Theoretical Computer Science 422 (2012) 39–51

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

# Coordinated scheduling of production and delivery with production window and delivery capacity constraints

Bin Fu [a], Yumei Huo [b,*], Hairong Zhao [c]

[a] Department of Computer Science, University of Texas—Pan American, Edinburg, TX 78539, USA

[b] Department of Computer Science, College of Staten Island, CUNY, Staten Island, NY 10314, USA

[c] Department of Mathematics, Computer Science & Statistics, Purdue University, Hammond, IN 46323, USA

ARTICLE INFO

*Article history:*
Received 25 July 2011
Received in revised form 19 November 2011
Accepted 29 November 2011
Communicated by D.-Z. Du

*Keywords:*
Coordinated scheduling
Production
Delivery
PTAS

ABSTRACT

This paper addresses the problem of coordinated scheduling of production and delivery subject to the production window constraint and the delivery capacity constraint. We have a planning horizon consisting of one or more delivery times each with a unique delivery capacity. There is a set of jobs each with a committed delivery time, processing time, production window, and profit. The company can earn the profit only if the job is processed in its production window and delivered before its committed delivery time. From the company point of view, we are interested in selecting a subset of jobs to process and deliver so as to maximize the total profit subject to the delivery capacity constraint.

We consider both the single delivery time case and the multiple delivery time case. In both cases, the problem is strongly NP-hard since the subproblems at the production stage and at the delivery stage are both strongly NP-hard. Our goal is to design approximation algorithms. Suppose the jobs are $k$-disjoint, that is, the jobs can be partitioned into $k$ lists of jobs such that the jobs in each list have disjoint production windows. When $k$ is a constant, we developed the first PTAS for the single delivery case. For multiple delivery time case, we also develop a PTAS when the number of delivery times is a constant as well.



## 1. Introduction

Under the current competitive manufacturing environment, companies tend to put more emphasis on the coordination of different stages of a supply chain, i.e. suppliers, manufacturers, distributors and customers. Among these four stages, the issue of coordinating production and distribution (delivery) has been widely discussed.

This paper addresses the problem of coordinated scheduling of production and delivery subject to the production window constraint and the delivery capacity constraint. In the stage of production, each order has a production window which may or may not be customer defined. For example, a company may rely on another company to complete a sub-process, or may rely on a manufacturer to make the products or semi-products. With some pre-scheduled jobs, the manufacturer can only provide partial production line or in some cases, a production window for each order. The windows of different orders may overlap. Another example of production window is when the raw materials are perishable. After the arrival of raw materials, the company has to start the manufacturing process before certain time. Given the arrival schedule of raw materials, the company creates a production window for each order. Similarly, these production windows may overlap.

In the stage of distribution (delivery), a company may have its own transportation vehicles which deliver products at periodic or aperiodic times, or the company may rely on a third party to deliver, which picks up products at regular or

* Corresponding author. Tel.: +1 7189822841; fax: +1 7189822856.
*E-mail addresses:* binfu@cs.panam.edu (B. Fu), huo@mail.csi.cuny.edu (Y. Huo), hairong@purduecal.edu (H. Zhao).

0304-3975/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2011.11.035

irregular times. The products incurred by different orders may be delivered together if the destinations are close to each other, e.g. delivery to same countries by ships, delivery to same states or cities by flights, or delivery to same areas by trucks. The delivery capacity may vary at different delivery times and is always bounded.

In summary, in the production stage, the company has the constraint of production window, and in the delivery stage, the company has the constraint of delivery capacity and promised delivery date. Therefore, it is possible that not all orders can be processed within their production windows and delivered before the promised delivery times. The company has to decide which orders to accept in order to maximize the total profit based on the production window, the committed delivery date, the potential profit of each order and the overall delivery capacity.

This paper addresses the problem faced by the company under the above scenarios. We consider the commit-to-ship model, i.e. if an order is accepted, the company guarantees the products be shipped to the customer before the committed time, we call this time the *committed delivery date*; and if a third party logistics company is used, then the committed delivery date is the date by which the company gives products to a third party logistics company. We focus on the single machine production environment. We have a set of orders, each associated with a processing time, a size, a potential profit, a promised delivery time and a production window before delivery time. The company can earn the profit only if the order is processed in its production window and delivered before its promised delivery time. From the company point of view, we are interested in selecting a subset of orders in order to maximize the total profit. When selecting the orders, both production schedule and delivery schedule should be considered simultaneously. Thus we have the "coordinated scheduling problem": generate a coordinated schedule, which consists of a production schedule and a delivery schedule subject to the production window, committed delivery date, and delivery capacity constraints.

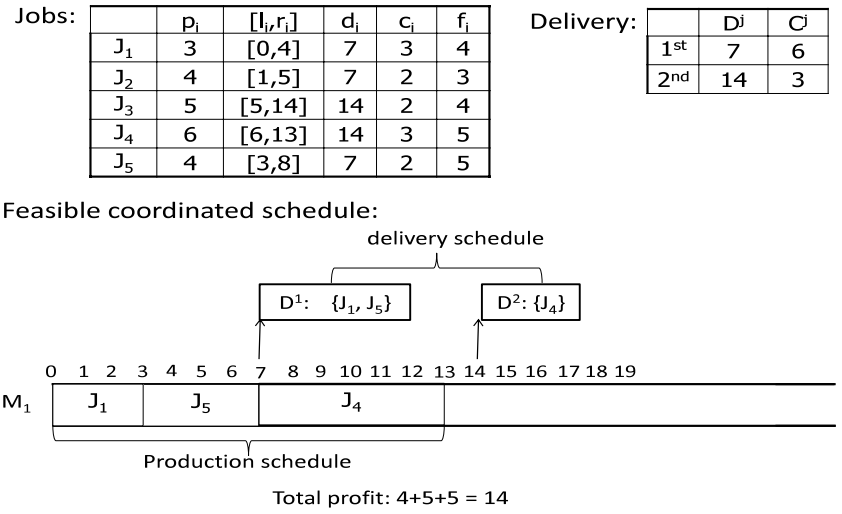
**Literature review.** Our problem combines two classical problems. The problem in the production stage generalizes a single machine real time scheduling problem: there are $n$ jobs each associated with a release time, a deadline, a weight, and a processing time and the goal is to find a nonpreemptive schedule that maximizes the total weight of the jobs that meet their deadline. In the standard notation for scheduling problems, the problem is $1 \mid r_i, d_i \mid w_i(1 - U_i)$. It is known that this problem is NP-hard in the strong sense (see [9,10]). Bar-Noy et al. [2] have shown that an LP formulation achieves a 2-approximation for polynomially bounded integral input and a 3-approximation for arbitrary input when the weights are arbitrary. They also showed that the problem is MAX-SNP hard when the jobs have identical weight and there are multiple unrelated machines. It remains open if this is the case for single machine with arbitrary job weights.

The problem in the delivery stage is the Multiple Knapsack Problem (MKP) with inclusive assignment restrictions, in which the assignment set of one item (i.e., the set of knapsacks that the item may be assigned to) must be either a subset or a superset of the assignment set of another item. Since MKP is a generalization of the classical knapsack problem, it is strongly NP-hard. Furthermore, it does not admit a Fully Polynomial Time Approximation Scheme (FPTAS) even if the number of bins is two [4]. Kellerer [14] gave a Polynomial Time Approximation Scheme (PTAS) for MKP with identical capacities and this result has been generalized by Chekuri and Khanna [4] who gave a PTAS for MKP with general capacities. Later, Jansen [13] developed an efficient PTAS for the MKP with general capacities. Recently, Kellerer et al. [15] developed a PTAS for multiple knapsack problem with inclusive assignment set restrictions when each job's weight equals to its size and the bins have identical capacity.

The coordinated production and delivery scheduling problems in general have received considerable interest in recent two decades. However, most of the research for this model is done at the strategic and tactical levels (see survey articles by Sarmiento and Nagi [17], Erenguc et al. [7], Goetschalckx et al. [11], Bilgen and Ozkarahan [3], and Chen [5] for example). At the operational scheduling level, Chen [6] gives a state-of-the-art survey of the models and results in this area. Based on the delivery mode, he classified the models into five classes: (1) models with individual and immediate delivery; (2) models with batch delivery to a single customer by direct shipping method; (3) models with batch delivery to multiple customers by direct shipping method; (4) models with batch delivery to multiple customers by routing method (5) models with fixed delivery departure date. In the first model, jobs have delivery windows, and thus production windows can be incurred, however, due to the immediate and individual delivery requirement, the problems under this model can be reduced to fixed-interval scheduling problems (without the delivery), which can be solved as a min-cost network flow problem [16]. For all other models, no production windows have been specially considered in the survey.

Several papers considered problems with time window constraints and/or delivery capacity constraints. Amstrong et al. [1] considered the integrated scheduling problem with batch delivery to multiple customers by routing method, subject to delivery windows constraints. The objective is to choose a subset of the orders to be delivered such that the total demand of the delivered orders is maximized. Garcia and Lozano [8] considered the production and delivery scheduling problems in which time windows are defined for the jobs' starting times. In their paper, orders must be delivered individually and immediately after they are manufactured, so delivery capacity is not an issue. In [12], Huo, Leung and Wang considered the integrated production and delivery scheduling problem with disjoint time windows where windows are defined for the jobs' completion times. In their paper, they assume a sufficient number of capacitated vehicles are available.

**New contribution.** Compared with existing models, our problem is more practical and thus more complicated. Since the problem is strongly NP-hard, our focus in this paper is to develop approximation algorithms. Suppose a set of jobs are $k$-disjoint, that is, the jobs can be partitioned into $k$ lists of jobs such that the jobs in each list have disjoint production windows. When $k$ is constant and there is a single delivery time, we develop the first PTAS. For multiple delivery times,




| | $p_i$ | $[l_i, r_i]$ | $d_i$ | $c_i$ | $f_i$ |
|---|---|---|---|---|---|
| $J_1$ | 3 | [0,4] | 7 | 3 | 4 |
| $J_2$ | 4 | [1,5] | 7 | 2 | 3 |
| $J_3$ | 5 | [5,14] | 14 | 2 | 4 |
| $J_4$ | 6 | [6,13] | 14 | 3 | 5 |
| $J_5$ | 4 | [3,8] | 7 | 2 | 5 |

| | $D^j$ | $C^j$ |
|---|---|---|
| 1st | 7 | 6 |
| 2nd | 14 | 3 |

**Fig. 1.** An example illustrating the feasible coordinated schedule.

when the number of delivery time is a constant as well, we also develop a PTAS which is extended from the PTAS for single delivery time.

The difficulty of our problems comes from the possible conflicts among the objective of maximizing total profit, the constraint of production window and the constraint of delivery capacity when selecting jobs. Our algorithm is designed to first satisfy the production window constraint, which may result in losing a small fraction of profit and violating the capacity constraint. To overcome this, we use the idea of treating large jobs and small jobs separately, where "large" and "small" are based on size. More importantly, we introduce the method of defining large jobs hierarchically and dynamically. This is the key to make sure that, when we need to remove some jobs with sufficient size and small profit at the same time in order to make the schedule feasible at the end, we can find such job (jobs). We expect this technique to be applied to some other similar problems.

The rest of the paper is organized as follows. In Section 2, we formally define our problems. In Section 3, we present an approximation scheme for single delivery time. In Section 4, we present an approximation scheme for multiple delivery times. In Section 5, we draw some conclusions.

## 2. Problem formulation

Our problem can be formally defined as follows. We have a planning horizon consisting of $z$ delivery times, $T = \{D^1, D^2, \ldots, D^z\}$. Each delivery time $D^j$ is associated with a delivery capacity $C^j$. We have a set of jobs $J = \{J_1, J_2, \ldots, J_n\}$. Each job $J_i$ has a promised delivery time $d_i \in T$, a processing time $p_i$, a production window $[l_i, r_i]$, a size $c_i$, and a profit $f_i$ which can be earned only if $J_i$ is processed at or before $r_i$, and delivered before or at $d_i$. Without loss of generality, we assume that $p_i \leq r_i - l_i$ and $r_i \leq d_i$ for all jobs $J_i$, $1 \leq i \leq n$. The problem is to select a subset of jobs from $J = \{J_1, J_2, \ldots, J_n\}$, and generate a feasible coordinated schedule $S$ of these jobs so as to maximize the total profit. A feasible coordinated schedule $S$ consists of a feasible production schedule and a feasible delivery schedule. A production schedule is feasible if all the jobs are processed within their production windows; and a delivery schedule is feasible if all jobs are delivered before the promised delivery time and the delivery capacities at all times are satisfied. Fig. 1 shows an example of a feasible coordinated schedule. There are two fixed delivery times $D^1 = 7$, $D^2 = 14$ with delivery capacity $C^1 = 6$ and $C^2 = 3$, respectively and there are five jobs. In the feasible coordinated schedule, only three jobs are selected and scheduled in their production windows and delivered by their promised delivery time, and the total size of jobs delivered at each delivery time $D^j$ ($j = 1, 2$) is not greater than the delivery capacity $C^j$.

## 3. Single delivery time

In this section, we study the coordinated production and delivery scheduling problem where the jobs have the same promised delivery time $D$ which is associated with a delivery capacity $C$. In this case, all jobs will be delivered at same time, thus no delivery schedule is necessary as long as the delivery capacity constraint is satisfied by the selected jobs and the production schedule of the selected jobs is feasible. Therefore the problem in this case becomes selecting a subset of orders and generate a feasible production schedule subject to the capacity constraint. Given a constant $\epsilon$, we develop an algorithm which generates a feasible production schedule of a subset of jobs subject to the capacity constraint, whose profit is at least $(1 - \epsilon)$ times the optimal. Our algorithm is a PTAS when the set of jobs $J = \{J_1, J_2, \ldots, J_n\}$ is $k$-disjoint and $k$ is a constant.

Our algorithm has four phases:

Phase I: select large jobs;
Phase II: schedule the large jobs selected from Phase I along with some small jobs selected in this phase;
Phase III: from the schedules generated in Phase II, search for the one with maximum total profits;
Phase IV: convert the schedule obtained from phase III to a feasible schedule.

### 3.1. Phase I: select large jobs

During this phase, we select large jobs for production and delivery, but we do not generate the production schedule. Let us define *large jobs* first. Given a constant parameter $0 < \delta < 1$ which depends on $\epsilon$ and will be determined later, we define a job to be a *large job* if its size is at least $\delta$ times the "available" delivery capacity; otherwise, it is a *small job*. From the definition, we can see that a job may be "small" at the beginning and becomes "large" later as the available delivery capacity becomes smaller due to more jobs are selected.

To select the large jobs, we use brute force. Specifically, we enumerate all the possible selections of large jobs subject to the available capacity constraint. We use $A$ to denote the set of all possible selections. The jobs in each selection $A_p \in A$ are selected in $\lceil \frac{1}{\delta} \rceil$ iterations. For each $A_p$, at the beginning of each iteration, the current available capacity $\bar{C}_p$ is calculated and the set of large jobs (and so small jobs) from the remaining jobs is identified, then a subset of large jobs is selected and added to $A_p$. If no large jobs is selected and added to $A_p$ at certain iteration, we mark $A_p$ as "finalized", which means no more large jobs will be selected and added to $A_p$ in later iterations.

**Phase1-Alg1**

Let $A$ be the set of all possible selections of large jobs so far; $A := \{\emptyset\}$.
For $i := 1$ to $\lceil \frac{1}{\delta} \rceil$
  Let $A' := \emptyset$
  For each selection of large jobs $A_p \in A$
    If $A_p$ is marked as "finalized", add $A_p$ directly to $A'$
    Else
    a. let $\bar{C}_p$ be the capacity available for jobs not in $A_p$, i.e. $\bar{C}_p := C - \sum_{J_i \in A_p} c_i$
    b. from the jobs not selected in $A_p$, find the large jobs with respect to $\bar{C}_p$
    c. generate all possible selections of these large jobs, say $X$, subject to the available capacity constraint
    d. for each $X_j \in X$
        generate a new large job selection $A_q := A_p \cup X_j$, and add $A_q$ into $A'$
        if $A_q = A_p$, mark $A_q$ as "finalized".
  $A := A'$
return $A$

**Lemma 1.** *There are at most $O(n^{O(1/\delta^2)})$ possible ways to select the large jobs in Phase1-Alg1 , where $0 < \delta < 1$ is a constant.*

**Proof.** By the definition of large jobs, it is easy to see that at each iteration $i$, there are at most $\frac{1}{\delta}$ new large jobs. So there are $O(n^{1/\delta})$ ways to select these large jobs at step (c). Since there are $\lceil \frac{1}{\delta} \rceil$ iterations, there are in total $O(n^{\lceil \frac{1}{\delta} \rceil / \delta}) = O(n^{O(1/\delta^2)})$ possible selections of the large jobs. □

### 3.2. Phase II: select and schedule

From Phase I, we get a set of large job selections $A$ without scheduling the jobs. For a job selection $A_p \in A$, although the delivery capacity constraint is satisfied by jobs in $A_p$, the production window constraint may not be satisfied, and thus, no feasible production schedule exists for jobs in $A_p$. In this case, we say $A_p$ is an infeasible job selection.

Our goal in this phase is to identify all feasible large job selections in $A$, and for each feasible selection $A_p$, to find a feasible production schedule $S$ that contains the large jobs in $A_p$, and some newly selected "small" jobs, where the small jobs are identified at the beginning of the last iteration of Phase1-Alg1, and each has a size less than $\delta \bar{C}_p$. Furthermore, the profit of this schedule, denoted by $\text{Profit}(S) = \sum_{J_i \in S} f_i$, is close to the maximum possible among all feasible schedules whose large job selection is exactly $A_p$. On the other hand, the generated schedule $S$ in this phase may violate the capacity constraint. Specifically, let $\widehat{C}_p = C - \sum_{J_i \in A_p} c_i$, be the available capacity at the end of Phase1-Alg1, which is the *available capacity for small jobs*. Let $\text{Load}(S/A_p) = \sum_{J_i \in S \setminus A_p} c_i$ be the total size of the small jobs in $S$, it is possible that $\widehat{C}_p < \text{Load}(S/A_p) \leq (1+\delta)\widehat{C}_p$. In this case, we say $S$ is a valid schedule. Note that $\widehat{C}_p$ is different from $\bar{C}_p$ which is the available capacity at the beginning of last iteration of Phase1-Alg1. Since some large jobs may be selected at the last iteration, we may have $\bar{C}_p > \widehat{C}_p$. Otherwise, $A_p$ is marked "finalized" at the end of algorithm, and then we have $\bar{C}_p = \widehat{C}_p$.

Even though we know that the large job selection in a schedule $S$ is exactly $A_p$, it is still unknown how to schedule the jobs in $A_p$ due to the production window constraint of the jobs and the unknown small jobs in $S$. We solve this problem by dynamic programming. We add jobs to the schedule one by one in certain order. For this, we assume the set of jobs $J = \{J_1, J_2, \ldots, J_n\}$ has been divided into $k$ job lists $L_1, L_2, \ldots, L_k$ such that production windows of jobs in the same list are disjoint. Let $n_u$ be the number of jobs in job list $L_u$ ($1 \le u \le k$). We relabel the jobs in each job list $L_u$ in increasing order of their production windows' starting time, and we use $J_{u,v}$, $1 \le v \le n_u$, to denote the $v$-th job in the job list $L_u$. It is easy to see that in any feasible schedule, one can always assume the jobs in the same list are scheduled in the order they appear in the list. So in our dynamic programming, the jobs in each list are considered in this order. In case the list $L_1, L_2, \ldots, L_k$ are not given, note that dividing the jobs in $J$ into minimum number of disjoint list of job $L_1, L_2, \ldots, L_k$, is the same problem as activity-selection problem, which can be solved greedily (See the Appendix).

For a given large job selection generated from Phase I, $A_p \in A$, if we can find all schedules whose large job selection is exactly $A_p$, we can easily find the best schedule. However, that will be both time and space consuming. To reduce space and time, we find a subset of schedules to approximate all possible schedules so that no two schedules in the set are "similar". Let us formally define "similar" schedules. Given a schedule $S$ with a large job selection $A_p$, let Profit$(S/A_p)$ be the total profit of small jobs in $S$. For two schedules $S_1$ and $S_2$ that both have the same set of large jobs $A_p$, we say they are *similar*, if Profit$(S_1/A_p)$ and Profit$(S_2/A_p)$ are both in the interval $[\omega^x, \omega^{x+1})$ where $\omega = 1 + \frac{\delta}{2n}$, and Load$(S_1/A_p)$ and Load$(S_2/A_p)$ are both in $[\omega^y, \omega^{y+1})$ for some integers $x$ and $y$.

In the following, we use $T(A_p, n'_1, \ldots, n'_k)$ to denote the set of valid schedules such that (a) no two schedules in the set are similar; (b) only the first $n'_u$ jobs from list $L_u$ ($1 \le u \le k$) are allowed to be scheduled; (c) and among the first $n'_u$ jobs in list $L_u$, all the jobs in $A_p$ must be scheduled. For each schedule $S$, we use $C_{\max}(S)$ to represent the last job's completion time. In $T(A_p, n'_1, \ldots, n'_k)$, from each group of schedules that are similar to each other, we only keep the schedule with the smallest $C_{\max}(S)$ in the group.

**Phase2-Alg1($J, A_p, \bar{C}_p$)**

- Input: a set of jobs $J$, which has been divided into $k$ disjoint job lists $L_1, L_2, \ldots, L_k$;
  $A_p$: a large job selection obtained from Phase1-Alg1;
  $\bar{C}_p$: the available capacity at the beginning of last iteration in Phase1-Alg1 for obtaining $A_p$.
- Let $\widehat{C}_p := C - \sum_{J_i \in A_p} c_i$, which is the available delivery capacity for small jobs
- Initialize $T(A_p, 0, \ldots, 0) := \{\emptyset\}$.
- Construct $T(A_p, n_1, \ldots, n_k)$ using dynamic programming
  To find the set $T(A_p, n'_1, \ldots, n'_k)$, do the following steps
  1. For $t := 1$ to $k$
     (a) Consider job $J_{t,n'_t}$, let $[l_{t,n'_t}, r_{t,n'_t}]$ be its production window, $p_{t,n'_t}$ be its processing time, $c_{t,n'_t}$ be its size
     (b) If $J_{t,n'_t} \in A_p$
         For each schedule $S$ in $T(A_p, n'_1, \ldots, n'_t - 1, \ldots, n'_k)$
             if $\max(C_{\max}(S), l_{t,n'_t}) + p_{t,n'_t} \le r_{t,n'_t}$
                 get a schedule $S'$ by adding $J_{t,n'_t}$ to $S$ and schedule it at $\max(C_{\max}(S), l_{t,n'_t})$;
                 add $S'$ to $T(A_p, n'_1, \ldots, n'_k)$;
     (c) Else
         For each schedule $S$ in $T(A_p, n'_1, \ldots, n'_t - 1, \ldots, n'_k)$
             add $S$ into $T(A_p, n'_1, \ldots, n'_k)$;
             if $c_{t,n'_t} < \delta\bar{C}_p$(i.e. a small job) and $\max(C_{\max}(S), l_{t,n'_t}) + p_{t,n'_t} \le r_{t,n'_t}$
             and Load$(S \cup \{J_{t,n'_t}\}/A_p) \le (1+\delta)\widehat{C}_p$
                 get a schedule $S'$ by adding $J_{t,n'_t}$ to $S$ at $\max(C_{\max}(S), l_{t,n'_t})$;
                 add $S'$ into $T(A_p, n'_1, \ldots, n'_k)$;
  2. From each group of schedules in $T(A_p, n'_1, \ldots, n'_k)$ that are similar to each other, delete all but the schedule $S$ with minimum $C_{\max}(S)$ in the group
- return the schedule $S$ from $T(A_p, n_1, \ldots, n_k)$ with maximum profit

It is easy to see that the production schedule generated at steps (b) and (c) must be valid. For any feasible large job selection $A_p$, we have the following lemma.

**Lemma 2.** *Suppose $A_p \in A$ is a feasible large job selection obtained from Phase I, and let $S'$ be the feasible schedule that has the maximum profit among all schedules whose large job selection is exactly $A_p$. Then Phase2-Alg1($J, A_p, \bar{C}_p$) returns a schedule $S$ such that: the large job selection in $S$ is $A_p$; $S$ is valid, i.e. Load$(S/A_p) \le (1+\delta)(C - \sum_{J_i \in A_p} c_i)$; and* Profit$(S) \ge (1-\delta)$Profit$(S')$.

**Proof.** Let $S'(n'_1, \ldots, n'_k)$ denote the partial schedule of $S'$ that contains only jobs $J_{u,v}$, for $1 \le u \le k$ and $1 \le v \le n'_u$. Let $n' = n'_1 + \cdots + n'_k$. In the following, we first prove that for any $S'(n'_1, \ldots, n'_k)$, there is a schedule $S'' \in T(A_p, n'_1, \ldots, n'_t, \ldots, n'_k)$ such that

(a) $S''$ has scheduled all jobs $J_{u,v} \in A_p$ from job list $L_u$ with $1 \le u \le k$ and $v \le n'_u$.
(b) $C_{\max}(S'') \le C_{\max}(S'(n'_1, \ldots, n'_k))$,

(c) $\text{Profit}(S''/A_p) \geq \frac{\text{Profit}(S'(n'_1,\ldots,n'_k)/A_p)}{\omega^{n'}}$, and

(d) $\text{Load}(S''/A_p) \leq \omega^{n'} \cdot \text{Load}(S'(n'_1,\ldots,n'_k)/A_p)$.

We prove by induction on $n'$. The hypotheses are obviously true when $n' = 0$. Assume this is true for $S'(x_1,\ldots,x_k)$ where $x_1 \leq n'_1,\ldots,x_k \leq n'_k$, and $x_1+\cdots+x_k < n'$. Now let us consider $S'(n'_1,\ldots,n'_k)$. If some job $J_{u,n'_u}$ is not selected in $S'(n'_1,\ldots,n'_k)$, then $S'(n'_1,\ldots,n'_k)$ is the same as $S'(n'_1,\ldots,n'_u-1,,\ldots,n'_k)$, the hypotheses are true by induction. So we assume that $J_{u,n'_u}$, for all $1 \leq u \leq k$, is contained in the partial schedule of $S'$. Assume that job $J_{t,n'_t}$ has the completion time $C_{\max}(S'(n'_1,\ldots,n'_k))$. Removing $J_{t,n'_t}$ from $S'(n'_1,\ldots,n'_k)$, we get the schedule $S'(n'_1,\ldots,n'_t-1,\ldots,n'_k)$.

By induction hypothesis, there exists a schedule $S''_1 \in T(A_p, n'_1,\ldots,n'_t-1,\ldots,n'_k)$ such that

(a) $S''_1$ has scheduled all jobs $J_{t,v} \in A_p$ from job list $L_t$ with $v \leq n'_t-1$, and jobs $J_{u,v}$ from list $L_u$ with $v \leq n'_u$ and $1 \leq u \leq k$ and $u \neq t$

(b) $C_{\max}(S''_1) \leq C_{\max}(S'(n'_1,\ldots,n'_t-1,\ldots,n'_k))$

(c) $\text{Profit}(S''_1/A_p) \geq \frac{\text{Profit}(S'(n'_1,\ldots,n'_t-1,\ldots,n'_k)/A_p)}{\omega^{n'-1}}$, and

(d) $\text{Load}(S''_1/A_p) \leq \omega^{n'-1}\text{Load}(S'(n'_1,\ldots,n'_t-1,\ldots,n'_k)/A_p)$.

Adding $J_{t,n'_t}$ to $S''_1$, we obtain a schedule $\widehat{S}$. Furthermore, $\widehat{S}$ must have a valid production schedule, since

$$\begin{aligned} C_{\max}(\widehat{S}) &= \max(C_{\max}(S''_1), l_{t,n'_t}) + p_{t,n'_t} \\ &\leq \max(C_{\max}(S'(n'_1,\ldots,n'_t-1,\ldots,n'_k)), l_{t,n'_t}) + p_{t,n'_t} \\ &= C_{\max}(S'(n'_1,\ldots,n'_t,\ldots,n'_k)). \end{aligned}$$

We have two cases: (1) $J_{t,n'_t} \in A_p$; (2) $J_{t,n'_t} \notin A_p$. For case (1), the small jobs in $\widehat{S}$ and $S''_1$ are same, i.e. $S''_1 \setminus A_p = \widehat{S} \setminus A_p$; similarly, $S'(n'_1,\ldots,n'_t-1,\ldots,n'_k) \setminus A_p = S'(n'_1,\ldots,n'_t,\ldots,n'_k) \setminus A_p$. Since $\omega > 1$, we have

$$\begin{aligned} \text{Profit}(\widehat{S}/A_p) = \text{Profit}(S''_1/A_p) &\geq \frac{\text{Profit}(S'(n'_1,\ldots,n'_t-1,\ldots,n'_k)/A_p)}{\omega^{n'-1}} \\ &= \frac{\text{Profit}(S'(n'_1,\ldots,n'_t,\ldots,n'_k)/A_p)}{\omega^{n'-1}} \end{aligned}$$

and

$$\begin{aligned} \text{Load}(\widehat{S}/A_p) = \text{Load}(S''_1/A_p) &\leq \text{Load}(S'(n'_1,\ldots,n'_t-1,\ldots,n'_k)/A_p)\omega^{n'-1} \\ &= \text{Load}(S'(n'_1,\ldots,n'_t,\ldots,n'_k)/A_p)\omega^{n'-1}. \end{aligned}$$

For case (2), we have $\widehat{S} \setminus A_p = S''_1 \setminus A_p \cup \{J_{t,n'_t}\}$ and $S'(n'_1,\ldots,n'_t,\ldots,n'_k) \setminus A_p = S'(n'_1,\ldots,n'_t-1,\ldots,n'_k) \setminus A_p \cup \{J_{t,n'_t}\}$

$$\begin{aligned} \text{Profit}(\widehat{S}/A_p) &= \text{Profit}(S''_1/A_p) + f_{t,n'_t} \geq \frac{\text{Profit}(S'(n'_1,\ldots,n'_t-1,\ldots,n'_k)/A_p)}{\omega^{n'-1}} + f_{t,n'_t} \\ &\geq \frac{\text{Profit}(S'(n'_1,\ldots,n'_t-1,\ldots,n'_k)/A_p) + f_{t,n'_t}}{\omega^{n'-1}} \\ &= \frac{\text{Profit}(S'(n'_1,\ldots,n'_t,\ldots,n'_k)/A_p)}{\omega^{n'-1}} \end{aligned}$$

and

$$\begin{aligned} \text{Load}(\widehat{S}/A_p) &= \text{Load}(S''_1/A_p) + c_{t,n'_t} \leq \text{Load}(S'(n'_1,\ldots,n'_t-1,\ldots,n'_k)/A_p)\omega^{n'-1} + c_{t,n'_t} \\ &\leq (\text{Load}(S'(n'_1,\ldots,n'_t-1,\ldots,n'_k)/A_p) + c_{t,n'_t})\omega^{n'-1} \\ &= \text{Load}(S'(n'_1,\ldots,n'_t,\ldots,n'_k)/A_p)\omega^{n'-1}. \end{aligned}$$

Let $S''$ be the schedule in the same group with $\widehat{S}$ that is selected in $T(A_p, n'_1, n'_2,\ldots,n'_k)$. If $J_{t,n'_t} \in A_p$, by our algorithm, $J_{t,n'_t}$ must be selected in all schedules $T(A_p, n'_1,\ldots,n'_t,\ldots,n'_k)$ including $S''$. Thus (a) is true. We also have that $C_{\max}(S'') \leq C_{\max}(\widehat{S}) \leq C_{\max}(S'(n'_1,\ldots,n'_t,\ldots,n'_k))$. Since $S''$ and $\widehat{S}$ are in the same group, we have that $\text{Profit}(S''/A_p) \geq \frac{\text{Profit}(\widehat{S}/A_p)}{\omega} \geq \frac{\text{Profit}(S'(n'_1,\ldots,n'_t,\ldots,n'_k)/A_p)}{\omega^{n'}}$ and $\text{Load}(S''/A_p) \leq \omega\text{Load}(\widehat{S}/A_p) \leq \omega^{n'}\text{Load}(S'(n'_1,\ldots,n'_t,\ldots,n'_k)/A_p)$. This completes the induction.

Since $\frac{1}{\omega^{n'}} \geq \frac{1}{\omega^n} \geq \frac{1}{1+\delta} \geq 1-\delta$, we have $\text{Profit}(S/A_p) \geq (1-\delta)\text{Profit}(S'/A_p)$, which implies $\text{Profit}(S) \geq (1-\delta)\text{Profit}(S')$; Similarly, since $\omega^{n'} \leq \omega^n = (1+\frac{\delta}{2n})^n \leq e^{\delta/2} \leq 1+\delta$,

$$\text{Load}(S/A_p) \leq (1+\delta)\text{Load}(S'/A_p) \leq (1+\delta)\left(C - \sum_{J_i \in A_p} c_i\right). \quad \square$$

**Lemma 3.** *For a given $A_p$, the running time of Phase2-Alg1($J$, $A_p$, $\bar{C}_p$) is $O(kn^k(\log_\omega \sum_{i=1}^n f_i)\log_\omega C)$.*

**Proof.** It is easy to see that the running time is dominated by the dynamic programming. $T(A_p, n_1, \ldots, n_k)$ is computed starting from $T(A_p, 0, \ldots, 0)$, which takes $n_1 \cdot n_2 \cdot \cdots \cdot n_k = O(n^k)$ iterations.

For any $T(A_p, n'_1, \ldots, n'_k)$, $n'_t \leq n_t$, $1 \leq t \leq k$, since we only keep one schedule from each group of similar schedules, the size of $T(A_p, n'_1, \ldots, n'_k)$ is at most $m_1 m_2$, where $m_1 = O(\log_\omega(\sum_{i=1}^n f_i))$, and $m_2 = O(\log_\omega C)$. To construct $T(A_p, n'_1, \ldots, n'_k)$, all schedules in $T(A_p, n'_1 - 1, n'_2, \ldots, n'_k), \ldots, T(A_p, n'_1, \ldots, n'_{k-1}, n'_k - 1)$ are considered. So it considers in total $O(km_1m_2)$ schedules to construct each $T(A_p, n'_1, \ldots, n'_k)$, and thus the running time of dynamic procedure for each large jobs selection $A_p$ is $O(n^k \cdot km_1m_2) = O(kn^k(\log_\omega \sum_{i=1}^n f_i)\log_\omega C)$. $\square$

### 3.3. Phase III: search for the best schedule

For each feasible large job selection $A_p \in A$ obtained from Phase I, the dynamic procedure of Phase II outputs a valid schedule whose large job selection is exactly $A_p$. In this phase, we find a good schedule to approximate the optimal schedule. This is done by selecting the schedule $S$ with the maximum total profit among all the schedules generated in Phase II for all feasible $A_p$-s.

**Lemma 4.** *Let $S^*$ be the optimal schedule and $S$ be the schedule with the maximum profit among all schedules generated from Phase II. Then $S$ must be valid and $\text{Profit}(S) \geq (1-\delta)\text{Profit}(S^*)$. Furthermore, the total running time to obtain $S$ is $O(\frac{k}{\delta^2} n^{O(k+1/\delta^2)}(\lg \sum_{i=1}^n f_i)(\lg C))$.*

**Proof.** Suppose that the set of large jobs selected by $S^*$ is $A_p$ which means $A_p$ is feasible. Since we enumerated all possible large job selections, we must have $A_p \in A$. By Lemma 2, Phase2-Alg1($A_p$, $J$, $\bar{C}_p$) must return a schedule $\bar{S}$ such that $\bar{S}$ is valid, i.e. $\text{Load}(\bar{S}/A_p) \leq (1+\delta)\text{Load}(S^*/A_p)$, $C_{\max}(\bar{S}) \leq C_{\max}(S^*)$, and $\text{Profit}(\bar{S}) \geq (1-\delta)\text{Profit}(S^*)$.

Since $S$ is the schedule with the maximum profit among all schedules generated from Phase II, $S$ must be a valid schedule and $\text{Profit}(S) \geq \text{Profit}(\bar{S}) \geq (1-\delta)\text{Profit}(S^*)$.

By Lemma 1, there are total $O(n^{O(\frac{1}{\delta^2})})$ possible selections of the large jobs. For each $A_p$, by Lemma 3, the running time of Phase2-Alg1($J$, $A_p$, $\bar{C}_p$) is $O(kn^k(\log_\omega \sum_{i=1}^n f_i)\log_\omega C)$. Note that $\omega = 1 + \frac{\delta}{2n}$ and $\ln \omega \approx \frac{\delta}{2n}$. So the total running time from Phase I to Phase III is

$$O\left(n^{O\left(\frac{1}{\delta^2}\right)} kn^k \left(\log_\omega \sum_{i=1}^n f_i\right) \log_\omega C\right) = O\left(kn^{O\left(k+\frac{1}{\delta^2}\right)} \frac{\ln \sum_{i=1}^n f_i}{\delta/2n} \frac{\ln C}{\delta/2n}\right)$$
$$= O\left(\frac{k}{\delta^2} n^{O\left(k+\frac{1}{\delta^2}\right)} \left(\lg \sum_{i=1}^n f_i\right)(\lg C)\right),$$

or $O(kn^{O(k)}(\lg \sum_{i=1}^n f_i)(\lg C))$ since $\delta$ is a constant. $\square$

### 3.4. Phase IV: convert to a feasible schedule

From Phase III, we get the schedule $S$ with the maximum total profit among all schedules generated from Phase II which is valid but may not be feasible, i.e. $\widehat{C}_p \leq \text{Load}(S/A_p) \leq (1+\delta)\widehat{C}_p$. To convert $S$ into a feasible schedule, we have to delete some jobs carefully so that the total profit will not be affected greatly. The following algorithm describes how to obtain a feasible schedule from $S$.

**Phase4-Alg1($S$)**

Input: $S$ is the best schedule returned by Phase III, which is valid but may not be feasible
Let $A_p$ be its corresponding large job selection in $S$.
Let $S' := S$
If $\text{Load}(S') > C$ (i.e. $\text{Load}(S'/A_p) > \widehat{C}_p$ )

If $A_p$ is marked as "finalized"
Delete a set of small jobs of total size of at least $\delta\widehat{C}_p$ and at most $2\delta\widehat{C}_p$ and with the least possible profit from $S'$
Else
Delete the large job in $A_p$ with least profit from $S'$
Return $S'$

**Lemma 5.** *Let $\delta$ be a constant of at most $\frac{\epsilon}{3}$. The schedule $S'$ returned by Phase4-alg1 is feasible, and* $\text{Profit}(S') \geq (1-\epsilon)\text{Profit}(S^*)$, *where $S^*$ is an optimal schedule.*

**Proof.** Assume $S$ is the schedule with the maximum total profit among all schedules in $T(A_p, n_1, n_2, \ldots, n_k)$ for all $A_p \in A$, and $S'$ is obtained from $S$ by Phase4-Alg1. By Lemma 4, we have $\text{Profit}(S) \geq (1-\delta)\text{Profit}(S^*)$ and $\text{Load}(S/A_p) \leq (1+\delta)\widehat{C}_p$.

Phase4-Alg1 considered two cases depending on whether $A_p$ is marked as "finalized" or not. In the case of $A_p$ is marked as "finalized" at the end of Phase I, we know starting from some iteration $i$ ($1 \leq i < \lceil \frac{1}{\delta} \rceil$), no more large job was selected and $\bar{C}_p = \widehat{C}_p$. Since $\text{Load}(S'/A_p) > \widehat{C}_p$, it means small jobs with total size of at least $\widehat{C}_p$ have been scheduled and delivered. Since each small job has size at most $\delta\bar{C}_p = \delta\widehat{C}_p$, we can partition these small jobs into at least $\lceil \frac{1}{2\delta} \rceil$ groups such that the total size of jobs in each group is in the range of $[\delta\widehat{C}_p, 2\delta\widehat{C}_p]$. We delete the group of jobs with smallest total profit, which is at most $\frac{1}{\lceil \frac{1}{2\delta} \rceil}$ times the total profit of small jobs in $S$, $\frac{\text{Profit}(S/A_p)}{\lceil \frac{1}{2\delta} \rceil} \leq \frac{\text{Profit}(S)}{\lceil \frac{1}{2\delta} \rceil}$.

In case $A_p$ is not marked as "finalized" at the end of Phase1-Alg1, we may not find a set of small jobs as in the first case. However, we know that at least one large job is selected from each iteration $1 \leq i \leq \lceil \frac{1}{\delta} \rceil$, so there are at least $\lceil \frac{1}{\delta} \rceil$ large jobs in $A_p$. The deleted large job has the smallest profit, which is at most $\frac{\text{Profit}(A_p)}{\lceil \frac{1}{\delta} \rceil} \leq \frac{\text{Profit}(S)}{\lceil \frac{1}{\delta} \rceil}$. Since it is large, its size is at least $\delta\bar{C}_p > \delta\widehat{C}_p$.

So in both cases the job(s) deleted has(have) size at least $\delta\widehat{C}_p$, thus the obtained schedule must be feasible; and the total profit of the deleted job(s) in both case is at most $\frac{\text{Profit}(S)}{\lceil \frac{1}{2\delta} \rceil}$, we have

$$\begin{aligned}
\text{Profit}(S') &\geq \text{Profit}(S) - \frac{\text{Profit}(S)}{\lceil \frac{1}{2\delta} \rceil} \\
&\geq \left(1 - \frac{1}{\lceil \frac{1}{2\delta} \rceil}\right) \text{Profit}(S) \\
&\geq \left(1 - \frac{1}{\lceil \frac{1}{2\delta} \rceil}\right) (1-\delta)\text{Profit}(S^*) \\
&\geq (1-2\delta)(1-\delta)\text{Profit}(S^*) \\
&\geq (1-3\delta)\text{Profit}(S^*) \\
&\geq (1-\epsilon)\text{Profit}(S^*). \quad \square
\end{aligned}$$

For any constant $\epsilon$, by Lemmas 5 and 4,we have the following theorem.

**Theorem 6.** *For any coordinated production and delivery scheduling problem with production window and delivery capacity constraints, if there is only one delivery time, and the job set is k-disjoint, where k is a constant, then there exists a PTAS that runs in time $O(n^{O(k+1/\epsilon^2)}(\lg \sum_{i=1}^{n} f_i)(\lg C))$.*

## 4. Multiple delivery times

In this section, we study the coordinated production and delivery scheduling problem with multiple delivery times $D^1, D^2, \ldots, D^z$ which have delivery capacity of $C^1, C^2, \ldots, C^z$, respectively. Our goal is to find a feasible coordinated production and delivery schedule whose total profit is close to optimal. As in the case of the single delivery time, a feasible production schedule is one that satisfies the production window constraint. A feasible delivery schedule, however, is more restricted than the single delivery time: for each selected job, we have to specify its delivery time which cannot be later than its promised delivery date; and the delivery capacity constraint has to be satisfied for all delivery times. Since the delivery schedule can be seen as an assignment of completed jobs to $z$ delivery times subject to the capacity constraint, in the following, we will use delivery time assignment and delivery schedule interchangeably.

Assume the job set $J = \{J_1, J_2, \ldots, J_n\}$ is $k$-disjoint, when $k$ and the number of delivery times $z$ are both constants, we develop a PTAS which is adapted from the PTAS for the case of single delivery time.

Phase I: select large jobs and assign them to delivery times;
Phase II: schedule the large jobs selected from phase I, along with some small jobs selected in this phase, and assign delivery times for the selected small jobs;

Phase III: search for the best schedule to approximate the optimal schedule;
Phase IV: remove some of the jobs to make the schedule feasible.

As one can see from above, the structure of the two PTASes are similar, but the details are different. In particular, in Phase I and Phase II, we have to consider the delivery schedule of the jobs; and in Phase IV, we have to make sure the capacity is satisfied for all delivery times. In the following, we will describe all phases, but we mainly concentrate on the differences of the two PTASes. We will follow the notations and definitions unless otherwise specified.

### 4.1. Phase I: select and assign large jobs

During this phase, we select the large jobs and assign them to delivery times without generating the production schedule.

First we need to redefine the *large jobs*. Given a constant parameter $0 < \delta < 1$, we define a job to be a *large job with respect to* $D^j$ if its size is at least $\delta$ times the available capacity of $D^j$; otherwise, it is a *small job with respect to* $D^j$. Note that a job may be large with respect to one delivery time with small delivery capacity while being small with respect to another with large delivery capacity. Even for one delivery time $D^j$, a job may be a small job at the beginning while be a large job later as the available delivery capacity at $D^j$ becomes smaller because more jobs are assigned to $D^j$.

As in single delivery case, we use brute force in this phase. But now, besides selecting large jobs to be processed and delivered, we also need to enumerate all possibilities of delivery time assignment for the selected jobs. Since each combination of selection and assignment can be simply represented by the set of jobs assigned to delivery time $D^j$ for all $1 \le j \le z$, we use $A$ to denote the set of all possible combinations. For a given combination $A_p \in A$, $A_p = \bigcup_{j=1}^{z} A_p^j$, where $A_p^j$ is the set of large jobs assigned to delivery time $D^j$. As before, the selection and assignment of jobs in each $A_p$ is still done in $\lceil \frac{1}{\delta} \rceil$ iterations. For a given $A_p$, if no large jobs are assigned to $D^j$ at certain iteration, i.e. $A_p^j = A_q^j$, where $A_q$ is a new combination derived from $A_p$, we mark $A_q^j$ as "finalized", which means no more large jobs will be assigned to $D^j$ in any combination produced from $A_q$ in later iterations. For each $A_p \in A$, let $\bar{C}_p = \{\bar{C}_p^1, \ldots, \bar{C}_p^z\}$ be the available capacity at the beginning of each iteration at $D^1, \ldots, D^z$ respectively. The $\bar{C}_p$ at the last iteration will be used in Phase II when small jobs are added.

**Phase1-Alg2**

For each delivery time $D^j$ $(1 \le j \le z)$, initialize the available capacity to be $C^j$
Let $A$ be the set of all combinations of selection and assignment so far, $A := \{\emptyset\}$.
For $i := 1$ to $\lceil \frac{1}{\delta} \rceil$
  Let $A' := \emptyset$
  For each $A_p \in A$, where $A_p := \bigcup_{j=1}^{z} A_p^j$

  (a) for each $1 \le j \le z$, let $\bar{C}_p^j$ be the available capacity at $D^j$
  (b) for each $j$ such that $A_p^j$ is not finalized, identify all jobs not in $A_p$ that are large with respect to $D^j$
  (c) generate all possible combinations of selection and assignment of the large jobs found above, $X$, subject to the available capacity constraint and the promised due date constraint
  (d) for any $X_q \in X$, $X_q := \bigcup_{j=1}^{z} X_q^j$
    (i) generate a new combination $A_q$ such that $A_q^j := A_p^j \cup X_q^j$ for $1 \le j \le z$
    (ii) for each $j$, $1 \le j \le z$, if $A_q^j = A_p^j$, mark $A_q^j$ as "finalized"
  (e) add all newly generated combinations into $A'$

  $A := A'$
return $A$

**Lemma 7.** *There are at most* $O(n^{\frac{z}{\delta^2}})$ *possible assignments of the large jobs in Phase1-Alg2, where* $0 < \delta < 1$ *is a constant.*

**Proof.** At each iteration $i$, at most $\frac{1}{\delta}$ large jobs with respect to each $D^j$ can be selected, so there are at most $n^{\frac{1}{\delta}}$ different ways to select large jobs for $D^j$, $1 \le j \le z$. Hence, the number of possible selections and assignments in one iteration is $O(n^{\frac{z}{\delta}})$. Since there are $\lceil \frac{1}{\delta} \rceil$ iterations, there are in total $O((n^{\frac{z}{\delta}})^{\lceil \frac{1}{\delta} \rceil}) = O(n^{O(\frac{z}{\delta^2})})$ possible selection and assignments of large jobs. □

### 4.2. Phase II: select, schedule and assign

As in the case of single delivery, no production schedule is generated in Phase I. In this phase, we will detect all feasible large job selection and assignment from $A$, and for each feasible $A_p \in A$, we will use dynamic programming to generate the feasible production schedules of large jobs along with newly selected small jobs, we will also assign the small jobs so that the delivery schedule is valid.

We have to extend some definitions from single delivery case. For a given large job selection and assignment $A_p$, let $\widehat{C}_p^j = C^j - \sum_{J_i \in A_p^j} c_i$ be the *available capacity for small jobs* that can be assigned to $D_j$. Assume $A_p$ is the corresponding

large job selection of a coordinated schedule $S$, we use $\text{Load}^j(S)$ to denote the total size of the jobs assigned to $D^j$ in $S$. We define $\text{Load}^j(S/A_p)$ to be the total size of small jobs assigned to $D^j$ in $S$. We say that the delivery schedule in $S$ is feasible if $\text{Load}^j(S/A_p) \le \widehat{C}_p^j$ for all delivery time $D^j$ $(1 \le j \le z)$ and the delivery schedule in $S$ is valid if $\text{Load}^j(S/A_p) \le (1+\delta)\widehat{C}_p^j$ for all $D^j$ $(1 \le j \le z)$.

As before, we assume the jobs are divided into $k$ disjoint lists $L_1, L_2, \ldots, L_k$. Let $n_u$ denote the number of jobs in job list $L_u$ $(1 \le u \le k)$. The jobs in each list are considered in the order of starting time of their production windows. For any $A_p \in A$, we use $T(A_p, n'_1, \ldots, n'_k)$ to represent a set of coordinated schedules such that in each schedule the production schedule is feasible, the delivery schedule is valid, only the first $n'_u$ jobs from list $L_u$ $(1 \le u \le k)$ can be scheduled, and among the first $n'_u$ jobs in list $L_u$, $1 \le u \le k$, all those jobs in $A_p$ must be scheduled and must be delivered as specified by $A_p$. Suppose two coordinated schedules $S_1$ and $S_2$ have the same large job selection and assignment $A_p$, for a given constant $\omega = 1+\frac{\delta}{2n}$, we say $S_1$ and $S_2$ are similar if $\text{Profit}(S_1/A_p)$ and $\text{Profit}(S_2/A_p)$ are both in the interval $[\omega^x, \omega^{x+1})$ for some integer $0 \le x \le \log_\omega \sum f_i$, and for all $1 \le j \le z$, $\text{Load}^j(S_1/A_p)$ and $\text{Load}^j(S_2/A_p)$ are both in $[\omega^{y_j}, \omega^{y_j+1})$ for some integer $0 \le y_j \le \log_\omega C^j$.

**Phase2-Alg2($J, A_p, \bar{C}_p, C$)**

- Input: $J := \{J_1, J_2, \ldots, J_n\}$: the job set which has been divided into $k$ job lists $L_1, L_2, \ldots, L_k$;
  $A_p := \bigcup_{j=1}^{z} A_p^j$: a large job selection and assignment obtained from Phase1-Alg2;
  $\bar{C}_p := \{\bar{C}_p^1, \ldots, \bar{C}_p^z\}$: the available capacity at the beginning of last iteration from Phase1-Alg2 for obtaining $A_p$
  $C := \{C^1, C^2, \ldots, C^z\}$: capacity at delivery times $D^1, D^2, \ldots, D^z$
- Initialize $T(A_p, 0, \ldots, 0) := \{\emptyset\}$
- Construct $T(A_p, n_1, \ldots, n_k)$ using dynamic programming
  To find the set $T(A_p, n'_1, \ldots, n'_k)$, do the following steps
  1. For $t := 1$ to $k$: consider job $J_{t,n'_t}$, let $[l_{t,n'_t}, r_{t,n'_t}]$ be its production window, $p_{t,n'_t}$ be its processing time, $c_{t,n'_t}$ be its size.
     If $J_{t,n'_t} \in A_p^j$ for some $j$ (i.e. $J_{t,n'_t}$ is a large job assigned to $D^j$ in $A_p$)
     for each schedule $S$ in $T(A_p, n'_1, \ldots, n'_t - 1, \ldots, n'_k)$ such that $\max(C_{\max}(S), l_{t,n'_t}) + p_{t,n'_t} \le r_{t,n'_t}$
     if $D^j \ge \max(C_{\max}(S), l_{t,n'_t}) + p_{t,n'_t}$
     build a new schedule $S'$ by adding job $J_{t,n'_t}$ to $S$ such that $J_{t,n'_t}$ is scheduled
     at $\max(C_{\max}(S), l_{t,n'_t})$ and assigned to $D^j$;
     add $S'$ to $T(A_p, n'_1, \ldots, n'_k)$;
     Else for each schedule $S$ in $T(A_p, n'_1, \ldots, n'_t - 1, \ldots, n'_k)$
     add $S$ into $T(A_p, n'_1, \ldots, n'_k)$;
     for each delivery time $D^j$ such that $D^j \le d_{t,n'_t}$ and $c_{t,n'_t} \le \delta\bar{C}_p^j$ (i.e. $J_{t,n'_t}$ is a small job with respect to $D^j$)
     if $\max(C_{\max}(S), l_{t,n'_t}) + p_{t,n'_t} \le r_{t,n'_t}$, $\max(C_{\max}(S), l_{t,n'_t}) + p_{t,n'_t} \le D^j$ and
     $\text{Load}^j(S \cup \{J_{t,n'_t}\}/A_p) \le (1+\delta)\widehat{C}_p^j$
     build a new schedule $S'$ by adding $J_{t,n'_t}$ to $S$ such that $J_{t,n'_t}$ is scheduled
     at $\max(C_{\max}(S), l_{t,n'_t})$ and delivered at $D^j$;
     add $S'$ into $T(A_p, n'_1, \ldots, n'_k)$;
  2. From each group of schedules in $T(A_p, n_1, \ldots, n_k)$ that are similar to each other, delete all but the schedule $S$ with minimum $C_{\max}(S)$ in the group
- return the schedule $S$ from $T(A_p, n_1, \ldots, n_k)$ with maximum profit

Similar to single delivery case, if a large job selection and assignment $A_p \in A$ is not feasible, the above dynamic procedure will return an empty schedule. For each feasible large job selection and assignment $A_p$, we have:

**Lemma 8.** *Suppose $A_p \in A$ is a feasible large job selection and assignment obtained from Phase1-Alg2, and let $S'$ be the feasible schedule that has maximum profit among all schedules whose large job selection and assignment is exactly $A_p$. Then Phase2-Alg2($J, A_p, \bar{C}_p, C$) returns a schedule $S$ such that $S$ is valid, i.e. $\text{Load}^j(S/A_p) \le (1+\delta)(C^j - \sum_{J_i \in A_p^j} c_i)$ for all $1 \le j \le z$, and* $\text{Profit}(S) \ge (1-\delta)\text{Profit}(S')$.

**Proof.** Let $S'(n'_1, \ldots, n'_k)$ denote the partial coordinated production and delivery schedule of $S'$ that contains only jobs $J_{u,v}$, $1 \le u \le k$ and $1 \le v \le n'_u$. Let $n' = n'_1 + \cdots + n'_k$. As in Lemma 2, it is sufficient to prove that for any $S'(n'_1, \ldots, n'_k)$, there is a schedule $S'' \in T(A_p, n'_1, \ldots, n'_t, \ldots, n'_k)$ such that

(a) $S''$ has scheduled all jobs $J_{u,v} \in A_p$ from job list $L_u$ with $1 \le u \le k$ and $v \le n'_u$, and if $J_{u,v} \in A_p^j$ in $A_p$, $J_{u,v}$ is assigned to $D_j$ in $S''$
(b) $C_{\max}(S'') \le C_{\max}(S'(n'_1, \ldots, n'_k))$,

(c) $\text{Profit}(S''/A_p) \geq \frac{\text{Profit}(S'(n'_1,\ldots,n'_k)/A_p)}{\omega^{n'}}$, and
(d) $\text{Load}^j(S''/A_p) \leq \omega^{n'} \cdot \text{Load}^j(S'(n'_1,\ldots,n'_k)/A_p)$ .

We can prove by induction and it is almost the same as that of Lemma 2, so we omit it. □

**Lemma 9.** *The running time of Phase2-Alg2(J, $A_p$, $\bar{C}_p$, C) is $O(kn^k(\log_\omega \sum_{i=1}^n f_i)\prod_{j=1}^z(\log_\omega C^j))$ for a given $A_p$.*

**Proof.** As in the single delivery time case, the running time is dominated by the dynamic programming, which takes $n_1 \cdot n_2 \cdot \ldots \cdot n_k = O(n^k)$ iterations.

For any $T(A_p, n'_1, \ldots, n'_k)$, $n'_t \leq n_t$, $1 \leq t \leq k$, the size of $T(A_p, n'_1, \ldots, n'_k)$ is at most

$$O\left(\left(\log_\omega \sum_{i=1}^n f_i\right)\prod_{j=1}^z(\log_\omega C^j)\right).$$

To construct $T(A_p, n'_1, \ldots, n'_k)$, all schedules in $T(A_p, n'_1 - 1, n'_2, \ldots, n'_k), \ldots, T(A_p, n'_1, \ldots, n'_{k-1}, n'_k - 1)$ are considered. So it considers in total $O(k(\log_\omega \sum_{i=1}^n f_i)\prod_{j=1}^z(\log_\omega C^j))$ schedules to construct $T(A_p, n'_1, \ldots, n'_k)$, and thus the running time of dynamic procedure for each large job selection $A_p$ is $O(kn^k(\log_\omega \sum_{i=1}^n f_i)\prod_{j=1}^z(\log_\omega C^j))$. □

### 4.3. Phase III: search for the best schedule

For each large job selection and assignment $A_p \in A$ obtained from Phase I, the dynamic procedure of Phase II returns a coordinated schedule $S$ such that the production schedule of $S$ is feasible and the delivery schedule of $S$ is valid, and all the jobs in $A_p$ have been selected and delivered as specified by $A_p$. In this phase, to find a good approximation, we will pick the schedule with the maximum total profit among all such schedules.

**Lemma 10.** *Let $S^*$ be the optimal schedule and S be the schedule with the maximum profit among all schedules produced from Phase III, we have* $\text{Profit}(S) \geq (1-\delta)\text{Profit}(S^*)$ *and* $\text{Load}^j(S/A_p) \leq (1+\delta)\widehat{C}_p^j$ *for all j, $1 \leq j \leq z$. Furthermore, the total running time to obtain S is*

$$O\left(\tfrac{k}{\delta^2}(n)^{O(\frac{z}{\delta^2}+k)}\left(\lg\sum_{i=1}^n f_i\right)\prod_{j=1}^z(\lg C^j)\right).$$

**Proof.** By Lemma 8, the first part is obviously true. The running time follows directly from Lemmas 7 and 9. □

### 4.4. Phase IV: convert to a feasible schedule

From Phase III, the schedule $S$ with the maximum total profit has been selected from all schedules in $T(A_p, n_1, n_2, \ldots, n_k)$ for all $A_p \in A$. Note that the production schedule of $S$ must be feasible but delivery schedule of $S$ may be valid but not feasible, i.e. for some delivery time $D^j$, $1 \leq j \leq z$, $\widehat{C}_p^j \leq \text{Load}^j(S/A_p) \leq (1+\delta)\widehat{C}_p^j$. To convert $S$ so that it has feasible delivery schedule, we delete some jobs from $S$.

**Phase4-Alg2($S$)**

Input: $S$ is the best schedule found by Phase III, which has a feasible production schedule and a valid delivery schedule.
Let $A_p := \bigcup_{j=1}^z A_p^j$ be its corresponding large job selection.
For each delivery time $D^j$ $(1 \leq j \leq z)$ such that $\text{Load}^j(S) > C^j$
  If $A_p^j$ is marked as "finalized"
    delete from $S$ the jobs that are small with respect to $D^j$ and
    have total size of between $\delta\widehat{C}_p^j$ and $2\delta\widehat{C}_p^j$ and with the least
    possible total profit
  Else
    delete from $S$ the large job that is in $A_p^j$ with least profit
Return $S$

**Lemma 11.** *Given any constant $0 < \epsilon < 1$, let $\delta$ be any constant at most $\epsilon/3$. The coordinated schedule S generated from Phase4-Alg2 has both feasible production schedule and feasible delivery schedule , and we have* $\text{Profit}(S) \geq (1-\epsilon)\text{Profit}(S^*)$, *where $S^*$ is the optimal schedule.*

**Proof.** Assume $S$ is the schedule with the maximum total profit among all schedules in $T(A_p, n_1, \ldots, n_k)$, $A_p = \bigcup_{j=1}^z A_p^j$ for all $A_p \in A$, and $S'$ is obtained from $S$ after Phase4-Alg2. By Lemma 10, we have $\text{Profit}(S) \geq (1-\delta)\text{Profit}(S^*)$ and $\text{Load}^j(S/A_p) \leq (1+\delta)\widehat{C}_p^j$ for all $1 \leq j \leq z$.

For each delivery time $D^j$ such that $\text{Load}^j(S) > C^j$ (i.e. $\widehat{C}_p^j < \text{Load}^j(S'/A_p)$), Phase4-Alg2 deletes either a large job or a set of small jobs assigned to $D_j$ depending on whether $A_p^j$ is marked as "finalized" or not. In case $A_p^j$ is marked as "finalized", we have $\widehat{C}_p^j = \bar{C}_p^j$ and we choose a set of small jobs to delete from $D_j$. We do the same as in the single delivery time case: partition these small jobs into groups such that the total size of jobs in each group is in the range of $[\delta\widehat{C}_p^j, 2\delta\widehat{C}_p^j]$, where $\widehat{C}_p^j = C^j - \sum_{J_i \in A_p^j} c_i$, there are at least $\lceil \frac{1}{2\delta} \rceil$ groups, delete the group of jobs with smallest total profit. In both cases, the total size of the deleted job(s) is at least $\delta\widehat{C}_p^j$. Therefore after the deletion, the final delivery schedule must be feasible.

All we need to show is that the total profit of $S$ is still close to that of $S^*$. Let $A^j$ be all the jobs assigned to $D^j$ in $S$, then the small jobs assigned to $D^j$ in $S$ is $A^j \setminus A_p^j$. As in the proof of Theorem 5, we can show that for a given delivery time $D^j$ whose large job assignment is $A_p^j$, if a large job assigned to $D^j$ is deleted from $S$, the profit of this large job is at most $\frac{\text{Profit}(A_p^j)}{\lceil \frac{1}{\delta} \rceil} \leq \frac{\text{Profit}(A^j)}{\lceil \frac{1}{\delta} \rceil}$; on the other hand, if a set of small jobs assigned to $D^j$ is deleted from $S$, then their total profit is at most $\frac{1}{\lceil \frac{1}{2\delta} \rceil}$ times the total profit of small jobs assigned to $D^j$, i.e. $\frac{\text{Profit}(A^j \setminus A_p^j)}{\lceil \frac{1}{2\delta} \rceil} \leq \frac{\text{Profit}(A^j)}{\lceil \frac{1}{2\delta} \rceil} \leq 2\delta\text{Profit}(A^j)$. Therefore in either case, we have the total profit deleted from $D^j$ is at most $2\delta\text{Profit}(A^j)$.

So we have

$$\begin{aligned}
\text{Profit}(S') &\geq \text{Profit}(S) - \sum_{j=1}^{z} 2\delta\text{Profit}(A^j) \\
&\geq \text{Profit}(S) - 2\delta\text{Profit}(S) \\
&\geq (1 - 2\delta)\text{Profit}(S) \\
&\geq (1 - \delta)(1 - 2\delta)\text{Profit}(S^*) \\
&\geq (1 - \epsilon)\text{Profit}(S^*). \quad \square
\end{aligned}$$

For any constant $\epsilon$, by Lemmas 9 and 11, we have the following theorem.

**Theorem 12.** *For any coordinated production and multiple delivery scheduling problem with production window and delivery capacity constraints, when there are constant number of delivery times z, and the job set is k-disjoint where k is a constant, there exists a polynomial time approximation scheme. Furthermore, that runs in time*

$$O\left(n^{O\left(\frac{z}{\epsilon^2}+k\right)}\left(\lg \sum_{i=1}^{n} f_i\right)\prod_{j=1}^{z}(\lg C^j)\right).$$

## 5. Conclusion

In this paper, we studied the problem of coordinated production and delivery scheduling problem with production window constraint and the delivery capacity constraint. When the jobs are $k$-disjoint and $k$ is a constant, we develop a PTAS for the case of single delivery time. We then extend the PTAS to solve the problem with constant number of delivery times. One open question is to develop constant approximation algorithms for the general problem.

## Appendix. Converting a list of jobs into $k$-disjoint lists

Our algorithms assume that the jobs are $k$-disjoint and the running time of both algorithms exponentially depends on the parameter $k$. It is essential to find the minimum $k$ and partition the jobs into $k$-disjoint lists. This can be solved by a simple greedy algorithm. For completeness, we will give the algorithm below.

**Theorem 13.** *Given a set of jobs, there exists an $O(n^2)$ time algorithm that partitions the jobs k-disjoint lists, where k is the minimum.*

**Proof.** We use greedy method. We start with an empty list. We repeatedly add into the list the job with the least production window's starting time such that no overlap of production windows is produced. Suppose in total, we build $k$ lists. Let $J_{k,1}$ be the first job in the $k$-th list, and the start time of its production window is $l_{k,1}$. Then it must be the case, that in each of the first $(k-1)$ lists, there is one job whose production window includes the time $l_{k,1}$. Otherwise, $J_{k,1}$ should be added to one of the $k-1$ lists due to the greedy approach. This means $k$ is the minimum number of lists required by any algorithm. $\square$

## References

[1] R. Amstrong, S. Gao, L. Lei, A zero-inventory production and distribution problem with a fixed customer sequence, Ann. Oper. Res. 159 (2008) 395–414.
[2] A. Bar-Noy, S. Guha, J. Naor, B. Schieber, Approximating the throughput of multiple machines in real-time scheduling, SIAM J. Comput. 31 (2001) 331–352.
[3] B. Bilgen, I. Ozkarahan, Strategic tactical and operational production–distribution models: a review, Internat. J. Tech. Management 28 (2004) 151–171.

[4] C. Chekuri, S. Khanna, A polynomial time approximation scheme for the multiple knapsack problem, SIAM J. Comput. 35 (2006) 713–728.
[5] Z.-L. Chen, Integrated production and distribution operations: taxonomy, models, and review, in: D. Simchi-Levi, S.D. Wu, Z.-J. Shen (Eds.), Handbook of Quantitative Supply Chain Analysis: Modeling in the E-Business Era, Kluwer Academic Publishers, Norwell, MA, 2004.
[6] Z.-L. Chen, Integrated production and outbound distribution scheduling: review and extensions, Oper. Res. 58 (1) (2010) 130–148.
[7] S.S. Erenguc, N.C. Simpson, A.J. Vakharia, Integrated production/distribution planning in supply chains: an invited review, Eur. J. Oper. Res. 115 (1999) 219–236.
[8] J.M. Garcia, S. Lozano, Production and delivery scheduling problem with time windows, Comput. Ind. Eng. 48 (2005) 733–742.
[9] M.R. Garey, D.S. Johnson, Two-processor scheduling with start-times and deadlines, SIAM J. Comput. 6 (1977) 416–426.
[10] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman, San Francisco, 1979.
[11] M. Goetschalckx, C.J. Vidal, K. Dogan, Modeling and design of global logistics systems: a review of integrated strategic and tactical models and design algorithms, Eur. J. Oper. Res. 143 (2002) 1–18.
[12] Y. Huo, J.Y-T. Leung, X. Wang, Integrated production and delivery scheduling with disjoint windows, Discrete Appl. Math 158 (2010) 921–931.
[13] K. Jansen, Parameterized approximation scheme for the multiple knapsack problem, in: Proceedings of the Twentieth ACM-SIAM Symposium on Discrete Algorithms, SODA'09, 2009, pp. 665–674.
[14] H. Kellerer, A polynomial time approximation scheme for the multiple knapsack problem, in: Proceedings of APPROX99, 1999, pp. 51–62.
[15] H. Kellerer, J.Y.-T. Leung, C.-L. Li, Multiple subset sum with inclusive assignment set restrictions, Naval Res. Logist (2011) doi:10.1002/nav.20466.
[16] L.G. Kroon, M. Salomon, L.N. Van Wassenhove, Exact and approximation algorithms for the operational fixed interval scheduling problem, Eur. J. Oper. Res. 82 (1995) 190–205.
[17] A.M. Sarmiento, R. Nagi, A review of integrated analysis of production-distribution systems, IIE Trans. 31 (1999) 1061–1074.