Coordinated Scheduling of Production and Delivery with Production Window and Delivery Capacity Constraints

Bin Fu[1], Yumei Huo[2], and Hairong Zhao[3]

[1] Department of Computer Science,
University of Texas–Pan American, Edinburg, TX 78539, USA
binfu@cs.panam.edu

[2] Department of Computer Science,
College of Staten Island, CUNY, Staten Island, New York 10314, USA
yumei.huo@csi.cuny.edu

[3] Department of Mathematics, Computer Science & Statistics,
Purdue University, Hammond, IN 46323, USA
hairong@calumet.purdue.edu

Abstract. This paper considers the coordinated production and delivery scheduling problem. We have a planning horizon consisting of z delivery times each with a unique delivery capacity. Suppose we have a set of jobs each with a committed delivery time, processing time, production window, and profit. The company can earn the profit if the job is produced in its production window and delivered before its committed delivery time. From the company point of view, we are interested in picking 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 times case.

Suppose the given 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 a constant, we developed a PTAS for the single delivery case. For multiple delivery times 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.

In the stage of distribution (delivery), the company may 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 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

B. Chen (Ed.): AAIM 2010, LNCS 6124, pp. 141–149, 2010.

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 the stage of production, each order may have a non-customer defined production window. For example, the 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, it provides a production window for each order where the windows of different orders may overlap. Another example of production window is that the company may have to wait for arrivals of raw materials to start the manufacturing process. If the raw materials are perishable, the company has to start the manufacturing process immediately. Given the arrival schedule of raw materials, the company creates a production window for each order.

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. The company has to decide which orders to accept based on these constraints and the potential profit of each order in order to maximize the total profit .

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*. We focus on the single machine production environment. We are interested in selecting a subset of orders in order to maximize the total profit. When orders are selected, both production schedule and delivery schedule should be considered simultaneously. Thus we face a "coordinated scheduling problem": generate a coordinated schedule, which consists of a production schedule and a delivery schedule subject to the production window, delivery date, and delivery capacity constraints.

Problem definition. Our problem can be formally defined as follows. We have a planning horizon consisting of z delivery times, $T = \{D^1, D^2, \cdots, D^z\}$. Each delivery time D^j is associated with a delivery capacity C^j. We have a set of jobs $J = \{J_1, J_2, \cdots, J_n\}$. 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 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 $p_i \leq d_i$ for all jobs J_i. We also assume $r_i \leq d_i$ for all jobs J_i. The problem is to select a subset of jobs from $J = \{J_1, J_2, \cdots, J_n\}$, and generate a feasible coordinated production and delivery 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.

Literature review. In recent two decades, coordinated production and delivery scheduling problems have received considerable interest. However, most of the research is done at the strategic and tactical levels (see survey articles [10], [5], [7], [2], [3]). At the operational scheduling level, Chen [4] 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 delivery and no delivery capacity constraints considered, 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 ([9]). 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 ([6]) 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 [8], 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 model is more practical and thus more complicated. The problem is NP-hard since the problem at each stage is already NP-hard by itself. So our focus 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(Polynomial Time Approximation Scheme) for the coordinated production and delivery scheduling problem. For multiple delivery times, we also develop a PTAS when the number of delivery time is a constant as well.

The paper is organized as follows. In Section 2, we present an approximation scheme for single delivery time. In Section 3, we present an approximation scheme for multiple delivery times. In Section 4, we draw some conclusions.

2 Single Delivery

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 ϵ, 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 $J = \{J_1, J_2, \cdots, J_n\}$ 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 profit;
Phase IV: convert the schedule from phase III to a feasible schedule.

2.1 Phase I: Select Large Jobs

In this phase, we select large jobs for production and delivery without generating the production schedule. Let us define *large jobs* first. Given a constant parameter $0 < \delta < 1$ which depends on ϵ and will be determined later, a job is said to be *large* if its size is at least δ times the "available" delivery capacity; otherwise, it is a *small job*. By definition, we can see that a job may be "small" at the beginning and becomes "large" later as the available 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 selection. 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, and 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-Alg

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, where* $0 < \delta < 1$ *is a constant.*

2.2 Phase II: Schedule Large and Small Jobs

From Phase I, we get a set of large job selections A without scheduling the jobs. However, for a job selection $A_p \in A$, it is possible no feasible production schedule exists for jobs in A_p. In this case, we say A_p is an infeasible job selection. In this phase, our goal is, to identify all feasible large job selections in A; and for each feasible selection A_p, find a feasible production schedule S that contains the large jobs in A_p, and some newly added "small" jobs, where the small jobs are identified at the beginning of the last iteration of Phase1-Alg, and each has a size less than $\delta\bar{C}_p$. Furthermore, the profit of S, denoted by $\text{Profit}(S) = \sum_{J_i \in S} f_i$, is close to the maximum 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 $\widehat{C}_p$ be the *available capacity for small jobs*, i.e. $\widehat{C}_p = C - \sum_{i \in A_p} C_i$, and 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 $\widehat{C}_p \leq \text{Load}(S/A_p) \leq (1+\delta)\widehat{C}_p$. In this case, we say S is a "valid" schedule.

Even though we know that the large job selection in a schedule S is exactly A_p, we still can not determine 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 build S using dynamic programming. We add jobs to the schedule one by one in certain order. For this, we assume the set of jobs $J = \{J_1, J_2, \cdots, J_n\}$ is k-disjoint, and J has been divided into k job lists $L_1, L_2, \cdots, L_k$ 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 $(1 \leq u \leq k)$. We relabel the jobs in each job list L_u in increasing order of their production windows' starting time, and we use $J_{u,v}$, $1 \leq v \leq n_u$, 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 lists $L_1, L_2, \cdots, L_k$ are not given, note that the problem can be solved greedily.

For a given large job selection produced 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 jobs 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, given a constant $\omega = 1 + \frac{\delta}{2n}$, we say they are *similar*, if Profit(S_1/A_p) and Profit(S_2/A_p) are both in $[\omega^x, \omega^{x+1})$, and Load(S_1/A_p) and Load$(S_2/A_p))$ are both in $[\omega^y, \omega^{y+1})$ for some integer x and y.

In the following, we use $T(A_p, n'_1, \cdots, n'_k)$ to denote a 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 $(1 \le u \le k)$ can be scheduled; (c) and among these 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 $T(A_p, n'_1, \cdots, n'_k)$, 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-Alg(A_p, J, $\bar{C}_p$)

- Input: a set of jobs J, which has been divided into k disjoint job lists $L_1, L_2, \cdots, L_k$;
  A_p: a large job selection obtained from Phase I;
  $\bar{C}_p$: the available capacity at the beginning of last iteration in Phase I for obtaining A_p.
- Let $\widehat{C}_p = C - \sum_{i \in A_p} C_i$, i.e., the available capacity for small jobs
- Initialize $T(A_p, 0, \cdots, 0) = \{\emptyset\}$.
- Construct $T(A_p, n_1, \cdots, n_k)$ using dynamic programming
  To find the set $T(A_p, n'_1, \cdots, n'_k)$, 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 $T(A_p, n'_1, \cdots, n'_t - 1, \cdots, n'_k)$
     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 $T(A_p, n'_1, \cdots, n'_k)$;
     (c) Else
     For each schedule S in $T(A_p, n'_1, \cdots, n'_t - 1, \cdots, n'_k)$
     add S into $T(A_p, n'_1, \cdots, n'_k)$;
     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 $T(A_p, n'_1, \cdots, n'_k)$;

2. From each group of schedules in $T(A_p, n'_1, \cdots, n'_k)$ 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 $T(A_p, n_1, \cdots, n_k)$ with maximum profit

One should note that in the algorithm $\bar{C}_p$ and $\widehat{C}_p$ are not exactly same. Since $\widehat{C}_p$ is available capacity for small jobs, while $\bar{C}_p$ is the available capacity at the beginning of last iteration of Phase1-Alg and some large jobs may be selected at the last iteration, we have $\bar{C}_p \geq \widehat{C}_p$. In particular, if A_p is marked "finalized" at the end of algorithm, then we have $\bar{C}_p = \widehat{C}_p$.

It is easy to see that if a large job selection $A_p \in A$ is not feasible, $T(A_p, n_1, \cdots, n_k)$ must be empty. 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-Alg(A_p, J, $\bar{C}_p$) returns a schedule S such that: the large job selection in S is A_p; S is valid,and* $\text{Profit}(S) \geq (1-\delta)\text{Profit}(S')$.

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

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

2.4 Phase IV: Convert to a Feasible Schedule

From Phase III, we get the schedule S with the maximum total profit 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.

Phase4-Alg(S)

Input: S is the best schedule produced after Phase III, which is valid but may not be feasible

Let A_p be its corresponding large jobs selection in S.

Let $S' = S$
If Load$(S') > C$ (i.e. 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 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 δ be a constant of at most $\frac{\epsilon}{3}$. The schedule S' returned by Phase4-alg is feasible, and* Profit$(S') \geq (1-\epsilon)$Profit(S^*)*, where S^* is an optimal schedule.*

For any constant ϵ, by Lemma 5 and 4, we have the following theorem.

Theorem 1. *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, there exists a polynomial time approximation scheme that runs in time $O(n^{O(k+1/\epsilon^2)}(\lg \sum_{i=1}^{n} f_i)(\lg C))$.*

3 Multiple Delivery

In this section, we study the coordinated production and delivery scheduling problem with multiple delivery times $D^1, D^2, \cdots, D^z$ which have delivery capacity of $C^1, C^2, \cdots, C^z$, respectively. Our goal is to find a feasible coordinated production and delivery schedule whose total profit is close to optimal. As 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 can not be later than its promised delivery date; and the delivery capacity constraint has to be satisfied for all delivery times.

When z is a constant, we develop a PTAS for this case which is based on the PTAS for the case of single delivery time. 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. Due to space limit, we will not given details of the algorithm.

Theorem 2. *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, for any constant ϵ, the algorithm runs in time*

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

4 Conclusion

In this paper, we study the problem of coordinated 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 problem.

References

1. Amstrong, R., Gao, S., Lei, L.: A zero-inventory production and distribution problem with a fixed customer sequence. Ann. Oper. Res. 159, 395–414 (2008)
2. Bilgen, B., Ozkarahan, I.: Strategic tactical and operational production-distribution models: A review. Internat. J. Tech. Management 28, 151–171 (2004)
3. Chen, Z.-L.: Integrated production and distribution operations: Taxonomy, models, and review. In: Simchi-Levi, D., Wu, S.D., Shen, Z.-J. (eds.) Handbook of Quantitative Supply Chain Analysis: Modeling in the E-Business Era. Kluwer Academic Publishers, Norwell (2004)
4. Chen, Z.-L.: Integrated production and outbound distribution scheduling: Review and extensions. In: Operations Research (2009) (to appear)
5. Erenguc, S.S., Simpson, N.C., Vakharia, A.J.: Integrated production/ distribution planning in supply chains: An invited review. Eur. J. Oper. Res. 115, 219–236 (1999)
6. Garcia, J.M., Lozano, S.: Production and delivery scheduling problem with time windows. Computers & Industrial Engineering 48, 733–742 (2005)
7. Goetschalckx, M., Vidal, C.J., Dogan, K.: Modeling and design of global logistics systems: A review of integrated strategic and tactical models and design algorithms. Eur. J. Oper. Res. 143, 1–18 (2002)
8. Huo, Y., Leung, J.Y.-T., Wang, X.: Integrated Production and Delivery Scheduling with disjoint windows. Discrete Applied Math. (2009) (to appear)
9. Kroon, L.G., Salomon, M., Van Wassenhove, L.N.: Exact and approximation algorithms for the operational fixed interval scheduling problem. Eur. J. Oper. Res. 82, 190–205 (1995)
10. Sarmiento, A.M., Nagi, R.: A review of integrated analysis of production-distribution systems. IIE Trans. 31, 1061–1074 (1999)