Complexity results and algorithms for an integrated single machine scheduling and outbound delivery problem with fixed sequence

Azeddine Cheref, Alessandro Agnetis, Christian Artigues, Jean-Charles Billaut

Complexity results and algorithms for an integrated single machine scheduling and outbound delivery problem with fixed sequence

Azeddine Cheref [*] Alessandro Agnetis [†] Christian Artigues [‡]
Jean-Charles Billaut [§]

Abstract

In this paper, we consider an integrated production and outbound delivery scheduling problem. In particular, we address the situation in which the scheduling sequence and the delivery sequence are the same and predefined. A set of jobs are processed on a single machine and finished jobs are delivered to the customers by a single capacitated vehicle. Each job has a processing time and transportation times between customers are taken into account. Since the sequence is given, the problem consists to form batches of jobs and our objective is to minimize the sum of the delivery times or general functions of the delivery times. The NP-hardness of the general problem is established and a pseudopolynomial time dynamic programming algorithm is given. Some particular cases are treated, for which NP-hardness proofs and polynomial time algorithms are given. Finally, a fixed-parameter tractability result is given.

1 Introduction

This paper deals with a model for coordinating production and delivery schedules. In many production systems, finished products are delivered from the factory to multiple customer locations, warehouses, or distribution centers by delivery vehicles. An increasing amount of research has been devoted, during the last twenty years, to devise integrated models for production and distribution. These models have been largely analyzed and reviewed by [Chen, 2010], who proposed a detailed classification scheme. The models reflect the variety of issues, including systems structure, vehicle/transportation system characteristics, delivery modes, various types of time constraints. In the large majority of the models in the literature, the coordination of production and distribution is achieved through the creation of *batches*, i.e., several parts are shipped together and delivered to their respective destinations during a single trip. When forming batches, one must therefore take into account both production information (such as processing time, release dates etc) and delivery information (such as customer location, time windows etc). Most of the models presented in the literature explicitly take into account transportation times to reach the customers' location, but there are no proper routing decisions, since the number of distinct customers is typically very small. Hence, the focus of the analysis is often on scheduling and batching.

[*] cheref@laas.fr
Université François-Rabelais Tours / CNRS, 64 av. J. Portalis, 37200 Tours, France
CNRS, LAAS-CNRS, Université de Toulouse, France
[†] agnetis@dii.unisi.it
Dipartimento di Ingegneria dell'Informazione, Universitá degli Studi di Siena, via Roma 56, 53100 Siena, Italy
[‡] artigues@laas.fr
CNRS, LAAS-CNRS, Université de Toulouse, France
[§] jean-charles.billaut@univ-tours.fr
Université François-Rabelais Tours / CNRS, 64 av. J. Portalis, 37200 Tours, France

Many studies consider delivery as a separate step after production, but do not model it in details, e.g. assuming that a sufficiently large number of vehicles is available to deliver the products at any time or assuming that it has only one customer. We briefly discuss some works related to integrated scheduling and delivery decisions. Lee and Chen in [Lee and Chen, 2001] consider delivery and scheduling problems for a situation in which jobs are transported between machines and another situation in which the jobs are transported to a single customer. [Li et al., 2005] analyze the joint problem of production sequencing and batch formation, in order to minimize total delivery time, given that delivery is performed by a single vehicle. Total delivery time is a meaningful indicator of the overall efficiency of the delivery process. They show that in general the problem is NP-hard, and then give polynomial time algorithms for the problem with a fixed number of distinct destinations. In [Chen and Vairaktarakis, 2005], the authors consider some particular cases of the integrated production and delivery problem in which jobs are produced on a single machine or on parallel machines and delivered by several vehicles to one or several customer locations. The authors minimize some objective functions including the delivery costs and the delivery times. For each case, the authors propose a polynomial dynamic programming algorithm or a heuristic with worst case analysis. This problem was extended by [Chen and Pundoor, 2006] to the case where the machines are located in different locations. In [Chen and Lee, 2008], there are various destinations but a batch can only contain jobs of the same destination. [Fan et al., 2015], consider a non availability time interval on the machine, several vehicles without capacity constraint and only transportation costs (no vehicle routing problem associated to the delivery). A single vehicle with a storage area and one or two customers is considered in [Chang and Lee, 2004]. The authors show that the problem is NP-hard for the makespan objective. Other complexity results are provided as well as polynomial time algorithms for special cases. For the same objective function, [Li and Ou, 2005] study the pickup and delivery problem in which a single vehicle travels between the machine and the warehouse, whereas [Wang and Cheng, 2009] study a similar problem in which three different locations and two vehicles are considered. The first vehicle transports unprocessed jobs between the warehouse and the factory and the second one transports finished jobs between the factory and the customer. In [Hall et al., 2001] and [Leung and Chen, 2013], the problem in which the delivery dates are fixed in advance is considered. In order to minimize the makespan, [Gao et al., 2015] consider the problem on a single machine and a unique capacitated vehicle with a no wait constraint, i.e., the batch must be delivered as soon as it is completed. They show that the problem is NP-hard, and then give polynomial time algorithms for the special case with constant travel times. In [Stecke and Zhao, 2007] and [Zhong et al., 2010], the authors study an integrated single machine production and distribution scheduling problem in which various shipping times and costs are specified.

Cases where the delivery problem is between machines (inbound delivery) are reported by [Lee and Chen, 2001], [Hurink and Knust, 2001] and [Zhong and Chen, 2015], where the authors consider multiple machines and transportation time between them. [Lee and Chen, 2001] consider also the case where the finished jobs are delivered to one customer location. Some complexity results are given. In [Agnetis et al., 2014], [Agnetis et al., 2015] there are two machines and the transportation is made between these machines, various cases are treated and the objective is to minimize the total transportation cost.

Using the terminology of [Chen, 2010], the models presented in this paper concern *batch delivery with routing*, i.e., orders going to different customers can be delivered together in the same shipment (batch).

However, a distinctive feature of the problems is studied. We consider a fixed sequence of production and delivery, i.e. the jobs must be delivered in the same order in which they are produced. Examples of situations in which the customer sequence is *fixed* are reported by [Armstrong et al., 2008] and [Viergutz and Knust, 2014]. For maximizing the total satisfied demand in a single round trip, the authors consider that the products expire in a constant time after their completion time and that a delivery time window exists for each product. In [Lenté and Kergosien, 2014], the authors search for a batching of jobs where a single capacitated vehicle is used for the delivery. Polynomial time algorithms are proposed for minimizing the

makespan, the maximum lateness and the number of tardy jobs but the sum of delivery times is not treated. [Tsirimpas et al., 2008] minimize the overall distance traveled for the single vehicle routing with a predefined customer sequence. This problem can be seen as one of the problems treated in [Lenté and Kergosien, 2014] for the makespan minimization with zero processing times (no scheduling problem). In [Agnetis et al., 2014] and [Agnetis et al., 2015] and [Li and Ou, 2005], special cases with a predefined sequence are addressed and for each of them, polynomial time algorithms are given.

Here we will mainly focus on the problem of deciding how to form batches with a given production and delivery sequence (basic problem).

Our contributions are as follows. In Section 2, we present the problem formally, we give the notations and we completely characterize the complexity of the basic problem, showing that when the objective function Z is to minimize the total delivery time, it is NP-hard in the ordinary sense. Then, we propose a pseudopolynomial time algorithm for any sum-type function of the delivery times. In Section 3, we focus on two problem extensions: the case where all transportation times are identical (constant travel times), and the case where the number of delivery locations is fixed. Finally, some conclusions and future research directions are presented in Section 4.

2 Problem definition and complexity

2.1 Problem definition and notation

The problem considered in this paper can be described as follows. A set of n jobs is given and their production sequence is known. Each job J_j, $j = 1, \ldots, n$, requires a certain *processing time* p_j on a single machine, and must be delivered to a certain location site. For the sake of simplicity, when it does not create confusion, we use j to refer to the destination of job J_j. We denote by $t_{i,j}$ the transportation time from destination i to destination j. For analogy with vehicle routing problems, we refer to the manufacturer's location as the *depot*. We use M to denote the depot (manufacturer), hence $t_{M,j} = t_{j,M}$ is the transportation time between the depot and destination j. Unless otherwise specified, we assume that transportation times are symmetric and satisfy the triangle inequality.

Deliveries are carried out by a single *vehicle*. The vehicle loads a certain number of jobs that have been produced and departs towards the corresponding destinations. Once all the jobs have been delivered, it returns to the depot, hence completing a *round trip*. The set of jobs delivered during a single round trip constitutes a *batch*. The capacity c of the vehicle is the maximum number of jobs it can load and hence deliver in a round trip (corresponding to the batch size). The delivery sequence of jobs follows the fixed production sequence. In other terms, the jobs must be delivered in the order in which they are produced. Notice that a more general case consists in considering that a size is associated to each job, and that the vehicle capacity is a maximum size. Because in our problem the sequences are fixed, the results that we propose in this paper are also valid for this general case.

The problem consists in calculating a partition of all jobs into *batches*, i.e., a *batching scheme*. Each batch will be routed according to the manufacturing sequence.

In general, the performance of the system depends on all the concurrent decisions: production scheduling, batching and vehicle routing. This requires therefore an *integrated model*, allowing one to coordinate all these aspects. A *solution* to our problem with fixed sequence, is the specification of a batching scheme.

Given a solution, we denote by C_j the *completion time* of job J_j on the single machine, which is also the time at which the job is released for delivery, i.e., the batch including job J_j cannot start before C_j. We denote by D_j the delivery time of J_j, i.e., the time at which the job J_j is delivered at its destination.

The performance measures we consider in this paper depend on such delivery times. In particular, denoting with Z the performance measure, in this paper we consider:

- the *total delivery time*, i.e., $Z = \sum_{j=1}^{n} D_j$

- a *general sum-type performance index*, i.e., $Z = \sum_{j=1}^{n} f_j(D_j)$, where $f_j(D_j)$ is a general, nondecreasing function of D_j, $j = 1, \ldots, n$.

Note that the latter case includes total (weighted) delivery time, total (weighted) tardiness, etc.

The problem we consider can be described by the $\alpha|\beta|\gamma$ notation due to [Lee and Chen, 2001]. There is a single manufacturing machine serving several customers ($1 \to D$), one vehicle ($v = 1$) having capacity c, and production and delivery sequences are identical and fixed (***fixed-seq***). In general, customers have different locations.

Problem $1 \to D|v = 1, c,$ ***fixed-seq***$|\sum D_j$**:** *Given n jobs of length p_j, $1 \leq j \leq n$, transportation times $t_{i,j}$ for all $i, j \in \{1, ..., n\}$, and a sequence σ of production and of delivery, find a batching scheme $\mathcal{B}$ such that $\sum D_j$ is minimized.*

2.2 Complexity of problem $1 \to D|v = 1, c,$ ***fixed-seq***$|\sum D_j$

Since the production sequence is given, and since jobs are delivered to the respective customers in the same given order, we assume that the job sequence is $\sigma = (J_1, J_2, ..., J_n)$. Only travel times $t_{j,j+1}$ are relevant, as well as times $t_{j,M} = t_{M,j}$, representing the travel time between customer j and the manufacturer and vice-versa.

For our purposes, we introduce the following problem.

Even-Odd Partition (EOP). A set of n pairs of positive integers $(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)$ is given, in which, for each i, $a_i > b_i$. Letting $H = \sum_{i=1}^{n}(a_i + b_i)$, is there a partition $(S, \bar{S})$ of the index set $I = \{1, 2, \ldots, n\}$ such that

$$\sum_{i \in S} a_i + \sum_{i \in \bar{S}} b_i = H/2? \tag{1}$$

EOP is NP-hard in the ordinary sense [Garey et al., 1988]. In the following, we will actually use the following slightly modified version of the problem.

Modified Even-Odd Partition (MEOP). A set of n pairs of positive integers $(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)$ is given, in which, for each i, $a_i > b_i$. Letting $Q = \sum_{i=1}^{n}(a_i - b_i)$, is there a partition $(S, \bar{S})$ of the index set $I = \{1, 2, \ldots, n\}$ such that

$$\sum_{i \in S}(a_i - b_i) = Q/2? \tag{2}$$

Note that the two problems are indeed identical. In fact, suppose that EOP has a partition $(S, \bar{S})$. The corresponding instance of MEOP also admits the same partition. In fact, subtracting $\sum_{i=1}^{n} b_i = \sum_{i \in S} b_i + \sum_{i \in \bar{S}} b_i$ from both sides of (1), one obtains

$$\sum_{i \in S}(a_i - b_i) = H/2 - \sum_{i=1}^{n} b_i \tag{3}$$

Now, from the definitions of K and Q it turns out that

$$Q = H - 2\sum_{i=1}^{n} b_i$$

and hence (3) is indeed (2). We next show the following result.

Theorem 2.1 *Problem $1 \to D|v = 1, c = 2,$ fixed-seq$|\sum D_j$ is NP-hard.*

Proof. The problem is obviously in NP. Given an instance of MEOP, we build an instance to our problem as follows. There are $3n+3$ jobs. The processing times of the jobs are defined as follows:

$$\begin{array}{rrl} p_1 = 0, & p_2 = 0, & p_3 = 0 \\ p_{3i+1} = 1, & p_{3i+2} = 1, & p_{3i+3} = 4x_i + b_i - 2 \text{ for all } i = 1, \ldots, n-1, \\ p_{3n+1} = 4x_n + b_n + Q/2, & p_{3n+2} = 0, & p_{3n+3} = 0 \end{array}$$

where the x_i are defined as.

$$x_i = (3a_i - 2b_i + 3(n-i)(a_i - b_i))/2 \text{ for all } i = 1 \ldots n \tag{4}$$

and $x_{n+1} = 0$.

In the following, we refer to the set of jobs $(J_{3i+1}, J_{3i+2}, J_{3i+3})$, $i = 0, \ldots, n$, as the *triple* T_{i+1}.

For what concerns the travel times, we let:

- for each $i = 0, 1, \ldots, n-1$, one has
  - $t_{M,3i+1} = t_{3i+1,M} = t_{M,3i+2} = t_{3i+2,M} = t_{M,3i+3} = t_{3i+3,M} = x_{i+1}$,
  - $t_{3i+1,3i+2} = a_{i+1}, t_{3i+2,3i+3} = b_{i+1}, t_{3i+3,3i+4} = x_{i+1} + x_{i+2}$.
- $t_{M,3n+1} = 0, t_{3n+1,M} = 0, t_{M,3n+2} = 0, t_{3n+2,M} = 0, t_{M,3n+3} = 0$.
- $t_{3n+1,3n+2} = 0, t_{3n+2,3n+3} = 0$.

The vehicle capacity is $c = 2$. The problem consists in determining whether a solution exists such that the total delivery time does not exceed

$$f^* = \sum_{i=1}^{n} (3C_{3i} + 7x_i + b_i) + C_{3n+1} + C_{3n+2} + C_{3n+3} - Q/2. \tag{5}$$

For shortness, we call *feasible* a schedule satisfying (5). The proof has the following scheme.

1. We first establish via Lemma 2.2 that if a feasible schedule exists, then there is one having a certain structure, called *triple-oriented*,
2. We analyze some properties of this structure,
3. We show that a triple-oriented schedule of value f^* exists if and only if the instance of MEOP is a yes-instance.

Lemma 2.2 *If a feasible schedule exists, then there exists one satisfying the following property: for all $i = 1, \ldots, n$, jobs J_{3i} and J_{3i+1} are NOT in the same batch.*

Proof. Suppose that a feasible schedule exists in which, for a certain i $(1 \leq i \leq n)$, jobs J_{3i} and J_{3i+1} are in the same batch. Since $c = 2$, the batch contains no other job. As a consequence, after delivering J_{3i+1}, the vehicle must go back to M in order to load the next jobs and start a new trip. If we denote by τ the start time of the round trip of jobs J_{3i} and J_{3i+1}, job J_{3i} is delivered at time $D_{3i} = \tau + t_{M,3i}$ and job J_{3i+1} is delivered at time $D_{3i+1} = \tau + t_{M,3i} + t_{3i,3i+1}$. Therefore we have $D_{3i} = \tau + x_i$ and $D_{3i+1} = \tau + x_i + (x_i + x_{i+1})$. The vehicle is back at M at time $\tau + 2x_i + 2x_{i+1}$. Now, if we replace this batch with two batches of one job each, the delivery times of both jobs as well as the time at which the vehicle is back at M are unchanged. Therefore, there is an equivalent solution where J_{3i} and J_{3i+1} are not in the same batch. □

We call *triple-oriented* a schedule satisfying Lemma 2.2. The reason of this name is that the schedule is decomposed according to triples. More precisely, since $c = 2$, for each triple $T_{i+1} = (J_{3i+1}, J_{3i+2}, J_{3i+3})$, $i = 0, \ldots, n-1$, a consequence of Lemma 2.2 is that there are *exactly* two batches, and only two possibilities, namely:

- either the first batch is $\{J_{3i+1}, J_{3i+2}\}$ and the second is $\{J_{3i+3}\}$,
- or the first batch is $\{J_{3i+1}\}$ and the second is $\{J_{3i+2}, J_{3i+3}\}$.

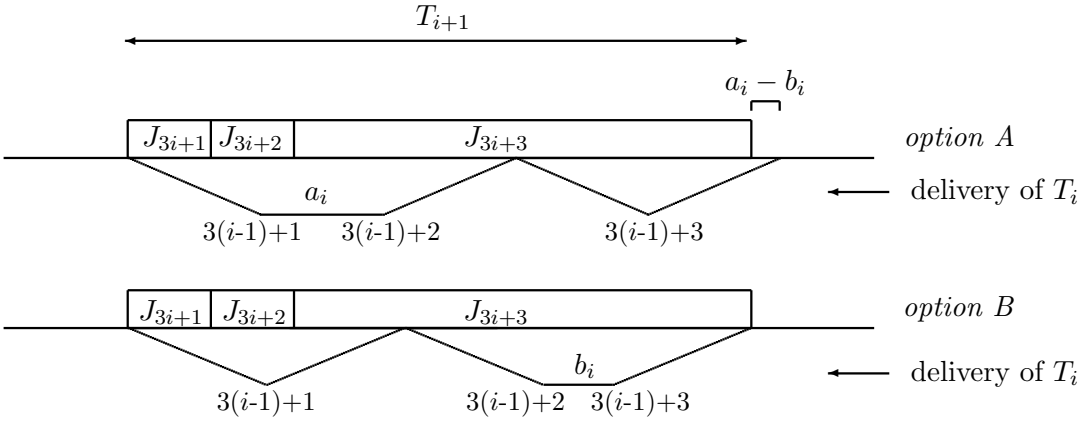

Figure 1: Round trips with options A and B.

We call these two possibilities *option A* and *option B* respectively (see Fig. 1). Namely, let us view option B as the *Base option*, and A as a variant to it.

Round trip length. Let M_i^A and M_i^B denote the round trip length of the jobs of T_i in the two cases. One has:

$$M_i^A = 4x_i + a_i \tag{6}$$

$$M_i^B = 4x_i + b_i \tag{7}$$

Since $a_i > b_i$, option A implies a longer round trip length than the Base option. The difference between the two (i.e., the additional time with option A with respect to B) is precisely $a_i - b_i$. Note in Fig. 1 that in option B, the delivery time of T_i is precisely the production completion time of T_{i+1} as, by definition, $p_{3i+1} + p_{3i+2} + p_{3i+3} = 4x_i + b_i$. This remark yields the following lemma.

Lemma 2.3 *In any triple-oriented schedule, the vehicle is never idle, except possibly before loading J_{3n+1}.*

Proof. Let us consider the first triple T_1. The vehicle starts at time 0 (to deliver batch $\{J_1\}$ or $\{J_1, J_2\}$), and is back at time $4x_1 + b_1$ or at time $4x_1 + a_1$. The completion time of T_2 is precisely equal to $C_6 = \sum_{j=1}^{6} p_j = 1 + 1 + 4x_1 + b_1 - 2 = 4x_1 + b_1$. Therefore, as $a_1 > b_1$, the vehicle can immediately start the delivery of the jobs of T_2 as soon as it is back to the depot. For the same reasons, the delivery of the jobs of T_i cannot be smaller than, the duration of the jobs of T_{i+1}, and the vehicle will be able to start immediately the delivery of the jobs of T_{i+1}. This reasoning does not hold for the last triple T_{n+1} because the duration of J_{3n+1} is different.

□

In view of Lemma 2.3, one can compute the total delivery time in the Base scenario, i.e., when option B is *always* chosen. From (7), one has that the vehicle delivering the last two jobs of T_i always returns to M exactly at time C_{3i+3} (see Fig. 2). Therefore, the last time the vehicle arrives at M (before delivering the jobs of T_{n+1}) is $C_{3n} + 4x_n + b_n$. Because of the definition of p_{3n+1}, and because J_{3n+1} starts at time C_{3n}, we have $C_{3n} + 4x_n + b_n = C_{3n+1} - Q/2$. In this case, the vehicle will stay idle from $C_{3n+1} - Q/2$ to C_{3n+1}, when job J_{3n+1} can be finally loaded and delivered at time C_{3n+1}. The three jobs of T_{n+1} have zero travel times and the last two jobs have also zero durations, so all jobs of T_{n+1} can be delivered at $C_{3n+1} = C_{3n+2} = C_{3n+3}$. Finally, recalling that $C_1 = C_2 = C_3 = 0$, the first job of each triple T_i, for $i \geq 1$, is delivered at $C_{3i} + x_i$, the second job is delivered at $C_{3i} + 3x_i$ and the last job at time $C_{3i} + 3x_i + b_i + x_i$, as

illustrated in Fig. 2. Hence, we have:

$$f^{BASE} = \sum_{i=1}^{n}(3C_{3i} + 7x_i + b_i) + C_{3n+1} + C_{3n+2} + C_{3n+3} \tag{8}$$

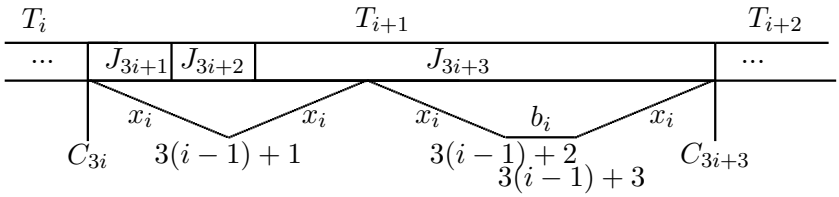

Figure 2: The base schedule (i.e., B is always chosen).

Contribution to total delivery time. Before computing the contribution of a certain triple to the total delivery time, let us consider schedules in which the deliveries of the last three jobs J_{3n+1}, J_{3n+2} and J_{3n+3} start exactly at their release time, i.e., at time $C_{3n+1} = C_{3n+2} = C_{3n+3}$ (options A and B are equivalent). Let us call *regular* a schedule in which such a condition holds.

Expression (8) refers to the scenario in which for all triples, the option B is chosen. We want now to compute the objective function of an arbitrary solution. Let us first consider the contribution of triple T_i to the objective function in the Base schedule, i.e., assuming that *the delivery of T_i started at time C_{3i}*, and let us denote this contribution as TDT_i^A and TDT_i^B depending on the selected option for T_i. One has:

$$\begin{aligned} TDT_i^A &= (C_{3i} + x_i) + (C_{3i} + x_i + a_i) + (C_{3i} + 3x_i + a_i) = 3C_{3i} + 5x_i + 2a_i \\ TDT_i^B &= (C_{3i} + x_i) + (C_{3i} + 3x_i) + (C_{3i} + 3x_i + b_i) = 3C_{3i} + 7x_i + b_i \end{aligned}$$

Note that

$$\begin{aligned} TDT_i^B - TDT_i^A &= 2x_i + b_i - 2a_i \\ &= (3a_i - 2b_i + 3(n-i)(a_i - b_i)) + b_i - 2a_i \\ &= a_i - b_i + 3(n-i)(a_i - b_i) \end{aligned} \tag{9}$$

which is positive, remembering that $a_i > b_i$. This means that choosing option A over B brings a benefit in terms of total delivery time. However, such favorable situation for option A is mitigated by the fact that, with option A, one has a longer round trip time than with option B, by the amount $(a_i - b_i)$ (see Fig. 3). In a regular schedule, such increased round trip time will be "paid" by all subsequent jobs, except the last jobs J_{3n+1}, J_{3n+2} and J_{3n+3}. Hence, in a regular schedule the total effect (in favor of option B) on the subsequent jobs of choosing option A for T_i is given by

$$3(n-i)(a_i - b_i) \tag{10}$$

In conclusion, the *net benefit* of choosing option A over B for T_i in terms of objective function value is obtained subtracting (10) from (9), and in view of the definition of x_i (4), one has therefore that

$$\begin{aligned} NetBenefit_i &= (2x_i + b_i - 2a_i) - 3(n-i)(a_i - b_i) \\ &= a_i - b_i \end{aligned} \tag{11}$$

In conclusion, it turns out that, when A is chosen over the Base option, one has a larger round trip time, by $(a_i - b_i)$, but also a smaller contribution to total delivery time (also by the amount $(a_i - b_i)$)(see Fig. 4). So, given any regular triple-oriented schedule in which the last three jobs depart at their completion time, let T_A be the set of triples for which the option A is chosen. Then, from the above considerations, the value f of the objective function is given by

$$f = f^{BASE} - \sum_{i \in T_A} (a_i - b_i) \tag{12}$$

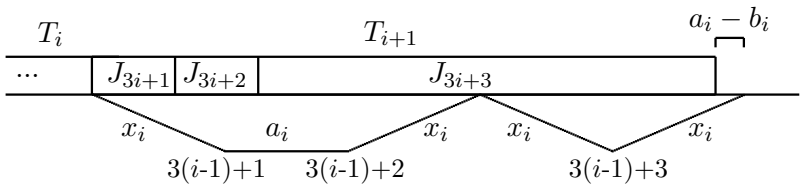

Figure 3: T_i is the first triple choosing option A.

On the other hand, the time at which the vehicle returns to M before loading the last three jobs (J_{3n+1}, J_{3n+2} and J_{3n+3}) is given by

$$C_{3n} + 4x_n + b_n + \sum_{i \in T_A} (a_i - b_i) \tag{13}$$

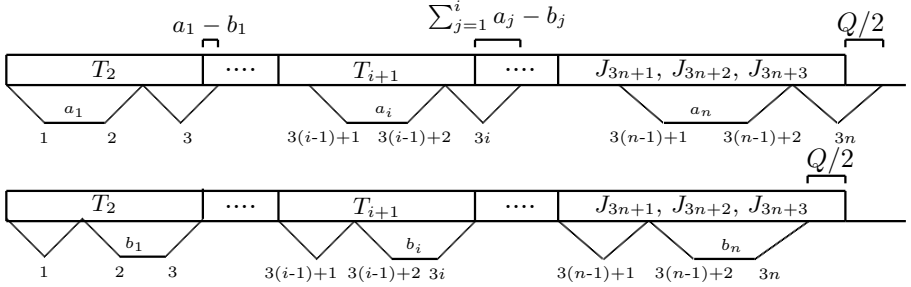

Figure 4: Round trips with option A only and option B only

Now, in a regular schedule the delivery of job J_{3n+1} (and also J_{3n+2} and J_{3n+3}) starts at time $C_{3n+1} = C_{3n} + 4x_n + b_n + Q/2$. Hence, from (13), in a regular schedule, it must hold:

$$\sum_{i \in T_A} (a_i - b_i) \leq Q/2$$

On the other hand, comparing (5), (8) it turns out that

$$f^* = f^{BASE} - Q/2$$

and hence, from (12), a regular schedule is feasible precisely if a subset T_A of indices exists such that $\sum_{i \in T_A}(a_i - b_i) = Q/2$, i.e., if and only if a feasible partition exists in the instance of MEOP. To conclude the proof, it is left to show that f^* can be attained only by a regular schedule. In fact, if a schedule is not regular, the departure time of the last batch is delayed by the amount $(\sum_{i \in T_A}(a_i - b_i) - Q/2)$ with respect to C_{3n+1}. As a consequence, the expression of f in (12) must be modified to take account of such delay of the last three jobs, i.e. it comes

$$f = f^{BASE} - \sum_{i \in T_A} (a_i - b_i) + 3(\sum_{i \in T_A} (a_i - b_i) - Q/2) = f^{BASE} + 2 \sum_{i \in T_A} (a_i - b_i) - 3Q/2 \tag{14}$$

Since, in a nonregular schedule,

$$\sum_{i \in T_A} (a_i - b_i) > Q/2,$$

from (14) one has

$$f = f^{BASE} + 2 \sum_{i \in T_A} (a_i - b_i) - 3Q/2 > f^{BASE} + Q - 3Q/2 = f^{BASE} - Q/2$$

and hence it cannot be feasible.

End of proof of theorem 2.1

□

By very similar arguments, it can be shown that the $1 \to D, k < n | v = 1, c,$ *fixed-seq*$| \sum D_j$ problem is NP-hard, when the number of locations denoted k is not fixed ('$k < n$' means that some consecutive jobs may have the same destination, leading at the end to $k < n$). On the other hand, we see in Section 3.2 that when the number of locations k is fixed (denoted K), the problem is polynomially solvable.

Remark 2.4 *Problem* $1 \to D, k < n | v = 1, c,$ *fixed-seq, split-deliv*$| \sum D_j$ *is also NP-hard. In this notation, 'split-deliv' denotes that split delivery is allowed, i.e. one can deliver a part of a job, come back to the depot, and take the reminder part of the job for another delivery. We remark that in this case, this problem is equivalent to the NP-hard problem* $1 \to D, k = n | v = 1, c,$ *fixed-seq*$| \sum D_j$*. Indeed, suppose that split delivery is allowed on the instance used to prove the complexity of* $1 \to D, k = n | v = 1, c,$ *fixed-seq*$| \sum D_j$*. One can see that each time the vehicle returns to the depot, the number of jobs already completed and ready for delivery is higher than the capacity c. Therefore, delivering a job in two successive trips has no interest and always increases the value of the objective function. The result can be extended to the case where the number of sites is fixed to* $K < n$*.*

2.3 Pseudopolynomial time algorithm for $1 \to D | v = 1, c,$ ***fixed-seq***$| \sum f_j(D_j)$

Theorem 2.1 implies that no optimal polynomial time algorithm can be found for problem $1 \to D | v = 1, c,$ *fixed-seq*$| \sum D_j$, and hence for more general objective functions, unless P=NP. In what follows, we show that problem $1 \to D | v = 1, c,$ *fixed-seq*$| \sum f_j(D_j)$ can be solved in pseudopolynomial time, proving that the problem is ordinary NP-hard.

We denote by $\{i, j\}$ the batch consisting of jobs $J_i, \ldots, J_j$. As usual, C_j is the completion time of job J_j (known because σ is known), and hence the release time for delivery. We denote by $M(i, j)$ the duration of the round trip of batch $\{i, j\}$, and, if the batch starts at time t, we call $K(i, j, t)$ its contribution to the objective function. Also, we assume that at the beginning, the vehicle is at the manufacturing location.

We denote by $F(i, j, t)$ the value of the optimal solution of the problem restricted to the first j jobs, in which the first job of the last batch is J_i, and such that the delivery of the last batch starts at time t. Then, $F(i, j, t)$ can be computed by means of a simple recursive formula. In the optimal solution of the subproblem, if the second last batch is $\{p, i-1\}$, and if it starts at time s, then we have:

$$F(i, j, t) = F(p, i-1, s) + K(i, j, t)$$

Note that, if the vehicle starts at time s, it must be back before or at time t, i.e., the following constraint must hold:

$$C_{i-1} \leq s \leq t - M(p, i-1)$$

In conclusion, the problem is solved by means of:

$$F(i, j, t) = \min_{\substack{\max(i-c,1) \leq p \leq i-1 \\ C_{i-1} \leq s \leq t - M(p,i-1)}} \{F(p, i-1, s)\} + K(i, j, t) \tag{15}$$

Let T be an upper bound on the latest possible departure time for the last batch. As long as the triangle inequality holds, this is given, for instance, by:

$$T = \max \left(\max_{1 \leq i \leq n-1} \{C_i + 2 \sum_{h=i}^{n-1} t_{hM}\}, C_n \right)$$

The optimal solution value is given by (assuming $c \leq n$):

$$z^* = \min_{n-c+1 \leq i \leq n, C_n \leq t \leq T} \big(F(i,n,t)\big)$$

A few boundary conditions must be imposed:

$$F(i,j,t) = +\infty \text{ for all } j < i \quad (16)$$

$$F(1,j,t) = K(1,j,t) \text{ for all } j,t \quad (17)$$

Condition (16) is obvious. Condition (17) allows to initialize the algorithm.

Let us turn to complexity. First, consider the computation of values $M(i,j)$ and $K(i,j,t)$. Both can be simply computed adding the contribution of the next job in the batch either to the round trip time (for $M(i,j)$) or to the objective function (for $K(i,j,t)$). More precisely, the transportation time d_h of job J_h is simply given by:

$$d_h = \begin{cases} d_{h-1} + t_{h-1,h}, & \text{if } i < h \leq j \\ t_{M,h}, & \text{if } h = i \text{ (in this case the vehicle starts from the depot)} \end{cases}$$

Notice that the delivery time of J_h is equal to $D_h = d_h + t$, with t the departure time of the batch.

Hence, $M(i,j)$ is simply given by $d_j + t_{j,M}$. Note that $M(i,j+1) = M(i,j) - t_{j,M} + t_{j,j+1} + t_{j+1,M}$. This means that all $M(i,j)$ can be computed in $O(nc)$ assuming $j \leq i+c-1$. Similarly, if batch $\{i,j\}$ starts indeed at time t, the contribution of job J_h to the objective function is given by:

$$f_h(t + d_h), \forall i \leq h \leq j$$

$K(i,j,t)$ is given by $\sum_{h=i}^{j} f_h(t+d_h)$. Again, assuming that $f_j(\cdot)$ can be computed in constant time, note that $d_{j+1} = d_j + t_{j,j+1}$ and $K(i,j+1,t) = K(i,j,t) + f_{j+1}(d_{j+1})$. So, all values $K(i,j,t)$ can be computed in $O(ncT)$.

Once all values $M(i,j)$ and $K(i,j,t)$ are known, one can compute formula (15) for all feasible triples (i,j,t). Each such computation requires comparing nT values. Finally, $O(cT)$ values are compared to find the optimal solution. Since there are $O(ncT)$ feasible triples, the computation of all values $F(i,j,t)$ clearly dominates the other phases, and the following result is proved.

Theorem 2.5 *Problem* $1 \to D|v = 1, c,$ *fixed-seq*$|\sum f_j(D_j)$ *can be solved in* $O(nc^2T^2)$.

Notice that c is bounded by n and that the pseudopolynomiality of the algorithm is due to T.

3 Special cases

In this section we address the complexity of two special cases of $1 \to D|v = 1, c,$ *fixed-seq*$|\sum D_j$, namely:

- The case in which the travel times are constant, denoted by $1 \to D|v = 1, c,$ *fixed-seq*$, t_{i,j} = t|\sum D_j$.
- The case in which the number of locations is known and fixed, denoted by $1 \to D, K$ *fixed* $|v = 1, c,$ *fixed-seq*$|\sum D_j$.

We show that these problems can be solved in polynomial time via dynamic programming algorithms.

3.1 Problem $1 \to D|v = 1, c,$ ***fixed-seq***$, t_{i,j} = t| \sum D_j$

In this section we investigate the special case of $1 \to D|v = 1, c,$ *fixed-seq*$| \sum D_j$ in which travel times (including the travel times to and from the depot) are equal to a constant t. Such a special case can be considered a reasonable approximation whenever the total time between two locations is dominated by some fixed-time loading/unloading activities.

We start by analyzing some properties of an optimal solution. Clearly, every time the vehicle is back at the depot, it can restart immediately with a new batch consisting of the jobs already completed, or it can wait for the completion of some jobs to be delivered (see Fig. 5).

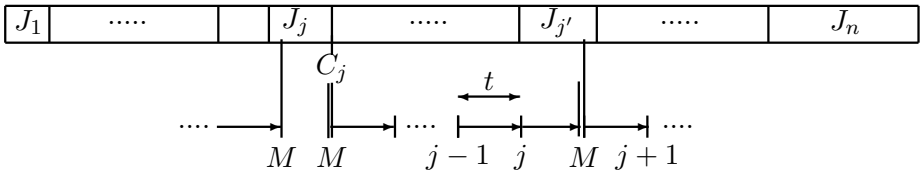

Figure 5: Start of a new round trip.

Following Li et al.(2005), we call NSS (*Non Stop Shipment*) a sequence of consecutive round trips during which the vehicle is never waiting at the depot (see Fig. 6), followed by a waiting time. We denote by $NSS[i, n_i, j]$ a NSS starting at time C_i, i.e. J_i is the last job of the first round trip of the NSS containing n_i jobs and ending strictly before C_j (when another NSS will start).

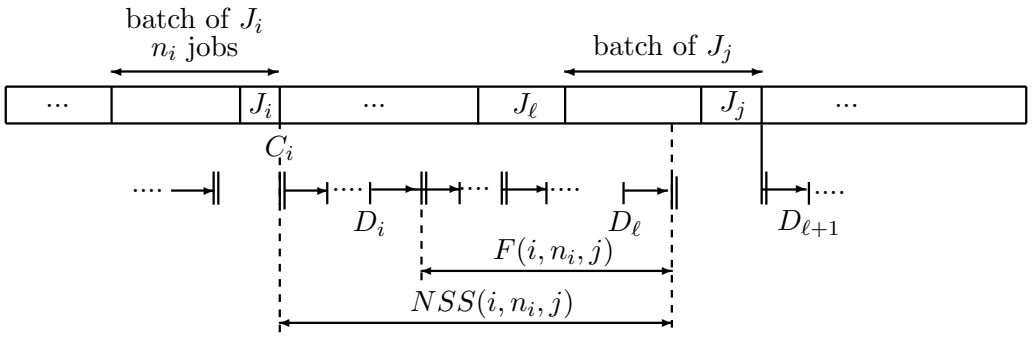

Figure 6: Illustration of a NSS

Suppose that a vehicle starts a round trip at a certain time τ. Let $\mathcal{J}$ be the set of jobs completed before τ (or at time τ) and not delivered. The round trip starting at time τ is called *maximal* if (i) the batch contains the first c jobs of $\mathcal{J}$, or (ii) it contains all the jobs of $\mathcal{J}$. The following proposition gives a key feature of an optimal solution.

Proposition 3.1 *There exists an optimal solution in which all round trips are maximal.*

Proof. Let us denote by R_q the round trip starting at τ. Let $J_i, \ldots, J_j$ be the jobs completed at or before τ and not yet delivered. The round trip R_q is maximal if it contains $\min\{c, j-i+1\}$ jobs. Suppose that R_q is not maximal, i.e. $|R_q| = k - i + 1 < \min\{c, j - i + 1\}$, i.e., the last job in R_q is job J_k, $k \leq j - 1$ (see Fig. 7(a)). This means that job J_{k+1} is delivered in the next batch R_{q+1}. One can move job J_{k+1} from R_{q+1} to R_q because R_q is not maximal. As a consequence, the delivery time of job J_{k+1} decreases by t, without changing the delivery times of *all* the subsequent jobs (see Fig. 7(b)). Hence, the new solution is better than the previous one. Suppose that R_q is maximal, then we are done. One can repeat the whole process for $k + 2, \ldots, \min\{c, j - i + 1\}$, and the proposition follows. □

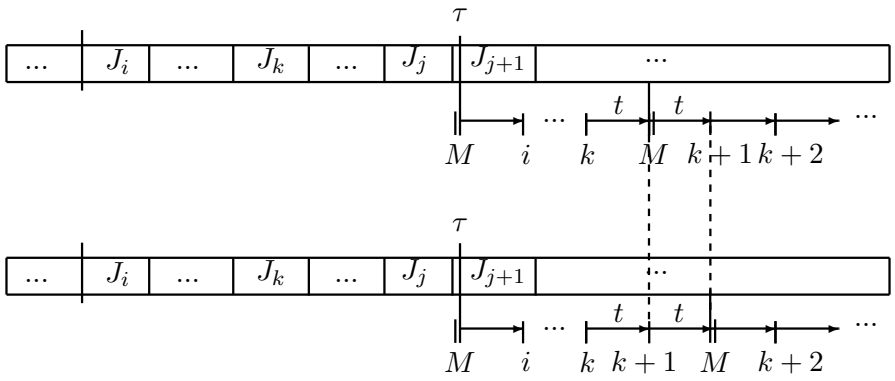

Figure 7: Maximal round trips

From this, we can see that a solution of the problem is composed by successive NSS. However, a round trip may have to wait for some additional jobs, before starting its route.

Example: Let us consider an instance with $n = 4$ jobs, processing times equal to $p = (1, 1, 10, 6)$, a travel time equal to 5 and a capacity of 2. The solution without waiting times where the vehicle starts after job J_1 has a total delivery time of 79, the solution without waiting times where the vehicle starts after job J_2 has a total delivery time of 73. The optimal solution consists in delivering $\{J_1, J_2\}$ in the same batch and then to wait for the completion of J_4 for delivering $\{J_3, J_4\}$ in a same batch too. The optimal solution has a total delivery time of 70. The three solutions are illustrated in Fig. 8.

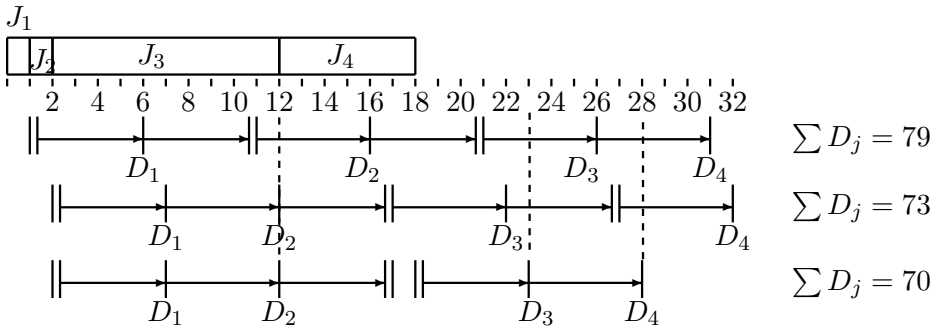

Figure 8: Illustration of the example

From the previous properties, one can propose a polynomial time algorithm to build any NSS.

Given jobs J_i, J_j, with $j > i$, and an integer n_i, we let $NSS[i, n_i, j]$ denote the following NSS. The vehicle starts at C_i, carrying n_i jobs (i.e., jobs $\{J_{i-n_i+1}, J_{i-n_i+2}, \ldots, J_i\}$), and goes back to the depot (first round trip). Thereafter, the vehicle performs a number of consecutive maximal round trips, without ever waiting at the depot (successive round trips). We let J_l denote the last job that can be delivered in the NSS, so that the vehicle is back at the depot before or at C_j (see Fig. 6). Notice that once J_i, n_i and J_j are known, J_l is uniquely determined and we denote by $\nu_{i,n_i,j}$ the number of undelivered jobs $\{J_{l+1}, \ldots, J_j\}$, i.e., $\nu_{i,n_i,j} = j - l$. At time C_j, another NSS starts in which the first round trip delivers the undelivered jobs $\{J_{l+1}, \ldots, J_j\}$. We denote by $B_{i,j}$ the number of round trips of $NSS[i, n_i, j]$ (which is also uniquely determined). The number of jobs delivered by the round trip number k ($1 \leq k \leq B_{i,j}$) is denoted by n'_k (we have $n'_1 = n_i$). The starting time of the first round trip of $NSS[i, n_i, j]$ is C_i, the starting time of the second round trip of $NSS[i, n_i, j]$ is $C_i + t(n_i + 1)$, of the third round trip is $C_i + t(n_i + 1 + n'_2 + 1)$,

..., the starting time of the round trip number k is equal to $C_i + t\sum_{k'=1}^{k-1}(n'_{k'}+1)$.

We denote by $F(i,n_i,j)$ the total contribution to the objective function of the second, third,..., $B_{i,j}$-th round trip, i.e., the round trips in which jobs $J_{i+1},\ldots,J_l$ $(j-c \le l < j)$ are delivered (see Fig. 6). Hence, we have:

$$\begin{aligned} F(i,n_i,j) &= \sum_{k=2}^{B_{i,j}} \sum_{r=1}^{n'_k} \left(C_i + t\sum_{k'=1}^{k-1}(n'_{k'}+1) + rt \right) \\ &= \sum_{k=2}^{B_{i,j}} \left(n'_k\big(C_i + t\sum_{k'=1}^{k-1}(n'_{k'}+1)\big) + t\frac{n'_k(n'_k+1)}{2} \right) \end{aligned} \tag{18}$$

Given n_i, J_i and J_j, Proposition 3.1 enables one to construct each $NSS[i,n_i,j]$ in $O(n)$, and therefore also each $F(i,n_i,j)$ can be calculated in $O(n)$.

We define $F(0,n_0,j)$ $(\forall j,\ 1 \le j \le c)$ as the contribution to the objective function of the first n_0 jobs $J_1,\ldots,J_j$, such that these jobs are delivered in the same first round trip, starting at time C_j. We have:

$$\begin{aligned} F(0,n_0,j) &= \sum_{l=1}^{j}(C_j + lt),\ \forall j, 1 \le j \le c, n_0 = j \\ &= \infty,\ \text{otherwise} \end{aligned}$$

Two particular cases are identified, where $NSS[i,n_i,j]$ cannot exist:

1. $C_i + t(n_i+1) > C_j$: in this case, the round trip delivering job J_i finishes after the completion time of C_j and thus $NSS[i,n_i,j]$ cannot exist.
2. $\nu_{i,n_i,j} > c$: job J_j cannot be delivered in the first tour of the next NSS starting at time C_j.

In both these cases we set $F(i,n_i,j) = +\infty$.

Let now $f(j,n_j)$ be the minimum total delivery cost of the problem restricted to jobs $\{J_1,...,J_j\}$, under the condition that the batch delivering J_j contains n_j jobs and starts at time C_j:

$$f(j,n_j) = \min_{\substack{0 \le i < j \\ 1 \le n_i \le c \\ \nu_{i,n_i,j} = n_j}} \left\{ f(i,n_i) + F(i,n_i,j) + \sum_{r=1}^{n_j}(C_j + rt) \right\} \tag{19}$$

with

$$\begin{aligned} f(0,n_j) &= 0, \quad \text{if } n_j = 0 \\ &= \infty, \quad \text{otherwise.} \end{aligned}$$

Since, in the optimal solution, the last round trip does not necessarily start at time C_n, we introduce a dummy job J_{n+1} in the last position with $C_{n+1} = C_n + 2tn$. Since $2tn$ is the time needed to deliver all the n jobs in n round trips, it is an upper bound on the time that may be needed after C_n to deliver all remaining jobs. This implies that there exists an optimal solution in which the last round trip delivers only this dummy job at time $D_{n+1} = C_{n+1} + t$. Hence, the optimal solution value is equal to

$$z^* = f(n+1,1) - D_{n+1} \tag{20}$$

Theorem 3.2 *The problem* $1 \to D|v=1,c,\ \text{fixed-seq}, t_{i,j} = t|\sum D_j$ *is solved to optimality by the dynamic programming algorithm in time* $O(cn^3)$.

Proof. Each $F(i,n_i,j)$ can be computed in $O(n)$ by (18). Since there are $O(cn^2)$ non stop shipments $NSS(i,n_i,j)$, all $F(i,n_i,j)$ can be computed in $O(cn^3)$. In turn, from (19) each $f(j,n_j)$ can be computed in $O(cn)$, and there are $O(cn)$ values $f(j,n_j)$. Hence, the dynamic programming algorithm can be implemented in $O(c^2n^2)$ and since $c < n$ the complexity is dominated by the computation of all values $F(i,n_i,j)$. □

3.2 Problem $1 \to D, k = K | v = 1, c,$ ***fixed-seq***$| \sum D_j$

In this section, we present a polynomial time algorithm to solve the special case where the jobs going to the same location are produced consecutively, and the number of locations is fixed to K (of course, the running time of the algorithm will be exponential in K). It means that for two consecutive jobs J_i and J_{i+1} having the same location, $t_{i,i+1} = 0$. This case occurs when a manufacturer has to deliver consecutively produced parts to the same warehouse. The number of warehouses is often small, which makes the study of this case particularly relevant.

As in the algorithm in Section 3.1, we introduce a dummy job J_{n+1} in the last position with $C_{n+1} = C_n + 2\sum_{i=1}^{n} t_{iM}$ and a location $n+1$ for J_{n+1} with $t_{n+1,M} = 0$. This implies that there exists an optimal solution in which the last round trip delivers only the dummy job J_{n+1}.

In what follows, we denote by $F(i, n_i, j, n_j)$ the contribution to the objective function of the jobs $\{J_{i+1}, \ldots, J_{j-n_j}\}$. These jobs are delivered by a *Non Stop Shipment* (call it $NSS(i, n_i, j, n_j)$) such that:

- (i) The first round trip starts after the vehicle has delivered J_i in a batch with n_i jobs and is back at the depot
- (ii) The last round trip returns to the manufacturer after the delivery of job J_{j-n_j} and strictly before C_j.

We will show that the function $F(i, n_i, j, n_j)$ can be computed in polynomial time. As in the algorithm in Section 3.1, $F(i, n_i, j, n_j)$ takes the value $+\infty$ if $NSS(i, n_i, j, n_j)$ does not exist.

We denote by $f(j, n_j)$ the value of an optimal solution of the problem restricted to the first j jobs, in which the last batch contains n_j jobs $\{J_{j-n_j+1}, J_{j-n_j+2}, \ldots, J_j\}$, and such that this batch starts at the completion time C_j of the job J_j. Then, $f(j, n_j)$ can be computed by the following recursive formula.

$$f(j, n_j) = \min_{\substack{1 \le n_i \le c \\ n_i \le i \le j - n_j}} \{f(i, n_i) + F(i, n_i, j, n_j)\} + K(j, n_j) \tag{21}$$

Where $K(j, n_j)$ is the contribution to the objective function of the jobs $J_{j-n_j+1}, \ldots, J_j$. These jobs are delivered by a single round trip starting at C_j:

$$K(j, n_j) = \sum_{r=j-n_j+1}^{j} D_r, \quad \text{and} \quad D_r = t_{M,j-n_j+1} + \sum_{s=j-n_j+1}^{r-1} t_{s,s+1} \tag{22}$$

The optimal solution value is given by:

$$z^* = f(n+1, 1) - D_{n+1} \tag{23}$$

In order to establish the complexity of the algorithm, we first propose an enumeration algorithm which returns the value $F(i, n_i, j, n_j)$ and we show that this value can be obtained in polynomial time. Let us first present the property on which the analysis is based, and suppose initially that $c = \infty$.

Property 3.3 *Whenever, during a NSS, a vehicle starts performing a round trip at time t, the batch contains either* all *released jobs at t that are bound to the same location, or* none *of them.*

Such property can be easily proved observing that, if there are no capacity restrictions, it does not make sense to deliver only *a few* jobs going to a given location ℓ, while others bound to ℓ are available. Doing so would only force the subsequent round trip to also visit location ℓ, with no convenience. Hence, if the available jobs at time t span locations $\ell, \ell+1, \ldots, \ell+q$, either the next batch will include all released jobs going to ℓ, or all released jobs going to ℓ and $\ell+1$, ..., or all released jobs going to $\ell, \ell+1, \ldots, \ell+q$. If $c < \infty$, the reasoning remains valid, except that now the jobs bound to the same location included in the same batch are either 0 or enough to fill vehicle capacity.

Algorithm 1 uses this property to list all batch partitions that give an NSS. For all $l \in \{i, \ldots, j - n_j\}$, we denote by $N(l)$ the set of *labels on* J_l. A label $(z, t) \in N(l)$ for $l \geq i + 1$ corresponds to a partition of jobs $J_{i+1}, \ldots, J_l$ in batches, z represents the contribution to the objective function of the jobs $J_{i+1}, \ldots, J_l$ and t is the vehicle return time after delivering job J_l. An initial label is set in $N(i)$, giving the contribution to the objective function of the first batch with n_i jobs and its return time, which are fixed. For each job $J_l \in \{J_i, \ldots, J_{j-n_j-1}\}$, the algorithm extends each label on $N(l)$ according to Property 3.3. We denote by $M(l + 1, l')$ the duration of the round trip of batch $\{J_{l+1}, \ldots, J_{l'}\}$, and by $Z(l + 1, l', t)$ its contribution to the objective function if the batch starts at time t. Finally, we consider the set $A(l, t)$ of relevant batches to consider (i.e., following Property 3.3), given that each of them starts at time t after delivering job J_l. So, the next batch is either composed of (i) jobs going to the same location as J_{l+1}, or (ii) all jobs going to the same location as J_{l+1} and jobs going to the next location, or (iii) all jobs going to the same location as J_{l+1}, all jobs going to the next location and jobs going to the subsequent location, ... and so on, until either we reach time t or vehicle capacity limit. Given job J_l, we let $A(l, t)$ denote the set of such *dominant* batches. Since $A(l, t)$ contains batches in which the first job is J_{l+1}, a batch of $A(l, t)$ is completely identified by its last job $J_{l'}$. Note that the largest batch in $A(l, t)$ is $\{J_{l+1}, \ldots, J_{l+s}\}$ where $s = \min(c, \max\{s' | l + s' \leq j - n_j, C_{l+s'} \leq t\})$.

Algorithm 1 Enumeration algorithm

$N(i) \leftarrow (Z(i - n_i + 1, i, C_i), C_i + M(i - n_i + 1, i))$
For $l = i$ to $j - n_j - 1$ do
  For $(z, t) \in N(l)$ do
    For l' such that $\{J_{l+1}, \ldots, J_{l'}\} \in A(l, t)$ do
      $t' = t + M(l + 1, l')$
      $z' = z + Z(l + 1, l', t)$
      $N(l') \leftarrow (z', t')$
    End
  End
End

Theorem 3.4 *Algorithm 1 returns the value of $F(i, n_i, j, n_j)$ by enumerating a polynomial number of labels.*

The proof has the following scheme.

1. We first define an upper bound of the number of labels generated in $N(j - n_j)$, i.e., the number of NSSs.
2. We give a polynomial expression of this bound for a capacity value c.

Point 1. Let $h(l, z, t)$ the number of labels on $N(j - n_j)$ obtained by extension of the label $(z, t) \in N(l)$. Following the algorithm, we have the following relation.

$$h(l, z, t) = \sum_{l' \in A(l,t)} h(l', z', t'), \text{ where } t' = t + M(l + 1, l') \text{ and } z' = z + Z(l + 1, l', t)$$

Let $g(l)$ denote an upper bound of $h(l, z, t)$ for any label $(z, t) \in N(l)$. For all $l \in \{i, \ldots, j - n_j - 1\}$, a valid upper bound can be given by the following recursive formula.

$$g(l) = \begin{cases} 0 & \text{for } l = j - n_j \\ 1 & \text{for } l = j - n_j - 1 \\ \max\limits_{\forall \tau \geq C_l} \sum_{l' \in A(l,\tau)} g(l') & \text{for } l \in \{i, \ldots, j - n_j - 2\} \end{cases} \tag{24}$$

Indeed, there is only one way to extend a label in $N(j-n_j-1)$ as only job J_{j-n_j} is available. The recursion then comes from the definition of $h(l,z,t)$. An upper bound of the total number of generated labels by the $NSS(i,n_i,j,n_j)$ on $N(j-n_j)$ is then given by $g(i)$.

Point 2. For the sake of simplicity, we use ℓ_j to refer to the destination of job J_j.

Lemma 3.5 *Let $c \in \{1,\ldots,n\}$. The number of batching schemes to deliver the jobs $J_l,\ldots,J_{j-n_j}$ of $NSS(i,n_i,j,n_j)$ such that $l>i$ is not larger than $2^{\ell_{j-n_j}-\ell_l}$.*

Proof. Note that $g(l-1)$ is by definition a valid upper bound on the number of batching schemes to deliver the jobs $J_l,\ldots,J_{j-n_j}$ of $NSS(i,n_i,j,n_j)$. We want to prove that, for all $l>i$,

$$g(l-1) \leq 2^{\ell_{j-n_j}-\ell_l}. \tag{25}$$

The proof of (25) is by induction on l. Following equation (24), $g(j-n_j-1)=1$, so (25) holds for $l=j-n_j$.
Let us now assume that (25) is true for all $q \in \{l+1,\cdots,j-n_j\}$ (i.e., $g(q-1) \leq 2^{\ell_{j-n_j}-\ell_q}$) and we want to show that is true also for $q=l$. We have :

$$\begin{aligned} g(l-1) &= \max_{\forall \tau \geq C_{l-1}} \sum_{l' \in A(l-1,\tau)} g(l') \\ &\leq \max_{\forall \tau \geq C_{l-1}} \sum_{l' \in A(l-1,\tau)\setminus\{j-n_j\}} 2^{\ell_{j-n_j}-\ell_{l'+1}} \text{ by induction, since } l' \geq l \text{ and } g(j-n_j)=0 \end{aligned}$$

In order to obtain the worst case, we assume that τ is such that c jobs are released at τ (or J_{j-n_j} is reached), which gives:

$$g(l-1) \leq \sum_{l' \in A(l-1,\tau)\setminus\{j-n_j\}} 2^{\ell_{j-n_j}-\ell_{l'+1}} \tag{26}$$

Replacing $\ell_{l'+1}$ by $\ell_l + (\ell_{l'+1}-\ell_l)$ in equation (26), one has

$$\begin{aligned} \sum_{l' \in A(l-1,\tau)\setminus\{j-n_j\}} 2^{\ell_{j-n_j}-\ell_{l'+1}} &= \sum_{l' \in A(l-1,\tau)\setminus\{j-n_j\}} 2^{\ell_{j-n_j}-\ell_l-(\ell_{l'+1}-\ell_l)} \\ &= 2^{\ell_{j-n_j}-\ell_l} \sum_{l' \in A(l-1,\tau)\setminus\{j-n_j\}} \frac{1}{2^{\ell_{l'+1}-\ell_l}} \end{aligned} \tag{27}$$

In order to compute the equation (27), let us observe that the only case where we can have $2^{\ell_{l'+1}-\ell_l}=2^0$ is the case where the set of dominant batches $A(l-1,\tau)$ contains exactly one job (identifying one batch) and such batch is bound to location ℓ_l. Let J_e the job identifying this batch. Note that this case is possible if all released jobs at τ are bound to the same location. Following equation (27), one have.

$$\frac{1}{2^{\ell_{e+1}-\ell_l}} = \begin{cases} 1 & \text{if } \ell_{e+1}=\ell_e \\ 1/2 & \text{if } \ell_{e+1}=\ell_e+1 \end{cases}$$

Then in any case, we have

$$\sum_{l' \in A(l-1,\tau)\setminus\{j-n_j\}} \frac{1}{2^{\ell_{l'+1}-\ell_l}} \leq 1, \tag{28}$$

and following equations (26)-(28), one has

$$g(l-1) \leq 2^{\ell_{j-n_j}-\ell_l}.$$

This proves that the number of labels on $N(j - n_j)$ (i.e., $g(i)$) is not larger than by 2^{K-1}. As we have n label sets $N(l)$, the total number of generated labels is upper bounded by $n2^{K-1}$, which yields the worst-case time complexity of Algorithm 1. □

Finally, z^* can be computed through formulas (21) and (23) in $O(n^3c^2 2^{K-1})$, which is polynomial for K fixed. We note here that this result shows that problem $1 \to D|v = 1, c,$ fixed-seq$|\sum D_j$ is fixed-parameter tractable, when parameterized by the number of different locations.

4 Conclusion

In this paper, we focus on the coordination of a single machine production scheduling problem and a single vehicle delivery problem. The jobs are processed on a single machine and delivered in batches to customers by a single vehicle with limited capacity. It is assumed that the production sequence is given, and is supposed to be the same as the delivery sequence. Therefore, the problem is to form batches of jobs and the objective function is to minimize the sum of the delivery times. We prove that the problem is NP-hard and propose a pseudopolynomial time dynamic programming algorithm. Some particular cases of the problem are studied. The case where transportation times are constant is solved in polynomial time by a dynamic programming algorithm, as well as the case where the number of locations is fixed. The latter result also shows that the problem is fixed-parameter tractable.

More cases can be investigated in the future under the hypotheses of fixed sequences, as for example the case where processing times of jobs are identical. We propose also to investigate the general case where the sequence is not fixed and has to be determined.

ACKNOWLEDGEMENT

This work was supported by the financial support of the ANR ATHENA project, grant ANR-13-BS02-0006 of the French Agence Nationale de la Recherche.

References

[Agnetis et al., 2014] Agnetis, A., Aloulou, M. A., and Fu, L.-L. (2014). Coordination of production and interstage batch delivery with outsourced distribution. *European Journal of Operational Research*, 238(1):130 – 142.

[Agnetis et al., 2015] Agnetis, A., Aloulou, M. A., Fu, L.-L., and Kovalyov, M. Y. (2015). Two faster algorithms for coordination of production and batch delivery: A note. *European Journal of Operational Research*, 241(3):927 – 930.

[Armstrong et al., 2008] Armstrong, R., Gao, S., and Lei, L. (2008). A zero-inventory production and distribution problem with a fixed customer sequence. *Annals of Operations Research*, 159(1):395–414.

[Chang and Lee, 2004] Chang, Y.-C. and Lee, C.-Y. (2004). Machine scheduling with job delivery coordination. *European Journal of Operational Research*, 158(2):470 – 487. Methodological Foundations of Multi-Criteria Decision Making.

[Chen and Lee, 2008] Chen, B. and Lee, C.-Y. (2008). Logistics scheduling with batching and transportation. *European Journal of Operational Research*, 189(3):871 – 876.

[Chen, 2010] Chen, Z.-L. (2010). Integrated production and outbound distribution scheduling: Review and extensions. *Operations Research*, 58(1):130–148.

[Chen and Pundoor, 2006] Chen, Z.-L. and Pundoor, G. (2006). Order assignment and scheduling in a supply chain. *Operations Research*, 54(3):555–572.

[Chen and Vairaktarakis, 2005] Chen, Z.-L. and Vairaktarakis, G. L. (2005). Integrated scheduling of production and distribution operations. *Management Science*, 51(4):614–628.

[Fan et al., 2015] Fan, J., Lu, X., and Liu, P. (2015). Integrated scheduling of production and delivery on a single machine with availability constraint. *Theoretical Computer Science*, 562:581 – 589.

[Gao et al., 2015] Gao, S., Qi, L., and Lei, L. (2015). Integrated batch production and distribution scheduling with limited vehicle capacity. *International Journal of Production Economics*, 160:13 – 25.

[Garey et al., 1988] Garey, M. R., Tarjan, R. E., and Wilfong, G. T. (1988). One-processor scheduling with symmetric earliness and tardiness penalties. *Mathematics of Operations Research*, 13(2):330–348.

[Hall et al., 2001] Hall, N. G., Lesaoana, M., and Potts, C. N. (2001). Scheduling with fixed delivery dates. *Operations Research*, 49(1):134–144.

[Hurink and Knust, 2001] Hurink, J. and Knust, S. (2001). Makespan minimization for flowshop problems with transportation times and a single robot. *Discrete Applied Mathematics*, 112(13):199 – 216. Combinatorial Optimization Symposium, Selected Papers.

[Lee and Chen, 2001] Lee, C.-Y. and Chen, Z.-L. (2001). Machine scheduling with transportation considerations. *Journal of Scheduling*, 4(1):3–24.

[Lenté and Kergosien, 2014] Lenté, C. and Kergosien, Y. (2014). Problème de livraison a séquence fixée. In *10ème conférence internationale de modélisation, optimisation et simulation (MOSIM'14), Nancy*.

[Leung and Chen, 2013] Leung, J. Y.-T. and Chen, Z.-L. (2013). Integrated production and distribution with fixed delivery departure dates. *Operations Research Letters*, 41(3):290 – 293.

[Li and Ou, 2005] Li, C.-L. and Ou, J. (2005). Machine scheduling with pickup and delivery. *Naval Research Logistics (NRL)*, 52(7):617–630.

[Li et al., 2005] Li, C.-L., Vairaktarakis, G., and Lee, C.-Y. (2005). Machine scheduling with deliveries to multiple customer locations. *European Journal of Operational Research*, 164(1):39 – 51.

[Stecke and Zhao, 2007] Stecke, K. E. and Zhao, X. (2007). Production and transportation integration for a make-to-order manufacturing company with a commit-to-delivery business mode. *Manufacturing & Service Operations Management*, 9(2):206–224.

[Tsirimpas et al., 2008] Tsirimpas, P., Tatarakis, A., Minis, I., and Kyriakidis, E. (2008). Single vehicle routing with a predefined customer sequence and multiple depot returns. *European Journal of Operational Research*, 187(2):483 – 495.

[Viergutz and Knust, 2014] Viergutz, C. and Knust, S. (2014). Integrated production and distribution scheduling with lifespan constraints. *Annals of Operations Research*, 213(1):293–318.

[Wang and Cheng, 2009] Wang, X. and Cheng, T. (2009). Production scheduling with supply and delivery considerations to minimize the makespan. *European Journal of Operational Research*, 194(3):743 – 752.

[Zhong and Chen, 2015] Zhong, W. and Chen, Z.-L. (2015). Flowshop scheduling with interstage job transportation. *Journal of Scheduling*, 18(4):411–422.

[Zhong et al., 2010] Zhong, W., Chen, Z.-L., and Chen, M. (2010). Integrated production and distribution scheduling with committed delivery dates. *Operations Research Letters*, 38(2):133 – 138.