Information Processing Letters 114 (2014) 718–722

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Lot scheduling on a single machine

Yung-Tsung Hou, Dar-Li Yang, Wen-Hung Kuo *

Department of Information Management, National Formosa University, Yun-Lin, 632, Taiwan, ROC

ARTICLE INFO

Article history:
Received 3 January 2014
Received in revised form 7 June 2014
Accepted 16 June 2014
Available online 5 July 2014
Communicated by James H. Anderson

Keywords:
Scheduling
Lot scheduling
Order
Total completion time
Single machine

ABSTRACT

In a practical situation, a manufacturer receives different orders from its customers. Different orders may contain different quantities of the product. Therefore, the manufacturer has to decide how to group these orders into different lots based on the capacity of the lot processing machine (such as integrated circuit tester, heated container, etc.) and then decides the sequence of these lots. In this paper, we study a lot scheduling problem with orders which can be split. The objective is to minimize the total completion time of all orders. We show that this problem can be solved in polynomial time.

1. Introduction

Lot processing is a very common production style in the manufacturing industry. One example is the IC (integrated circuit) test in a semiconductor factory, in which some chips are tested simultaneously in a burn-in oven. Another one is the production of adhesives and glues, which normally keeps the mixture of different raw materials in a heated container for a period of time to form the end products.

There are two main classes of batch scheduling problems. One class is that batch sizes can be split. For example, Santos and Magazine [1] considered a single batch machine scheduling problem in which batch sizes are not constrained to be integer, that is, the job number in a batch may not be integer. They assumed that each job has the same processing time and the processing time of a batch depends on how many jobs are grouped into the batch. Also, there is a constant setup time for each batch. The objective is to minimize the total flowtime of all jobs. Naddef and Santos [2], Dobson et al. [3] and Coffman et al. [4] studied different batch scheduling problems for the relaxed version, that is, the integer batch sizes are not required. The other class is that batch sizes cannot be split. For example, Shallcross [5] considered a problem of batching identical jobs on a single machine in which batch sizes cannot be split. He provided a polynomial time algorithm to minimize the total batched completion time. Mosheiov et al. [6] also considered the same problem and proposed a simple and efficient ($O(n)$) transformation (rounding procedure) of the solution given by Santos and Magazine [1] to solve the integer version. Mor and Mosheiov [7] further considered the batch scheduling problem with identical job processing time and identical setup on parallel identical machines for both integer and non-integer versions. For more related studies, the reader is referred to the survey papers (Potts and Kovalyov [8], and Allahverdi et al. [9]).

In this paper, we will consider a similar lot scheduling problem in a real situation. In practical cases, a manufacturer receives different orders from its customers. Different orders may contain different quantities of the products. Therefore, the manufacturer has to decide how to group these orders into different lots based on the capacity of the lot processing machine (such as IC tester, heated container, etc.) and then decides the sequence of these lots. This lot scheduling problem is very similar to the above

* Corresponding author. Tel.: +886 5 6315733; fax: +886 5 6364127.
E-mail address: whkuo@nfu.edu.tw (W.-H. Kuo).

http://dx.doi.org/10.1016/j.ipl.2014.06.016
0020-0190/

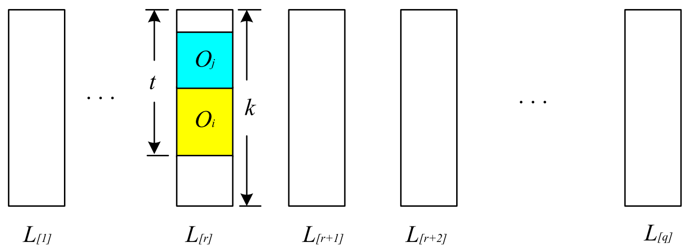

Fig. 1. The original situation of O_i and O_j in Case 1.

Table 1
Notations.

Symbol	Definition
$L_{[r]}$	the lot arranged in position r
k	the capacity of each lot
u	the processing time of each lot
O_i	order i, $i = 1, 2, \dots, n$
σ_i	the size of order i, $i = 1, 2, \dots, n$ ($\sigma_i \le k$)
$C_L(O_i)$	the completion time of O_i in lot L
$\sum C_i$	the total completion of all orders

batch scheduling problems. However, the maximum capacity of one lot is fixed here. The processing time of each lot is fixed no matter how many orders are assigned to the lot. Also, no setup time is considered here.

2. Problem description

There are n orders to be processed on a single lot processing machine. Each order has its own size. The size of each order is no more than the capacity of one lot. Every order can be split if the remaining capacity of the lot is less than the size of the order. Besides, the split order has to be processed in consecutive lots. Therefore, several orders or part of any order can be grouped into one lot and can be simultaneously processed on the machine as long as the total size of these orders does not exceed the capacity of one lot. In addition, the orders in the same lot have the same processing time (u). Therefore, the whole orders in the same lot have the same completion time. (For notations, see Table 1.)

However, for one certain order which is split into different lots, the first part of the order, which is arranged in the first lot, can be delivered to the customer immediately when it is finished. Therefore, it is reasonable to take the completion time of the first /second lot as the completion time of the first/second part of the order. Hence, the order's actual completion time is the sum of the products of the completion times and the percentages of order size in different lots. (For example, if an order is split into two parts, say 30% in the rth lot and 70% in the $(r+1)$th lot, then its actual completion time is $0.3ru + 0.7(r+1)u$.) The machine can handle at most one lot at a time and cannot stand idle until the last lot assigned to it has finished processing. The objective is to minimize the total completion time of all orders. Thus, using the three-field notation, this scheduling problem is denoted by $1/Lot, split/\sum C_i$.

3. Minimizing the total completion time of all orders

In this section, we show that the problem $1/Lot, split/\sum C_i$ is polynomially solvable.

Theorem 1. *For the problem $1/Lot, split/\sum C_i$, there exists an optimal schedule in which orders are sequenced in non-decreasing order of σ_i (i.e. Least order size first rule) and then arranged to lots sequentially.*

Proof. Let $S = (L_{[1]}, L_{[2]}, \dots, L_{[r]}, L_{[r+1]}, L_{[r+2]}, \dots, L_{[q]})$ denote a lot schedule. Suppose the capacity of each lot is k. Each lot contains one or several orders. The size of an order is no more than k. Two orders O_i and O_j are arranged in these lots. Let t be the available capacity of the first lot which O_i and O_j are planned to arrange (see Fig. 1). Also, let the corresponding sizes of the two orders be σ_i and σ_j, respectively. If $\sigma_i > \sigma_j$, we show that swapping orders' positions in these lots does not increase the total completion time of the orders in all possible cases as follows.

Case 1: $\sigma_i + \sigma_j \le t$

Let $L_{[r]}$ contain both O_i and O_j. Since O_i and O_j are in the same lot (see Fig. 1), swapping the two orders does not affect their completion times.

Case 2: $\sigma_i + \sigma_j > t$

Case 2.1: $\sigma_i \le t$

Let $L_{[r]}$ contain O_i and part of O_j, and $L_{[r+1]}$ contain the remaining part of O_j (see Fig. 2). Then, the total completion time of O_i and O_j in $L_{[r]}$ and $L_{[r+1]}$ is calculated as follows.

$$C_L(O_i) + C_L(O_j) = ur\left(1 + \frac{t - \sigma_i}{\sigma_j}\right) + u(r+1)\left(1 - \frac{t - \sigma_i}{\sigma_j}\right) \tag{1}$$

Let $L'_{[r]}$ and $L'_{[r+1]}$ be the new lots by swapping O_i and O_j. It results in the following situation.

Situation (A): $\sigma_j \le t$

In this situation, $L'_{[r]}$ contains O_j and part of O_i, and $L'_{[r+1]}$ contains the remaining part of O_i (see Fig. 3). Since swapping the two orders does not change the sequence of the other orders in these lots, the completion times of the other orders are not changed. Therefore we only need to consider the total completion time of O_i and O_j. The