The 7th International Symposium on Operations Research and Its Applications (ISORA'08)
Lijiang, China, October 31–Novemver 3, 2008

Scheduling jobs on a single machine with inventory operations*

Bao-Qiang Fan[1,†] Guo-Chun Tang[2,‡] Shu-Xia Zhang[3]

[1]Department of Mathematics, Ludong University, Yantai, Shandong 264025, China
[2]Institute of Management Engineering, Shanghai Second Polytechnic University, Shanghai 201209, China
[3]Department of Watercraft CommandčňZhenjiang Watercraft College, Zhenjiang, Jiangsu 212003, China

Abstract In the classical scheduling problems, it is always assumed that jobs would be delivery immediately when they are completed. However, in many production-distribution systems, the jobs are required to be delivered by their due date but completed times. Then those jobs completed ahead of their due dates must be stored. In this paper, we consider the single machine scheduling problem with inventory operations. The objective is to minimize makespan subject to minimize $\sum U_j$. We show the problem is strongly NP-hard. A polynomial 2-approximation scheme for the problem is presented and a special case of the problem, in which each job is one unit in size, is provided an optimal algorithm.

Keywords scheduling; strongly NP-hard; approximation algorithm; performance ratio; makespan

1 Introduction

We consider a maker-to-order production-distribution system consisting of one supplier and more customers. At the beginning of a planning horizon, each customer places a order with the supplier. The supplier needs to process these orders and deliver the completed orders to the customers. Each order has a due date specified by the customer and is required to be delivered by its due date. However, that all orders would just be completed at their respective due dates by the supplier is great difficulty. Some orders have to be scheduled to complete ahead of their due dates so that all orders can be delivered on time.

The problem is often faced by the manufactures who make time-sensitive products such as perishable food, which must be stored in the special storage for those jobs(products) completed ahead of their due dates. Another factor the manufacturer has to consider is that the capacity of storage is limited. The total size of the jobs stored should not be more than such capacity in any time. The problem we study in this paper is to find a schedule for the jobs so that some objectives are optimized. For example, to minimize the total tardy orders or minimize the makespan. To be able to refer to the problems under study

*Supported by the National Science Foundation of China (No. 70618001).
†Corresponding author. E-mail: baoqiangfan@163.com
‡E-mail: gtang@sh163.net

in a concise manner, we shall use the notation of Graham et al.[8], extended to job field with inventory operations. The problem of scheduling jobs on single machine with inventory operations is represented by $1|inven|\gamma$, where *inven* stands for the job with inventory operations.

Consider a single machine scheduling problem, where n jobs $\{J_1, J_2, \cdots, J_n\}$ are ready for processing at time 0. Each job J_j has a processing time p_j, a size v_j and a due date d_j. If π is a schedule of the n jobs, we let C_j denote the completion time of job J_j in π. If $C_j < d_j$, the job needs to be stored until its due date. If $C_j \geq d_j$, the job would be delivered immediately by its completion time. We are given a storage of capacity c meaning that the total size of inventory is up to c at any time. The objective is to minimize makespan subject to minimize $\sum U_j$. We represent the two dual criteria scheduling problems by $1|inven|Les(\sum U_j, C_{max})$.

In the literature, dual criteria scheduling problems have been studied under three approaches; see [12] and [10]. The first approach is to have one criterion designated as the primary criterion and the other one designated as the secondary criterion. Here, we seek a schedule that minimizes the primary criterion and choose, from among all the schedules that minimize the primary criterion, the one that also minimizes the secondary criterion. For example, $1||Les(\sum U_j, C_{max})$ denotes the problem where the primary criterion is the number of tardy jobs and the secondary criterion is makespan. The second approach is to efficiently generate the Pareto curve which enables the decision maker to make explicit trade-offs between these schedules. The final approach is to minimize a cost function which is a linear combination of the two criteria. In this article we consider only the first approach. Although there are numerous work done under the second and the third approaches, we shall not dwell on them in this article.

When the capacity of the storage is unlimited or $\sum_{j=1}^{n} p_j \leq c$, our problem becomes a normal dual criteria scheduling problems $1||Les(\sum U_j, C_{max})$, which is equal to scheduling problem $1||\sum U_j$. For the problem, a schedule with minimum number of tardy jobs can be obtained by the Hodgson-Moore algorithm [15], which schedules jobs in ascending order of due dates. As early as 1956, Smith [16] developed a polynomial-time algorithm for the problem $1||Lex(T_{max} = 0, \sum C_j)$. In 1975, Emmons [6] studied the problem $1||Lex(\sum U_j, \sum C_j)$. He proposed a branch-and-bound algorithm which in the worst case runs in exponential time. Later, Chen and Bulfin [3] proved that the problem is NP-hard with respect to id-encoding. Vairaktarakis and Lee [17] studied the problem $1||Lex(\sum U_j, \sum T_j)$. They gave a polynomial-time algorithm when the set of tardy jobs is specified. As well, a branch-and-bound algorithm was given for the general problem. In 2007, Huo et al. [9] proved the problem is binary NP-hard.

If we consider the inventory cost instead of capacity constrain, and the tardiness penalty instead of $\sum U_j$, the problem $1|inven|Lex(\sum U_j, C_{max})$ becomes a normal JIT(just-in-time) scheduling problem $1||\sum(E_j + T_j)$. Hall and Posner [11] showed the problem is NP-hard. When all jobs have a common due date $d_j \equiv d$, they provided an $O(n\sum p_j)$ pseudo-polynomial time algorithm. For such problem, Bagchi et al. [2] proved that the number of optimal schedules is $2^{\lfloor \frac{n}{2} \rfloor}$. Further results about single machine JIT scheduling problems can be found in [1, 5, 14, 13, 4].

Interestingly, the problem $1||Lex(\sum U_j, C_{max})$ is still open problem. In this paper, we consider the problem with inventory operations. In Section 2, we prove that the problem

$1|inven|Lex(\sum U_j = 0, C_{max})$ is strongly NP-hard and a polynomial approximation scheme for the problem is presented. A special case of the problem, in which each job is one unit in size, is provided an optimal algorithm. Section 3 is a brief conclusion.

2 Optimal $Lex(\sum U_j, C_{max})$

Without loss of generality, we assume that $v_j \leq c, p_j \leq d_j (j = 1, 2, \cdots, n)$ and $\sum_{j=1}^{n} p_j > c$. In many applications, job has a larger size if its processing time is larger. In the following, we consider the case that $v_j = p_j (j = 1, 2, \cdots, n)$. Firstly, we show that the problem of minimizing makespan on single machine with inventory operations is strongly NP-hard.

2.1 The proof of the NP-hard

The problem $1|inven|Lex(\sum U_j = 0, C_{max})$ is strongly NP-hard. This is done by reducing the strongly NP-hard 3-Partition[7] to the decision version of our problem.

3-Partition. Given positive integers t, A and a set of integers $S = \{a_1, a_2, \cdots, a_{3t}\}$ with $\sum_{j=1}^{3t} a_j = tA$ and $A/4 < a_j < A/2$ for $1 \leq j \leq 3t$, does there exit a partition $\langle S_1, S_2, \cdots, S_t \rangle$ of S into 3-element sets such that

$$\sum_{a_j \in S_i} a_j = A$$

for each i?

Theorem 1. *The problem* $1|inven|Lex(\sum U_j = 0, C_{max})$ *is strongly NP-hard.*

Proof. Given the 3-partition problem n, A and a set of integers $\{a_1, a_2, \cdots, a_{3t}\}$. We will first describe the decision version I of the problem $1|inven|Lex(\sum U_j = 0, C_{max})$.

There are basically two classes of jobs in I. The first class, $\{J_j^1 | 1 \leq j \leq t\}$, where job lengths and due date times are specified as follows:

$$\begin{cases} p_j^1 = tA + 1, j = 1, 2, \cdots, t, \\ d_j^1 = j(t+1)A + j, j = 1, 2, \cdots, t. \end{cases}$$

The second class, $\{J_j^2 | 1 \leq j \leq 3t\}$, with job lengths and due dates specified as follows:

$$\begin{cases} p_j^2 = a_j, j = 1, 2, \cdots, 3t, \\ d_j^2 = t^2 A + tA + t, j = 1, 2, \cdots, 3t. \end{cases}$$

The job sizes $v_j = p_j, j = 1, 2, \cdots, 4t$. We define the capacity of the storage is c, where $c = tA$. The bound is given by $\delta = t^2 A + tA + t$. All the remains is to show that the desired partition of S exists if and only if a schedule for I exists, which length less than or equal to δ and all jobs are on time.

Firstly, suppose a partition $\langle S_1, S_2, \cdots, S_t \rangle$ exists which has the desired form. That is each set S_i consists of three elements a_{i1}, a_{i2} and a_{i3}, such that for all $1 \leq i \leq t, \sum_{j=1}^{3} a_{ij} = A$. Then the following schedule π has length $\delta = t^2 A + tA + t$. About the first class jobs in such schedule, the job J_j^1 is processed with the completed time

$$C(J_j^1) = d_j^1 = j(t+1)A + j, j = 1, 2, \cdots, t.$$

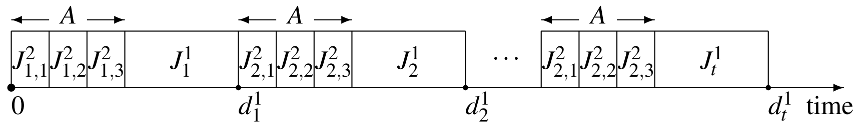

Figure 1: Illustration of the scheduling π

From Fig.1, we note that this basic framework just leaves a series of t "*time slots*" open before the time $t^2A+tA+t$, each of which length exactly A, and the due date of the second class jobs is $t^2A+tA+t$. These are precisely tailored so that we can fit the second class jobs as follows. For each $i=1,2,\cdots,t$,

$$\begin{aligned} s(J^2_{i1}) &= d^1_{i-1} \\ s(J^2_{i2}) &= d^1_{i-1}+a_{i1} \\ s(J^2_{i3}) &= d^1_{i-1}+a_{i1}+a_{i2} \end{aligned}$$

Since $\sum_{j=1}^{3} a_{ij} = A(1 \le i \le t)$ and the total size of jobs in storage is less than the capacity c in any time, this yields a valid schedule with $C_{max}(\pi)=\delta$ and no tardy job.

Conversely, suppose a schedule π with $C_{max}(\pi)=\delta$ does exist. Because the total length of jobs in I is $\sum p_j = t^2A+tA+t$, we must have $C_{max}(\pi)=\delta=t^2A+tA+t$, and the machine is no idle in π. Because of no tardy job in π, from the constructor of the jobs and the capacity $c=tA$, the first class jobs must be scheduled as the same way they are in Fig.1. Thus there are again t slots of length A into which the second class jobs can be placed.

Since the total length of the second class jobs is $\sum_{j=1}^{3t} p^2_j = tA$, every one of these t slots must be filled completely, and hence must contain a set of the second class jobs whose total length is exactly A. Now since every $a_j > A/4$, no such set can contain more than three jobs. Similarly, since $a_j < A/2$, no such set can contain less than three jobs. Thus each set contains exactly three jobs of the second class. Hence, by setting $S_i = \{a_i | d^1_{i-1} < S_{J^2_i} < d_i - p^1_i\}, i=1,2,\cdots,t$, we obtain our desired partition. $\square$

2.2 Approximating optimal makespan in polynomial time

Since the problem $1|inven|Lex(\sum U_j = 0, C_{max})$ is strongly NP-hard, we design an approximation algorithm for the problem. Firstly, we introduce some useful properties associated with optimal schedules as follows.

Lemma 2. *For the problem* $1|inven|Lex(\sum U_j = 0, C_{max})$*, if*

$$\begin{cases} \sum_{j=k}^{n} p_j > c \\ \sum_{j=k+1}^{n} p_j \le c \end{cases}$$

then

$$C^*_{max} \ge d_k,$$

where $k \in \{1, 2, \cdots, n-1\}$ *and* C^*_{max} *is the value of the optimal schedule.*

Proof. If every job is completed before the time d_k, there are at less $n-k$ stocking jobs such as $\{J_k, J_{k+1}, \cdots, J_n\}$ at the time d_k. Since $\sum_{j=k}^{n} p_j > c$, it follows that there must exist job to be processed after the time d_k. Thus the optimal value of the problem is $C^*_{max} \geq d_k$. □

If the problem exists a feasible schedule, there exists a partly feasible schedule about the jobs $\{J_1, J_2, \cdots, J_j\}$ before the time d_j, $j = 1, 2, \cdots, n$. However, tardy job is possible if the jobs $\{J_{k+1}, \cdots, J_n\}$ begin to process at time d_k. To avoid the tardy job, we need to look for an more useful boundary $d_r (k+1 \leq r < n)$. From the Lemma 2, there is a conclusion as follows.

Lemma 3. *For the problem* $1|inven|Lex(\sum U_j = 0, C_{max})$*, let*

$$d_k + \sum_{i=k+1}^{r} p_i - d_r = max\{d_k + \sum_{i=k+1}^{j} p_i - d_j, j = k+1, \cdots, n\}.$$

If

$$d_k + \sum_{i=k+1}^{r} p_i - d_r > 0,$$

then there exists an optimal schedule π *with jobs* $\{J_{r+1}, \cdots, J_n\}$ *begin to process at time* d_r *and*

$$C_{max}(\pi) \leq d_k + \sum_{j=k+1}^{n} p_j.$$

Where $k+1 \leq r \leq n$ *and* k *subject to the Lemma 2.*

Based on the Lemma 2 and Lemma 3, we now provide an approximating algorithm for the problem $1|inven|Lex(\sum U_j = 0, C_{max})$.

Heuristic Alg.1

Step 1. To search for the $k \in \{1, 2, \cdots, n\}$ subject to $\sum_{j=k}^{n} p_j > c$ and $\sum_{j=k+1}^{n} p_j \leq c$.

Step 2. Let $d_k + \sum_{i=k+1}^{r} p_i - d_r = max\{d_k + \sum_{i=k+1}^{j} p_i - d_j, j = k+1, \cdots, n\}$. If $d_k + \sum_{i=k+1}^{r} p_i - d_r > 0$, then $k \doteq r$.

Step 3. Let $C_k = d_k, C_j = min\{d_j, C_{j+1} - p_{j+1}\}, j = k-1, \cdots, 1$, and $s_j = C_{j-1}, j = k+1, \cdots, n$.

Theorem 4. *For* $1|inven|Lex(\sum U_j = 0, C_{max})$*, the* Alg.1 *has a worst-case competitive ratio of* 2*, and the time complexity of the* Alg.1 *is* $O(n^2 logn)$.

2.3 Each job is one unit in size

Consider the following inventory operations. Each job is one umit in size, $v_j \equiv 1$. We are given a storage of capacity c meaning that is capable of storing up to c jobs in any time.

We first provide a optimality property for the problem.

Lemma 5. *For the problem $1|v_j \equiv 1, inven|Lex(\sum U_j = 0, C_{max})$, there exists an optimal schedule π, in such schedule the jobs are processed by the EDD rule.*

Proof. The result is established by a standard job interchange argument. □

For the remainder of this section, let the jobs be indexed so that $d_1 \leq d_2 \leq \cdots \leq d_n$. For any schedule π for jobs $\{J_1, J_2, \cdots, J_n\}$, in which the jobs is processed by the EDD rule, let $I_{d_k} = \{J_i | C_i < d_k, i \geq k\}$ denote the jobs stored at the time d_j. If the number of the jobs stored is not more than c such that $\|I_{d_k}\| \leq c(j = 1, 2, \cdots, n)$, π is feasible.

Algorithm Qusia-EDD

Step 1. Set $d_0 = 0, C_0 = 0$ and $C_1 = p_1$.
Step 2. For $j = 1, 2, \cdots, n$.
Let $d_{k-1} \leq C_{j-1} + p_j < d_k, k \leq j$, and compute $I_{d_k} = \{J_i | C_i < d_k, i \geq k\}$.
If $\|I_{d_j}\| \leq c$, set $C_j = C_{j-1} + p_j$.
Else, set $C_j = d_k$.
Set $j = j + 1$.

Given Lemma 5, the optimality of this algorithm can be easily proved. Hence we state the following result without proof.

Theorem 6. *Algorithm Qusia-EDD can find an optimal schedule for the problem $1|v_j \equiv 1, inven|Lex(\sum U_j = 0, C_{max})$ in $O(n^2)$ time.*

3 Concluding remarks

In this paper, we address the problem $1|inven|Lex(\sum U_j, C_{max})$ and given the proof of strongly NP-hard. A polynomial 2-approximation scheme for the problem is presented and a special case of the problem is provided an optimal algorithm. We will go on researching this problem with other objective (i.e. $Les(\sum U_j, \sum C_j), Les(\sum U_j, T_{max})$ or $Les(T_{max}, \sum C_j)$). Another research topic is about the open problem $1||Lex(\sum U_j, C_{max})$.

References

[1] K.R. Baker, G.D. Scudder. Sequencing with earliness and tardiness penalties: A review. Operations Research, 38, 22-36, 1990.

[2] U. Bagchi, R.S. Sullivan and Yi-Long Zhang. Minimizing absolute and squared deviation of completion times about a common due date. Naval Research Logistics, 33, 227-240, 1986.

[3] C.L. Chen and R.L. Bulfin. Complexity of single machine multicriteria scheduling problems. European Journal Operation Research, 70, 115-125, 1993.

[4] Zhi-Long Chen and W.B. Powell. A column generation based decomposition algorithm for a parallel machine just-in-time scheduling problem. European Journal of Operational Research, 116, 220-232, 1999.

[5] J.S. Davis and J.J. Kanet. Single-machine scheduling with early and tardy completion costs. Naval Research Logistics, 40, 85-101, 1993.

[6] H. Emmons. One machine sequencing to minimize mean flow time with minimum number tardy. Naval Research Logistics, 22, 585-592, 1975.

[7] M.R. Garey and D.S. Johnson. Computers and intactability: A guide to the theory of NP-completeness. Freeman, New York, 1979.

[8] R. L. Graham, E. L. Lawler, J. K. Lenstra and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling. Annals of Discrete Mathematics, 5, 287ÍC326, 1979.

[9] Y-M. Huo, J.Y-T. Leung and H-R. Zhao. Complexity of two dual criteria scheduling problems. Operations Research Letters, 35, 211-220, 2007.

[10] H. Hoogeveen. Multicriteria scheduling. European Journal Operations Research, 167, 592ÍC623, 2005.

[11] N.G. Hall, M.E. Posner. Earliness-tardiness scheduling problems, I: Weighted deviation of completion times about a common due date. Operations Research, 39, 836-846, 1991.

[12] V. Tąŕkindt and J-C. Billaut. Multicriteria Scheduling: Theory, Models and Algorithms. Springer, Berlin, 2002.

[13] Y-D. Kim, C.A. Yano. Minimizing mean tardiness and earliness in single-machine scheduling problems with unequal due dates. Naval Research Logistics, 41, 913-933, 1994.

[14] C-L. Li, Zhi-Long Chen and T.C.E. Cheng. A note on one processor scheduling with asymmetric earliness and tardiness penalties. Operations Research Letters, 13, 45-48, 1993.

[15] J. M. Moore. An *n* job, one machine sequencing algorithm for minimizing the number of late jobs. Management Science, 15, 102-109, 1968.

[16] W.E. Smith. Various optimizers for single stage production. Naval Research Logistics, 3, 59-66, 1956.

[17] G.L. Vairaktarakis and C-Y. Lee. The single-machine scheduling problem to minimize total tardiness subject to minimum number of tardy jobs. IIE Transactions, 27, 250-256, 1995.