Sun Yat-Sen Management Review
Vol.3, No.4, December 1995

開放式排程遲延成本之啟發解
A Heuristic Solution To The Total Tardy Cost Of An M-Machine Nonpreemptive Open-Shop Schedule

吳雪馥 *Hsueh-foo Lin*
國立屏東商業專科學校
National Ping Tung Institute of Commerce
(Received June 1995, revised November 1995, accepted December 1995)

摘　要

物流中心之撿貨系統與汽車測試中心之測試作業皆為一開放式排程，開放式排程之別於流程式與變序式排程乃是其各工作之操作順序無限制。本研究嘗試以數學規劃模式來描述一 n 件貨品工作訂單，m 步操作作業程序之開放式排程中遲延罰金成本最小化，再以擴張樹之網路結構簡化數學規劃模式，並發展一閒置時間法則作為排程決策評估選擇之基準。本研究嘗試採用貨品工作訂單數、操作作業程序數、貨品工作訂單遲延因數和預定交貨期分布情況不同之十五種實驗組合中 7,200 個案，來比較本研究發展之啓發式解法與最小作業時間作業先排法、最小預定交貨期作業先排法、最小總寬裕工作作業時間作業先排法、最小寬裕作業時間作業先排法四種派工法之績效。實驗結果顯示本研究發展之啓發式解法比另四種派工法好出甚多，本研究發展之啓發式解法非常適用於多工作、多機和高遲延因數之開放式排程。貨品工作訂單遲延因數是此啓發式解法績效之主要影響因素，預定交貨期分布則較為次之。

關鍵字: 數學規劃，網路模式，開放式排程

Abstract

This open-shop schedule has no restriction on the processing order of the jobs. The problem for minimizing the total tardy cost in an m-machine nonpreemptive open-shop is NP-hard. A network structure based on the spanning tree is constructed in this work to simplify the mathematical model of a nonpreemptive open-shop problem whose objective is to minimize total tardy cost. The idle time rule developed in this work serves as the decision criterion to test the performance. The proposed Heuristic A is compared with SPT, EDD, SLACK, and MWSTR/EST (Minimum Weighted Slack Time Remaining rule and Early Starting Time rule) methods using 7,200 cases, which are constructed by different features. Experimental results indicate that Heuristic A performs much better than the other four methods under the total tardy cost criterion and even better for complex problems. The results also reveal that the tardiness factor has significant influence on Heuristic A, and the due-date range has minor influence.

Key Words : Mathematical Programming, Network Model, Open-shop, Scheduling.

1. INTRODUCTION

Open-shop schedules differ from flow-shop and job-shop schedules in that no restrictions are placed on the processing order for any job in the open-shop.

The open-shop scheduling problem can be described as follows. Suppose a number of jobs must be performed, each of which consists of m operations, by using m processors, and there is no prescribed order for any job using these processors. In many situations, operations making up a job can be performed in any order. For example, consider an order picking system in a physical distribution center. Each new order from contracted stores may require to pick products from different stock shelves. These products picking, however, may be carried out in any sequence. In this particular example, the manager may wish to schedule the picking to meet due dates of customers' orders. Minimizing total tardy cost in the nonpreemptive open-shop problem is a problem particularly prevalent in testing and repair, even though it has not been extensively studied.

Each job is processed by one processor at a time, and each processor executes one job at a time. Let n denote the number of jobs and p_{ij} be the time to process job i on processor j. Given the results of performance by which the cost of each possible solution can be measured, finding a feasible schedule that minimizes the corresponding total tardy cost is of primary concern. Using the three-field notation of Graham et al.[5], minimizing tardy cost in an open-shop with an arbitrary number of machines is denoted by $O//\Sigma\alpha_i T_i$, where α_i is the unit cost of tardiness penalty for job i. The NP-hardness of the minimizing maximum lateness problem in a two-machine nonpreemptive open-shop ($O2//L_{\max}$) has been established by Lawler, Lenstra and Rinnooy Kan[6]. Graham, Lawler, Lenstra and Rinnooy Kan have mentioned that $O2//L_{\max}$ is a reducible case of $O//\Sigma T_i$, and $O//\Sigma T_i$ is a special case of $O//\Sigma\alpha_i T_i$ for α_i all equals to 1, so $O//\Sigma\alpha_i T_i$ is obviously NP-hard and a heuristic approach to obtain a good solution is recommended by the knowledge of problem complexity [2].

Job-shop scheduling is among the most difficult combinatorial problems. Removing the process sequencing constraints from the job-shop that yields the seemingly simpler open-shop. Previous studies by Gonzalez and Sahni[3], Gonzalez[4], and Adiri and Amit[1] have revealed that some NP-complete job-shop problems become efficiently solvable by removing the ordering constraints. When ordering constraints are relaxed, however, the number of feasible schedules increases enormously. For example, there are approximately $4\text{x}10^{20}$ feasible schedules for a five-job, five-processor open-shop problem compared to approximately $2\text{x}10^{10}$ feasible schedules for an equivalent job-shop problem. Furthermore, if a job-shop problem could be solved in one second by complete enumeration using a high speed computer, the same computer would take about 6.4 centuries to solve the open-shop problem. Obviously, the problem of finding optimal open-shop schedule by enumeration is even more difficult than that of the job-shop. Therefore, open-shop scheduling has a simple problem structure, but a complex solution structure.

Minimizing total tardiness in the nonpreemptive open-shop problem is a problem particularly prevalent in testing and repair, even though it has not been extensively studied. Liu and Bulfin[7] developed polynomial algorithms to minimize the total tardiness and the number of jobs causing tardiness in the nonpreemptive open-shop with identical processing time jobs. The purpose of this research is to evaluate the efficacy of five heuristics in minimizing the total tardy cost in a nonpreemptive open-shop with an arbitrary number of

processor. A network structure based on the spanning tree was constructed to simplify the mathematical model of this NP-hard problem and an idle time rule is developed to serve as the decision criterion to test the performance. More specifically, the objectives of this research are (1) to develop a new heuristic that exploits the problem structure, and (2) to conduct empirical evaluation of the proposed heuristic in comparison with SPT (job priority equals processing time of the imminent operation), EDD (job priority equals its due-date), SLACK (job priority equals the time remaining before the job due-date minus all remaining processing for the job), and the MWSTR/EST (a mixture of the minimum time remaining before the job due-date/ the weighted tardiness rule and early starting time rule) methods using a large number of simulated cases, that consist of different features, such as the number of jobs, the number of processors, tardiness factor, and the range of due date.

The rest of the paper is organized as follows: a mathematical model for determining the total tardy cost is presented in section two to provide an explicit statement of the problem, and the new Heuristic A is proposed in section three. Computational evaluation is described in section four, and conclusions are given in section five.

2. AN N-JOB, M-PROCESSOR OPEN-SHOP

The orders picking system in a physical distribution center is a typical open-shop problem. The orders from contracted stores can be treated as jobs, the clerks can be treated as the processors, the operations of picking the orders from a line of stock shelves by a clerk can be treated as a task. The sequence in picking products from the lines of stock shelves by clerks can be immaterial. Other examples of open shop scheduling include the teacher-class assignment problem, examination scheduling, and some operations of manufacturing testing and repair.

The open-shop problem can be stated as follows. There are n jobs ($J_1, J_2, \cdots\cdots, J_n$) which have to be processed by m processors ($P_1, P_2, \cdots\cdots, P_n$). Each job has m tasks associated with it. For $i = 1$, ..., n and $j = 1$, ..., m, task j of job i is to be executed by P_j. The execution time for task j of job i is given, and is denoted by p_{ij}. We assume the execution times to be nonnegative integers. The m tasks of a job may be processed in any order, but two of them must not be executed at the same time. All tasks must be executed without interruption. Let task ij be the operation of job i executing on processor j. A 4-job, 3-processor open shop can be illustrated as the disjunctive graph structure in Figure 1. The problem in this research is to find, for each jobs, the schedule of operation that obeys the constraints of the problem and minimizes a given objective function the total tardy cost. Before formulating the model, some notations are introduced as follows:

s_{ij} = the start time of job i on processor j,
p_{ij} = the processing time of job i on processor j,
d_i = the due date of job i,
T_i = the tardiness of job i,

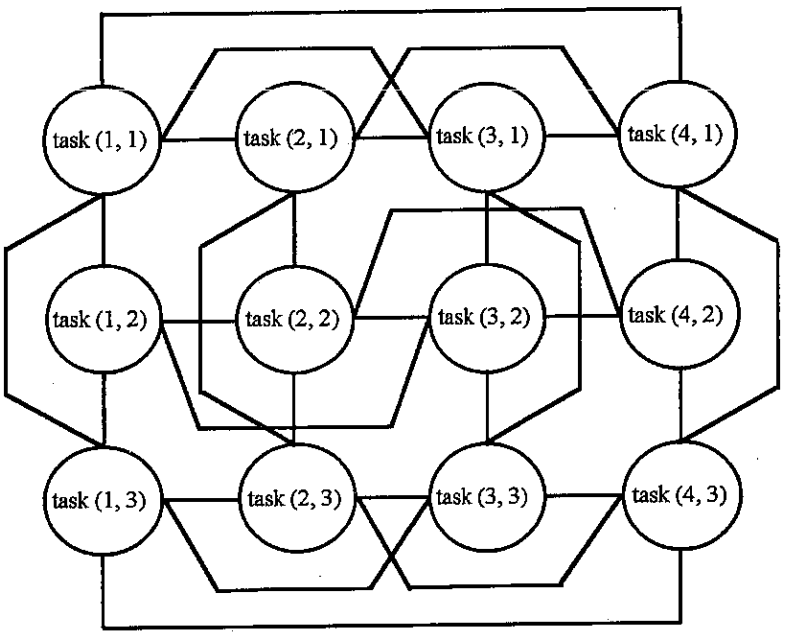


Figure 1. Task relationships within the 4-job, 3-processor open shop

α_i = the unit cost of tardiness penalty for jobi, and
H = the upper bound on makespan.

Indicator variables of an operation sequence are set as following:

$$X_{ikj} = \begin{cases} 1, \text{ if job i precedes job k on processor j,} \\ 0, \text{ otherwise.} \end{cases}$$

$$Y_{ijh} = \begin{cases} 1, \text{ if task j precedes task h of job i,} \\ 0, \text{ otherwise.} \end{cases}$$

The n-job, m-processor open-shop in the total tardy cost model is to be formulated in Model 1 as follows:

[Model 1]

Minimize $\sum_{i=1}^{n} \alpha_i T_i$

s.t.

$$s_{ij} + p_{ij} - d_i \le T_i \qquad 1 \le i \le n, 1 \le j \le m \tag{1}$$

$$s_{kj} - s_{ij} + H(1 - X_{ikj}) \ge p_{ij} \qquad 1 \le i \le k \le n, 1 \le j \le m \tag{2}$$

$$s_{ij} - s_{kj} + HX_{ikj} \ge p_{kj} \qquad 1 \le i \le k \le n, 1 \le j \le m \tag{3}$$

$$s_{ih} - s_{ij} + H(1 - Y_{ijh}) \ge p_{ij} \qquad 1 \le i \le n, 1 \le j \le h \le m \tag{4}$$

$$s_{ij} - s_{ih} + HY_{ijh} \ge p_{ih} \qquad 1 \le i \le n, 1 \le j \le h \le m \tag{5}$$

$$s_{ij} \geq 0 \quad 1 \leq i \leq n, 1 \leq j \leq m$$
$$T_i \geq 0 \quad 1 \leq i \leq n, 1 \leq j \leq m$$
$$X_{ikj}, Y_{ijh} \in \{0,1\} \quad 1 \leq i \leq k \leq n, 1 \leq j \leq h \leq m$$

This model closely reflects the disjunctive graph structure as found in Figure 1. Constraints (2) and (3) (also (4) and (5)) are pairwise disjoint constraints because exact one of the pair holds. There are mn(n+m-1) constraints, n(m+1) linear variables, and mn(n+m-2)/2 indicator binary variables in this model. For a small size problem, e.g., 4 jobs and 3 machines, it comes out that the formulation has 72 constraints and 46 variables. The problem can be solved by fixing the indicator binary variables using a branch-and-bound method. Given a partial solution of fixed indicator variables, a lower bound can be calculated by either relaxing the free integer variables or aggregating the constraints to a single linear constraint. However, these bounds are either too weak to fathom a partial solution or too expensive to be computed. Thus, the computational results appear to be particularly discouraging for the branch-and-bound method.

The problem can be simplified by Benders' Partitioning. All indicator variables are assumed to be fixed, and the dual variables of each constraint in Model 1 are permitted to be $w_{ij}, u_{ikj}, u'_{ikj}, v_{ijh}$, and$v'_{ijh}$, respectively. Model 1 can then be rewritten as Model 2.

[Model 2]

$$\text{Min Max } \Sigma_{i=1}^{n}\Sigma_{j=1}^{m} \alpha_i W_{ij}(p_{ij} - d_i) +$$
$$\Sigma_{i=1}^{n}\Sigma_{k=i+1}^{n}\Sigma_{j=1}^{m} u_{ikj}[p_{kj} - H(1 - X_{ikj})] + \Sigma_{i=1}^{n}\Sigma_{k=i+1}^{n}\Sigma_{j=1}^{m} u'_{ikj}[p_{kj} - HX_{ikj}] +$$
$$\Sigma_{i=1}^{n}\Sigma_{j=1}^{m}\Sigma_{h=j+1}^{m} v_{ijh}[p_{ij} - H(1 - Y_{ijh})] + \Sigma_{i=1}^{n}\Sigma_{j=1}^{m}\Sigma_{h=j+1}^{m} v'_{ijh}[p_{ih} - HY_{ijh}]$$

s.t.

$$\Sigma_{j=1}^{m} w_{ij} \leq 1 \quad \forall i,$$
$$-w_{ij} - \Sigma_{k=i+1}^{n} u_{ikj} + \Sigma_{k=i+1}^{n} u'_{ikj} + \Sigma_{k=1}^{i-1} u_{ikj} - \Sigma_{k=1}^{i-1} u'_{ikj}$$
$$-\Sigma_{h=j+1}^{m} v_{ijh} + \Sigma_{h=j+1}^{m} v'_{ijh} + \Sigma_{h=1}^{j-1} v_{ijh} - \Sigma_{h=1}^{j-1} v'_{ijh} \leq 0 \quad \forall i, j,$$
$$w_{ij} \geq 0 \quad \forall i, j,$$
$$u_{jkj}, u'_{ikj} \geq 0 \quad 1 \leq i \leq k \leq n, 1 \leq j \leq m$$
$$v_{ijh}, v'_{ijh} \geq 0 \quad 1 \leq i \leq n, 1 \leq j \leq h \leq m$$

Since H is a very large value, the following conditional statements can be set for simplifying Model 2 via:

if $X_{ikj} = 1$, then $u'_{ikj} = 0$;
if $X_{ikj} = 0$, then $u_{ikj} = 0$;
if $Y_{ijh} = 1$, then $v'_{ijh} = 0$; and
if $Y_{ijh} = 0$, then $v_{ijh} = 0$.

Next, some parts of the objective function can be deleted because the following equations are equal to zero.

$$\Sigma_{i=1}^{n}\Sigma_{k=i+1}^{n}\Sigma_{j=1}^{m} u_{ikj}H(1 - X_{ikj}) + \Sigma_{i=1}^{n}\Sigma_{k=i+1}^{n}\Sigma_{j=1}^{m} u'_{ikj}HX_{ikj} = 0$$

$$\Sigma_{i=1}^{n}\Sigma_{j=1}^{m}\Sigma_{h=j+1}^{m} v_{ijh}H(1-Y_{ijh}) + \Sigma_{i=1}^{n}\Sigma_{j=1}^{m}\Sigma_{h=j+1}^{m} v'_{ijh}HY_{ijh} = 0$$
$$u_{ikj} * u'_{ikj} = 0 \quad 1 \le i \le k \le n, 1 \le j \le m$$
$$v_{ijh} * v'_{ijh} = 0 \quad 1 \le i \le n, 1 \le j \le h \le m$$

Therefore, Model 2 can be rewritten as Model 3:

[Model 3]

$$\text{Min Max } \Sigma_{i=1}^{n}\Sigma_{j=1}^{m} \alpha_i W_{ij}(p_{ij} - d_i) +$$
$$\Sigma_{i=1}^{n}\Sigma_{k=i+1}^{n}\Sigma_{j=1}^{m} u_{ikj}p_{ij} + \Sigma_{i=1}^{n}\Sigma_{k=i+1}^{n}\Sigma_{j=1}^{m} u'_{ikj}p_{kj} +$$
$$\Sigma_{i=1}^{n}\Sigma_{j=1}^{m}\Sigma_{h=j+1}^{m} v_{ijh}p_{ij} + \Sigma_{i=1}^{n}\Sigma_{j=1}^{m}\Sigma_{h=j+1}^{m} v'_{ijh}p_{ih}$$

s.t.

$$\Sigma_{j=1}^{m} w_{ij} \le 1 \quad \forall i,$$
$$-w_{ij} - \Sigma_{k=i+1}^{n} u_{ikj} + \Sigma_{k=i+1}^{n} u'_{ikj} + \Sigma_{k=1}^{i-1} u_{ikj} - \Sigma_{k=1}^{i-1} u'_{ikj}$$
$$-\Sigma_{h=j+1}^{m} v_{ijh} + \Sigma_{h=j+1}^{m} v'_{ijh} + \Sigma_{h=1}^{j-1} v_{ijh} - \Sigma_{h=1}^{j-1} v'_{ijh} \le 0 \quad \forall i, j$$
$$u_{ikj} * u'_{ikj} = 0 \quad 1 \le i \le k \le n, 1 \le j \le m$$
$$v_{ijh} * v'_{ijh} = 0 \quad 1 \le i \le n, 1 \le j \le h \le m$$
$$w_{ij} \ge 0 \quad \forall i, j$$
$$u_{ikj}, u'_{ikj} \ge 0 \quad 1 \le i \le k \le n, 1 \le j \le m$$
$$v_{ijh}, v'_{ijh} \ge 0 \quad 1 \le i \le n, 1 \le j \le h \le m$$

The maximum objective function ensures that a busy schedule is generated and the minimum maximum objective function ensures that a busy schedule with minimum tardy cost is generated. Model 3 is a nonlinear programming formulation with n(m+1) linear constraints, mn(m+n-2)/2 nonlinear constraints, and mn(m+n-1) real variables. For a small size problem, e.g., 4 jobs and 3 processors, Model 3 requires 16 linear constraints, 30 nonlinear constraints, and 72 real variables to be constructed. The mn(m+n-2)/2 nonlinear constraints indicate that only one of the disjoint arcs in Figure 1 is selected. This causes the problem to be computationally difficult. A new heuristic algorithm A is proposed in the next section on the basis of the formulation of minimizing the total tardy cost in an open-shop with arbitrary numbers of jobs and processors.

3. DEVELOPING A HEURISTIC ALGORITHM

Model 3 can be transformed into a network model. An example of a 3-job, 2-processor open-shop with minimizing the total tardy cost is shown in Figure 2. The (m x n) task nodes and n due date nodes represent the constraints, the (m x n) directed arcs represent w_{ij}, and the mn(m+n-2) directed arcs represent $u_{ikj}, u'_{ikj}, v_{ijh}$, and v'_{ijh}, respectively. The information on the node-arc in the network can be stored in a binary adjacent matrix to conserve the memory

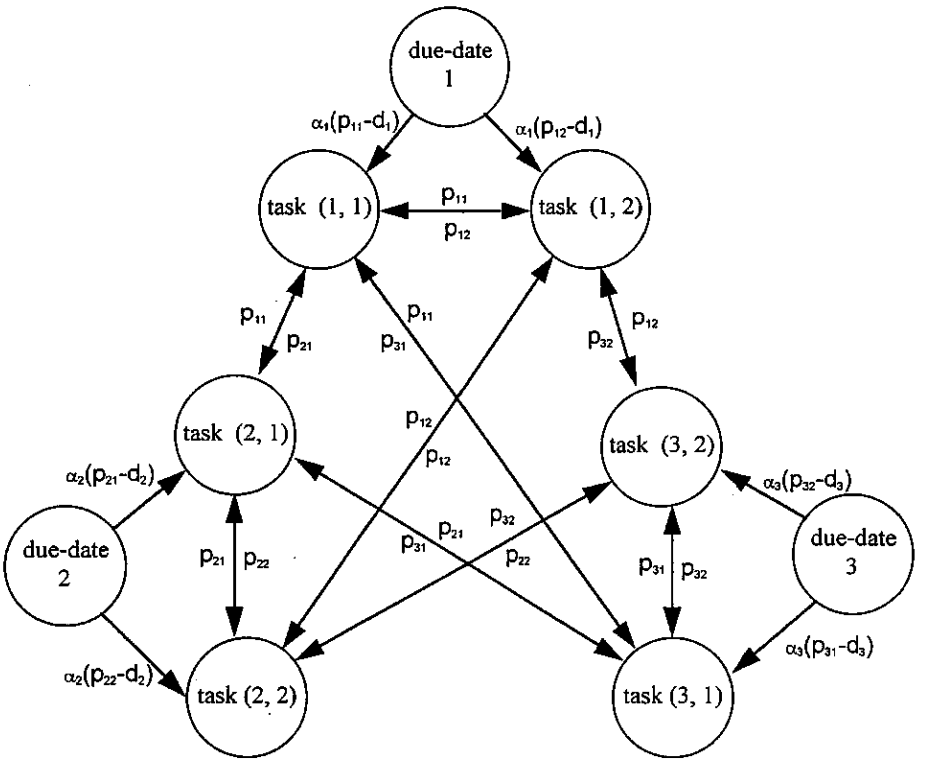


Figure 2. The network model for 3-job, 2-processor open shop in Model 3

space for computation. The cost of each arc is the coefficient of the objective function. The objective of Model 3 is to select the combination of arcs that links all of the (m x n) task nodes and n due-date nodes in the network to minimize the total tardy cost. Figure 2 reveals that a network model can be used to represent an n-job, m-processor open-shop problem, so the properties of a spanning tree can be used to search the local optimal of such open-shop network by adding a dummy root node. The first level of the spanning tree (level 0) consists of only one node termed as root, which is a dummy node. Below level 0, for $n \geq m$, at most m different task nodes can be assigned; otherwise, at most n different task nodes can be scheduled on the same level t. Each task node may have 0, 1, or 2 successors. Of the two possible successors of a task node, one must have the same job index as its predecessor and the other one must have the same processor index as its predecessor. After constructing a complete scheduling tree with a root node and (m x n) task nodes, n due date nodes are attached to the last task nodes of each job. Therefore, the network of an n-job, m-processor problem achieves a spanning tree. The Gantt chart and the tree structure of a 3-job, 3-processor feasible schedule are displayed in Figure 3 and Figure 4.

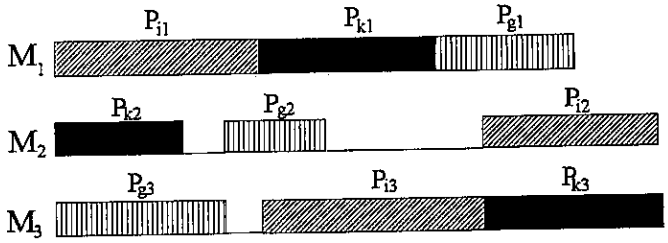


Figure 3. Gantt chart of a 3-job, 3-processor feasible schedule

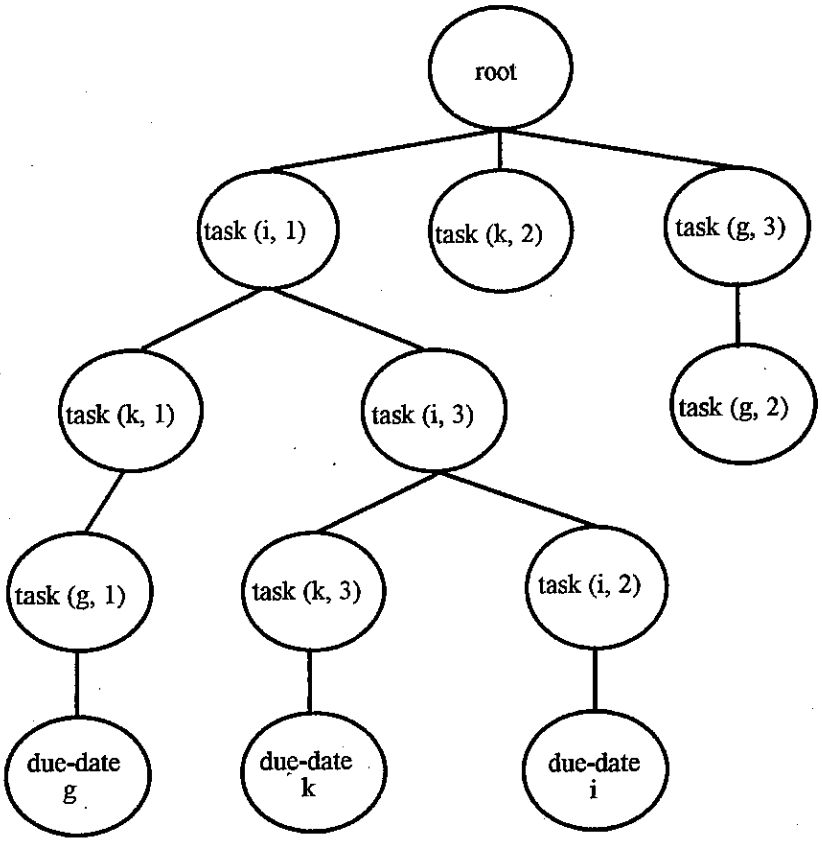


Figure 4. Tree structure of a 3-job, 3-processor feasible schedule

For minimizing the tardy cost of a job is to expand the scalar of the spanning tree to cut the height of the spanning tree, so an idle time rule is developed to make decision. This rule states that when a processor becomes free at level t of a scheduling search tree, schedule the next task(i,j) with the minimum idle time. The idle time between task node(i,j) and (i,h) is defined by I_{ij}. Task node(i,j) is the task to be selected at level t, and task node (i,h) is the task of the same job i selected just before the node(i,j). C_i denotes the finishing time of job i. The total tardy cost can be formulated as below:

$$T=\Sigma_{i=1}^{n}\alpha_i T_i=\Sigma_{i=1}^{n}\alpha_i\{\max(0,C_i-d_i)\}=\Sigma_{i=1}^{n}\alpha_i\{\max\{0,[(\Sigma_{j=1}^{m}p_{ij}+\Sigma_{j=1}^{m}I_{ij})-d_i]\}\};$$

The value of I_{ij} is the critical factor of the total tardiness T, and d_i is the indicator of the urgency of the jobs. Therefore, they are both used to compute the priorities for evaluating the alternatives of a partial scheduling tree; ω_{ij} denotes the idle time of an unassigned task(i,j) which will be postponed to level t. Three priorities must be computed on the basis of the idle time rule for evaluating the alternatives of a partial scheduling tree.

The idle times among the task nodes in Figure 4 are as follows:

Level 0-1. $I_{i1}=I_{k2}=I_{g3}=0$

Level 1-2. $I_{i3}=0,\ I_{k1}=p_{i1}-p_{k2},\ I_{g2}=0$

Level 2-3. $I_{i2}=0,\ I_{k3}=(\omega_{i3}+p_{i3})-(\omega_{k1}+p_{k1}),\ I_{g1}=(\omega_{k1}+p_{k1})-(\omega_{g2}+p_{g2})$

The priorities are dependent on the level of a partial scheduling search tree and must be computed for each pair of tasks. The following variables are defined as :

t= the level of a scheduling search tree, t $\geq$ 1,

p_{ij}= the executing time of job i on processor j,

d_i= the due date of job i,

α_i= the unit cost of tardiness penalty for job i,

ω_{ij}= the idle time of an unscheduled task(i,j) which will be postponed to level t,

z_{ij}= the slack time or tardy time of an unscheduled task(i,j) at level t, or slack time or tardy time of a scheduled task(i,j),

$Pr_t1[i,j]$ = the total tardy cost of other unscheduled tasks(k,j) when the unscheduled task(i,j) is selected at level t,

$Pr_t2[(i,j),(k,h)]$ = the slack time or the tardy penalty of job k when the first scheduled task(i,j) and the unscheduled task(k,h) are both selected at level t,

$Pr_t3[(i,j),(k,h)]$ = the slack time or the tardy penalty of job k when the first scheduled task(i,j) is selected at level t but the unscheduled task(k,h) is not,

$e_t[(i,j),(k,h)]$ = the possible link between task(i,j) and task(k,h) at level t.

The value of $e_t[(i,j),(k,h)]$ is set to 1 when task(i,j) can be linked with task(k,h) in its network; otherwise, the value is set to 0. Two tasks in an open-shop can be linked together only when i = k or j = h.

The functions for calculating the priorities on Heuristic A are derived as below:

$Pr_t1[i,j] = \Sigma_{\forall k}\alpha_k \max\{0, z_{kj} + |(\omega_{ij} + p_{ij}) - \omega_{kj}|\}, when\ e_t[(i,j),(k,j)] = 1$

$Pr_t2[(i,j),(k,h)] = z_{ij} + z_{kh} + |(\omega_{ij} + p_{ij}) - (\omega_{kh} + p_{kh})|, when\ e_t[(i,j),(k,h)] = 0$

$Pr_t3[(i,j),(k,h)] = z_{ij} + z_{kh} + |(\omega_{ij} + p_{ij}) - \omega_{kh}|, when\ e_t[(i,j),(k,h)] = 0$

If the value of $Pr_t2[(i,j),(k,h)]$ is greater than 0, then let $Pr_t2[(i,j),(k,h)] = \alpha_k Pr_t2[(i,j),(k,h)]$.

If the value of $Pr_t3[(i,j),(k,h)]$ is greater than 0, then let $Pr_t3[(i,j),(k,h)] = \alpha_k Pr_t3[(i,j),(k,h)]$.

The idle time rule is used to select the next m tasks from different jobs to be processed at level t of a partial scheduling search tree. The procedure of the proposed Heuristic A is:

Step 1. Initially, the shop schedule is empty and all jobs are unscheduled. A binary adjacent matrix **E** is used to store the information of the links between two tasks, a matrix **Z** is used to store the slack time or tardy time of each task at level t, and a matrix **W** is used to update the idle time of an unscheduled task (i,j) which will be postponed to level t.

Step 2.1. Evaluate the alternatives of the first task at level t of the partial scheduling search tree by computing the value of $Pr_t1[i,j]$ for each unscheduled task (i,j).

Step 2.2. Select the alternatives of the first task at level t of the partial scheduling search tree. Select a task (i,j) with the minimum value of $Pr_t1[i,j]$ and add this task to the shop schedule. Term the task "scheduled". If there is still an unscheduled task existed, then go to Step 3.1; otherwise, execute Step 4 to compute the total tardy cost. If a tie exists, the EDD (Earliest Due Date first) rule is used. The LPT (Longest Processing Time first) rule is used to break the tie of the same due date.

Step 3.1. Evaluate the alternatives of the second to the mth tasks at level t of the partial scheduling search tree. Compute the values of $Pr_t2[(i,j),(k,h)]$ and $Pr_t3[(i,j),(k,h)]$ for each pair of the first selected task (i,j) in Step 2.2 and unscheduled tasks (k,h).

Step 3.2. Select the alternatives of the second to the mth tasks at level t of the partial scheduling search tree. Select an unscheduled task (k,h) with the maximum value of $Pr_t3[(i,j),(k,h)]$ and add this task to the shop schedule. Term the task "scheduled". If the number of selected task nodes at level t is equal to m (the size of processors), go to Step 2.1; otherwise, repeat Step 3.2. If task (k,h) has a value of $Pr_t2[(i,j),(k,h)]$ which is greater than the value of $Pr_t3[(i,j),(k,h)]$, then it will not be selected at level t. After that another task (k,h) with the next largest value of $Pr_t3[(i,j),(k,h)]$ is to be chosen. If there is a tie in the value of $Pr_t3[(i,j),(k,h)]$, task (k,h) with the minimum value of $Pr_t2[(i,j),(k,h)]$ is selected. Repeat this step m-1 times.

Step 4. When the unscheduled task is empty, compute the total tardy cost. $T = \Sigma_{i=1}^{n}\alpha_i T_i = \Sigma_{i=1}^{n}\alpha_i\{\max_{\forall 1}(0, z_{ij})\}$.

Table 1. The data of a 4-job, 3-clerk open shop in the total tardy cost model

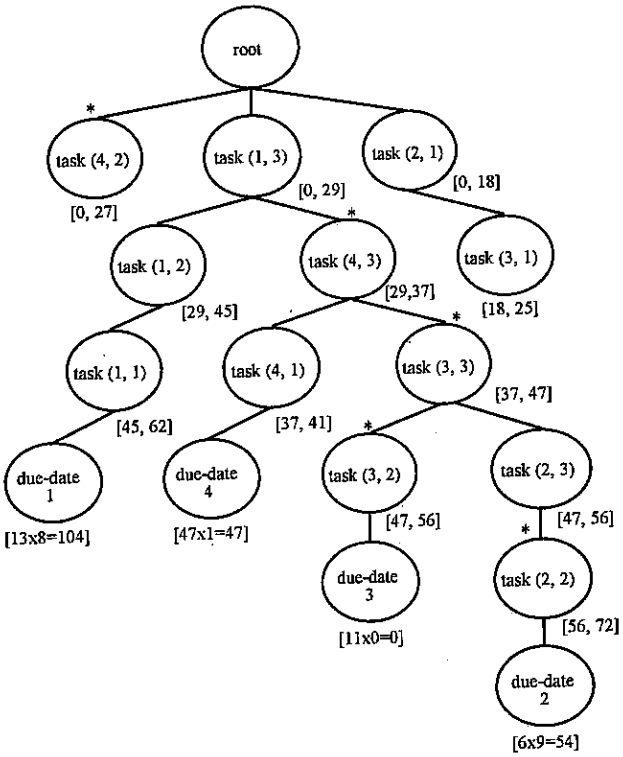
Clerk (j)	Job (i)			
	1	2	3	4
1	17	18	17	4
2	16	16	9	27
3	29	9	10	8
Due-date	54	63	63	40
Tardy cost	13	6	11	47

For example, Table 1 the processing time, due-date, and tardiness unit cost of a 4-job, 3-processors open shop. A tree schedule of the problem can be constructed by the procedures of the proposed Heuristic A:

1. Step 1. Initialize all data.
2. Step 2.1. Evaluate the value of $Pr_t1[i,j]$ for twelve unassigned task nodes.
3. Step 2.2. The value of $Pr_t1[i,j]$ of task (4, 2) is minimum, so task (4, 2) is selected as the first task in Level 1. It is marked as "*" in Figure 5. The start time and the finished time of task (4, 2) are shown in Figure 5 as [0,27].
4. Step 3.1. Evaluate the value of $Pr_t2[(4,2),(k,h)]$ and $Pr_t3[(4,2),(k,h)]$ for each pair of the first selected task (4, 2) and unassigned task (1, 1), task (1, 3), task (2, 1), task (2, 3), task (3, 1), and task (3, 3).
5. Step 3.2. Task (1, 3) has the maximum value of $Pr_t3[(4,2),(k,h)]$, so it is selected as the second task node at level 1.
6. Step 3.1. Evaluate the value of $Pr_t2[(1,3),(k,h)]$ and $Pr_t3[(1,3),(k,h)]$ for each pair of the previous selected task (1, 3) and unassigned task (3, 1) and task (2, 1).
7. Step 3.2. Task (2, 1) has the maximum value of $Pr_t3[(1,3),(k,h)]$, so it is selected as the third task node at Level 1. The number of task nodes reachs 3, so go to Step 2.1.
8. Step 2.1. Evaluate the value of $Pr_t1[i,j]$ for nine unassigned task nodes.
9. Step 2.2. The value of $Pr_t1[i,j]$ of task (4, 3) is minimum, so task (4, 3) is selected as the first task in Level 2.

As the procedure goes on, a tree structure of this example can be constructed as shown in Figure 5. The total tardiness penalty is 205. An optimal solution of 157 units tardy cost was run by Super LINDO. It took more than 58,000 iterations (about 8 minutes) to reach the optimal solution. Comparing with the solutions produced by SPT (2,227), EDD (770), SLACK (2,810), and MWSTR/EST (614) methods, the result of the proposed Heuristic A is the best among the five methods.

In Heuristic A, there are (n x m) "unscheduled" tasks at the beginning of level 1, so mn(m+n-2) calculations and comparisons are required to select



Total tardy cost = 104+54+0+47 = 205

Figure 5. Tree structure of a 4-job, 3-clerk schedule in the total tardy cost model

the minimum value of $Pr_t1[i,j]$ in Step 2.1 and Step 2.2 for assigning the first selected task (i,j) at level 1. Since there are (n-1)(m-1) "unassigned" tasks that can be scheduled with the first selected task (i,j) at level 1, (m-1)(mn-n-m+1) calculations and comparisons are required to select the maximum value of $Pr_t3[(i,j),(k,h)]$ in Step 3.1 and Step 3.2 for assigning m-1 task nodes (k,h) at level 1. Because the numbers of unscheduled task node decrease from nm to 0 by m at each level, the procedure requires n{[mn(m+n-2)] + [(m-1)(mn-n-m+1)]} calculations and comparisons, and it works out that the complexity for Heuristic A is $O(n^3m)$.

4. COMPUTATIONAL RESULTS

4.1 Design of the Test Data

4.1.1 Shop Layout

The shop layout defines the set of clerks in the open-shop. Four scenarios are:

(1). two clerks shop,
(2). three clerks shop,
(3). four clerks shop, and
(4). five clerks shop.

4.1.2 Job Operation Times

All the data, processing times and due-dates, are integers, and all the jobs arrive simultaneously at time 0. The operation time of job i on clerk j, p_{ij}, is distributed uniformly within the range [1,30], i.e., $p_{ij} \sim$ U(1,30). The number of jobs in each set of tests, n, is 5, 10, 20, 30, 40, and 50. The tardiness costs/per time unit (α_i) is distributed uniformly within the range [1,60], i.e., $\alpha_i \sim$ U(1,60).

4.1.3 Job Due Date

Concerning the tardiness costs in deterministic problems, Srinivasan [9] and Ow [8] have found that it is important to control two main factors, i.e., the tardiness factor of a set of jobs and the due date range of the jobs. The tardiness factor, τ, is the average number of jobs that would be late if jobs were randomly scheduled. The due date range, R, refers to the dispersion of the due dates of the jobs drawn up.

In the following computational study, these two factors were also controlled to determine if they would influence the performance of the procedures. In the open-shop, however, controlling the tardiness factor is more difficult because of the forced idle time and the immaterial orders in the schedules. Nevertheless, the mean due date was determined by $\bar{d} = 1 - \tau[(n-1)]\bar{p}_m + \Sigma_{j=1}^{m}\bar{p}_j]$, where $\bar{p}_j$ is the mean operation processing time at clerk j and $\bar{p}_m$ is the maximum value of $\bar{p}_j$. The actual tardiness factor is consistently higher than the τ value, but it is allowed to control the tardiness factor for a set of jobs in the open-shop. The due date is also uniformly distributed within $[\bar{d}(1 - R/2), \bar{d}(1 + R/2)]$.

The τ values varied from 0.0 to 0.8 with 0.2 units increments. R was set at 0.6, 1.0, and 1.6, given 30%, 50%, and 80% respectively. Twenty cases of each layout with various numbers of jobs were generated in each of the above 15 combinations. Summarizing the above experimental design, 7,200 cases were tested. The proposed heuristic algorithm is compared with SPT (job priority equals processing time of the imminent operation), EDD (job priority equals its due-date), SLACK (job priority equals the time remaining before the job due-date minus all remaining processing for the job), and the MWSTR/EST (a mixture of the minimum time remaining before the job due-date/ the weighted tardiness rule and early starting time rule) methods.

4.2 Summary of the Results

The mean values (μ_R) in Table 2 show that the proposed Heuristic A is the best among these five methods. Furthermore, MWSTR/EST is the second, EDD is the third, SPT is the fourth, and SLACK is the worst. Table 3 shows that 92 percent of the 7,200 test cases having the first ranking under Heuristic A. The results indicate the effectiveness of the proposed Heuristic

A. The average performance and the best performance of the five methods under 15 combinations of different tardiness factor and due-date ranges are also summarized in Figure 6 and Figure 7. The average trends of five methods under five different tardiness factors in Figure 6 show that the performance of the proposed Heuristic A is very stable. The trend of the first ranking among five methods under five different tardiness factors in Figure 7 shows that the performance of the proposed Heuristic A is varying, the lowest performance is 86% at τ= 0.2. The percentage of the first ranking decreases from 96% to 86% when the tardiness factor increases from 0.0 to 0.2, it goes up from 86% to 95% when τ increases from 0.2 to 0.6, then it drops down to 94% when τ increases from 0.6 to 0.8. The performance of Heuristic A under 15 combinations is shown in Table 4. The trend of the first ranking among three due-date ranges, 30%, 50%, and 80%, under five different tardiness factors in Figure 8 shows that the performance of Heuristic A under these three due-date ranges has the same pattern. But their lowest performance points have small variance. The 30% case is 81% at $\tau = 0.2.$, the 50% case is 89% at $\tau = 0.4$, and the 80% case is 82% at $\tau = 0.4$. The fact reveals that the tardiness factor has significant influence on Heuristic A, and the influence of due-date ranges is minor.

Table 2. The average performance of five methods under 15 combinations

τ	R	SPT	EDD	SLACK	MWSTR	Heuristic A
0.0	0.6	2.0	1.1	1.3	1.2	1.0
	1.0	2.1	1.1	1.3	1.1	1.0
	1.6	2.7	1.4	1.7	1.4	1.1
	average	2.3	1.2	1.4	1.2	1.0
0.2	0.6	4.1	2.5	3.8	2.7	1.3
	1.0	3.2	1.7	2.4	1.8	1.1
	1.6	3.5	1.9	2.4	1.8	1.2
	average	3.6	2.0	2.8	2.1	1.2
0.4	0.6	3.5	3.2	4.5	2.7	1.1
	1.0	4.1	2.9	4.2	2.7	1.1
	1.6	4.4	2.6	3.8	2.7	1.3
	average	4.0	2.9	4.2	2.7	1.2
0.6	0.6	3.3	3.4	4.8	2.4	1.1
	1.0	3.5	3.4	4.6	2.4	1.1
	1.6	3.8	3.3	4.4	2.4	1.0
	average	3.5	3.4	4.6	2.4	1.1
0.8	0.6	3.2	3.6	5.0	2.2	1.1
	1.0	3.3	3.5	4.9	2.2	1.1
	1.6	3.4	3.5	4.9	2.2	1.1
	average	3.3	3.5	4.9	2.2	1.1
average		3.3	2.6	3.6	2.1	1.1

Table 3. The best performance of five methods under 15 combinations

(Unit =%)

τ	R	SPT	EDD	SLACK	MWSTR	Heuristic A
0.0	0.6	23	93	82	91	100
	1.0	10	91	85	93	99
	1.6	6	75	67	78	91
	average	13	86	78	87	96
0.2	0.6	2	17	5	13	81
	1.0	3	60	40	53	92
	1.6	2	51	38	55	85
	average	2	42	28	40	86
0.4	0.6	4	1	0	2	93
	1.0	3	3	0	5	89
	1.6	0	10	3	10	82
	average	2	4	1	5	88
0.6	0.6	3	1	0	3	93
	1.0	4	0	0	2	94
	1.6	1	0	0	2	97
	average	3	0	0	2	95
0.8	0.6	2	1	0	5	93
	1.0	2	0	0	5	93
	1.6	2	0	0	4	95
	average	2	0	0	4	94
average		4	22	18	24	92

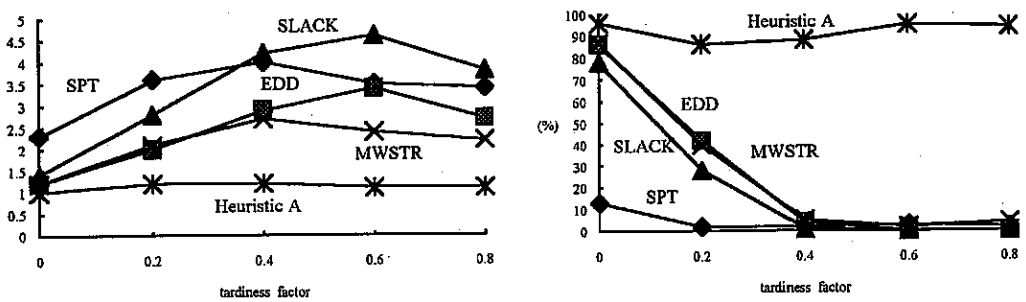


Figure 6. The average trend and the trend of 1st ranking order of five methods under 5 tardiness factors

Table 4. The performance of the proposed Heuristic A under 15 combinations

τ	R	1st	2nd	3rd	4th	5th	μ_R
0.0	0.6	100	0	0	0	0	1.0
	1.0	99	1	0	0	0	1.0
	1.6	91	6	2	1	0	1.1
	average	96	3	1	0	0	1.0
0.2	0.6	81	13	5	1	0	1.3
	1.0	92	6	2	0	0	1.1
	1.6	85	9	5	1	0	1.2
	average	86	9	4	1	0	1.2
0.4	0.6	93	6	1	0	0	1.1
	1.0	89	8	2	1	0	1.1
	1.6	82	12	5	1	0	1.3
	average	88	9	2	1	0	1.2
0.6	0.6	93	5	2	0	0	1.1
	1.0	94	5	0	1	0	1.1
	1.6	97	3	0	0	0	1.0
	average	95	4	1	0	0	1.1
0.8	0.6	93	6	1	0	0	1.1
	1.0	93	6	1	0	0	1.1
	1.6	95	5	0	0	0	1.1
	average	94	6	0	0	0	1.1
average		92	6	2	0	0	1.1

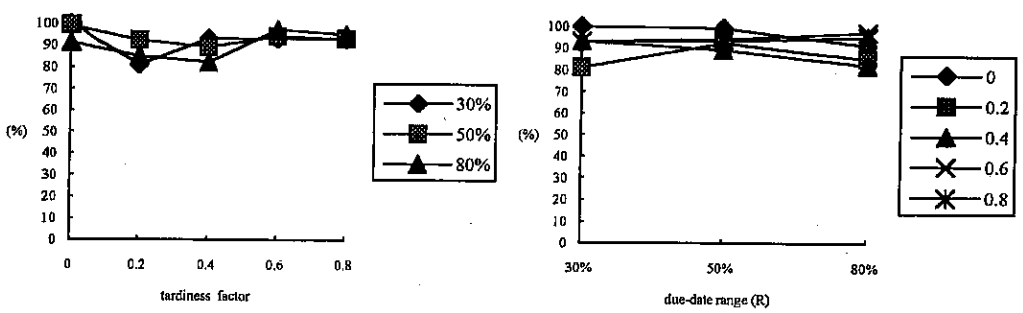


Figure 8. The trend of 1st ranking using the proposed Heuristic A

Table 5. Average effect of the number of jobs on Heuristic A compared to the MWSTR /EST method (Unit = %)

	n																	
	5			10			20			30			40			50		
τ	B	E	W	B	E	W	B	E	W	B	E	W	B	E	W	B	E	W
0.0	14	63	23	16	71	13	5	87	8	3	88	9	5	88	7	22	63	15
0.2	50	16	34	65	15	20	61	23	16	55	25	20	47	31	22	56	28	16
0.4	84	0	16	95	0	5	94	0	6	95	0	5	90	0	10	90	0	10
0.6	87	0	13	99	0	1	98	0	2	98	0	2	100	0	0	98	0	2
0.8	85	0	15	97	0	3	98	0	2	98	0	2	95	0	5	96	0	4
average	64	16	20	74	17	9	71	22	7	70	22	8	65	26	9	72	18	10

A triple-value (B,E,W) representation is used to demonstrate the relative performance of Heuristic A over and against the MWSTR/EST method. Table 5 shows the average effect of the number of jobs on Heuristic A compared to the MWSTR/EST method. The results show that the average percentage of better cases ranges from 64% to 74% when the number of jobs ranges from 5 to 50. The average percentage of worse cases from 20% to 7% when the number of jobs increases from 5 to 20, then ranges from 7% to 10% as the number of jobs increases from 20 to 50. The trend of better cases on Heuristic A compared to MWSTR/EST is shown on Figure 9, the trends reveal that the percentage of better cases on Heuristic A compared to MWSTR/EST is up to 80% for $\tau > 0.2$. The fact indicates that the proposed heuristic algorithm performs ever better for complex problems, but the number of jobs has little impact on the performance of Heuristic A when $\tau > 0.2$. The tardiness factor has the significant influence on the performance of Heuristic A.

Table 6 shows the average effect as per number of clerks on Heuristic A compared to the MWSTR/EST method. The results show that the average percentage of better cases increases from 60% to 78% and the percentage of worse cases drops from 17% to 6% when the number of clerks increases from 2 to 5. The percentage of improvement on the total tardy cost for the better cases increases from 34% to 53% when the number of clerks increases from 2 to 5. Figure 10 shows the trend of better cases on Heuristic A compared to the MWSTR/EST method, the trends reveal that the percentage of better cases on Heuristic A compared to MWSTR/EST is up to 80% for $\tau > 0.2$. This occurrence reveals that Heuristic A performs ever better for complex problems, and the tardiness factor also is a significant factor on the performance of Heuristic A.

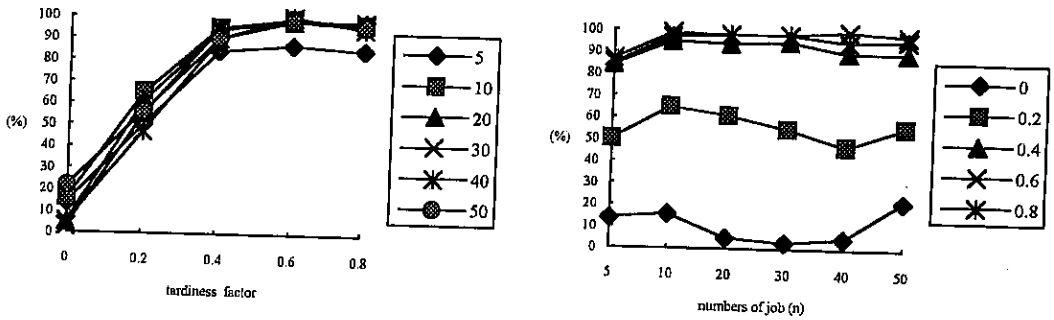


Figure 9. The trend of better cases on Heuristic A compared to the MWSTR/EST method

Table 6. Average effect of the number of clerks compared to the MWSTR/EST method (Unit = %)

	m											
	2			3			4			5		
τ	B	E	W	B	E	W	B	E	W	B	E	W
0.0	4	81	15	6	81	13	14	75	11	20	69	11
0.2	36	30	34	50	26	24	63	21	16	73	14	13
0.4	81	0	19	96	0	4	94	0	6	98	0	2
0.6	94	0	6	97	0	3	98	0	2	98	0	2
0.8	88	0	12	95	0	5	98	0	2	99	0	1
average	60	23	17	69	21	10	74	19	7	78	16	6

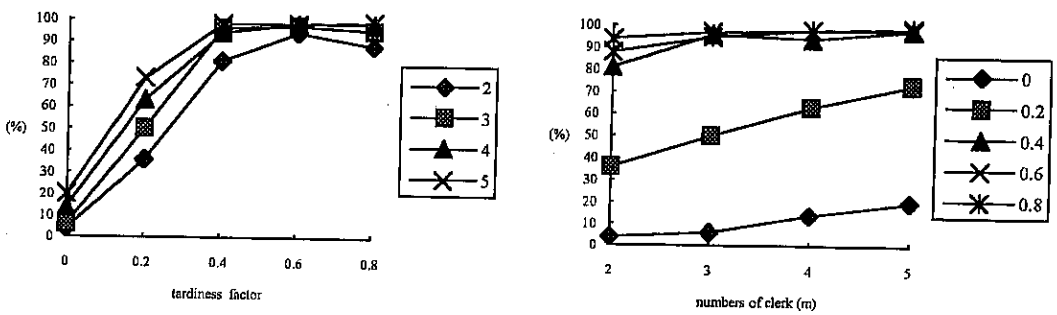


Figure 10. The trend of better cases on Heuristic A compared to the MWSTR/EST method

Table 7. Average effect of due-date range under five different tardiness factors (Unit = %)

	R									average		
	30%			50%			80%					
τ	B	E	W	B	E	W	B	E	W	B	E	W
0.0	9	90	1	6	88	6	17	51	32	11	76	13
0.2	89	5	6	45	41	14	34	23	43	55	23	22
0.4	97	0	3	96	0	4	81	0	19	91	0	9
0.6	96	0	4	97	0	3	97	0	3	97	0	3
0.8	94	0	6	95	0	5	96	0	4	95	0	5
average	77	19	4	68	26	6	65	15	20	70	20	10

Table 7 shows the average effect of the tardiness factor on Heuristic A compared to the MWSTR/EST method. The results show that the percentage of better cases increases sharply from 11% to 97% when the tardiness factor increases from 0.0 to 0.6, with the percentage of better cases decreasing steadily from 97% to 95% for τ= 0.6 to 0.8. The percentage of equal cases decreases sharply from 76% to 0% when τ increases from 0.0 to 0.4, with the percentage of equal cases becoming 0 for τ= 0.4 to 0.8. The percentage of worse cases ranges from 13% to 22% for τ= 0.0 to 0.2, with the percentage of worse cases ranges from 3% to 9% for τ= 0.4 to 0.8. Summarizing all data, the results show that 70% of the 7200 tested cases Heuristic A is preferred, as opposed to only 10% for the MWSTR/EST method, whereas the rest are indifferent among the 7200 tested cases. The high percentage of better cases and the low percentage of worse cases illustrate the superior performance of Heuristic A. The trend of better cases on Heuristic A compared to the MWSTR/EST method in Figure 11 shows that the influence of the tardiness factor on the proposed method is radical. For the problem of no job tardiness (τ= 0.0) only 11% of the test cases are better, while 76% are the same. For the problems with 20% job tardiness, 55% of the test cases are better and 23% are the same. For the problems with 40% to 80% job tardiness, the proposed heuristic performs very well.

The trend in Figure 11 also reveals the influence of the due-date range under five different tardiness factors. When the tardiness factor is from 0.0 to 0.4, the performance of Heuristic A for the 30% due-date range is the best among the 30%, 50%, and 80% due-date ranges. Yet, for τ= 0.6 to 0.8, the differences in the better cases and the worse cases among the three ranges are very slight. This implies that the performance of Heuristic A remains unaffected by the range of due-dates when $\tau > 0.4$.

The percentages of improvement on the total tardy cost for the better cases under these 15 combinations are shown in Table 8 and Figure 12. The results show that the percentage of improvement decreases when the value of tardiness factor increases. For τ= 0.0 to 0.2, the percentage of improvement on the total tardy cost for the better cases is about 80%, with the percentage of improvement on the total tardiness for the better cases has decreased sharply

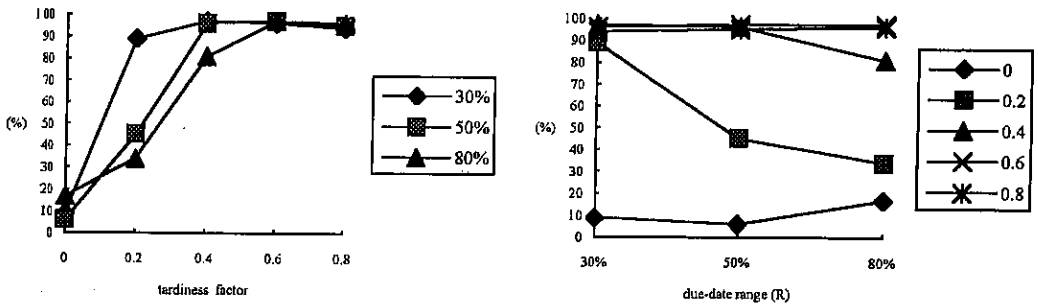


Figure 11. The trend of better cases on Heuristic A compared to the MWSTR/EST method

Table 8. The percentage of improvement compared to the MWSTR/EST method (Unit =%)

	R			
τ	30%	50%	80%	average
0.0	100	67	69	77
0.2	95	68	63	81
0.4	60	52	62	58
0.6	38	33	44	38
0.8	23	19	26	23
average	63	48	53	54

from 81% to 23% for τ= 0.2 to 0.8. However, the differences of improvement on the total tardiness among the ranges of due-dates are trivial for τ= 0.4 to 0.8. The average percentage of improvement in the total tardy cost for the better cases on Heuristic A compared to the MWSTR/EST method is around 54%.

On the basis of the above facts, experimental results show that the proposed Heuristic A performs much better than the other four methods under the total tardy cost criterion and it are ever better for complex problems. The results also reveal that the tardiness factor has significant influence on Heuristic A, and the due-date range is minor.

5. CONCLUSIONS

The superior performance of the proposed Heuristic A in open-shop scheduling when compared to the four methods may be attributed to the idle time rule and the structure of network. The proposed heuristic algorithm A performs much better than the other four methods, and generates the best results among 92% of the 7,200 test cases. Computational results also clearly indicate

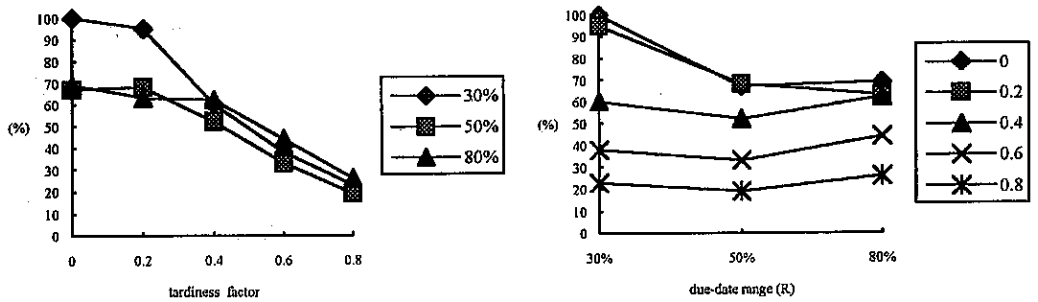


Figure 12. The percentage of improvement on Heuristic A compared to the MWSTR/EST method

that Heuristic A performs much better for an open-shop having more than or equal to 40% late jobs in its system. The complexity of Heuristic is $O(n^3m)$, which implies that Heuristic A can generate a suboptimal solution within a polynomial time. The proposed heuristic was coded in FORTRAN and executed on a SUN workstation for this experimental study. The average running time of a 50-job, 5-processor open-shop scheduling is around 11 seconds, and the running time of the worst case in all test cases is around 16 seconds. The results verify the high efficiency of the proposed Heuristic A.

Business logistics management has become more and more important to meet the strategy of marketing to satisfy the need of customers. The criterion of minimizing the tardy cost in the open-shop scheduling is particularly fitted in the picking order system in the physical distribution centers. Furthermore, for reducing the cost of inventory and perished products, the proposed Heuristics A should be able to expand for the real applications and then be converted into modules to embed into the decision support systems in the business logistics management for the near future.

Order insertion is prevalent in the small & medium enterprises of Taiwan. The property of the network model and the concept of the spanning tree will let the open-shop model or the job-shop model easy to be modified for handling the order insertion to promote the efficiency and the quality of scheduling in the future research.

References

1. Adiri, I. and Amit, N., "Route Dependent Open Shop Scheduling," IIE Transactions, 15(3), 1983, pp.231–234.
2. Garey, M. R. and Johnson D. S., Computers and Intractability: a Guide to the Theory of NP-Completeness, Freeman, San Francisco, 1979.
3. Gonzalez, T. and Sahni, S., "Open Shop Scheduling to Minimize Finish Time," Journal of the ACM, 23, 1976, pp.665–679.
4. Gonzalez, T., "Unit Execution Time Shop Problems," Mathematics of Operations Research, 7, 1982, pp.57–66.

5. Graham, R. L., Lawler, E. L., Lenstra, J. K. and Rinnooy Kan, A.H.G., "Optimization and Approximation in Deterministic Sequencing and Sched uling : A Survey," Annals of Discrete Mathematics, 5, 1979, pp.287–326.
6. Lawler, E.L., Lenstra, J. K. and Rinnooy Kan, A.H.G. , "Minimizing Maximum Lateness in a Two- Machine Open Shop," Mathematics of Operations Research, 6, 1981, pp.153–158.
7. Liu, C. Y. and Bulfin, R. L., "Scheduling Open Shops with Unit Execution Times to Minimize Functions of Due Dates," Operations Research, 36(4), 1988, pp.553–559.
8. Ow, P. S. , "Focused Scheduling in Proportionate Flowshops," Management Science, 31(7) 1985, pp.852–869.
9. Srinivasan, V., "A Hybrid Algorithm for the One Machine Sequencing Problem to Minimize Total Tardiness," Naval Research Logistics Quarterly, 18, 1971, pp.317–327.