European Journal of Operational Research 207 (2010) 70–77

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Discrete Optimization

Scheduling multiple orders per job in a single machine to minimize total completion time

Scott J. Mason [a,*], Jen-Shiang Chen [b]

[a] Department of Industrial Engineering, University of Arkansas, 4207 Bell Engineering Center, Fayetteville, AR 72701, USA
[b] Department of Business Administration, Far East University, 49 Junghua Road, Shinshr Shiang, Tainan 744, Taiwan, ROC

ARTICLE INFO

Article history:
Received 8 December 2008
Accepted 15 March 2010
Available online 30 March 2010

Keywords:
Scheduling
Multiple orders per job
Integer programming
Heuristics
Semiconductor manufacturing

ABSTRACT

This paper deals with a single-machine scheduling problem with multiple orders per job (MOJ) considerations. Both lot processing machines and item processing machines are also examined. There are two primary decisions that must be made in the proposed problem: (1) how to group the orders together, and (2) how to schedule the jobs once they are formed. In order to obtain the optimal solution to a scheduling problem, these two decisions should be made simultaneously. The performance measure is the total completion time of all orders. Two mixed binary integer programming models are developed to optimally solve this problem. Also, two efficient heuristics are proposed for solving large-sized problems. Computational results are provided to demonstrate the efficiency of the models and the effectiveness of the heuristics.

1. Introduction

A front-opening unified pod (FOUP) is an automatic wafer transfer facility in the semiconductor manufacturing industry. To avoid contamination, wafers must be held in FOUPs with an inert nitrogen atmosphere. There are two frequently processing machine types in the semiconductor manufacturing factory, including the lot processing machine and the item processing machine. For a lot processing machine, the FOUP processing time is independent of the number of wafers in the FOUP and is equal to the time to process a single wafer on that machine. On an item processing machine, the FOUP processing time is the sum of the processing times for all wafers in all orders in the FOUP (Laub et al., 2007). Thus, semiconductor manufacturers often have the need to group orders from different customers into one FOUP. Once multiple orders are grouped into the same FOUP, these FOUPs must then be scheduled on the machine in the wafer fab, so that effective FOUP processing can reduce the work-in-process and promote on-time delivery of customer orders. Note that there are two primary decisions that must be made in this problem: (1) how to group the orders together, and (2) how to schedule the FOUPs once they are formed. In order to obtain the optimal solution to the proposed scheduling problem, these two decisions should be made simultaneously. We refer to this problem as *multiple orders per job* (MOJ) scheduling problems. The MOJ scheduling problems are also encountered in the TFT–LCD (thin-film transistor–liquid-crystal display) industry except the semiconductor manufacturing industry. The two processing machine types are really met in the real setting, the lot processing machine as the wet sink and the item processing machine as the stepper/scanner.

To match the scheduling terminology, we call a wafer as an item and a FOUP as a job. Several items should be assembled to one order and several orders should be assembled to one job. For conciseness, the notation of Graham et al. (1979) is extended here to include the constraint of MOJ scheduling problems. This notation represents three fields, $\alpha|\beta|\gamma$, where α refers to the machine environment, β refers to the processing characteristics and constraints, and γ specifies the objective to be minimized. $\alpha = 1, Pm, F2$ denote the single machine, identical parallel machines, and two-machine flow shop, respectively. $\beta = moj(lot)$ and $moj(item)$ represent MOJ scheduling problem with the lot and item processing machine, respectively. Moreover, $\gamma = \sum_{i=1}^{n} w_i C_i$, $\sum_{i=1}^{n} w_i T_i$, C_{max}, and $\sum_{i=1}^{n} C_i$ denote the performance measures of the total weighted completion time, total weighted tardiness, makespan, and total completion time, respectively.

Qu and Mason (2005) examined the problems $1|moj(lot)|\sum_{i=1}^{n} w_i C_i$ and $1|moj(lot)|\sum_{i=1}^{n} w_i T_i$ and applied genetic algorithms to solve them. Erramilli and Mason (2008) studied the problems $1|moj(lot)|\sum_{i=1}^{n} w_i C_i$ and $Pm|moj(lot), r_i|\sum_{i=1}^{n} w_i C_i$, where r_i denoted ready time of order i. They presented column generation heuristics to solve the proposed problems. Jia and Mason (2009) investigated the problem $Pm|moj(lot), r_i, s_{jl}|\sum_{i=1}^{n} w_i C_i$, where s_{jl} was the setup

* Corresponding author.
E-mail address: mason@uark.edu (S.J. Mason).

0377-2217/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2010.03.034

time of job j on machine l. A mixed-integer program was proposed and a number of polynomial-time heuristic approaches were presented. Erramilli and Mason (2006) investigated the problem $1|moj(lot), batch|\sum_{i=1}^{n} w_i T_i$, where "*batch*" denoted the lots needed to be grouped batches. A mixed integer program and a new simulated annealing-based heuristic were proposed to solve the problem. Erramilli and Mason (2008) studied the problem $1|moj(lot), batch, incompatible|\sum_{i=1}^{n} w_i T_i$, where "*incompatible*" denoted the incompatible job family case. A mixed-integer programming formulation and a number of simple heuristic approaches were presented. Laub et al. (2007) studied the problems $F2|moj(lot)|C_{max}$ and $F2|moj(item)|C_{max}$. An optimization model was presented that addresses both job formation and job sequencing. They also designed an efficient heuristic to solve the problem $F2|moj(item)|C_{max}$.

Related MOJ scheduling literature falls into three production environments:

(i) Single machine. See for example, Qu and Mason (2005), Erramilli and Mason (2006, 2008), Jampani and Mason (2008).
(ii) Parallel machines. See Jia and Mason (2009) and Jampani and Mason (2008).
(iii) Two-machine flow-shop. See Laub et al. (2007).

Additionally, related MOJ scheduling literature falls into three performance measures:

(i) Total weighted completion times. See for example, Qu and Mason (2005), Jia and Mason (2009) and Jampani and Mason (2008).
(ii) Total weighted tardiness. See Qu and Mason (2005), Erramilli and Mason (2006, 2008).
(iii) Makespan. See Laub et al. (2007).

The previous MOJ scheduling environments focus single machine, parallel machines, and two-machine flow-shop, while the performance measures concentrate on $\sum_{i=1}^{n} w_i C_i$, $\sum_{i=1}^{n} w_i T_i$, and C_{max}. To date, there are not any MOJ scheduling articles using $\sum_{i=1}^{n} C_i$ as the performance measure. This paper considers $\sum_{i=1}^{n} C_i$, a sub class of performance measure $\sum_{i=1}^{n} w_i C_i$, as the objective criterion on the idea that all items carry equal weights. The problems $1|moj(lot)|\sum_{i=1}^{n} C_i$ and $1|moj(item)|\sum_{i=1}^{n} C_i$ are investigated in this study. $1|moj(lot)|\sum_{i=1}^{n} C_i$ and $1|moj(item)|\sum_{i=1}^{n} C_i$ are the special cases of $1|moj(lot)|\sum_{i=1}^{n} w_i C_i$ and $1|moj(item)|\sum_{i=1}^{n} w_i C_i$, respectively. Generally, the special case has important properties and they can improve the computational efficiency. The important properties for problems $1|moj(lot)|\sum_{i=1}^{n} w_i C_i$ and $1|moj(item)|\sum_{i=1}^{n} w_i C_i$ have not been found, so meta-heuristics are employed to solve $1|moj(lot)|\sum_{i=1}^{n} w_i C_i$ and $1|moj(item)|\sum_{i=1}^{n} w_i C_i$. This paper first provides two *NP*-hardness proofs and presents some properties for the two proposed problems. Two mixed binary integer programming (BIP) models are provided to derive the optimal solution, one model for $1|moj(lot)|\sum_{i=1}^{n} C_i$ and the other for $1|moj(item)|\sum_{i=1}^{n} C_i$. Additionally, two efficient heuristics are proposed for finding the near-optimal solution for large-sized problems, one heuristic for $1|moj(lot)|\sum_{i=1}^{n} C_i$ and the other for $1|moj(item)|\sum_{i=1}^{n} C_i$. Computational results are provided to demonstrate the efficiency of the models and the effectiveness of the heuristics.

2. Problem description and notation definition

2.1. The notation

The following notation was used throughout the study:

Symbol definition	
O_i	order number i
J_j	job number j
$J_{[q]}$	the job in the sequence position q
π_k	feasible schedule number k
Input parameters	
M_L	a relative large positive number for the lot processing case
M_I	a relative large positive number for the item processing case
N	the number of orders
K	the capacity of FOUP
F	the number of FOUPs for all orders to assign
ϕ	the number of jobs for all orders to assign, where $\phi = \min\{N, F\}$
σ_i	the size of O_i
ρ_L	the unit lot processing time
ρ_I	the unit item processing time
$TC_L(\pi_k)$	the total completion time under the feasible schedule π_k and the lot processing case
$TC_I(\pi_k)$	the total completion time under the feasible schedule π_k and the item processing case
Decision variables	
$n_{[q]}$	the number of orders of $J_{[q]}$
$s_{[q]}$	the order sizes of $J_{[q]}$
$p_{[q]}$	the processing times of $J_{[q]}$
C_i	the completion time of O_i, that is, equals the completion time of the job to which O_i is assigned
$\tau_{[q]}$	the completion time of $J_{[q]}$
$X_{i[q]}$	1 if O_i is assigned to $J_{[q]}$; 0 otherwise
$W_{[q]}$	1 if any order is assigned to $J_{[q]}$; 0 otherwise

2.2. Problem description

Fig. 1 illustrates an example for the single machine MOJ scheduling problem. There are five orders, and three FOUPs for all orders to assign. To obtain a feasible schedule, we set the FOUP capacity at $K = 3$ and the number of jobs at $\phi = \min\{N, F\} = \min\{5, 3\} = 3$. The three jobs include J_1, J_2, and J_3, where J_1 consists of O_1, J_2 consists of O_2, O_3, and O_4, and J_3 consists of O_5. Assume these three jobs sequence on a single machine in the sequence of J_2, J_3, and J_1.

This study considers the single machine MOJ scheduling problem in which a machine includes both lot and item processing environments. MOJ scheduling in a lot processing environment means all the items in a job are simultaneously processed and all the orders in that job have the same processing time. However,

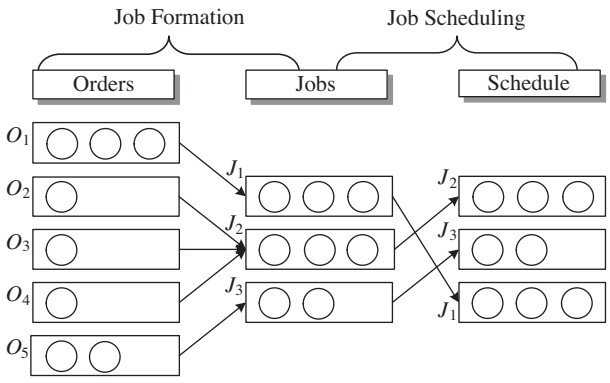

Fig. 1. An example for the single machine MOJ scheduling problem.