT orders, we can see that $C_{[k],1} \leqslant p_{2h+1} + kh^2A + kA/h = (1 + 2k)t$, and $C_{[k],2} \leqslant kh^2A + kA/h = 2kt$. Let the vehicle deliver only one job in each delivery trip. Let odd-numbered delivery trip i deliver the jth job completed on machine 1 for some j with $j = (i + 1)/2$, and let even-numbered delivery trip i deliver the jth job completed on machine 2 for some j with $j = i/2$.

In the above schedule, the first delivery trip delivers job $2h + 1$ and departs at time $p_{2h+1} = t$. It is easy to see that by the time the vehicle is ready to deliver a job $j \in H$, the job is already completed. Hence, the vehicle is always busy starting from time t. Thus, the final completion time (i.e. the time when it arrives at the customer) of the kth job delivered is $kt + t_1 = (k + 1/2)t$. So, the total completion time of all of the jobs is $\sum_{j \in N}(j + 1/2)t = [(2h + 1)/2 + (h + 1)(2h + 1)]t = Y$.
$\leftarrow$ Now we show that if there exists a schedule with a total completion time of no more than Y, then there must be a solution to the ESPP instance. First, it is easy to see that, in an optimal schedule for the given instance of the problem $P_2 \to D|v = 1, c \geqslant 1|\sum C_j$, the following properties hold. (i) Jobs on either machine are processed in the SPT order, for otherwise, pair-wise interchange argument can be applied to improve the solution. (ii) Job $2h + 1$ is processed first on a machine (let it be machine 1 without loss of generality) since it is the job with the shortest processing time. (iii) Besides job $2h + 1$, there are h jobs processed on machine 1 (denote the set of these jobs as G), and h jobs processed on machine 2.

To show (iii), note that $C_{[j],1} \geqslant t + jh^2A$, for $j = 0, 2, \ldots, n_1$ and $C_{[j],2} \geqslant jh^2A$, for $j = 1, 2, \ldots, n_2$ where $C_{[j],i}$ is the completion time of the jth job on machine i, and $n_1 + 1$ and n_2 are numbers of jobs on machines 1 and 2, respectively and $n_2 = (2h - n_1)$. Note that even if we ignore the transportation constraint and allow each job to be delivered immediately after it is completed on a machine, the total completion time of the schedule is at least

$$Z = \sum_{j=1}^{n_1+1} (C_{[j],1} + t_1) + \sum_{j=1}^{n_2} (C_{[j],2} + t_1) \geqslant \sum_{j=0}^{n_1} (t + jh^2A + t_1) + \sum_{j=1}^{2h-n_1} (jh^2A + t_1)$$

It can be shown that

$$\sum_{j=0}^{n_1} (t + jh^2A + t_1) + \sum_{j=1}^{2h-n_1} (jh^2A + t_1)$$

is convex in n_1. If $n_1 \neq h$, then it can be shown that

$$\sum_{j=1}^{n_1} (t + jh^2A + t_1) + \sum_{j=1}^{2h-n_1} (jh^2A + t_1) > Y$$

hence $Z > Y$, which is a contradiction. Thus, $n_1 = n_2 = h$.

Given a schedule with a total completion time of no more than Y, we first show that the vehicle must deliver only one job on each delivery trip. We will only consider the solution that satisfies (i)–(iii) above. From the discussion above,

$$Z \geqslant \sum_{j=0}^{h} (t + jh^2A + t_1) + \sum_{j=1}^{h} (jh^2A + t_1) = Y - A(h + 1)$$

On the other hand, we can see that the smallest difference of completion times of any two jobs in the schedule is at least $h^2A/2 - 2A$. Thus, if there is a delivery trip that delivers more than one job, then the resulting total completion time will be more than $Z + h^2A/2 - 2A \geqslant Y + h^2A/2 - (Ah + 3A) > Y$ when $h \geqslant 4$. Thus the vehicle can deliver only one job per delivery trip. Hence, job $2h + 1$ is the first job delivered, and the vehicle makes exactly n delivery trips.

Since $Y = (t + t_1) + (2t + t_1) + \cdots + (nt + t_1)$, the vehicle must depart at time t for the first trip and have no idle time after t. Furthermore, we can see that the departure time for the kth delivery trip is kt, for $k = 1, 2, \ldots, n$. On the other hand, we can assume that jobs processed earlier are delivered earlier because otherwise we could follow this rule to improve the schedule. It is easy to see that $C_{[h+1],1} \geqslant C_{[h],2} \geqslant C_{[j],1}$, for all $j \leqslant h$. Thus, the last job on machine 2 is delivered in the $(n - 1)$th trip, and the last job on machine 1 is delivered in the nth trip. This means that $C_{[h+1],1} \leqslant nt$ and $C_{[h],2} \leqslant (n - 1)t$. Note that $C_{[h+1],1} \leqslant nt$ is equivalent to $t + h(h^2A) + \sum_{j\in G} a_j \leqslant nt = (2h + 1)t = t + h(h^2A) + A$. Hence, $\sum_{j\in G} a_j \leqslant A$. Similarly, we can show that $C_{[h],2} \leqslant (n - 1)t$ implies $\sum_{j\in H\setminus G} a_j \leqslant A$. Thus, $\sum_{j\in G} a_j = \sum_{j\in H\setminus G} a_j = A$ since $\sum_{j\in H} a_j = 2A$. □

We note that Theorem 2 implies that problem $Pm \to D|v = 1, c \geqslant 1|\sum C_j$ is at least NP-hard. However, whether it is strongly NP-hard is still an open question.

4.4. *Problem* $F_2 \to D|v = 1, c = 1|C_{\max}$

The corresponding classical problem without transportation $F_2||C_{\max}$ is polynomially solvable by the well-known Johnson's algorithm [2]. In this section, we prove that this problem with type-2 transportation is strongly NP-hard.

Theorem 3. The problem $F_2 \to D|v = 1,\ c = 1|C_{\max}$ is NP-hard in the strong sense even if $t_1 = t_2$.

Proof. We prove the theorem by a reduction from 3-PP. Given an instance of 3-PP, we construct the following instance for our scheduling problem.

Number of jobs, $n = 4h + 1$, $N = H \cup \{3h + 1, \ldots, 4h + 1\}$.
Processing times

$$p_{j1} = b/2 + a_j/2, \text{ for } j \in H;\ p_{3h+1,1} = 0;\ p_{j,1} = 2b, \text{ for } j = 3h + 2, \ldots, 4h + 1.$$

$$p_{j2} = a_j, \text{ for } j \in H;\ p_{j2} = 3b, \text{ for } j = 3h + 1, \ldots, 4h + 1.$$

One-way transportation time, $t_1 = t_2 = b/2$.
Makespan threshold value, $Y = (4h + 3)b + b/2$.

$\to$ Given a solution to the 3-PP instance, we can construct a schedule for processing jobs on the two machines as shown in Figure 2, where jobs are scheduled in the sequence $(3h + 1, H_1, 3h + 2, \ldots, H_h, 4h + 1)$. Based on this schedule, we let the vehicle deliver one job on each delivery trip, and the kth delivery trip departs at time $(k + 2)b$, for $k = 1, 2, \ldots, 4h + 1$; and hence, the vehicle has no idle time after it first departs at time $3b$. Job $3h + j$, for $j = 1, 2, \ldots, h + 1$, is delivered on the $(4j - 3)$th delivery trip, and the three jobs in H_j, for $j = 1, 2, \ldots, h$, are delivered, respectively, on the $(4j - 2)$th, $(4j - 1)$th, and $4j$th trips. It is easy to see that this schedule is feasible and its makespan is $(4h + 3)b + b/2 = Y$.
$\leftarrow$ Suppose that there is a schedule for the instance of our problem with a makespan of no more than Y. We first show that in any feasible schedule for the instance of our problem with a makespan of no more than Y, the following must be true: (i) there is no idle time on machine 2; (ii) job $3h + 1$ is processed first; and (iii) the vehicle starts at time $3b$ and has no idle time between consecutive deliveries, i.e. the kth delivery trip departs at time $3b + (k - 1)b$, for all $k = 1, 2, \ldots, n$. The total processing time of jobs on machine 2 is $(4h + 3)b = Y - b/2$. It takes $b/2$ units of time to deliver the last job to the destination, and hence machine 2 should have no idle time. This shows (i). If a job other than $3h + 1$ is processed first, then the completion time of the last job will be greater than $(4h + 3)b$, which implies that the makespan of the problem will be greater than Y. This shows (ii). Since the first job completes at time $3b$, the vehicle starts no earlier than that time. Furthermore, since there are $4h$ round trips and one one-way trip, the total transportation time is at least $4hb + b/2$. Hence, the vehicle has to start at time $3b$ and there is no idle time after it starts. This shows (iii).

Now, we prove that there is a solution to the instance of 3-PP. Note that we only need to consider permutation schedules because other types of schedules can be transformed into permutation schedules without increasing the objective value. This can be done by simply

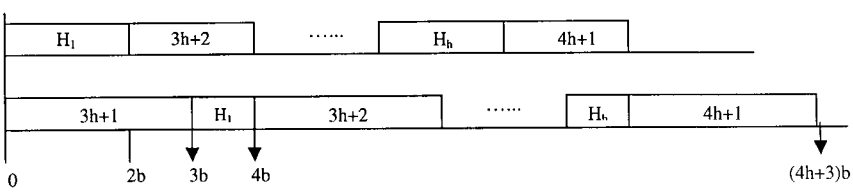

Figure 2. A schedule for the instance of the problem $F_2 \to D|v = 1,\ c = 1|C_{\max}$.

rescheduling jobs on machine 2 in the same order as on machine 1. Given a permutation schedule with a makespan of no more than Y, by the results (i)–(iii) shown earlier, we can see that in this schedule jobs are scheduled in the following sequence: $3h+1$, H_1, $3h+2$, $H_2, \ldots, 4h$, H_h, $4h+1$, H_{h+1} where H_i, for $i = 1, 2, \ldots, h+1$, is a subset of jobs from H (H_1 and H_{h+1} could be empty). Let the number of items in H_1 be k. In the following, we show that $k = 3$, and the total processing time of jobs in H_1 on machine 1 is $2b$. This implies that $\sum_{j\in H_1} a_j = b$. By the same argument, it can be shown that each other H_k, for $k = 2, \ldots, h$, contains exactly 3 jobs and $\sum_{j\in H_k} a_j = b$. Hence, H_{h+1} is empty and the subsets $H_1, H_2, \ldots, H_h$ give a solution to the 3-PP instance.

Suppose that $k \leqslant 2$. Then the completion time of job $3h+2$ on machine 2 is $6b + \sum_{j\in H_1} p_{j2} > 4b + kb$. On the other hand, job $3h+2$ is the $(k+2)$th job delivered and hence it must be completed by time $3b + (k+1)b = 4b + kb$. This results in a contradiction.

Now, suppose that $k \geqslant 4$. Then the completion time of job $3h+2$ on machine 1 is $C_{3h+2,1} = (k/2+2)b + g/2$, and the completion time of the last job of H_1 on machine 2 is $C_L = 3b + g$, where $g = \sum_{j\in H_1} a_j$. When $k \geqslant 4$, $g < (k-2)b$, and hence $C_L - C_{3h+2,1} = b - kb/2 + g/2 < 0$. Thus, there is an idle time on machine 2 between the last job of H_1 and job $3h+2$. This violates the property (i) derived earlier.

We thus conclude that $k = 3$. We now prove that $\sum_{j\in H_1} a_j = b$. Suppose that $\sum_{j\in H_1} a_j < b$. Then $\sum_{j\in H_1} p_{j2} < 2b$. The completion time of job $3h+2$ on machine 1 is equal to $(\frac{3}{2})b + (\frac{1}{2})\sum_{j\in H_1} a_j + 2b$ which is greater than $3b + \sum_{j\in H_1} a_j$, the completion time of the last job of H_1 on machine 2. Hence, there is an idle time on machine 2 between the last job of H_1 and job $3h+2$. This violates the property (i). On the other hand, if $\sum_{j\in H_1} a_j > b$ then the completion time of job $3h+2$ on machine 2 is greater than $7b$, and it is thus impossible for the vehicle to deliver job $3h+2$ at time $7b$. It must be true that $\sum_{j\in H_1} a_j = b$. □

Note that the problem $F_2 \to D|v = 1,\ c = 1|C_{\max}$ is similar to the classical problem $F_3|p_{j3} \equiv p|C_{\max}$. The latter problem can be shown to be strongly NP-hard by a proof similar to that of $F_2 \to D|v = 1,\ c = 1|C_{\max}$. Hurink and Knust [27] have independently showed that $F_3|p_{j3} \equiv p|C_{\max}$ is strongly NP-hard.

4.5. *Problem* $F_2 \to D|v = 1,\ c \geqslant 4$ *fixed*$|C_{\max}$

In this section, we show that the problem $F_2 \to D|v = 1,\ c = k|C_{\max}$ with any fixed $k \geqslant 4$ is strongly NP-hard. We have seen in Section 4.4 that the problem $F_2 \to D|v = 1,\ c = 1|C_{\max}$ is strongly NP-hard. Although we believe that the problems $F_2 \to D|v = 1,\ c = 2|C_{\max}$ and $F_2 \to D|v = 1, c = 3|C_{\max}$ are also strongly NP-hard, we are not able to prove that now. Namely, the complexity of these two problems is still unknown.

Theorem 4. The problem $F_2 \to D|v = 1, c = k|C_{\max}$ with any fixed $k \geqslant 4$ is strongly NP-hard.

Proof. To prove this theorem, we first show that the following problem, which we call *k-Partition Problem* (k-PP), is strongly NP-hard, for any fixed $k \geqslant 4$. Then the theorem will be proved by a reduction from $(k-1)$-PP.

K-partition problem (*k-PP*). Given ku items, $G = \{1, 2, \ldots, ku\}$, each item $j \in G$ has a positive integer size x_j satisfying $\sum_{j\in G} x_j = uq$, for some integer q, the question asks whether there are u disjoint subsets $G_1, G_2, \ldots, G_u$ of G such that each subset contains exactly k items and its total size is equal to q.

We show that k-PP is strongly NP-hard by transforming 3-PP to k-PP. Given an instance of 3-PP, we construct the following instance for k-PP.

Number of items, ku with $u = h$, $G = H \cup \{3h + 1, \ldots, kh\}$
Sizes of items, $x_j = (k-3)a_j$, for $j \in H$, $x_j = A - b$, for $j \in G \backslash H$, where A is sufficiently large such that $A > khb$
In this instance, $q = (k-3)A$.

(i) If there is a solution to the 3-PP instance, $H_1, \ldots, H_h$, then add $k-3$ items from $\{3h+1, \ldots, kh\}$ to each H_i, for $i = 1, \ldots, h$. Let the resulting sets be denoted as $G_1, \ldots, G_h$. It is easy to see that there are exactly k items in each G_i and the total size of the items in G_i is exactly q. Hence, $G_1, \ldots, G_u$ form a solution to the k-PP instance.
(ii) If there is a solution to the k-PP instance, $G_1, \ldots, G_u$, then first we can see that each G_i must contain exactly $k-3$ items from $\{3h+1, \ldots, kh\}$. Thus, each G_i contains exactly 3 items from H. Let H_i denote the set of these three items. The total size of H_i is thus $q - (k-3)(A-b) = (k-3)b$. Hence, $\sum_{j \in H_i} a_j = b$, for each i. This implies that $H_1, \ldots, H_h$ form a solution to the 3-PP instance.

Combining the results of (i) and (ii), we have shown that k-PP with any fixed $k \geqslant 4$ is strongly NP-hard. In the following, we show the strong NP-hardness of our scheduling problem $F_2 \to D|v = 1, c = k|C_{\max}$ with any fixed $k \geqslant 4$ by a reduction from $(k-1)$-PP. Given an instance of $(k-1)$-PP, we construct an instance of the scheduling problem as follows:

Number of jobs, $n = ku + 1$, $N = G \cup \{(k-1)u + 1, \ldots, ku + 1\}$
Processing times, $p_{j1} = 0$, $p_{j2} = 2(x_j + Q)$, for $j \in G$, where Q is sufficiently large such that $Q > nq$,

$$p_{(k-1)u+1,1} = 0,\ p_{(k-1)u+1,2} = 2$$

$$p_{j1} = 2R,\ p_{j2} = 2, \text{ for } j \in \{(k-1)u + 2, \ldots, ku + 1\}, \quad \text{where } R = (k-1)Q + q + 1$$

One-way transportation time, $t_1 = t_2 = R$
Makespan threshold value, $Y = 2uR + R + 2$.

$\to$ If there is a solution to the $(k-1)$-PP instance, we show that there is a schedule to the instance of our scheduling problem with the makespan no more than Y. Given a solution to the $(k-1)$-PP instance, we construct a schedule for processing jobs on the two machines as shown in Figure 3. Based on this schedule, we form $u+1$ delivery batches, $B_1 = \{(k-1)u + 1\}$, and $B_{j+1} = G_j \cup \{(k-1)u + j + 1\}$, for $j = 1, 2, \ldots, u$. The departure times of these batches are, $d_1 = 2$, and $d_{j+1} = 2jR + 2$, for $j = 1, 2, \ldots, u$. It is easy to see that this schedule is feasible and the makespan of this schedule is exactly Y.
$\leftarrow$ We now show that if there is a schedule for the instance of our problem with the makespan no more than Y, then there is a solution to the instance of $(k-1)$-PP. We first prove that in any feasible schedule with the makespan no more than Y, the following must be true:

(1) The vehicle makes exactly $u + 1$ deliveries.
(2) The first delivery batch B_1 contains job $(k-1)u + 1$ only and its departure time $d_1 = 2$.
(3) The vehicle has no idle time between consecutive deliveries.
(4) There is no idle time between jobs on machine 2.

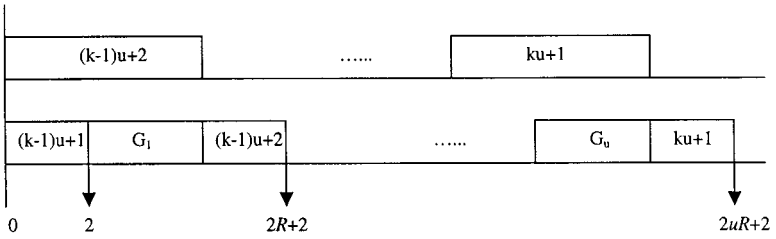

Figure 3. A schedule for the instance of the problem $F_2 \to D|v = 1, c = k|C_{\max}$.

(1) If the vehicle makes more than $u + 1$ deliveries, then the makespan will be at least $(u + 1)(t_1 + t_2) + t_1 = 2(u + 1)R + R > Y$. This means that the vehicle makes at most $u + 1$ deliveries. On the other hand, since its capacity is k and there are $ku + 1$ jobs, at least $u + 1$ deliveries are necessary. This shows that there are exactly $u + 1$ deliveries.
(2) If the first batch contains a job from $N \backslash \{(k-1)u + 1\}$, then the departure time of the first batch must be later than 2. Since the vehicle makes exactly $u + 1$ deliveries, the makespan will be more than $2 + u(t_1 + t_2) + t_1 = Y$. This shows that the first batch can only contain job $(k-1)u + 1$ and its departure time must be 2.
(3) By (1) and (2).
(4) If there is an idle time on machine 2, then the completion time of the last job on machine 2 will be more than $\sum_{j \in G} p_{j2} = 2uR + 2$, which means that the makespan will be more than $2uR + 2 + t_1 = Y$. Thus there should be no idle time on machine 2.

Now, we are ready to prove the main result. By (2) and (3), it is easy to see that the departure time of delivery batch j is $d_j = 2 + (j-1)R$, for $j = 1, \ldots, u + 1$. By (1) and (2), we can see that each delivery batch B_j, for $j = 2, \ldots, u + 1$, contains exactly k jobs. Without loss of generality, let us assume that the u jobs that require processing on machine 1, i.e. jobs $(k-1)u + 2, \ldots, ku + 1$, are scheduled on both machines in this order. (We can assume so because these jobs are identical.) Now, let us consider B_2. Clearly, all the jobs in B_2 are processed and completed in the time interval $(d_1, d_2] = (2, 2R + 2]$ on machine 2. In this interval, at most $k - 1$ jobs from G can be processed and completed on machine 2 because any k or more jobs of G will have a total processing time more than $2kQ > 2R = d_2 - d_1$. On the other hand, except jobs from G, the only other job that can be completed on machine 2 in this interval is job $(k-1)u + 2$. Thus, B_2 must contain $k - 1$ jobs from G and job $(k-1)u + 2$. Furthermore, job $(k-1)u + 2$ is the last job in the batch and is completed at time $2R + 2$ because $2R + 2$ is the earliest possible time it can be completed on machine 2. Let G_1 denote the set of the $k - 1$ jobs from G. Consider the total processing time of these $k - 1$ jobs on machine 2, $P(G_1) = \sum_{i \in G_1} p_{i2} = 2(k-1)Q + 2\sum_{i \in G_1} x_i$. We can see $P(G_1) \leqslant 2R - 2$ because they must be completed no later than $2R$ at which job $(k-1)u + 2$ starts processing. This implies that $\sum_{i \in G_1} x_i \leqslant q$. On the other hand, by equation (4), we can see that $\sum_{i \in G_1} x_i \geqslant q$ because otherwise there will be some idle time on machine 2 between job $(k-1)u + 2$ and the last job of G_1. All this shows that G_1 contains $k - 1$ jobs and $\sum_{i \in G_1} x_i = q$.

Applying the same arguments to each delivery batch B_j and the time interval $(d_{j-1}, d_j]$, for $j = 3, 4, \ldots, u + 1$, we can easily show that each B_j contains exactly $k - 1$ jobs from G and job $(k-1)u + j$ which are all processed and completed in the interval $(d_{j-1}, d_j]$ on machine 2. Let G_{j-1} denote the set of the $k - 1$ jobs from G in the batch B_j. It can be proved in the same way that

$\sum_{i \in G_{j-1}} x_i = q$, for each $j = 3, 4, \ldots, u+1$. Therefore, $G_1, \ldots, G_u$ form a solution to the $(k-1)$-PP instance. □

The following property holds for the problem $F_2 \to D|v = 1, c = k|C_{\max}$ with any k.

Property 3. There exists an optimal schedule for the problem $F_2 \to D|v = 1, c = k|C_{\max}$ such that (1) earlier processed jobs are delivered earlier; and (2) the first delivery trip delivers $n - (\lceil n/k \rceil - 1)k$ jobs, and each other delivery trip delivers k jobs.

Proof. It is easy to see that (1) is true. To show that (2) is true, we first observe that the last delivery batch must contain k jobs; otherwise, some jobs in the previous batch can be delayed and put into this last batch without increasing the objective value. The argument applies to other batches except the first one. □

As discussed above, the case with $c = 1$, and the case with a fixed $c \geqslant 4$, respectively, are NP-hard in the strong sense, and it is an open question for the case with $c = 2$ or 3. Now, the interesting question is whether the problem is NP-hard in the case with the capacity c not fixed in advance. If $c \geqslant n$, then an obvious optimal delivery schedule is to deliver all the jobs together when they are completed on machine 2, and Johnson's algorithm is optimal for scheduling jobs on the machines. Thus, it is a polynomially solvable problem. If $c = n - 1$, we still have a polynomial solution. This can be done by first fixing one job to be processed first and to be delivered alone. We then schedule the remaining jobs by Johnson's Algorithm and deliver all of them in one batch. Depending which one is selected as the first job, there are n schedules generated. We pick the one with the smallest makespan. Hence, combining these two cases, we can see that the problem $F_2 \to D|v = 1, c \geqslant n - 1|C_{\max}$ is polynomially solvable. Following the same argument, it can be shown that $F_2 \to D|v = 1, c \geqslant n - k|C_{\max}$ is polynomially solvable for fixed k. The next section shows that this problem becomes NP-hard in the ordinary sense when $c = n/2$.

4.6. Problem $F_2 \to D|v = 1, c = n/2|C_{\max}$

Theorem 5. The problem $F_2 \to D|v = 1, c = n/2|C_{\max}$ is NP-hard even if $t_1 = t_2$.

Proof. We prove the case with n as an even number. The proof technique can be easily extended to the other case where n is an odd number. We use a reduction from the ESPP. Given an instance of ESPP, we construct the following instance for our problem with an even n.

Number of jobs $n = 2h + 2$, $N = H \cup \{2h+1, 2h+2\} = \{1, 2, \ldots, 2h+2\}$
Processing times:

$$p_{j1} = 0, \text{ for } j \in H, \quad p_{2h+1,1} = p_{2h+2,1} = (h+1)A,$$

$$p_{j2} = A + a_j, \quad \text{for } j \in H; \; p_{2h+1,2} = p_{2h+2,2} = 2$$

Capacity of the vehicle $c = h + 1$
Transportation time $t_1 = t_2 = (h+1)A/2 + 1$
Makespan threshold value $Y = 5(h+1)A/2 + 5$

$\to$ If there is a solution to the ESPP instance, we show that there is a schedule for our problem with a makespan of no more than Y. Let G be a subset of H that solves the ESPP instance. We can construct a schedule for the instance of the problem $F_2 \to D|v = 1, c = n/2|C_{\max}$ by processing

jobs on both machines in the order $(G, 2h+1, G\backslash H, 2h+2)$. Deliver all the jobs in G together with job $2h+1$ in one batch at time $(h+1)A+2$, and deliver all remaining jobs in another batch at time $2(h+1)A+4$. Hence, the makepsan is equal to $2(h+1)A+4+t_1 = Y$.

$\leftarrow$ Now, we show that if there exists a schedule with a makespan of no more than Y, then there must be a solution to the ESPP instance. First, it is easy to see that in a schedule with a makespan of no more than Y, the following must be true: (i) there is no idle time on machine 2; (ii) there are only two deliveries, one at the completion time of the $(h+1)$th job and the other at the completion time of all jobs; and (iii) the first delivery must be no later than $(h+1)A+2$.

Without loss of generality, let job $2h+1$ be processed earlier than job $2h+2$. Let the sequence of jobs on machine 2 be $(H_1, 2h+1, H_2, 2h+2, H_3)$, where H_1, H_2, and H_3 are subsets of H and H_1 and H_3 may be empty. If H_1 contains more than h jobs, then the completion time of the first $h+1$ jobs on machine 2 is greater than $(h+1)A+(h+1) > (h+1)A+2$ (recall that a_j is a positive integer), contradicting the fact that the first $h+1$ jobs are delivered no later than $(h+1)A+2$. On the other hand, if H_1 contains less than h jobs, then the completion time of the last job of H_1 on machine 2 is smaller than $(h+1)A$ while the completion time of job $2h+1$ on machine 1 is $(h+1)A$. Hence, there is an idle time on machine 2, a contradiction. Hence, H_1 contains exactly h items. Using a similar argument, we can see that the total processing time of the jobs of H_1 on machine 2 is equal to $(h+1)A$, which implies $\sum_{j\in H_1} a_j = A$. Similarly, we can show that H_2 contains h jobs and the total processing time of the jobs of H_2 on machine 2 is $(h+1)A$, and hence H_3 is empty and $\sum_{j\in H_2} a_j = A$. This shows that there is a solution to the ESPP instance. □

It can be shown easily that there exists an optimal schedule for the problem $F_2 \to D|v=1, c=n/2|C_{\max}$ such that jobs are delivered by two trips (each one carries $n/2$ jobs) and jobs delivered on the first and second trips are processed in their respective orders determined by Johnson's algorithm. Hence, we can first apply Johnson's algorithm to our problem and then partition the jobs into two groups. The first group is delivered in the first trip and the remaining jobs are delivered in the second trip. In order to partition jobs into two groups we can design a pseudo-polynomial dynamic programming algorithm to solve our problem optimally. The recursive equations in the algorithm are quite standard [31] and are omitted here.

The existence of such a pseudo-polynomial algorithm means that the problem $F_2 \to D|v=1, c=n/2|C_{\max}$ is NP-hard in the ordinary sense.

4.7. A heuristic for the problem $F_2 \to D|v=1, c=k|C_{\max}$

Heuristic 1. Get a schedule for job processing on the two machines by applying Johnson's Algorithm. Then deliver $n-(\lceil n/k \rceil - 1)k$ jobs in the first delivery trip and k jobs in each other delivery trip.

Lemma 2. If we apply Heuristic 1 to the problem $F_2 \to D|v=1, c=k|C_{\max}$ with $t_1 \geqslant t_2$ and let C_{H} be the makespan obtained, then $C_{\mathrm{H}} \leqslant (1+(2n-2k)/(2n-k))C^*$ where C^* is the optimal makespan, and the bound is tight.

Proof. Let A be the makespan for applying Johnson's Algorithm to the classical problem $F_2||C_{\max}$. We have $C_{\mathrm{H}} \leqslant A + \lceil n/k \rceil (t_1+t_2) - t_2$ and $C^* \geqslant \max\{A+t_1, \lceil n/k \rceil (t_1+t_2) - t_2)\}$.

Hence,

$$\frac{C_H}{C^*} \leqslant 1 + \frac{(\lceil n/k \rceil - 1)(t_1 + t_2)}{C^*}$$

$$\leqslant 1 + \frac{(\lceil n/k \rceil - 1)(t_1 + t_2)}{(\lceil n/k \rceil - 1)(t_1 + t_2) + t_1}$$

$$\leqslant 1 + \frac{\lceil n/k \rceil - 1}{\lceil n/k \rceil - 1/2}$$

$$\leqslant 1 + \frac{2n - 2k}{2n - k}$$

Consider an example with the following instance: $p_{j1} = \varepsilon$ for $j = 1, \ldots, n$; $p_{j2} = \varepsilon$ for $j = 1, \ldots, n-1$ and $p_{n2} = n - k$, where ε is a very small number. Furthermore, let $t_1 = t_2 = k/2$. The optimal solution is to sequence jobs in the order: $J_1, J_2, \ldots, J_n$, and deliver $n - (\lceil n/k \rceil - 1)k$ jobs in the first delivery and then k jobs each in the remaining deliveries, resulting $C^* = \varepsilon(n - (\lceil n/k \rceil - 1)k + 1) + (\lceil n/k \rceil - 1)k + k/2$. Heuristic 1 may sequence jobs in the order: $J_n, J_1, \ldots, J_{n-1}$, and deliver $n - (\lceil n/k \rceil - 1)k$ jobs in the first delivery and then k jobs each in the remaining deliveries, resulting $C_H = \varepsilon(n - (\lceil n/k \rceil - 1)k) + (n - k) + (\lceil n/k \rceil - 1)k + k/2$. When ε approaches 0, C_H/C^* approaches $(1 + (2n - 2k)/(2n - k))$. □

Remark. If $k = n/2$, then $C_H/C^* \leqslant 5/3$.

Table I. Summary of complexity status of scheduling problems with transportation

Problem	Complexity	Reference
Type-1 *transportation*		
$TF_2 \mid v = 1, c = 1 \mid C_{\max}$	SNP	Hurink and Knust [27]
$TF_2 \mid v = 1, c = 2 \mid C_{\max}$	Open	
$TF_2 \mid v = 1, c \geqslant 3 \mid C_{\max}$	SNP	Section 3.2
$TF_2 \mid p_{j1} = p_1, v \geqslant 1, c \geqslant 1 \mid C_{\max}$	P	Section 3.3
$TF_2 \mid p_{j2} = p_2, v \geqslant 1, c \geqslant 1 \mid C_{\max}$	P	Section 3.3
$TF_2 \mid v \geqslant n, c \geqslant 1 \mid C_{\max}$	P	Maggu and Das [1]
Type-2 *transportation*		
$1 \to D \mid v \geqslant 1, c \geqslant 1 \mid C_{\max}$	P	Section 4.1
$1 \to D \mid v = 1, c \geqslant 1 \mid \sum C_j$	P	Ahmadi et al. [30]
$1 \to D \mid v \geqslant 1, c \geqslant 1 \mid \sum C_j$	P	Section 4.2
$P_2 \to D \mid v = 1, c \geqslant 1 \mid \sum C_j$	NP	Section 4.3
$F_2 \to D \mid v = 1, c = 1 \mid C_{\max}$	SNP	Section 4.4
$F_2 \to D \mid v = 1, c = 2 \mid C_{\max}$	Open	
$F_2 \to D \mid v = 1, c = 3 \mid C_{\max}$	Open	
$F_2 \to D \mid v = 1, c \geqslant 4$, fixed $c \mid C_{\max}$	SNP	Section 4.5
$F_2 \to D \mid v = 1, c = n/2 \mid C_{\max}$	NP	Section 4.6
$F_2 \to D \mid v = 1, c \geqslant n - k$, fixed $k \mid C_{\max}$	P	Section 4.5

Note: P: polynomially solvable; SNP: NP-hard in the strong sense; NP: NP-hard; Open: unknown complexity.

5. CONCLUSION

Motivated by logistics management in practice, the machine scheduling problems we have studied take into account explicitly both transportation capacity and transportation times. We have classified the computational complexity of various scheduling problems with type-1 or -2 transportation by either proving their NP-hardness or providing polynomial algorithms. The results of this paper, together with related existing results, are summarized in Table I. Note that the dynamic programming algorithms (polynomial or pseudo-polynomial) developed in the paper may be refined so that a possibly lower complexity can be achieved. However, since we are mainly interested in classifying the complexity status of the problems, we leave possible refinement of these algorithms for future research.

Many interesting topics remain for future exploration. First of all, the open problems posed in this paper need to be resolved. Secondly, various polynomially solvable special cases need to be identified. Thirdly, more realistic models need to be investigated, including problems with type-1 transportation that consider constraints on buffer space and problems with type-2 transportation that involve multiple customers such that vehicle routing decisions have to be addressed as well. We have already begun conducting related research in this direction.

ACKNOWLEDGEMENTS

This work was supported in part by NSF Grants DMI-9610229, DMI-9908221 and DMI-9988427 and the University of Pennsylvania Research Foundation. The authors are grateful to Prof. Michael Pinedo and three anonymous referees for their constructive comments.

REFERENCES

1. Maggu PL, Das G. On $2 \times n$ sequencing problem with transportation times of jobs. *Pure and Applied Mathematika Sciences* 1980; **12**:1–6.
2. Johnson SM. Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly* 1954; **1**:61–68.
3. Maggu PL, Das G, Kumar R. On equivalent job-for-job block in $2 \times n$ sequencing problem with transportation times. *Journal of the OR Society of Japan* 1981; **24**:136–146.
4. Kise H. On an automated two-machine flowshop scheduling problem with infinite buffer. *Journal of the Operations Research Society of Japan* 1991; **34**:354–361.
5. Stern HI, Vitner G. Scheduling parts in a combined production-transportation work cell. *Journal of the Operational Research Society* 1990; **41**:625–632.
6. Ganesharajah T, Hall NG, Sriskandarajah C. Design and operational issues in AGV-served manufacturing systems. *Annals of Operations Research* 1998; **76**:109–154.
7. Panwalkar SS. Scheduling of a two–machine flowshop with travel time between machines. *Journal of the Operational Research Society* 1991; **42**:609–613.
8. Stevens JW, Gemmill DD. Scheduling a two-machine flowshop with travel times to minimize maximum lateness. *International Journal of Production Research* 1997; **35**:1–15.
9. Mitten LG. Sequencing n jobs on two machines with arbitrary time lags. *Management Science* 1959; **5**: 293–298.
10. Maggu PL, Singhal ML, Mohammad N, Yadav SK. On n-job, 2-machine flow-shop scheduling problem with arbitrary time lags and transportation times of jobs. *Journal of the OR Society of Japan* 1982; **25**:219–227.
11. Langston MA. Interstage transportation planning in the deterministic flow-shop environment. *Operations Research* 1987; **35**:556–564.
12. Yu W. The two-machine flow shop problem with delays and the one-machine total tardiness problem. *Ph.D. Dissertation*, Eindhoven University of Technology, 1996.
13. Potts CN. Analysis of a heuristic for one machine sequencing with release dates and delivery times. *Operations Research* 1980; **28**:1436–1441.

14. Hall LA, Shmoys DB. Jackson's rule for single-machine scheduling: making a good heuristic better. *Mathematics of Operations Research* 1992; **17**:22–35.
15. Woeginger GJ. Heuristics for parallel machine scheduling with delivery times. *Acta Informatica* 1994; **31**:503–512.
16. Herrmann JW, Lee C-Y. On scheduling to minimize earliness-tardiness and batch delivery costs with a common due date. *European Journal of Operational Research* 1993; **70**:272–288.
17. Chen Z-L. Scheduling and common due date assignment with earliness-tardiness penalties and batch delivery costs. *European Journal of Operational Research* 1996; **93**:49–60.
18. Cheng TCE, Gordon VS, Kovalyov MY. Single machine scheduling with batch deliveries. *European Journal of Operational Research* 1996; **94**:277–283.
19. Yuan J. A note on the complexity of single-machine scheduling with a common due date, earliness-tardiness, and batch delivery costs. *European Journal of Operational Research* 1996; **94**:203–205.
20. Manda BS, Palekar US. Recent advances in the design and analysis of material handling systems. *Journal of Manufacturing Science and Engineering* 1997; **119**:841–848.
21. Raman N, Talbot FB, Rachamadugu RV. Simultaneous scheduling of machines and material handling devices in automated manufacturing. In *Proceedings of the Second ORSA/TIMS Conference on Flexible Manufacturing Systems*: *Operations Research Models and Applications*, Stecke KE, Suri R (eds). Elsevier: Amsterdam, 1986; 455–465.
22. Jaikumar R, Solomon MM. Dynamic scheduling of automated guided vehicles for a certain class of systems. *Journal of Manufacturing Systems* 1990; **9**:315–323.
23. Kise H, Shioyama T, Ibaraki T. Automated two-machine flowshop scheduling: a solvable case. *IIE Transactions* 1991; **23**:10–16.
24. Levner E, Kogan K, Maimon M. Flowshop scheduling of robotic cells with job-dependent transportation and set-up effects. *Journal of the Operational Research Society* 1995; **46**:1447–1455.
25. Bilge U, Ulusoy G. A time window approach to simultaneous scheduling of machines and material handling systems in an FMS. *Operations Research* 1995; **43**:1058–1070.
26. Agnetis A, Pacciarelli D, Rossi F. Lot scheduling in a two-machine cell with swapping devices. *IIE Transactions* 1996; **28**:911–917.
27. Hurink J, Knust S. Flow-shop problems with transportation times and a single robot. *Preprints* 1998, Fachbereich Mathematik/Informatik, Universitat Osnabruck.
28. Pinedo M. *Scheduling*: *Theory*, *Algorithms*, *and Systems*. Prentice-Hall: Englewoods Cliffs, NJ, 1995.
29. Garey MR, Johnson DS. *Computers and Intractability*: *A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company: New York, 1979.
30. Ahmadi JH, Ahmadi RH, Dasu S, Tang CS. Batching and scheduling jobs on batch and discrete processors. *Operations Research* 1992; **39**:750–763.
31. Lee C-Y, Chen Z-L. Machine scheduling with transportation considerations. *Working Paper*, #98–08, Department of Systems Engineering, University of Pennsylvania, 1998.

