Journal of the Society of Korea Industrial and Systems Engineering
Vol, 29, No. 1, pp.109－115, March 2006.

분할법을 이용한 가중납기지연 최소화 문제

변의석* · 홍성욱**

*선문대학교 지식정보산업공학과
**한라대학교 경영학과

Minimizing Weighted Tardiness using Decomposition Method

Eui-Seok Byeon* · Sung-Wook Hong**

*Department of Knowledge & Industrial Engineering, Sun Moon University
**Department of Business Administration, Halla University

Exact solutions for practical-size problems in job shop will be highly inefficient. Scheduling heuristics, therefore, are typically found in the literature. If we consider real-life situations such as machine breakdowns, the existing scheduling methods will be even more limited. Scheduling against due-dates addresses one of the most critical issues in modern manufacturing systems. In this paper, the method for weighted tardiness schedule using a graph theoretic decomposition heuristic is presented. It outstands the efficiency of computation as well as the robustness of the schedule.

Keywords : Decomposition, Weighted Tardiness, Robustness

1. Introduction

The Job-shop scheduling problem(JSP) for the weighted tardiness is well known to be NP-hard in a strong sense(Ullman, 1975). Exact solutions for practical-size problems will be highly inefficient. Scheduling heuristics, therefore, are typically found in the literature. If we consider real-life situations such as machine breakdowns, rush orders, power failure, and other distributions, the existing scheduling methods will be even more limited. Scheduling against due-dates addresses one of the most critical issues in modern manufacturing systems. Tardy shipments can result in contractual penalties and eventually cause significant losses to the company.

In this chapter, we present a new method for weighted tardiness schedule using a graph theoretic decomposition heuristic. This heuristic decomposes a given JSP into a series of sub-problem by solving a variant of the assignment problem(VAP). The assignment defines, in turn, a partially solved job-shop schedule that retains a great deal of local flexibility, and a much reduced complexity. The local flexibility offers additional opportunities for the schedule to handle possible disturbances in the systems. Thus, a non-myopic view of the scheduling function is maintained throughout the planning, while the attained local flexibility allows timely responses to the shop disturbances.

Roundy et al.(1991) have also addressed the issue of scheduling flexibility. They reformulated Pritsker (1968) Integer Programming formulation and implemented it as a two-phase approach including in a planning and a dispatching module. The planning module calculates the price of using each machine over the planning horizon with the dispatching module schedule based the price. Their approach is designed to handle minor system disturbances. Bean et al.(1991) developed a matchup scheduling algorithm, which computed a transient schedule upon the occurrence of a system disruption. The schedule eventually matches up with the pre-schedule in a finite amount of time while assuming an infinite planning horizon.

Morton et al.(1988), Lawrence and Morton(1993) and Vepsalainen and Morton(1987) have developed dispatching heuristics based on look ahead pricing schemes. These heuristics were shown to be quite effective for realistic size problems. This research has shown that when implemented in an iterative fashion, the performance of these heuristics can be much improved. We have used, in turn, a heuristics modified from Vepsalainen and Morton(1987) to compute the upper bounds of our problem.

Kong and Kim (2000) studied that Resource-constrained project scheduling was to allocate limited resources to activities to optimize certain objective functions and to determine a start time for each activity in the project such that precedence constraints and resource requirements are satisfied.

Sidney(1975) proposed an optimal decomposition procedure for minimizing weighed flow time in single machine problems with precedence constraints. The problem is decomposed into subsets based on the precedence network. Nonetheless, the number of subsets to be considered is not polynomially bounded by the problem size. Potts and Van Wassenhove(1982) have applied Lawler's decomposition theorem(1977) to solve a one-machine total tardiness problem. The author stated, however, that the approach was limited to the objective and could not be generalized to other scheduling problems. In most recent researches, Kim (2004) presented 0-1 IP formulation and reduced the problem as an assignment sub problem which can be solvable in polynomial time.

2. Problem Statement

Our variant of the assignment problem(VAP) is defined by reference to the disjunctive graph associated with a job shop scheduling problems. We adopt the following notation :

N	: a set of operation to be scheduled, including a dummy source, o, and sink,*
A	: a set of conjunctive arcs representing precedence constraints
E	: a set of disjunctive arcs representing operations which may compete for the same machine
E_m	: a set of disjunctive arcs for machine m
a_k	: the number of operations to be assigned to subset k
O	: the set of last operations of each job
C_{ik}	: An estimation cost of assigning operation i to subset k
x_{ik}	: a binary variable equals to 1 operation i is assigned to subset k, 0 otherwise
w_j	: weight of job j
dd_j	: due date of job j
r_i	: ready time of operation i
p_i	: processing time of operation i
t_i	: actual starting time of operation i

Given a job shop scheduling problem and its associated disjunctive graph G(N,A,E)(Adams et al. 1988), we first assign each of the $|N|$ operations into p subsets. This assignment problem can be expressed mathematically as follows :

$$\text{(VAP) : Minimize} \sum_i \sum_k cikxik \quad \cdots\cdots (1)$$

s.t.

$$\sum_k xik = 1 \qquad i \in N \quad \cdots\cdots (2)$$

$$\sum_i x_{ik} = 1 \qquad k=1,\dots,p \quad \cdots\cdots (3)$$

$$x_{jl} \le \sum_{k=1}^{l} x_{ik} \quad (i,j) \in A \text{ and } l=1,\dots,p \quad \cdots\cdots (4)$$

$$x_{ik} \in \{0,1\} \qquad i \in N \text{ and } k=1,\dots,p \quad \cdots\cdots (5)$$

Constraints (2) and (5) would be familiar with a set partitioning problem. The precedence constraints with respect to the job-shop scheduling problem(JSP) are represented in (4) by stating that operation j subject to (i,j) ∈ A can be assigned only if its predecessor i has already been assigned to the current or an earlier subset. We term (4) the partial ordering constraints. The problem with (2), (3) and (5) is known as a generalized assignment problem(Ross and Soland, 1975).

Note that solution of VAP will resolve some disjunctive arcs in the set E. In general, an arc (i,j) with i,j not assigned to a same subset will have its direction fixed. Suppose $\overline{E}^{k}_{m}$ is the remaining, unresolved set of disjunctive arcs in subset k, on machine m, then $\overline{E}^{k}_{m} = \{(i,j) \in E_m \mid x_{ik} = x_{jk} = 1\}$. The resolved set of disjunctive arcs on machine m may be expressed as,

$$\sigma_k=\{(i,j)\in E_m \mid x_{ik}=x_{jl}=1 \text{ and } k<l\} \cdots\cdots (6)$$

and

$$\sigma = U_{k\in M_k^c} \cdots\cdots (7)$$

Consequently, we could associate a disjunctive graph, $G_\psi = (N, A\cup\sigma, U_{m,k}\overline{E}_m^k)$ with each VAP solution ψ. Denote $A\cup\sigma$ as A_ψ, $U_{m,k}\overline{E}_m^k$ as E_ψ. We can then associate the disjunctive graph $G_\psi(N, A_\psi, E_\psi)$ with each solution ψ. Each disjunctive graph G_ψ defines a job-shop scheduling subproblem (P_ψ) as follows :

$$(P_\psi): Min\sum_{j\in 0} w_j(t_j+p_j-dd_j)^+ \cdots\cdots (8)$$

s.t. (Balas, 1979)

$$t_j - t_i \geqq p_i \quad (i,j) \in A_\psi \cdots\cdots (9)$$

$$t_j - t_i \geqq p_i \bigvee t_i - t_j \geqq p_i (i,j)\in E_\psi \cdots\cdots (10)$$

$$t_i \geqq 0 \quad i \in N \cdots\cdots (11)$$

(P_ψ) can be viewed as a variation of the original JSP with some additional precedence constraints.

As a distinct property of the above decomposition, an assignment decision is followed by a detail scheduling decision. The basic idea is to localize the sequencing decisions via a well-informed, higher level, operations assignment decision. For planning and control purpose, this specific property offers important planning stability, since much flexibility is retained for the schedule to handle shop dynamics.

3. Problem Analysis

Each solution of VAP(ψ) can be associated with a partially solved job-shop scheduling problem(P_ψ). The lower and upper bounds on problem P_ψ provide a performance index for each VAP solution, ψ. Suppose an operation assignment rather than a schedule is released at the beginning of a planning horizon, then the above index provides a pessimistic as well as optimistic estimation of the scheduling performance.

We have studied several lower bounds. A trivial lower bound can be computed by first dropping the unresolved disjunctive arcs in graph $G_\psi(N,A_\psi,E_\psi)$, then compute the ready time for each operation i by calculating the longest path from the source *o* to *i*. Denote this operation ready time LB1(r_i), thus, we have the following relationship between each pair of arcs in A_ψ :

$$LB1(ri) = maxj(LB1(rj) + pj) \quad \forall \ (i,j) \in A\psi. \cdots\cdots (12)$$

A lower bound in weighted tardiness can be then computed as follows :

$$LB1(WT) = \sum_{k\in O} wk(LB1(rj) + pk - ddk)+. \cdots\cdots (13)$$

LB1 totally disregards the set E_ψ, thus it can be quite weak. Rather than ignoring E_ψ, we could improve the lower bound by using the processing times information of each subset. Define D_i as the set of operations assigned to an immediately preceding subset in the same machine of i. Thus the ready times of operation *i* is no less than the total processing time for D_i. That is, in general the earliest start time(EST) of subset k is no less than the EST of subset k-1 plus the sum of operation processing times in subset k-1. The corresponding lower bound on the operation ready time is as follows :

$$LB2(r_i)=\max\{\min_{j\in D_i}\{LB2(r_j)\}+\sum_{j\in Di} p_j,\ LB1(r_i)\} \cdots\cdots (14)$$

A new lower bound on weighted tardiness LB2(WT) can be thus computed by (13) while replacing LB1 with LB2. In other words,

$$LB2(WT) = \sum_{k\in O} w_k(LB2(r_j) + p_k - dd_k)^+ \cdots\cdots (15)$$

LB2 can be further tightened by considering following. Denote L_m the last subset of operations on machine m and C_m the earliest completion of machine m, thus,

$$C_m = \min_{i\in L_m}\{LB2(r_i)\} + \{\sum_{i\in Lm} p_i\} \cdots\cdots (16)$$

Once each machine's earliest possible completion time is computed by C_m, we select the minimum possible weighted tardiness among the jobs which belong to machine m. A lower bound of weighted tardiness can be thus defined as follows :

$$LB3(WT) = \sum_{m \in M} \min_{j \in O_m} \{w_j \cdot (C_m - dd_j)^+\} \cdots\cdots (17)$$

where $O_m \subseteq O$ is the set of last operations on machine m. We then use the maximum of LB2(WT) and LB3(WT) as the lower bound LB(WT).

As previous stated, associated with each VAP solution, is a more restricted disjunctive graph $G_\psi(N,A_\psi,E_\psi)$ and a scheduling problem P_ψ. An upper bound on P_ψ can be computed by completing the schedule. We use the ATC dispatching heuristic of Vepsalainen and Morton(1987) to compute the upper bound of P_ψ.

4. Solution Methodology

As proposed by our formulation, to solve a job shop scheduling problem we first solve a variant assignment problem(VAP) which puts operations in distinct subsets, we then solve a scheduling subproblem(P_ψ) over time by generating the detailed schedule in a dynamic fashion. Thus, when solving VAP we assign operations to subsets so as to guarantee a certain level of scheduling performance. This can be achieved by using an assignment objective that reflects directly the scheduling objective.

We first propose a heuristic which uses a linear approximation $\sum_i \sum_k c_{ik}x_{ik}$ of the true assignment objective. We have developed a pricing structure which estimates the assignment costs c_{ik} based on the cost of assigning operation i to subset k. The VAP is then solved as a linear integer program. Each solution of this VAP represents a feasible assignment of operations to subsets. We iteratively adjust the linear assignment costs based on the lower or the upper bound computed for the corresponding scheduling problem(P_ψ). In the rest of this paper, we will refer to the heuristics developed under this scheme as the Price Directed Heuristics, or PDH.

4.1 The Price Directed Heuristic(PDH)

Given the job weight(w_j) and due-date(dd_j), we prioritize each operation corresponding to the weighted tardy objective. Since a due-date is giver to each job, not to operations, we define an operation due-date as its latest finishing time(LFT) for meeting the job due-date. Similar approaches can be found in Baker and Kanet (1983). Based on the due-date dd_j of job j, an artificial due-date of operation i(LFT_i) can be defined as follows :

$$LFT_i = dd_j - \sum_k p_k \quad \forall (i.k) \in A \cdots\cdots (18)$$

Like most well-known dispatching rules(EDD, MINSLK etc.), more priority is assigned to job operation which has a heavier weight, tight due-date, or both. Normalized with job weight and operation due date(LFT), we define a priority index for each operation as follows :

$$\left(\frac{wi - \min w}{\max w - \min w}\right) \cdot \left(1 - \frac{LFTi - \min LFT}{\max LFT - \min LFT}\right) \cdots (19)$$

As previously indicated, the linear cost c_{ik} in (1) is an estimation of the true assignment cost, or the corresponding cost of assigning operation i to subset k. In this section, we describe a linear, additive pricing structure which compute the price of having operation i assigned subset k based on its relative importance to the overall scheduling cost. The description of the pricing structure procedure is as follows :

Procedure *PRICING STRUCTURE* :

```
Step 0. Compute a priority index $\rho_i$ for each operation $i \in N$.
        Sort the operations according to $\rho_i$ in a non-increasing order, and let S be the sorted list of operations without violating the precedence constrains,
        i.e., if $(i,j) \in A$, i precedes j in S.
        For i=1, · · · $|S|$
          Given p subsets
          For k=1, · · ·, p
Step 1. Assign operation i to subset k, set $a_k \leftarrow (a_k - 1)$, and assign the remaining S-{i}
        operations as follows :
        For j=1, · · ·, p
          Assign the next $a_j$ operation in S-{i}
          to subset j with the following
          constraints :
          (a) the predecessors of operation I can
              be only assigned to subsets $j \le k$
          (b) the successors of operation $i$ can be
              only assigned to subsets $j \ge k$
          end for
Step 2. Resolve disjunctive arcs $(i,j) \in E$ as follows :
        if $i$ is assigned to set m, j is assigned
        set n, and m 〈n,
        then fix direction i→j.
Step 3. Compute a weighted tardy lower bound for this
        graph, and set it as $c_{ik}$.
          end for
        end for.
```

A critical component of PDH is an iterative search scheme. The basic idea is to update at each iteration the assignment index(p_i) based on the current VAP solution. Since the index is a function of job weight, due date and processing time, the value of these parameters can be adjusted such that the VAP solutions may be improved. We adjust the due-dates and weights for tardy jobs such that they are scheduled earlier in the iteration that follows. This method leads to schedules that give higher priority to tardy jobs in the previous iteration. The step-size of each adjustment is sufficiently small so as to allow fine tuning.

```
Procedure ITERATIVE SEARCH :
Step 0. Set Itrn =1, and denote NTI be the number of to-
        tal iterations. Compute priority index using original
        due-date and weight. Go to step 2.
Step 1. Compute priority index using updated due-date and
        weight.
Step 2. Solve VAP by PDH, and compute Bounds of $P_\psi$.
Step 3. If Itrn > NTI, then stop.
Step 4. Update due-dates and weights.

        $dd_i^{t+1} = dd_i^t + \text{stepsize} \times dd_i^t \times w_i^t \times \left(\frac{wi \cdot si}{WT}\right)^-$
        $w_i^{t+1} = w_i^t + \text{stepsize} \times w_i^t \times \left(\frac{wi \cdot si}{WT}\right)^-$

where $s_i$ is a slack for each job i.
        Go to step 1.
End.
```

4.2 The Index Based Heuristic(IBH)

The index-based heuristic assigns operation to subsets based on a priority index as in (19). This assignment heuristic is implemented in an iterative fashion similar to that of PDH. This can be done easily by using the iterative search procedure.

We have implemented, in addition to the index in (19), a priority index originally proposed by Vepsalainen and Morton(1987). This priority index was originally developed for weighted tardy job shop scheduling problems. We use this index for operation assignment. The index can be briefly described as follows :

$$\frac{wj}{pi} \cdot \exp\left(-\left[\frac{ddj - t - pi - \sum q\in(Wq+pq)}{kp}\right]^+\right) \cdots\cdots\cdots (20)$$

where, operation i is in job J and has the set of successors. p in the denominator is the average processing time, and k is a look-ahead parameter(we set k=3). Wi is leadtime estimator where we set Wi=2 · pi. As in the PDH method, an assignment from this procedure leads to a solution to VAP, and a corresponding lower and upper bounds can be computed.

5. Experimental Results

We have implemented the PDH and IBH procedures. Three sets of test problems, 10x10, 30x10 and 20x15, were generated from the makespan problems in Applegate and Cook(1991). Job weights were generated by uniform random numbers in the range of [1,10]. The above scheme generates rather challenging tight due-date problems. The 30x10 and 20x15 problems were generated in a similar fashion except that a non-delay, instead of optimal, schedule were used for due-date assignment.

The computational results for PHD and IBH are reported in Table 1. As shown in the table, associated with each VAP solution is a lower and upper bounds to the corresponding scheduling problem(p_ψ). The upper bounds(UB) were generated by completing the schedule using the Vepsalainen and Morton's heuristic(VMH). As previously mentioned, if we view the VAP solution as a partial schedule then the lower and upper bounds provide a performance index of the assignment.

As discussed earlier, both heuristics could iterate on either the upper bound(i.e., PDH(UB), IBH(UB)) or lower bounds(i.e., PDH(LB), IBH(LB)). To illustrate the effectiveness of the iterative search procedure, we have implemented an iterative version of Vepsalainen and Morton's heuristic. Observed from columns VMH(0) and VMH(I) the iterative method appears to be very effective with an aver-

<Table 1> Comparison of Heuristic Performance of weighted tardiness on test problems.

	PDH(LB)		PDH(UB)		IBH(LB)		IBH(UB)		VMH(0)	VMH(I)
10×10	[LB	UB]	[LB	UB]	[LB	UB]	[LB	UB]	UB	UB
abz5	2886	8844	1792	8893	4077	8541	2334	9922	8532	7863
al6	117	4187	1870	4431	488	1781	519	3265	3688	1878
la20	2454	6044	1215	6225	1557	4019	1138	4613	5485	4533
mt10	875	7141	845	8260	1263	5172	1103	4510	6413	3486
orb6	447	3646	1171	5377	2052	4312	1760	5585	5572	3240
30x10										
la31	2256	22581	1482	20815	922	7583	5969	7814	6558	4607
la33	4528	18868	4528	21120	2671	6143	4528	6645	5949	2708
la34	1743	16471	5590	17407	2144	16951	5376	13714	8780	6058
20x15										
abz7	1114	10877	994	12463	1317	6711	1896	6031	6251	3677
abz8	2938	14973	3172	15399	1662	9842	3286	10939	9706	7297
abz9	179	10678	2672	11945	977	6848	2731	8314	8958	6918

PDH(UB) : Price Directed Heuristic with iterative search based on Upper Bound.
PDH(LB) : Price Directed Heuristic with iterative search based on Lower Bound.
IBH(UB) : Index Based Heuristic with iterative search based on Upper Bound.
IBH(LB) : Index Based Heuristic with iterative search based on Lower Bound.
VMH(0) : Morton's ATC-Priority heuristic without iterative search.
VMH(I) : Morton's ATC-Priority heuristic with iterative search.

age improvement of 30%.

The results in Table 1 establish a rough comparison between the two VAP heuristics (PDH and IBH) and the more traditional scheduling heuristic VMH. Nevertheless, the results are rather difficult to interpret since the VAP heuristic generates only a partial schedule. A basic thesis of this experiment is that this partial schedule provides global performance while offering local flexibility to handle shop disturbances. To verify this point we have conducted a set of Monte Carlo experiments which test robustness of the partial schedule under a wide range of shop disturbances. The results are then compared to the traditional static, and dynamic scheduling methods.

For each partial schedule Ψ generated in Table 1, we start with its corresponding disjunctive graph GΨ and make the remaining scheduling decisions over time using a dynamic dispatching rule. To establish a fair comparison, we use Vepsalainen and Morton's ATC heuristic for the above dynamic dispatching, and we use the same heuristic to generate pure dynamic schedules where all scheduling decisions are made dynamically. Moreover, we use the iterative version of the ATC heuristic(VMH(I)) to generate static schedules. These scheduling procedures are then simulated under various levels of processing time variations. Specifically, we generate perturbed processing times p_i' as follows : $p_i' = p_i \pm \text{Exp}(\tau)$ where $\tau=5, 10, 15, 20, ..., 60$. In the case where p_i' becomes negative, we set it to 1. Table 2 summarizes the simulation results for the case where $\tau=15$. Each entry in the table represents the average total weighted tardiness of 100 simulation runs. As can be seen from the table IBH methods in general, outperform the PDH. Furthermore, IBH(UB) is slightly better than IBH(LB). In the simulation study, we further investigated the effect of different levels of disturbances. We compare the performance of VAP heuristics with the traditional static and dynamic scheduling methods. The performance of static schedules deteriorates rapidly as the level of uncertainty increases. The VAP and the dynamic schedules, on the other hand, appear to be much more robust.

6. Conclusions

This research describes a new decomposition method for the job shop scheduling problem using a variant of the classical assignment formulation. There are some characteristics of this approach which differentiate it from existing ideas dealing with JSP. First of all, using a graph theoretical approach, the proposed decomposition method offers computational efficiency. A machine scheduling and sequencing problem can be partitioned into a number of smaller problems so that less computational effort is required.

<Table 2> Simulation Results for τ=15.

10×10	PDH(LB)	PDH(UB)	IBH(LB)	IBH(UB)
abz5	10414	11823	10595	12878
al6	7386	6515	4414	4185
la20	8228	7556	6019	5287
mt10	10252	12650	8331	8715
orb6	6944	7326	8554	7645
30x10				
la31	28980	34019	18917	20789
la33	30542	30959	18333	16036
la34	28574	36573	28183	22878
20x15				
abz7	28367	31710	22044	18881
abz8	29233	30690	28228	26243
abz9	26462	23804	21986	22139

Since many JSPs are too large to be solved economically by existing algorithms, the decomposition approach provides an obvious improvement. Secondly, for on-line control purposes, decomposition provides more flexible and more robust schedules in situations involving shop disturbances.

References

[1] Adams. J., Balas, E. and Zawack, D. (1988), The Shifting Bottleneck Procedure for Job Shop Scheduling, *Management Science*, 34, 391-401.

[2] Applegate, D. and Cook, W. (1991), A Computational Study of the Job-Shop scheduling Problem, *ORSA Journal on Computing*, 3, 149-156.

[3] Baker, K. and Kanet, J. (1983), Job Shop Scheduling with Modified Due Dates, *Journal of Operations Management*, 4, 11-22.

[4] Balas, E. (1979), Disjunctive Programming, *Annals of Discrete Mathematics*, 5, 3-51.

[5] Bean, J., Birge, J., Mittenthal, J. and Noon, C. (1991), Matchup Scheduling with Multiple Resources, Release Dates and Disruptions, *Operations Research*, 39, 470-483.

[6] Kim, D. (2004), Unrelated Parallel Machine Scheduling for PCB Manufacturing, *Journal of the Society of Korea Industrial and Systems Engineering*, 27, 141-146.

[7] Kong, M. and Kim, J. (2000), A Heuristic Algorithm for Resource Constrained Multi-Project Scheduling, *IE Interfaces*, 13, 110-119.

[8] Lawler, E. (1977), A 'Pseudopolynomial' Algorithm for Sequencing Jobs to Minimize Total Tardiness, *Annals of Discrete Mathematics*, 1, 331-342.

[9] Lawrence, S. and Morton, T. (1993), Resource Constrained Multi-Project Scheduling with Tardy Costs : Comparing Myopic, Bottleneck, and Resource Pricing Heuristics," *European Journal of Operational Research*, 64, 168-187.

[10] Morton, T., Lawrence, S., Rajagopalan, S. and Kerke, S. (1988), SCHED-STAR : A Price-Based Shop Scheduling Module, *Journal of Manufacturing and Operations Management*, 1, 131-181.

[11] Potts, C. and van Wassenhove, L. (1982), A Decomposition Algorithm for the Single Machine Total Tardiness Problem, *Operations Research Letters*, 1, 177-181.

[12] Pritsker, A. and Watters, L. (1968), A Zero-One Programming Approach to Scheduling with Limited Resources, The RAND Corporation, RM-5561-PR.

[13] Ross, G. and Soland, R. (1975), A Branch and Bound Algorithm for the Generalized Assignment Problem, *Mathematical Programming*, 8, 91-103.

[14] Roundy, R., Maxwell, Y., Herer, S. and Getzler, A. (1991), A Price-Directed Approach to Real Time Scheduling of Production Operations, *IIE Transaction*, 23, 149-160.

[15] Sidney, J. (1975), Decomposition Algorithms for Single Machine Sequencing with Precedence Relations and Deferral Costs, *Operations Research*, 23, 283-298.

[16] Ullman, J. (1975), NP-Complete Scheduling Problems, *Journal of Computers and Systems Science*, 10, 384-393.

[17] Vepsalainen, A. and Morton, T. (1987), Priority Rules for Job Shops with Weighted Tardy Costs, *Management Science*, 33, 95-103.