Lagrangian relaxation algorithms for hybrid flow-shop scheduling problems with limited buffers

Takashi Irohara

Faculty of Science and Technology
Sophia University
7-1 Kioi-cho, Chiyoda-ku,
Tokyo, 102-8554, Japan
E-mail: irohara@sophia.ac.jp

Abstract: In this research, Lagrangian relaxation algorithms are proposed for Hybrid Flow-Shop (HFS) scheduling problems. Conventional HFS consists of a series of production stages, each of which has several identical parallel machines and no buffer spaces are considered. Jobs are processed through all stages in the same direction. In the previous researches, they are assumed that the capacity of buffer is infinite. But in the actual manufacturing environment, the maximum capacity of buffer is limited. That's why HFS with limited buffer is studied in this research. Unrelated parallel machine is the general and important model of parallel machine, because this is the model which can consider the difference of machining performance with large flexibility. But most studies of HFS deal with identical parallel machines which do not consider the machine abilities. These are the motivation to study HFS scheduling problem with unrelated parallel machine in this study. In this research, the objective function is to minimize the total weighted tardiness and the earliness for each job. Three methods of Lagrangian relaxation algorithms are proposed to solve the HFS scheduling problem with limit buffers. In most studies about Lagrangian relaxation algorithm, the machine capacity constraints are relaxed and each stage is scheduled separately. But in this study, not only the machine capacity but also the precedence constraints are relaxed to schedule all stages together. The results of numerical experiments showed that the proposed methods perform very well especially for large scale problems.

Keywords: Hybrid flow-shop scheduling, Lagrangian relaxation algorithm, Combinatorial optimization

1. INTRODUCTION

We focus on a scheduling problem in a hybrid flow shop (HFS), also called flexible flow shop with multiple processors. In a typical HFS, there are several serial stages and there are one or more parallel machines at each stage. HFS scheduling problems are quite common in practice, especially in the process industry where multiple machines are available at each stage as well as in certain flexible manufacturing environments (Gupta et al. 1997).

The HFS considered here consists of a series of production stages and each stage composed of several unrelated parallel machine (UPM). Figure 1 illustrates a schematic view of HFS model. UPM can be characterized as machines that perform the same function but have different capabilities or capacities. A company may invest in similar machines that have different capabilities, taking into consideration the capital cost, operation cost and variability in demand. A fair amount of the research that has been performed on UPM scheduling has focused on a variety of objectives, ranging from minimizing the maximum completion time (Lamica 2000, Srivastava 1998), due date related objective (Bank et al.2001, Kim et al.2002, Kim et al.2003).

As surveyed in Linn and Zhang (1999), there are many studies on HFS scheduling problems. However, most of them deal with the objective function to minimize makespan or mean flow time (Dessouky et al.1998, Portman et al.1998, Kyparisis et al.2006, Jin et al.2006, Tang et al.2006). Despite the importance of due dates in today's market environment, there are very limited number of papers on the HFS scheduling problem with the performance measures related to due dates (Tang et al.2002, Gupta et al.1998), to the best of our knowledge. Generally, each job has a given due date. If it finishes too late, it cannot be delivered on time. If it finishes too early, it has to be stored as inventory. The objective of scheduling is to meet the products' due dates just in time (Zhang et al. 2000).

A new solution methodology is developed for the above-mentioned problem based on the Lagrangian Decomposition and Coordination method (LDC). LDC has recently emerged as a promising method for solving large-scale integer programming problems including complex scheduling problems. It introduces the constraints into the objective function by using a vector of Lagrangian multipliers to form a relaxed problem of the primal problem. For a given value of the vector of Lagrangian multipliers, the relaxed problem is usually much easier to solve than the original problem. The optimal values of the Lagrangian multipliers are searched through solving the Lagrangian dual problem by using a subgradient algorithm. The technique has successfully been used to obtain near-optimal solutions for parallel machine scheduling problems (Luh et al. 1993), flowshop scheduling (Tang et al.2002) and job shop scheduling (Hoitmt et al. 1993, Luh et al. 1998).

The originality of this research is to consider the new problem, which is a HFS scheduling problem with limited buffers in which each stage contains UPM. The objective is to find a schedule, which minimizes the sum of weighted earliness and tardiness penalty of the due date for each job. A new solution methodology is developed for the above-mentioned problem based on the LDC. We decompose our HFS with UPM scheduling problems into smaller sub-problems at each stage, and we solve the subproblems by the LDC.

The rest of the paper is organized as follows. Section 2 first outlines the requirements on the scheduling model. An integer programming formulation of the HFS scheduling problem with limited buffer is then presented. Section 3 describes the solution methodology based on the LDC algorithm. The proposed algorithms and other algorithms are tested on randomly generated problems, and the results are given in section 4. Finally, section 5 concludes with a short summary.

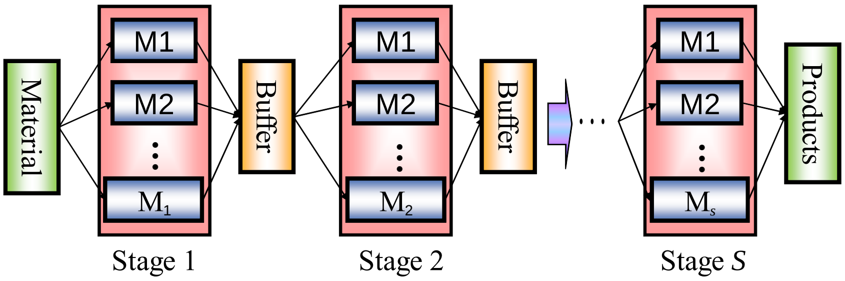

Figure 1: A schematic view of HFS model

2. MATHEMATICAL FORMULATION

2.1 Problem description

The problem considered in this paper is described as follows. There are $\boldsymbol{N}$ jobs to be processed through $\boldsymbol{S}$ stages in series. Each stage $\boldsymbol{j}$ ($\boldsymbol{j}$=1,2,…,$\boldsymbol{S}$) has unrelated parallel machines. Each job $\boldsymbol{i}$ ($\boldsymbol{i}$=1,2,…,$\boldsymbol{N}$) requires a processing time $\boldsymbol{p_{ij}}$ at stage $\boldsymbol{j}$ and has a due date $\boldsymbol{D_i}$, tardiness penalty $\boldsymbol{W_i}$, earliness penalty $\boldsymbol{U_i}$, all of which are assumed to be positive. Job processing is assumed to be non-preemptive so that a contiguous block of time of length $\boldsymbol{p_{ij}}$ is needed to process job $\boldsymbol{i}$ at stage $\boldsymbol{j}$. All jobs are considered available for processing at time zero and the scheduling time horizon $\boldsymbol{L}$ is assumed to be long enough to complete all the jobs. Each machine can handle at most one job at a time while a job can be processed on at most one machine at any time. The maximum capacity of buffer between stage $\boldsymbol{j}$ and stage $\boldsymbol{j}+1$ is limited to $\boldsymbol{V_j}$. The objective is to find a schedule that minimizes the sum of weighted earliness and tardiness penalty of the due date for each job.

2.2 The model

Parameters:
S : the number of stages
N : the number of jobs
D_i : the due date of job i
L : the planning horizon
W_i : the tardiness penalty of job i
U_i : the earliness penalty of job i
P_{ijk} : the processing time of job i at stage j on machine type k
M_{jk} : the number of machine type k at stage j
T_j : the number of machine types at stage j
V_j : the maximum capacity of buffer between stage j and stage $j+1$

Decision variables:
t_i : the tardiness of job i
e_i : the earliness of job i
c_{ij} : the completion time of job i at stage j
b_{ij} : the beginning time of job i at stage j
p_{ij} : the processing time of job i at stage j

$$\delta_{ijkl} = \begin{cases} 1 & \text{if job } i \text{ is processed on machine type } k \text{ in term } l \text{ at stage } j \\ 0 & \text{otherwise} \end{cases}$$

$$z_{ijk} = \begin{cases} 1 & \text{if job } i \text{ is processed on machine type } k \text{ at stage } j \\ 0 & \text{otherwise} \end{cases}$$

$$\gamma_{ijl} = \begin{cases} 1 & \text{if job } i \text{ stays between stage } j \text{ and stage } j+1 \text{ in term } l \\ 0 & \text{otherwise} \end{cases}$$

With the above notation the HFS scheduling problem under consideration can be formulated as follows. Note that c_{is} in the model is the final completion time of job *i*.

$$\min \quad J = \sum_{i}^{N} (W_i t_i + U_i e_i) \tag{1}$$

subject to

$$t_i \geq c_{is} - D_i \quad \forall i \tag{2}$$

$$t_i \geq 0 \quad \forall i \tag{3}$$

$$e_i \geq D_i - c_{is} \quad \forall i \tag{4}$$

$$e_i \geq 0 \quad \forall i \tag{5}$$

$$\sum_{i=1}^{N} \delta_{ijkl} \leq M_{jk} \quad \forall j,k,l \tag{6}$$

$$c_{ij} = b_{ij} + p_{ij} - 1 \quad \forall i,j \tag{7}$$

$$p_{ij} = \sum_{k=1}^{T_j} P_{ijk} z_{ijk} \quad \forall i,j \tag{8}$$

$$\sum_{k=1}^{T_j} z_{ijk} = 1 \quad \forall i,j \tag{9}$$

$$\sum_{k=1}^{T_j} \sum_{l=1}^{L} \delta_{ijkl} = p_{ij} \quad \forall i,j \tag{10}$$

$$\delta_{ijkl} \leq z_{ijk} \quad \forall i,j,k,l \tag{11}$$

$$z_{ijk} \leq \sum_{l=1}^{L} \delta_{ijkl} \quad \forall i,j,k \tag{12}$$

$$c_{iS} \leq L \quad \forall i \tag{13}$$

$$b_{i(j+1)} \geq b_{ij} + p_{ij} \quad \forall i,j \tag{14}$$

$$\sum_{i=1}^{N} \gamma_{ijl} \leq V_j \quad \forall j,l \tag{15}$$

$$\sum_{l=1}^{L} \gamma_{ijl} = b_{i(j+1)} - c_{ij} - 1 \quad \forall i,j \tag{16}$$

Equation (1) is used to minimize the total weighted tardiness and the total weighed earliness for each job. Constraints (2) ~ (5) define ti and ei. Constraints (6) are the machine capacity constraints. Constraints (7) define the completion time of each job. Constraints (8), (9) define the processing time for each job. Constraints (10) define that jobs are processed on one machine type in a stage. (11), (12) define relationship between δijkl and zijk. Constraints (13) guarantee that all jobs are completed in the planning horizon. Constraints (14) are the precedence constraints. (15), (16) are the constraints of capacity of the buffer.

3. SOLUTION METHODOLOGY

LDC introduces the constraints into an objective function by using a vector of the Lagrangian multipliers to form a relaxed problem for the original problem. The optimal values of the Lagrangian multipliers are searched through solving the Lagrangian dual problem by using a subgradient algorithm. The following three methods (A, B and C) to relax the constraints are proposed in this paper.

For a given set of multipliers, all the sub-problems of the relaxed problem are solved. Because the resulting solution may be infeasible for the original problem, a heuristic is applied to covert the infeasible schedule into a feasible schedule. The set of multipliers is then adjusted based on the degree of constraints violation of the relaxed solution. With the new set of multipliers, a new relaxed problem is then formulated and solved. This process continues until the end criterion is reached. The solution quality is measured by the relative duality gap($=\left(Z_{UB}^{n}-Z_{LB}^{n}\right)/Z_{LB}^{n}$).

The dual objective value provides a lower bound $\boldsymbol{Z_{LB}}$ on the optimal value for the original problem while the objective value $\boldsymbol{Z_{UB}}$ of the feasible schedule is an upper bound for the problem. The details for the solution of the sub-problems, the construction of feasible schedules, and updating the Lagrangian multipliers are presented as follows.

3.1 Lagrangian relaxation

In the each proposed relaxation method, the objective function can be formed as $\boldsymbol{R^A}$, $\boldsymbol{R^B}$ and $\boldsymbol{R^C}$ by using a vector of the Lagrangian multipliers λ_{ij} and π_{jkl}, which is non-negative value. For given values of λ_{ij} and π_{jkl}, the relaxed problem can be decomposed into smaller sub-problems, each for one job. The sub-problems for each job ***i*** is given as $\boldsymbol{L_i^A}$, $\boldsymbol{L_i^B}$ and $\boldsymbol{L_i^C}$.

Method A: Relaxation of precedence constraints (14)

$$\begin{aligned}\min R^A &= \sum_{i=1}^{N}\left(W_i t_i + U_i e_i\right) + \sum_{i=1}^{N}\sum_{j=1}^{S-1}\lambda_{ij}\left(b_{ij}+p_{ij}-b_{i(j+1)}\right)\\ &= \sum_{i=1}^{N}\sum_{j=1}^{S-1}\lambda_{ij}p_{ij} + \sum_{i=1}^{N}\left\{\left(W_i t_i + U_i e_i\right) + \sum_{j=1}^{S-1}\lambda_{ij}\left(b_{ij}-b_{i(j+1)}\right)\right\}\end{aligned} \tag{17}$$

This leads to the following decomposed sub-problem for each job ***i*** (given λ_{ij})

$$L_i^A = \left(W_i t_i + U_i e_i\right) + \sum_{j=1}^{S-1}\lambda_{ij}\left(b_{ij}-b_{i(j+1)}\right) \quad \forall i \tag{18}$$

From the ($\boldsymbol{L_i^A}$) in equation (18), the scheduling of job ***i*** becomes the selection of the optimal beginning time $\boldsymbol{b_{ij}}$*******. To do this, ($\boldsymbol{L_i^A}$) is computed for each possible value of $\boldsymbol{b_{ij}}$, and $\boldsymbol{b_{ij}}$******* is the one yielding the lowest value of these ($\boldsymbol{L_i^A}$). The computational complexity of the problem ($\boldsymbol{L_i^A}$) is linear in ***L*** (planning horizon) since at most ***L*** times evaluations of ($\boldsymbol{L_i^A}$) is needed to determine $\boldsymbol{b_{ij}}$*******. This optimization problem is solved from stage ***S*** and the $\boldsymbol{b_{iS}}$******* is assumed to be the due date for stage ***S***-1. Next, the optimization problem at stage ***S***-1 is solved and the $\boldsymbol{b_{iS-1}}$******* is assumed to be the due date for stage ***S***-2. These processes are repeated until we solve the first stage and determine the $\boldsymbol{b_{ij}}$*******(***j***=1,2,…,***S***).

In the each stage, unrelated parallel machine (UPM) scheduling problem is solved by the LDC based algorithm proposed by (Watanabe et al.2006). In the UPM scheduling problem, a processing time of a job is different from the machine type which the job processed. In proposed method, this UPM scheduling problem is decomposed into two problems. One is the assignment problem which is to assign jobs to machine types, the other is scheduling problem on the machine to which jobs are assigned. That enables to solve this problem as identical parallel machine (IPM) scheduling problem.

The characteristic of proposed method based on LDC is to solve the assignment problem efficiently by using the Lagrangian multiplier obtained the LDC. The value of the Lagrangian multiplier obtained when the LDC is applied to

IPM scheduling problem has the feature that the value is small in the term when production demand is small and vice versa. We solve the assignment problem by using this feature to equalize the each machine type's operational rate.

These processes are also applied for the following relaxation method B and C.

Method B: Relaxation of machine capacity constraints (6)

$$\begin{aligned}\min R^B &= \sum_{i=1}^{N}\left(W_i t_i + U_i e_i\right) + \sum_{j=1}^{S}\sum_{k=1}^{T_j}\sum_{l=1}^{L}\pi_{jkl}\left(\sum_{i=1}^{N}\delta_{ijkl} - M_{jk}\right) \\ &= -\sum_{j=1}^{S}\sum_{k=1}^{T_j}\sum_{l=1}^{L}M_{jk}\pi_{jkl} + \sum_{i=1}^{N}\left\{\left(W_i t_i + U_i e_i\right) + \sum_{j=1}^{S}\sum_{k=1}^{T_j}\sum_{l=1}^{L}\pi_{jkl}\delta_{ijkl}\right\}\end{aligned} \tag{19}$$

$$L_i^B = \left(W_i t_i + U_i e_i\right) + \sum_{j=1}^{S}\sum_{k=1}^{T_j}\sum_{l=1}^{L}\pi_{jkl}\delta_{ijkl} \quad \forall i \tag{20}$$

Method C: Relaxation of machine capacity constraints (6) and precedence constraints (14)

$$\begin{aligned}\min R^C &= \sum_{i=1}^{N}\left(W_i t_i + U_i e_i\right) + \sum_{j=1}^{S}\sum_{k=1}^{T_j}\sum_{l=1}^{L}\pi_{jkl}\left(\sum_{i=1}^{N}\delta_{ijkl} - M_{jk}\right) + \sum_{i=1}^{N}\sum_{j=1}^{S-1}\lambda_{ij}\left(b_{ij} + p_{ij} - b_{i(j+1)}\right) \\ &= -\sum_{j=1}^{S}\sum_{k=1}^{T_j}\sum_{l=1}^{L}M_{jk}\pi_{jkl} + \sum_{i=1}^{N}\sum_{j=1}^{S-1}\lambda_{ij}p_{ij} + \sum_{i=1}^{N}\left\{\left(W_i t_i + U_i e_i\right) + \sum_{j=1}^{S}\sum_{k=1}^{T_j}\sum_{l=1}^{L}\pi_{jkl}\delta_{ijkl} + \sum_{j=1}^{S-1}\lambda_{ij}\left(b_{ij} - b_{i(j+1)}\right)\right\}\end{aligned} \tag{21}$$

$$L_i^C = \left(W_i t_i + U_i e_i\right) + \sum_{j=1}^{S}\sum_{k=1}^{T_j}\sum_{l=1}^{L}\pi_{jkl}\delta_{ijkl} + \sum_{j=1}^{S-1}\lambda_{ij}\left(b_{ij} - b_{i(j+1)}\right) \quad \forall i \tag{22}$$

3.2 Constructing a feasible solution

Because of the stopping criterion used, the solution in the dual space is generally associated with an infeasible schedule, because the capacity and/or precedence constraint might be violated for a few time units. To construct a feasible schedule, a heuristic approach based on the "list scheduling" concept is developed as follows.

In the optimal dual solution, each job is uniquely associated with a beginning time ***b***$\boldsymbol{b_{ij}}$*****. A list is created by arranging jobs in the descending order of their respective completion times, and jobs are scheduled on machines according to this list as machines become available from the due date of each job. This backward scheduling mechanism is applied to satisfy the just-in-time requirement of the completion of jobs. After that, a forward scheduling mechanism is also applied after the backward scheduling, when the schedule violates the release time constraint. Besides that, if the completion times of more than two jobs are the same, we consider the amount of slack to the due date, job weight and expected processing time and decide the priority in that list. Once a feasible schedule is obtained, the corresponding value of the objective function is an upper bound on the optimal objective function value.

3.3 Updating Lagrangian multipliers

In order to solve the dual problem, the subgradient method (Luh et al.1993) is adopted for updating the Lagrangian multipliers. In this way, the vectors of Lagrangian multipliers π_{jkl}^n and λ_{ij}^n are updated by (23) and (24).

$$\pi_{jkl}^{n+1} = \pi_{jkl}^{n} + \alpha^n g\left(\pi_{jkl}^{n}\right) \tag{23}$$

$$\lambda_{ij}^{n+1} = \lambda_{ij}^{n} + \beta^n h\left(\lambda_{ij}^{n}\right) \tag{24}$$

where α^n and β^n are the step size at the ***n*** the iteration and given by (25) and (26).

$$\alpha^n = f \frac{Z_{UB}^n - Z_{LB}^n}{\sum_{l=1}^{L} \left\{ g\left(\pi_{jkl}^n\right) \right\}^2} \quad 0 < f \le 2 \tag{25}$$

$$\beta^n = a \frac{Z_{\mathbf{UB}}^n - Z_{LB}^n}{\sum_{l=1}^{L} \left\{ h\left(\lambda_{ij}^n\right) \right\}^2} \quad 0 < a \le 2 \tag{26}$$

where $Z_{\mathbf{UB}}^n$ is an upper bound value of the optimal value ***R*** obtained from the best feasible solution found so far and Z_{LB}^n is an lower bound value of ***R*** at the ***n***th iteration.

$g\left(\pi_{jkl}^n\right)$ and $h\left(\lambda_{ij}^n\right)$ are the subgradient of the Lagrangian multipliers π_{jkl}^n and λ_{ij}^n and calculated in (27) and (28).

$$g\left(\pi_{jkl}^n\right) = \sum_{i=1}^{N} \delta_{ijkl} - M_{jk} \tag{27}$$

$$h\left(\lambda_{ij}^n\right) = \sum_{i=1}^{N} \left(b_{ij} + p_{ij} - b_{i(j+1)}\right) \tag{28}$$

4. COMPUTATIONL EXPERIMENTS

To test the performance of the method and to study the characteristics of the solutions, a computational experiment has been carried out on randomly generated problem instances. In this experiment, proposed three relaxation methods are compared.

4.1 Generation of problem instance

For each data set, ten instances are randomly generated, therefore resulting in a total of 230 test problems used in this experiment. Here, [x, y] indicates the uniform distributions with range [x, y]. Table 1 shows the input data summary.

$$D_i = \sum_{j}^{S} \frac{\sum_{k}^{T_j} P_{ijk}}{ST_j} \times D_{range}$$

$$P_{ijk} = \left[1, P_{\max}\right], W_i = \left[1, W_{\max}\right], U_i = \left[1, U_{\max}\right], V_j = \left[1, V_{\max}\right], T_j = \left[1, T_{\max}\right], M_{jk} = \left[1, M_{\max}\right]$$

Table1. Parameters for each data set

Data set	Jobs	Stages	D_range	Pmax	Wmax	Umax	Vmax	Tmax	Mmax
1	3	2	0.8-1.2	3.0	5.0	5.0	2	2	3
2	4								
3	5								
4	6								
5	7								
6	8								
7	3	3							
8	4								
9	5								
10	6								
11	7								
12	30	2	0.8-1.6	5.0			3	4	3
13		4							
14		6					4	6	5
15		8							
16	50	2						6	6
17		4							
18		6						8	8
19		8							
20	100	2						6	6
21		4							
22		6						8	8
23		8							

4.2 Computational results

The performance of each proposed algorithm was examined for small scale and large scale problems. All the objective function values and CPU time shown in the Table 2 and 3 represent the average performance measures for the ten instances of the corresponding data set.

4.2.1 Experiment 1

In the small size problem shown in Table 2, all three proposed relaxation algorithm (Precedence, Capacity and Both constraints relaxation) could find optimal solutions for all the data set (1 through 11). The optimal solutions were confirmed by solving the integer programming formulation of HFS scheduling problem by using mathematical programming software (GLPK). The computation time of proposed algorithm is less than 1 second but the time of GLPK increase exponentially with the increase of number of jobs.

Table 2. Experimental results for small size problem

Data set	Jobs	Stages	Average Objective Function Value				Average CPU time			
			Precedence	Capacity	Both	GLPK	Precedence	Capacity	Both	GLPK
1	3	2	7.1	7.1	7.1	7.1	0.1	0.1	0.1	**20.7**
2	4		16.0	16.0	16.0	16.0	0.1	0.1	0.1	**72.4**
3	5		47.9	47.9	47.9	47.9	0.1	0.1	0.1	**112.3**
4	6		22.8	22.8	22.8	22.8	0.1	0.1	0.1	**402.6**
5	7		52.9	52.9	52.9	52.9	0.1	0.1	0.1	**1111.3**
6	8		59.6	59.6	59.6	59.6	0.1	0.1	0.1	**2621.8**
7	3	3	11.2	11.2	11.2	11.2	0.1	0.1	0.1	**38.2**
8	4		14.0	14.0	14.0	14.0	0.1	0.1	0.1	**114.7**
9	5		28.9	28.9	28.9	28.9	0.1	0.1	0.1	**249.7**
10	6		29.7	29.7	29.7	29.7	0.1	0.1	0.1	**855.6**
11	7		56.1	56.1	56.1	56.1	0.1	0.1	0.1	**2337.1**

4.2.2 Experiment 2

In the large size problems shown in Table3, it is impossible to use the GLPK to find optimal solution because of too much time consuming. Then we compare the performance among those proposed algorithms. The second method, relaxation of capacity constraints, was the best among them when the number of stage is small, such as data set 12, 13, 16 and 20. The reason would be that it is more important to consider the relationship between machines in each stage than the relationship between stages. However, if the number of stage becomes larger, such as data set 14,15, 18,19,22 and 23, the third method, relaxation of both capacity and precedence constraints, performs very well. Because it would be more important to consider not only the relationship between machines in each stage, but also the relationship between stages to optimize all stages together in the large number of stages problems. The computation time of those methods are also different as shown in Table 3. However, even the longest time is about 700 seconds for 100 jobs. Such a calculation time is acceptable for real world use.

Table 3. Experimental results for large size problem

Data set	Jobs	Stages	Average Objective Function Value			Average CPU time		
			Precedence	Capacity	Both	Precedence	Capacity	Both
12	30	2	798.9	492.6	506.2	3.8	4.8	5.5
13		4	838.8	419.4	420.4	10.7	10.5	16.1
14		6	552.6	270.8	266.5	25.4	20.3	37.8
15		8	365.2	239.3	232.5	50.6	43.6	58.3
16	50	2	3367.5	1829.3	1881.9	15.0	19.7	26.6
17		4	2487.0	1239.8	1130.1	49.5	50.0	76.5
18		6	2423.6	1131.2	1066.5	68.3	62.5	105.2
19		8	2289.6	1156.6	1103.3	122.3	103.3	199.8
20	100	2	3026.0	2056.3	2080.5	39.0	36.5	45.3
21		4	2685.4	1482.1	1479.0	173.6	175.8	300.5
22		6	2033.1	1133.3	1049.6	258.3	209.6	457.5
23		8	1565.8	953.2	842.3	586.2	493.6	700.6

5. CONCLUSION

In this paper, hybrid flow shop scheduling problem with limited buffer was studied to minimize the total weighted tardiness and earliness for all jobs. The problem was formulated as an integer programming model and three Lagrangian relaxation methods were proposed. Results of numerical experiment for small size problem showed that all of three proposed methods could find optimal solutions in just a second. Large size problem experiment showed that the performances of those methods depend on the problem characteristics, such as number of jobs and stages. If the number of stage is relatively small, the relaxation of capacity constraints performs well. However if the number of stage increase, the relaxation of both capacity and precedence constraints is the best among them. For further research, setup time between jobs should be considered in the HFS scheduling model.

ACKNOWLEDGMENT
The author would like to thank Mr. Koji Watanabe who was the master course student in my research laboratory for conducting the numerical experiments.

REFERENCES

1. Bank, J. and Werner, F., (2001), Heuristic Algorithms for Unrelated Parallel Machine Scheduling with a Common Due Date, Release Dates, and Liner Earliness and Tardiness Penalties, Mathematical and Computer Modelling, 33, 363-383
2. Dessouky, M.M., Dessouky, M.I., and Verma, S.K., (1998) Flowshop scheduling with identical jobs and uniform parallel machines, European Journal of Operational Research, 109, 620-631
3. Gupta, J.D.D., Hariri, A.M.A., and Potts, (1997), Scheduling a two-stage hybrid flow shop with parallel machines at the first stage, Annals of Operations Research, 69, 171-191
4. Gupta, J.N.D., and Tunc, E.A. , (1998), Minimizing tardy jobs in a two-stage hybrid flowshop, International Journal of Production Research, 36, 2397-2417
5. Hoitmt, D.J., Luh, P.B., Max, E. and Patipati, K., (1993), A practical approach to job shop scheduling, IEEE Transactions on Robotics and Automation, 9,1-13.
6. Jin, Z., Yang, Z. and Ito, T.(2006). Metaheuristic algorithms for the multistage hybrid flowshop scheduling problem.

International Journal of Production Economics, 100: 322 – 334

7. Kim, D.W., Kim, K.H., Jang, W., and Chen, F.F., (2002), Unrelated parallel machine scheduling with setup times using simulated annealing, Robotics and Computer Integrated Manufacturing, 18, 223-231
8. Kim, D.W., Na, D.G., and Chen, F.F., (2003), Unrelated parallel machine scheduling with setup times and a total weighted tardiness objective, Robotics and Computer Integrated Manufacturing, 19, 173-181
9. Kyparisis, G.J., and Koulamas, C. (2006), Flexible flow shop scheduling with uniform parallel machines", European journal of operational research, Vol.168/, pp.985-997
10. Lamica, G., (2000), Scheduling jobs with release dates and tails on two unrelated parallel machines to minimize the makespan, European Journal of Operational Research, 120, 277-288
11. Linn, R. and Zhang, W., (1999), Hybrid flow shop scheduling: a survey, Computers and Industrial Engineering, 37, 57-61.
12. Luh, P.B., and Hoitomt, D.J., (1993), Scheduling of Manufacturing Systems Using the Lagrangian Relaxation Technique, IEEE Transactions on automatic control, 38, 1066-1079
13. Luh, P.B., Gou, L., Zhang, Y., Nagahora, T., Tsuji, M., Yoneda, K., Hasegawa,T., Kyoya, Y., and Kano, T., (1998), Job shop scheduling with group-dependent setups, finite buffers and long time horizon, Annals of Operations Research, 76, 233-259
14. Portmann, M.C., Vignier, A., Dardilhac, D., and Dezalay, D., (1998), Branch and bound crossed with GA to solve hybrid flowshops, European Journal of Operational Research, 107, 389-400
15. Srivastava, B., (1998), An effective heuristic for minimizing makespan on unrelated parallel machines, Journal of the Operational Research Society, 49, 886-894
16. Tang, L., Luh, P.B., Liu, J., and Fang, L. (2002), Steel-making process scheduling using Lagrangian relaxation, International Journal of Production Research, 40, 55-70
17. Tang, L., Xuan, H. and Liu, J.(2006). A new Lagrangian relaxation algorithm for hybrid flowshop scheduling to minimize total weighted completion time. Computers & Operations Research, 33: 3344 – 3359.
18. Watanabe, K., Toizumi, K. and Irohara, T.(2006). Lagrangian Decomposition and Coordination based Algorithm to Solve an Unrelated Parallel Machine Scheduling Problem, International Symposium on Flexible Automation, Osaka, Japan, July 10-12
19. Zhang, Y., Luh, P.B., Yondeda, K., Kano, T., and Kyoya, Y., (2000), Mixed-model assembly line scheduling using the Lagrangian relaxation technique, IIE Transactions, 32, 125-134