Available online at www.sciencedirect.com
ScienceDirect
Int. J. Production Economics 111 (2008) 368–377
international journal of production economics
www.elsevier.com/locate/ijpe

Heuristics for scheduling in a no-wait open shop with movable dedicated machines

Hung-Tso Lin[a,*], Hong-Tau Lee[b], Wen-Jung Pan[b]

[a]*Department of Distribution Management, National Chin-Yi University of Technology, 35, Lane 215, Chung-Shan Road, Section 1, Taiping City, Taichung County, 41101, Taiwan, ROC*
[b]*Department of Industrial Engineering and Management, National Chin-Yi University of Technology, Taichung County, Taiwan, ROC*

Received 6 June 2005; accepted 9 January 2007
Available online 21 February 2007

Abstract

In this paper, we address a scheduling problem related to people's livelihood, such as road construction works that include laying pipes for gas, water and phone. The problem is a multi-processing-stage open shop with the characteristics of movable dedicated machines and no-wait restriction, also known as no intermediate queue. The objective is to schedule the jobs such that the total occupation time for all the processing stages is minimized. Some two-phase heuristic algorithms are proposed for solving the problem. Computational results show that the heuristic is fairly effective in finding an optimal or a near-optimal solution for small-sized problems. Results of the heuristic for experiments tallied with the real-life environment demonstrate the potential of the heuristic to efficiently deal with the scheduling problems.

Keywords: Open shop; No-wait; Mixed integer programming; Heuristic

1. Introduction

The problems of how to handle properly road construction works such as laying pipes for water supply, gas supply and power networks are frequently encountered by section chiefs. Suppose there are n construction units such as companies of gas supply, water supply, and power supply, each proposes a job for laying pipes on m sections of road. The m sections of road can be regarded as m processing stages. Each job processed on each section of road can then be regarded as an operation on the processing stages. The processing time of operations may be assumed to be zero if such operations need not be implemented at some processing stages. For each construction unit, the sequence of operations that will be processed on those sections of road is adjustable. Each job is processed with a dedicated machine owned by each construction unit. These construction units move the dedicated machines to the sections of road when the operations are processed. With a consideration of efficiency, the m operations of each job should be processed continuously on the m sections of road meaning no-wait restriction, and hence, the dedicated machines will be shifted to the pre-determined sections of road immediately as the need arises. It is possible for different jobs to be processed

*Corresponding author. Tel.: +886 4 23924505x7805; fax: +886 4 23932065.
E-mail address: htl@ncit.edu.tw (H.-T. Lin).

0925-5273/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.ijpe.2007.01.005

simultaneously on the same section of road. The aspiration to reduce the period of social costs such as air pollution, traffic congestion and others depends on an adequate arrangement of the n jobs that minimize the total occupation time of the m sections of road.

The scenario of the previous problem can be illustrated as a scheduling problem of n jobs with dedicated machines to be processed on m processing stages of an open shop. The constraint of the problem is that each job has to be processed on the m processing stages with no-wait. The scheduling objective is to minimize the total occupation time of the m processing stages, denoted by D. The performance measure D expresses the idea of social costs or environmental responsibility. Concerning the scheduling study of taking into account environmental cost, Subai et al. (2006) particularly emphasize the environmental criterion, which is characterized by energy consumption and creation of polluted water and sludge, in a treatment surface line scheduling problem.

According to the classification of Graham et al. (1979), the addressed problem can be expressed as $Om/no\text{-}wait/D$. Lots of researchers have devoted efforts to the open shop scheduling problems with one machine fixed on each processing stage. With respect to the open shop scheduling problems with the objective of minimizing makespan ($C_{\max}$), Pinedo (1995) proposed the scheduling rules using the longest alternate processing time first (LAPT) and the longest total processing time (LTPT) for the problems with two processing stages and more than two processing stages, respectively. Adiri and Aizikowitz (1989) proposed a scheduling rule under the specific production condition when one processing stage dominates other processing stages. Liaw (2000) developed a hybrid genetic algorithm that comprises tabu search and genetic algorithm. Concerning open shop with total tardiness or total completion time performances, Liaw (2003) proposed an efficient tabu search approach to deal with the preemptive scheduling problem of minimizing total tardiness. Later, he proposed a dynamic programming algorithm to solve the problem of minimizing total completion time ($\sum C_j$) (Liaw, 2004). Lauff and Werner (2004) investigated the complexity and properties of the scheduling problem with earliness and tardiness penalties. Concerning multi-criteria problems, Kyparisis and Koulamas (2000) proposed a scheduling rule with complexity of polynomial time that first minimizes $C_{\max}$ and then minimizes $\sum C_j$. Gupta et al. (2003) proposed insertion and iterative heuristic algorithms for the problems with secondary criteria where the primary criterion is the minimization of $C_{\max}$ and the secondary criterion is the minimization of the total flow time, total weighted flow time, or total weighted tardiness time. Konno and Ishii (2000) investigated open shop scheduling problem of maximizing the minimum value of satisfaction degrees with respect to the processing intervals of jobs and the resource amounts used in the processing intervals with flexible deadline and resource.

Due to some characteristics of the circumstances and processing technology, the operations of a job must be performed without any waiting between stages, which is known as no-wait restriction. For the case of an open shop no-wait scheduling problem, most of the previous research has been focused to two-processing-stage or m-processing-stage problems, where all operations have equal processing times. For the two-processing-stage open shop no-wait scheduling with $C_{\max}$ as the objective, Sidney and Sriskandarajah (1999), Yao and Soewandi (2000) and Liaw et al. (2005) proposed heuristics to solve the problem. Adiri and Amit (1984) proposed a scheduling rule for the m-processing-stage open shop to minimize $\sum C_j$ with no-wait and all operations have equal processing times. They also developed a heuristic rule to minimize $\sum C_j$ and $C_{\max}$ simultaneously where all operations have equal processing times.

Concerning linear programming (LP) formulations for solving the open shop scheduling problems, Liaw (2005) presented a LP formulation for the preemptive open shop scheduling problem to minimize total tardiness if the sequence in which jobs are completed is known. Noda et al. (2006) proposed a LP formulation to find feasible schedules for the preemptive open shop scheduling problem with time-windows, where the release times are always satisfied and lateness is not permitted, and then they made use of the formulation to solve the problem where the objective is to minimize $C_{\max}$.

The paper is organized as follows. The properties of the addressed problem are illustrated by a numerical example in Section 2. The mixed integer programming (MIP) formulations for modeling the scheduling problem are presented in Section 3. Some properties of an optimal schedule will be explored in Section 4 for the purpose of developing heuristic algorithms. Some efficient two-phase

heuristic algorithms are presented in Section 5. Computational results and analysis are presented in Section 6 and conclusions are made in Section 7.

2. Illustration of problem scenario

Suppose there are two construction units: a gas supply company and a water supply company. Each of them proposes a construction work of laying pipes named jobs 1 and 2, respectively. These two jobs will be processed at three road sections regarded as processing stages 1, 2 and 3, respectively. Each of the two companies processes its works with dedicated equipment named machines A and B, respectively, which can be moved to these three road sections as the need arises. The sequence of the construction works at these three road sections, for the two construction units, is arranged arbitrarily. For example, the gas supply company can schedule its works at these three road sections with the sequence of processing stages 1-2-3 or 2-1-3 or others. Table 1 depicts the data for each job of each construction unit. Fig. 1 illustrates an example of two jobs processed at three processing stages.

The rectangles with bold frame represent that job 1 is processed with the operation sequence of O_{11}-O_{12}-O_{13}-, or with the processing stage sequence of 1-2-3. The other rectangles with normal frame represent that job 2 is processed with the operation sequence of O_{21}-O_{22}-O_{23}, or with the processing stage sequence of 2-1-3. The operations of both jobs satisfy the no-wait restriction. O_{11} and O_{21} are processed simultaneously at processing stage 1 in the period [7,8]. O_{13} and O_{23} are also processed simultaneously at processing stage 3 in the period [20,26]. From this example, the occupation times of processing stages 1–3 are in the periods [0, 16], [0, 20] and [16, 32], respectively. The total occupation time of the three processing stages is calculated as $(16-0)+(20-0)+(32-16)=52$.

Table 1
The data for two jobs processed at three processing stages

Construction unit	Gas supply company			Water supply company		
Job j	Job 1			Job 2		
Dedicated machine	A			B		
Processing stage (k)	1	2	3	1	2	3
Operation k of job j (O_{jk})	O_{11}	O_{12}	O_{13}	O_{21}	O_{22}	O_{23}
Processing time (t_{jk})	8	12	6	9	7	16

3. MIP model

In this section, a MIP model is presented for the addressed scheduling problem. For notation used, see Table 2.

$$\text{Minimize } D=\sum_{k=1}^{m}(C_{\max,k}-A_{\min,k}). \tag{1}$$

$$\text{Subject to } \sum_{s=1}^{m} y_{jks}=1,\ j\in J,\ k\in K. \tag{2}$$

$$\sum_{k=1}^{m} y_{jks}=1,\ j\in J,\ s\in S. \tag{3}$$

$$C_{jk}=t_{jk}+\sum_{s=2}^{m}\sum_{l=1,l\neq k}^{m}\sum_{h=1}^{s-1}(y_{jks}\times y_{jlh}\times t_{jl}),\ j\in J,\ k\in K. \tag{4}$$

$$A_{jk}=C_{jk}-t_{jk},\ j\in J,\ k\in K. \tag{5}$$

$$C_{\max,k}\geqslant C_{jk},\ j\in J,\ k\in K. \tag{6}$$

$$A_{\min,k}\leqslant A_{jk},\ j\in J,\ k\in K. \tag{7}$$

$$C_{jk}\geqslant 0,\ j\in J,\ k\in K. \tag{8}$$

$$A_{jk}\geqslant 0,\ j\in J,\ k\in K. \tag{9}$$

$$y_{jks}\in\{0,\ 1\},\ j\in J,\ k\in K,\ s\in S. \tag{10}$$

The objective function (1) is to minimize the total occupation time of the m-processing stages. Constraint (2) indicates that each operation of a job must be arranged exactly in one position of the sequence. Constraint (3) indicate s that only one of the m operations of job j can be arranged in a specific position of the sequence. Constraint (4) identifies the completion time of O_{jk} under the no-wait restriction. The meaning of other constraints is evident. The total number of variables in this MIP model is $m(nm+2n+2)$, in which there are nm^2 binary variables. Consequently, the total number of constraints is $8nm$.

According to the data in Table 1 and the schedule in Fig. 1, the values of variables in this model are as follows:

Objective function (1) $D=(16-0)+(20-0)+(32-16)=52$

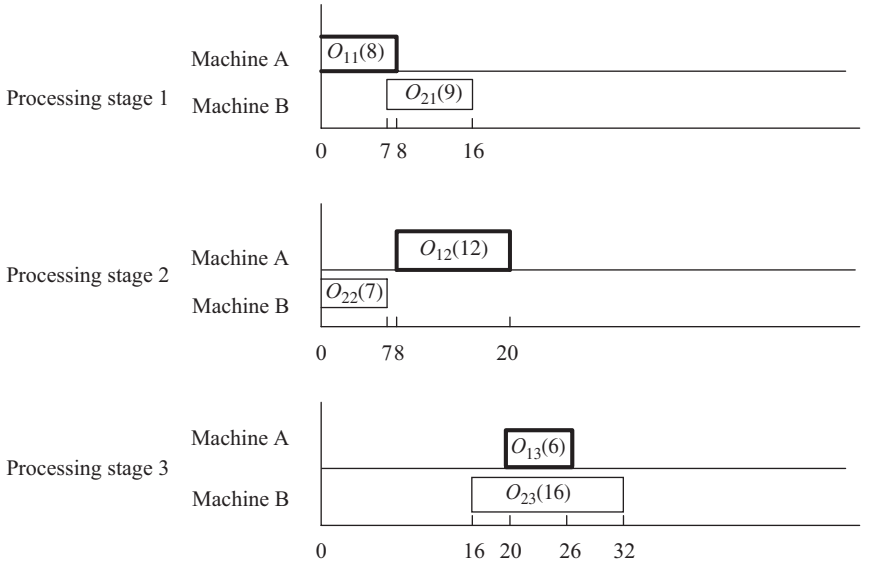

Fig. 1. Example of two jobs processed at three processing stages.

Table 2
Notation

Indices
$j =$ job, $j \in J = \{1, 2, \ldots, n\}$.
$k =$ processing stage, $k \in K = \{1, 2, \ldots, m\}$.
$s =$ position of the sequence, $s \in S = \{1, 2, \ldots, m\}$

Input parameters
$t_{jk} =$ processing time of O_{jk}. Set $t_{jk} = 0$ if job j has no operation at processing stage k

Decision variables
$y_{jks} = 1$, if operation O_{jk} is in the sth position of the sequence; otherwise $y_{jks} = 0$.
$C_{jk} =$ the completion time of O_{jk}.
$A_{jk} =$ the starting time of O_{jk}.
$C_{\max,k} =$ the latest completion time of processing stage k.
$A_{\min,k} =$ the earliest starting time of processing stage k.

Constraints (2) and (3)

$$y_{111} = 1, \quad y_{112} = 0, \quad y_{113} = 0,$$
$$y_{121} = 0, \quad y_{122} = 1, \quad y_{123} = 0,$$
$$y_{131} = 0, \quad y_{132} = 0, \quad y_{133} = 1,$$
$$y_{211} = 0, \quad y_{212} = 1, \quad y_{213} = 0,$$
$$y_{221} = 1, \quad y_{222} = 0, \quad y_{223} = 0,$$
$$y_{231} = 0, \quad y_{232} = 0, \quad y_{233} = 1.$$

(4) $C_{11} = 8, \ C_{12} = 20, \ C_{13} = 26, \ C_{21} = 16, \ C_{22} = 7, \ C_{23} = 32.$

(5) $A_{11} = 0, \ A_{12} = 8, \ A_{13} = 20, \ A_{21} = 7, \ A_{22} = 0, \ A_{23} = 16.$

(6) $C_{\max,1} = 16, \ C_{\max,2} = 20, \ C_{\max,3} = 32.$

(7) $A_{\min,1} = 0, \ A_{\min,2} = 0, \ A_{\min,3} = 16.$

The optimal schedule for the illustrating example is with the operation sequence of O_{11}-O_{12}-O_{13} for job 1 and O_{21}-O_{22}-O_{23} for job 2. Fig. 2 describes the situation of each job processed at each stage. The minimal total occupation time of the three processing stages is 37 ($= 9-0+20-8+32-16$).

4. Exploring the phenomenon of optimal schedules

With respect to the complexity of the open shop problem, Gonzalez and Sahni (1976) proved that the complexity for minimizing $C_{\max}$ of the three-processing-stage is NP hard. Later, Sahni and Cho (1979) showed the complexity for minimizing $C_{\max}$ of the two-processing-stage with no-wait restriction is strongly NP-hard. The performance measure of the problem under consideration, $D = \sum_{k=1}^{m} (C_{\max,k} - A_{\min,k})$, is equivalent to $D = \sum_{k=1}^{m} w_k C_{\max,k} - \sum_{k=1}^{m} v_k A_{\min,k}$, where the values of weights w_k and v_k ($k = 1, 2, \ldots, m$) equal to 1. By setting all the values of weights equal to zero, except for the weight of $C^*_{\max} = \max_{k \in K}\{C_{\max,k}\}$, the problem becomes a special case of the addressed problem. The performance measure $D = C^*_{\max}$ is equivalent to makespan. The number of processing stages of the addressed problem is usually greater than three. As this special case of open shop no-wait scheduling

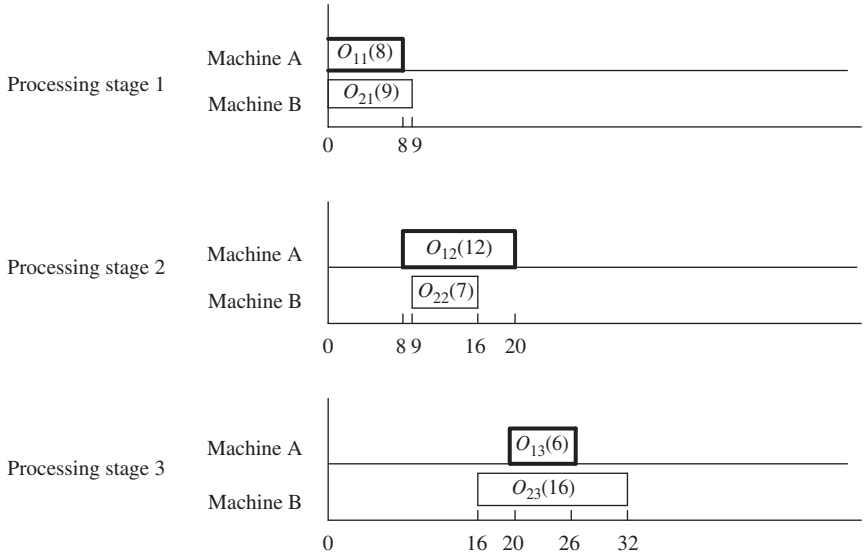

Fig. 2. The optimal schedule for two jobs processed at three processing stages.

problem is shown to be NP-hard, the addressed problem is also NP-hard. Hence, we dedicate our efforts to develop heuristic algorithms to find an approximate solution.

Exploring the properties of the optimal schedule will contribute to the development of heuristic algorithms. Owing to the complexity of the addressed problem, small-sized problems are designed deliberately for the experiment. The number of jobs and the number of operations in the test problems are set at three levels of 3–5, respectively. Consequently, nine combinations of the test modules are available. The MIP model is employed to formulate the designed problems. A package of LINGO is used to solve the formulated problems. If the sequences of operations of each job are identical for all jobs in a test problem, we can conclude that the property of the optimal schedule is of flow shop type. Otherwise, it will be regarded as having the property of job shop type. In order to explore the property of an optimal schedule influenced by the distribution of operation processing time, two approaches are used to generate the processing time.

4.1. Complete random design

The processing time of each operation of each job is generated randomly using a uniform distribution of parameters [1,10], which follows the methods proposed by Schaller (2001). Twenty test problems are generated for each test module. The experimental results indicate that 162 of the 180 test problems obtained the optimal schedules as of flow shop type. For the other 18 test problems currently with the property of job shop type, if we give them another run, some of them may get the same value of performance measure but with the property of flow shop type. Consequently, we can conclude that there are more than 90% (= 162/180) of the complete random design (CRD) test problems having the property of flow shop type.

4.2. Block random design

Using the analog of the method proposed by Schaller (2001) to generate the family set-up time, we generate randomly the processing time for the operations using a uniform distribution with the following three sets of parameter values: (i) [1, 20], (ii) [21, 50], (iii) [51, 100]. Twenty test problems are generated for each test module. The deployment of the processing time is displayed in Table 3.

The experimental results indicate that 175 of the 180 test problems obtained the optimal schedules as of flow shop type. Because optimal schedules of both flow shop and job shop types may exist for the other five test problems, we can conclude that more than 97% (= 175/180) of the block random design (BRD) test problems possess the property of flow shop type.

Table 3
The deployment of the processing time

Operation	Job				
	1	2	3	4	5
1	[1, 20]	[21, 50]	[51, 100]	[1, 20]	[21, 50]
2	[21, 50]	[51, 100]	[1, 20]	[21, 50]	[51, 100]
3	[51, 100]	[1, 20]	[21, 50]	[51, 100]	[1, 20]
4	[1, 20]	[21, 50]	[51, 100]	[1, 20]	[21, 50]
5	[21, 50]	[51, 100]	[1, 20]	[21, 50]	[51, 100]

5. Heuristic algorithms

The philosophy of two-phase heuristic approach to the scheduling problems (Suliman, 2000; Lin and Liao, 2003) is employed to develop a solution procedure for the addressed problem. An initial sequence is developed by proper methods in the first phase and then this sequence is improved iteratively in the second phase.

5.1. Phase I—developing an initial sequence

In the previous section, the 360 test problems reveal that more than 93% [(162+175)/360] of the optimal schedules have the property of flow shop type. For this reason, we develop the forward and backward recursive algorithms to generate the operation sequences according to flow shop type. The sequence with smaller value of the two performance measures will be selected as the initial sequence.

5.1.1. Forward recursive algorithm

An algorithm that arranges forwardly the operation sequence of each job is named forward recursive algorithm. For each of the processing stages, reducing the difference between the latest completion time and earliest completion time will shorten the occupation time. Therefore, when we are considering the operation which will be arranged in the gth ($g = 1, 2, \ldots, m-1$) position of the sequence, we check each of the $m-g+1$ undetermined operations and estimated the latest completion times and earliest completion times of the processing stages. The operation that gives the minimal difference of the estimated latest completion time and earliest completion time is arranged in the gth position of the sequence. The steps of the algorithm are described as follows:

Step 1: Set $g = 1$, $J = \{1, 2, \ldots, n\}$, $K = \{1, 2, \ldots, m\}$ and $K' = \varnothing$.

Step 2: Let $T_{jk} = t_{jk} + \sum_{k' \in K'} t_{jk'}, j \in J, k \in K$.

Step 3: Calculate $d_k = \max_{j \in J}\{T_{jk}\} - \min_{j \in J}\{T_{jk}\}$, $k \in K$.

Step 4: Let $d_{k^*} = \min_{k \in K}\{d_k\}$, if there is more than one index k^* taking on this value, perform the calculation in Step 3 in which the $\min\{T_{jk}\}$ will be replaced with the second minimum and so on until only one index k^* is identified.

Step 5: The operation k^* of each job, denoted by $O_{\bullet k^*}$, is arranged in the gth position of the sequence, reset $g = g + 1$.

Step 6: Reset $K = K \backslash k^*$ and $K' = K' \cup \{k^*\}$.

Step 7: If $g = m$, terminate the algorithm. The operation sequence along with the associated total occupation time is obtained. Otherwise, go back to Step 2.

5.1.2. Backward recursive algorithm

An algorithm that arranges the operation sequence backwardly for each job will be called backward recursive algorithm. For each of the processing stages, reducing the difference between the earliest starting time and latest starting time will shorten the occupation time. Therefore, when we are considering the operation which will be arranged in the gth ($g = m, m-1, \ldots, 2$) position of the sequence, the earliest starting time and latest starting time occurring somewhere on the undetermined g processing stages are estimated. The operation that gives the minimal difference of the estimated earliest starting time and latest starting time is arranged in the gth position of the sequence. The steps of the algorithm are described as follows:

Step 1: Set $g = m$ and $K = \{1, 2, \ldots, m\}$.

Step 2: Calculate

$$d_k = \max_{j \in J}\left\{\sum_{\substack{l \in K \\ l \neq k}} t_{jl}\right\} - \min_{j \in J}\left\{\sum_{\substack{l \in K \\ l \neq k}} t_{jl}\right\},\ k \in K.$$

Step 3: Let $d_{k^*} = \min_{k \in K}\{d_k\}$, if there is more than one index k^* taking on this value, perform the calculation in Step 2 in which the $\min\{\sum t_{jl}\}$ will be replaced with the second minimum and so on until only one index k^* is identified.

Step 4: The operation k^* of each job, denoted by $O_{\bullet k^*}$, is arranged in the gth position of the sequence, reset $g = g-1$.

Step 5: Reset $K = K \backslash k^*$.

Step 6: If $g = 1$, terminate the algorithm. The operation sequence along with the associated total occupation time is obtained. Otherwise, go back to Step 2.

5.2. Phase II—the iterative improvement procedure

The procedure is performed by altering the operations in the adjacent positions of the sequence. We now elaborate the principle in detail. Consider two operations of job x, $O_{x\alpha}$ and $O_{x\beta}$, where $O_{x\alpha}$ and $O_{x\beta}$ are in the sth and $s+1$st positions of the sequence, respectively. That is, $O_{x\alpha}$ is processed immediately preceding $O_{x\beta}$, denoted by $O_{x\alpha} \rightarrow O_{x\beta}$. If the sequence is altered to $O_{x\beta} \rightarrow O_{x\alpha}$, the performance measure D may be varied. The change in the value of D is analyzed as follows:

(i) $A_{x\beta}$ and $C_{x\beta}$ are advanced by the period $t_{x\alpha}$ because of the alteration. As $A_{\min,\beta}$ and $C_{\max,\beta}$ are probably advanced, and hence, the value of D may be varied. The increment in the value of D caused by advancing $A_{x\beta}$ and reduction in the value of D caused by advancing $C_{x\beta}$are calculated as $D_{\beta}^{+} = \max\{A_{\min,\beta} - (A_{x\beta} - t_{x\alpha}), 0\}$ and $D_{\beta}^{-} = \min_{j \in J, j \neq x} \{C_{\max,\beta} - C_{j\beta}, t_{x\alpha}\}$, respectively.

(ii) $A_{x\alpha}$ and $C_{x\alpha}$ are postponed for the period $t_{x\beta}$ because of the alteration. As $A_{\min,\alpha}$ and $C_{\max,\alpha}$ are probably postponed, and hence, the value of D may be varied. The increment in the value of D caused by postponing $C_{x\alpha}$ and reduction in the value of D caused by postponing $A_{x\alpha}$ are calculated as $D_{\alpha}^{+} = \max\{C_{x\alpha} + t_{x\beta} - C_{\max,\alpha}, 0\}$ and $D_{\alpha}^{-} = \min_{j \in J, j \neq x} \{A_{j\alpha} - A_{\min,\alpha}, t_{x\beta}\}$, respectively.

(iii) The change in the value of D caused by the alteration is calculated as $D^{\bullet} = (D_{\alpha}^{-} + D_{\beta}^{-}) - (D_{\alpha}^{+} + D_{\beta}^{+})$. If the value of $D^{\bullet}$ is greater than zero, meaning the performance measure is improved, the alteration is performed.

Consider the example illustrated in Fig. 1. If the sequence $O_{22} \rightarrow O_{21}$ is altered to $O_{21} \rightarrow O_{22}$, the change in the value of D is analyzed as follows: (i) $D_1^{+} = 0$ and $D_1^{-} = 7$ caused by advancing A_{21} from 7 to 0 and C_{21} from 16 to 9, respectively. (ii) $D_2^{+} = 0$ and $D_2^{-} = 8$ caused by postponing C_{22} from 7 to 16 and A_{22} from 0 to 9, respectively. (iii) As $D^{\bullet} = 7 + 8 - 0 = 15$, meaning reduction of 15 in the value of D, the alteration is performed. The updated sequence is illustrated in Fig. 2.

The iterative improvement procedure is performed as follows:

Step 0: Initially, set $I = 1$, $j = 1$ and $s = 1$.

Step 1: Identify the operations of job j in the sth and $s+1$st positions of the sequence, denoted by $O_{j\alpha} \rightarrow O_{j\beta}$. Consider to alter sequence $O_{j\alpha} \rightarrow O_{j\beta}$ to $O_{j\beta} \rightarrow O_{j\alpha}$ and calculate the associated value of $\bullet$.

Step 2: If the value of $D^{\bullet}$ is greater than zero, perform the alteration and calculate $D = D - D^{\bullet}$. Otherwise, maintain the original sequence.

Step 3: If $s < m-1$, let $s = s+1$, go back to Step 1; otherwise, continue.

Step 4: If $j < n$, let $j = j+1$ and reset $s = 1$, go back to Step 1; otherwise, continue.

Step 5: If $I = 5$, the procedure is terminated and the heuristic is obtained; otherwise, let $I = I+1$ and reset $j = 1$ and $s = 1$, go back to Step 1.

5.3. Numerical example

A three-job, four-operation open shop scheduling problem shown in Table 4 is solved to illustrate the heuristic algorithms. By performing the forward recursive algorithm, the three jobs are processed with the operation sequence of $O_{\bullet 2} - O_{\bullet 1} - O_{\bullet 4} - O_{\bullet 3}$, the total occupation time of the four processing stages is 210. By performing the backward recursive algorithm, the operation sequence and the total occupation time are identical with the results for the forward recursive algorithm. As a result, the initial sequence along with the associated performance measure of this problem is $O_{\bullet 2} - O_{\bullet 1} - O_{\bullet 4} - O_{\bullet 3}$ and 210. By performing the iterative improvement procedure, the sequence $O_{24} \rightarrow O_{23}$ is altered to $23 \rightarrow O_{24}$ which causes reduction of four in the performance measure. As a result, the

Table 4
Processing time for three jobs at four processing stages

	t_{jk}	Job j		
		1	2	3
Processing stage (k)	1	33	18	20
	2	43	33	42
	3	44	7	8
	4	6	3	41

operation sequences are O_{12}-O_{11}-O_{14}-O_{13}, O_{22}-O_{21}-O_{23}-O_{24} and O_{32}-O_{31}-O_{34}-O_{33} for jobs 1–3, respectively, and the value of heuristic performance measure, D_{heu}, is 206. This problem is also formulated with the MIP model and solved with the software LINGO. The optimal operation sequences for jobs 1–3 along with the optimal performance measure, D_{opt}, are identical with the heuristic.

6. Computational results and analysis

To test the effectiveness of the heuristic, the heuristic algorithms are employed to solve the previous 360 CRD and BRD test problems. The performance of the heuristic is evaluated by comparing its solution with the optimal solution obtained by the MIP model. The computational results are summarized in Table 5, where the percentage error of heuristic from optimum is calculated as $(D_{heu} - D_{opt}) \times 100\%/D_{opt}$. The average % error of the heuristic are 0.27%, 0.22% and 0.25% of the CRD, BRD and overall test problems, respectively. The heuristic produced a solution with optimum in approximate 91% ($=164/180$), 87% ($=157/180$) and 89% ($=321/360$) of the CRD, BRD and overall test problems, respectively.

The heuristic algorithms perform better with BRD test problems in respect of aggregate % error, comparing to CRD test problems. This is probably due to the development of the initial sequence, which is based on flow shop type, and the proportion of BRD test problems possessed the property of flow shop type (97%) is greater than the proportion of CRD test problems (90%). In general, the results indicate that the heuristic is fairly effective in finding an optimal or a near-optimal solution for small-sized problems.

To evaluate the performance of the heuristic algorithms for real-life environment, experiments on six larger test modules are conducted. In practice, the construction units proposed jobs for laying pipes on roads probably include water supply, gas supply, power supply, phone and TV cable companies, and hence, it is reasonable to set the maximal number of jobs as five. In the larger test modules, the number of jobs is set as five and the numbers of sections of road, meaning numbers of processing stages, are set as 8, 10, 12, 15, 18 and 20, respectively. Twenty test problems are generated for each of the test modules, where the processing time follows the previous distributions for CRD and BRD test problems, respectively. Due to the complexity of the addressed problem, to find a precise lower bound is very difficult, and hence, the sum of maximal processing time of each processing stage is used as a benchmark for evaluating the performance of the heuristic. The benchmark, B, is calculated as $B = \sum_{k=1}^{m} \max_{j \in J}\{t_{jk}\}$. We define the percentage error of heuristic from benchmark as $B_{heu} = (D_{heu} - B) \times 100\%/B$, the percentage error of optimum from benchmark $B_{opt} = (D_{opt} - B) \times 100\%/B$. For the numerical example illustrated in Table 4, $B = 33+43+44+41 = 161$ and $B_{heu} = (206 - 161) \times 100\%/161 = 27.95\%$. The value of

Table 5
Computational results of the heuristic

Scales of module		CRD				BRD			
No. of jobs	No. of operations	% Error				% Error			
		Min.	Mean.	Max.	e_0[a]	Min.	Mean.	Max.	e_0[a]
3	3	0	0.00	0.00	20	0	0.00	0.00	20
3	4	0	0.09	1.82	19	0	0.00	0.00	20
3	5	0	0.56	5.41	16	0	0.53	4.42	14
4	3	0	0.00	0.00	20	0	0.00	0.00	20
4	4	0	0.41	6.67	18	0	0.08	0.89	18
4	5	0	0.76	5.08	16	0	0.84	5.92	12
5	3	0	0.00	0.00	20	0	0.00	0.00	20
5	4	0	0.10	2.00	19	0	0.00	0.00	20
5	5	0	0.51	4.26	16	0	0.57	3.87	13
Aggregate		0	0.27	6.67	164	0	0.22	5.92	157

[a] e_0 denotes the number of times that the optimal solution is obtained.

Table 6
Computational experience of the heuristic for further experiments

Scales of module		CRD			BRD		
No. of jobs	No. of operations	B_{heu} (B_{opt})			B_{heu} (B_{opt})		
		Min.	Mean.	Max.	Min.	Mean.	Max.
3	3	0 (0)	19.50 (19.50)	65.22 (65.22)	0 (0)	5.98 (5.98)	16.06 (16.06)
3	4	0 (0)	25.33 (25.19)	76.47 (76.47)	0 (0)	7.38 (7.38)	18.21 (18.21)
3	5	0 (0)	19.54 (18.92)	51.06 (51.06)	0 (0)	6.42 (5.87)	14.83 (14.83)
4	3	0 (0)	19.35 (19.35)	40.00 (40.00)	0 (0)	6.01 (6.01)	17.39 (17.39)
4	4	8.57 (8.57)	37.14 (36.50)	97.22 (94.44)	0 (0)	12.64 (12.55)	20.30 (20.30)
4	5	4.08 (4.08)	31.84 (30.88)	74.36 (74.36)	3.65 (3.65)	12.91 (11.97)	26.03 (26.03)
5	3	12.00 (12.00)	31.82 (31.82)	58.33 (58.33)	2.20 (2.20)	9.53 (9.53)	17.84 (17.84)
5	4	10.81 (10.81)	34.76 (34.61)	72.22 (72.22)	2.91 (2.91)	13.93 (13.93)	29.14 (29.14)
5	5	20.83 (20.83)	42.41 (41.70)	79.07 (74.42)	5.31 (5.31)	17.47 (16.79)	34.44 (33.88)
5	8	11.29	45.73	94.03	5.48	15.36	26.10
5	10	11.76	43.79	86.36	4.29	13.04	21.88
5	12	11.96	44.02	93.06	4.33	10.47	25.75
5	15	16.67	47.76	98.36	5.12	12.25	23.21
5	18	17.91	43.92	80.29	5.18	13.37	23.78
5	20	14.10	47.68	95.65	7.69	15.65	36.38
Aggregate[a]		11.29	45.48	98.36	4.29	13.36	36.38

[a]Includes the six larger test modules only.

B_{opt} equals to the value of B_{heu}. The values of B_{heu} and B_{opt} of the nine small-sized test modules and the values of B_{heu} of the six larger test modules are summarized in Table 6. For the nine small-sized test modules of overall test problems, the values of B_{heu} are close on the values of B_{opt} in maximal, average and minimal percentage error, respectively. The values of B_{heu} and B_{opt} for BRD test problems are much less than the values for CRD test problems, respectively. This is probably due to the processing time generating methods, which generate larger values of B for BRD test problems than for CRD test problems.

For the six larger test modules of the CRD test problems, the average value of B_{heu} is 45.48%, which is close on 42.41% of the five-job, five-operation test module. The maximal value of B_{heu} is 98.36%, which is close on 97.22% of the four-job, four-operation test module. For the six larger test modules of the BRD test problems, the average value of B_{heu} is 13.36%, which is close on 17.47% of the five-job, five-operation test module. The maximal value of B_{heu} is 36.38%, which is close on 34.44% of the five-job, five-operation test module. Computational results demonstrate the potential of the heuristic to efficiently deal with the scheduling problems in real-life environment.

7. Conclusions

In this paper, we have addressed an open shop scheduling problem with the characteristics of movable dedicated machines and no-wait restriction. With a consideration of efficiency, the m operations of each job should be processed continuously on the m sections of road meaning no-wait restriction, and hence, the dedicated machines will be shifted to the pre-determined sections of road immediately as the need arises. It is possible for different jobs to be processed simultaneously on the same section of road. The objective of the scheduling problem is to minimize the total occupation time for all the processing stages meaning to reduce the period of social costs such as air pollution, traffic congestion and others. Some two-phase heuristic algorithms are proposed for solving the problem. An initial sequence is developed according to flow shop type in the first phase, and then, the iterative improvement procedure is performed in the second phase. The heuristic produced an optimum solution in approximate 89% of the 360 test problems with the average and maximum % error of 0.25% and 6.67%, respectively. Computational results show that the heuristic is fairly effective in finding an optimal or a near-optimal solution for small-sized problems. Results of the heuristic for experiments

tallied with the real-life environment demonstrate the potential of the heuristic to efficiently deal with the scheduling problems.

In practice, section chiefs frequently encounter problems related to people's livelihood such as road construction works that include laying pipes for gas, water and phone can be dealt with by this approach. The period of social costs such as air or noise pollution, traffic congestion caused by road works can be reduced by adequate arrangement of road construction works that decreases the total occupation time. In this paper, some two-phase heuristic algorithms that can reach satisfied solutions within reasonable execution time are developed and can be applied to the real-life problems.

Acknowledgments

The authors wish to thank the referees for their suggestions. This research was supported by the National Science Council (NSC) of the Republic of China under Grant NSC 93-2416-H-167-003.

References

Adiri, I., Aizikowitz, N., 1989. Open-shop scheduling problems with dominated machines. Naval Research Logistics 36, 273–281.

Adiri, I., Amit, N., 1984. Openshop and flowshop scheduling to minimize sum of completion times. Computers and Operations Research 11, 275–284.

Gonzalez, T., Sahni, S., 1976. Open shop scheduling to minimize finish time. Journal of the Association for Computing Machinery 23, 665–679.

Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G., 1979. Optimization and approximation in deterministic sequencing and scheduling: A survey. Annals of Discrete Mathematics 5, 287–326.

Gupta, J.N.D., Werner, F., Wulkenhaar, G., 2003. Two-machine open shop scheduling with secondary criteria. International Transactions in Operational Research 10, 267–294.

Konno, T., Ishii, H., 2000. An open shop scheduling problem with fuzzy allowable time and fuzzy resource constraint. Fuzzy Sets and Systems 109, 141–147.

Kyparisis, G.J., Koulamas, C., 2000. Open shop scheduling with makespan and total completion time criteria. Computers and Operations Research 27, 15–27.

Lauff, V., Werner, F., 2004. On the complexity and some properties of multi-stage scheduling problems with earliness and tardiness penalties. Computers and Operations Research 31, 317–345.

Liaw, C.F., 2000. A hybrid genetic algorithm for the open shop scheduling problem. European Journal of Operational Research 124, 28–42.

Liaw, C.F., 2003. An efficient tabu search approach for the two-machine preemptive open shop scheduling problem. Computers and Operations Research 30, 2081–2095.

Liaw, C.F., 2004. Scheduling two-machine preemptive open shops to minimize total completion time. Computers and Operations Research 31, 1349–1363.

Liaw, C.F., 2005. Scheduling preemptive open shops to minimize total tardiness. European Journal of Operational Research 162, 173–183.

Liaw, C.F., Cheng, C.Y., Chen, M., 2005. Scheduling two-machine no-wait open shops to minimize makespan. Computers and Operations research 32, 901–917.

Lin, H.T., Liao, C.J., 2003. A case study in a two-stage hybrid flow shop with setup time and dedicated machines. International Journal of Production Economics 86, 133–143.

Noda, A.S., Alcaide, D., Martin, C.G., 2006. Network flow approaches to pre-emptive open-shop scheduling problems with time-windows. European Journal of Operational Research 174, 1501–1518.

Pinedo, M., 1995. Scheduling: Theory, Algorithms, and Systems. Prentice-Hall, Englewood Cliffs, NJ.

Sahni, S., Cho, Y., 1979. Complexity of scheduling shops with no-wait in process. Mathematics of Operations Research 4, 448–457.

Schaller, J., 2001. A new lower bound for the flow shop group scheduling problem. Computers and Industrial Engineering 41, 151–161.

Sidney, J.B., Sriskandarajah, C., 1999. A heuristic for the two-machine no-wait openshop scheduling problem. Naval Research Logistics 46, 129–145.

Subai, C., Baptiste, P., Niel, E., 2006. Scheduling issues for environmentally responsible manufacturing: The case of hoist scheduling in an electroplating line. International Journal of Production Economics 99, 74–87.

Suliman, S.M.A., 2000. A two-phase heuristic approach to the permutation flow-shop scheduling problem. International Journal of Production Economics 64, 143–152.

Yao, M.J., Soewandi, H., 2000. Simple heuristics for the two-machine openshop problem with blocking. Journal of the Chinese Institute of Industrial Engineers 17, 537–547.