Sādhanā, Vol. 21, Part 4, August 1996, pp. 415–433. © Printed in India.

Real-world extensions to scheduling algorithms based on Lagrangian relaxation

Y NARAHARI[1] and R SRIGOPAL[2]

[1]Computer Science and Automation, Indian Institute of Science, Bangalore 560012, India
[2]Computer Science and Engineering, University of North Carolina, Raleigh, NC, USA
e-mail: hari@csa.iisc.ernet.in

MS received 18 February 1996

Abstract. Recently, efficient scheduling algorithms based on Lagrangian relaxation have been proposed for scheduling parallel machine systems and job shops. In this article, we develop real-world extensions to these scheduling methods. In the first part of the paper, we consider the problem of scheduling single operation jobs on parallel identical machines and extend the methodology to handle multiple classes of jobs, taking into account setup times and setup costs. The proposed methodology uses Lagrangian relaxation and simulated annealing in a hybrid framework. In the second part of the paper, we consider a Lagrangian relaxation based method for scheduling job shops and extend it to obtain a scheduling methodology for a real-world flexible manufacturing system with centralized material handling.

Keywords. Jobshop scheduling; Lagrangian relaxation; simulated annealing; flexible manufacturing systems; weighted tardiness; subgradient methods.

1. Introduction

The problem of scheduling arises in situations where scarce resources have to be optimally allocated to activities over time. Most scheduling problems belong to the class of NP hard combinatorial optimization problems. Any scheduling methodology should aim to (Luh *et al* 1990)

(i) generate efficiently near optimal solutions with measurable performance,

(ii) facilitate rapid "what if" analysis to examine the impact of dynamic changes, and

(iii) support efficient methods for schedule reconfiguration to accommodate these changes.

In the area of discrete activity scheduling, it is generally accepted that a gap exists between scheduling theory and practice. Practical methods react to dynamic changes without the ability to produce good solutions and theoretical methods produce good schedules without the ability to react to dynamic changes. Recently, Luh *et al* (1990) and Hoitomt *et al* (1990, 1993) have developed a Lagrangian relaxation based suboptimal algorithm for scheduling of non-preemptive single/multi-operation jobs on parallel identical machines and for job shop scheduling. Their method performs very well in a wide variety of scheduling situations and is also amenable to carrying out extensive "what-if" analysis.

In this paper, we consider the above scheduling methodologies for parallel identical machines (Luh *et al* 1990) and for jobshops (Hoitomt *et al* 1993) and extend these to take into account real-world features.

1.1 *Extension to a multiclass environment*

The scheduling methodolgy for parallel identical machines, developed by Luh *et al* (1990) does not take into account setup times and setup costs that are very important in multiclass manufacturing system scheduling. The first part of our work attempts to extend the scheduling methodology to multiclass production systems comprising parallel identical machines and taking into account setup times and setup costs.

In a multiclass production setting, the jobs are divided into a number of mutually exclusive part types. *Setup* operations are an important feature of such production environments. A significant setup time is incurred when a machine changes from processing one type of parts to a different type of parts. The setup time generally includes times for fixturing tool changing and preparing the workplace. Thus, a setup cost is incurred, since the setup operations do not contribute to productivity. To minimize the setup times and costs, a batch of products belonging to the same part type is manufactured after a single setup. Large batch sizes, on the other hand, result in high inventory levels. The economic lot sizing problem (ELSP) (Fleishmann 1990) addresses this problem of minimizing the sum of inventory and setup costs. The problem is known to be NP hard (Lawler *et al* 1989).

The extension proposed here follows a hybrid approach that combines the techniques of *Lagrangian relaxation* (Fischer 1973, 1981; Luenberger 1984) and *simulated annealing* (Kirkpatrick *et al* 1983; Aarts & Van Laarhoven 1985; Van Laarhoven *et al* 1992). The objective is to minimize the sum of the total weighted tardiness and setup costs (assumed to be a monotonically increasing function of the setup times).

1.2 *Extension to an FMS*

Scheduling is an important issue in the planning and operation of flexible manufacturing systems. The paper by Hoitomt *et al* (1993) proposes a Lagrangian relaxation-based scheduling methodology for job shops. However, it cannot be applied directly to the scheduling of a typical FMS. In the second part of this paper, we describe the scheduling problem of a particular flexible manufacturing system and develop an extension of the approach presented by Hoitomt *et al* (1993) to schedule jobs on the machines and the material

handling equipment of the given FMS. The objective is to minimize the total quadratic weighted tardiness of the schedule. The problem is believed to be NP hard and the effort here is to only design an efficient suboptimal algorithm with performance measured with the help of a lower bound.

1.3 *Organization of the paper*

The next section is a survey of the relevant literature and is in two parts. Section 2.1 deals with the scheduling of jobs in a single class production environment as described by Luh *et al* (1990). It essentially summarises the integer programming formulation of the scheduling problem and the solution methodology. Section 2.2 describes the job-shop scheduling problem and presents the Lagrangian relaxation approach of Hoitomt *et al* (1990, 1993). Section 3 proposes a hybrid methodology to a multiclass parallel identical machine problem with setup times included. The proposed methodology employs simulated annealing and Lagrangian relaxation. Three examples are discussed to demonstrate the working of the proposed methodology and detailed numerical results are provided. Section 4 describes a particular FMS, and describes our extended methodology for scheduling the resources of the FMS. Numerical results are also provided. Section 5 presents conclusions and directions for future work.

2. Scheduling methods based on Lagrangian relaxation

2.1 *The case of parallel identical machines*

Lagrangian relaxation (Fischer 1973, 1976, 1981; Luenberger 1984) provides an efficient way of scheduling independent jobs with due dates on identical parallel machines. The special integer programming formulation facilitates the application of the Lagrangian relaxation technique. Decomposition of the dual problem serves to simplify solution at the lower level. The high level problem is solved via a subgradient method. Dynamic changes can easily be accommodated in this approach. In this section, we provide a review of the Lagrangian relaxation technique as applied to scheduling of non-preemptive, single operation jobs on parallel identical machines. The material is mostly taken from the paper by Luh *et al* (1990).

2.1a *Problem formulation* An integer programming formulation as described by Luh *et al* (1990) is a common way to represent a scheduling problem. The following is a static, discrete time, integer programming formulation of the scheduling problem. We shall use the following notation.

N total number of jobs,

i index of jobs, $i = 1, 2, \ldots, N$,

K time horizon under consideration,

k index of time, $k = 1, 2, \ldots, K$,

w_i weight of job i,

t_i processing time of job i,

D_i due date of job i,

M_k number of machines available at time k,

b_i beginning time of job i,

c_i completion time of job i,

δ_{ik} integer variable, equals 1 if job i is active at time k, and 0 otherwise,

J objective function to be minimized.

Among the above variables, the number of jobs N, time horizon K, weights of jobs $\{w_i\}_{i=1}^N$, time requirements $\{t_i\}_{i=1}^N$, due dates $\{D_i\}_{i=1}^N$ and machine availability $\{M_k\}_{k=1}^K$ are assumed to be given. Also the job processing is non-preemptive so that a contiguous block of time length t_i is needed to process job i. The decision variables are $\{b_i\}_{i=1}^N$. Once the b_is are selected, $\{c_i\}_{i=1}^N$, $\{T_i\}_{i=1}^N$ and $\{\delta_{ik}\}_{i=1,k=1}^{N,K}$ can easily be derived. For example, for $b_i = 2$, $t_i = 3$, and $D_i = 3$, we have $\delta_{i2} = \delta_{i3} = \delta_{i4} = 1$, $c_i = b_i + t_i - 1 = 4$, and $T_i = c_i - D_i = 1$. We also assume for the sake of simplicity that all jobs are available for processing at time 1 (this can easily be relaxed) and that the time horizon K is long enough to complete all the jobs.

The objective function of interest is

$$J = \sum_i w_i T_i. \tag{1}$$

Such an objective function accounts for the weight of jobs, the importance of meeting due dates, and the fact that completing a job becomes critical with each time unit after passing its due date (Luh *et al* 1990). A static and deterministic parallel machine scheduling problem can now be formulated as follows.

$$P : \min_{b_i} J = \sum_i w_i T_i, \tag{2}$$

subject to capacity constraints

$$\sum_i \delta_{ik} \leq M_k \quad (k = 1, 2, \ldots, K), \tag{3}$$

and processing time constraints

$$c_i - b_i + 1 = t_i \quad (i = 1, 2, \ldots, N). \tag{4}$$

Note that in (4), adding 1 to $c_i - b_i$ is required to obtain t_i in view of the definitions of b_i and c_i.

The single machine sequencing problem can be solved as a weighted bipartite matching problem that is NP hard (Lawler *et al* 1989). Consequently, the parallel machine weighted tardiness problem is also NP hard. The additivity of the objective function facilitates the decomposition approach.

2.1b *Solution methodology* Relaxing the capacity constraints (3) using Lagrangian multipliers π_k $(k = 1, 2, \ldots, K)$ to form the relaxed problem,

$$R : \min_{b_i} \left[\sum_i w_i T_i + \sum_k \pi_k \left(\sum_i \delta_{ik} - M_k \right) \right] \tag{5}$$

subject to (4), the dual problem is

$$D : \max_{\pi} \; L, \tag{6}$$

with

$$L = - \sum_k \pi_k M_k + \min_{b_i} \sum_i \left(w_i T_i + \sum_k \pi_k \delta_{ik} \right), \tag{7}$$

subject to

$$\pi \geq 0. \tag{8}$$

This leads to the following decomposed subproblems for each job i (given π).

$$R_i : \min_{1 \leq b_i \leq K - t_i + 1} L_i, \tag{9}$$

with

$$L_i \equiv \left(w_i T_i + \sum_k \pi_k \delta_{ik} \right), \tag{10}$$

subject to

$$c_i - b_i + 1 = t_i. \tag{11}$$

K is assumed to be large enough to complete all the jobs.

For convex programming problems, the maximum of the lag (dual cost) equals the minimum of the original objective function and a saddle point exists. However, there are several difficulties in utilizing this technique for solving discrete variable problems. First, the saddle point may or may not exist and it may be difficult to determine when the algorithm has terminated. Second, even if the dual optimum were obtained, the corresponding schedule at that point may not be feasible. Heuristic adjustment is generally required to ensure that the once relaxed constraints are obeyed. Therefore, the various steps to obtaining a near optimum solution are

(1) solving the subproblems,

(2) solving the dual problem,

(3) constructing a feasible solution, and

(4) finding a (sub) optimal solution.

Each of these steps is discussed by Luh *et al* (1990).

The optimized Lagrangian multipliers π_k are interpreted as a shadow price for using the resource (machine) at k. Therefore, they reflect the sensitivity of the objective function with respect to resource levels. This can be used to provide answers to "what if" questions and to reconfigure an existing schedule when changes occur in resource availability. Thus, Lagrangian relaxation has the ability to react effectively to dynamic changes and at the same time produce good suboptimal schedules.

2.2 *The case of job shops*

A discrete time, integer programming formulation of the scheduling problem is given below. The variable definitions, and the constraint statements are all influenced by the work of Hoitomt *et al* (1990, 1993). Let (i, j) refer to the jth operation of the ith job.

N total number of jobs,

i index of jobs, $i = 1, 2, \ldots, N$,

K time horizon under consideration,

k index of time, $k = 1, 2, \ldots, K$,

w_i weight if job i,

N_i number of operations of job i,

j index of operation, $j = 1, 2, \ldots, N_i$,

t_{ij} processing time of (i, j),

D_i due date of job i,

H number of machine types,

h index of machine type, $h = 1, 2, \ldots, H$,

m_{ij} machine chosen to process (i, j),

b_{ij} beginning time of (i, j),

c_{ij} completion time of (i, j),

δ_{ijk} integer variable equals 1 if (i, j) is active at time k, 0 otherwise,

C_i completion time of job i,

M_{kh} number of machines of type h available at time k,

h_j type of machine processing operation j,

T_i tardiness of job $i = \max(0, C_i - D_i)$,

J objective function to be minimized.

The precedence constraints of every job form a simple directed acyclic graph and $C_i = c_{iN_i}$, for all i. All jobs are assumed to be available for processing from time 1 (not a crucial requirement). K is assumed to be large enough to complete all the jobs.

Among the above variables, the number of jobs N, time horizon K, weights of jobs $\{w_i\}_{i=1}^{N}$, time requirements $\{t_{ij}\}_{i=1,j=1}^{N,N_i}$, due dates $\{D_i\}_{i=1}^{N}$ and machine availability $\{M_{kh}\}_{k=1,h=1}^{K,H}$ are assumed to be given. Also the job processing is non-preemptive so that a contiguous block of time length t_{ij} is needed to process (i, j). The decision variables are $\{b_{ij}\}_{i=1,j=1}^{N,N_i}$. Once the b_{ij}'s are selected, $\{c_{ij}\}_{i=1,j=1}^{N,N_i}$, $\{T_i\}_{i=1}^{N}$ and $\{\delta_{ijk}\}_{i=1,j=1,k=1}^{N,N_i,K}$ can easily be derived. The objective function of interest is

$$J = \sum_i w_i T_i^2. \tag{12}$$

The above function accounts for the weight of the jobs, the importance of meeting due dates and the fact that a job becomes more critical with each time after passing its due date. Compare the function here with the objective function in (1). Whether we choose $\sum w_i T_i$ or choose $\sum w_i T_i^2$ does not alter either the formulation or the solution methodology, but only helps explore different weightages to the individual tardiness values. A static and deterministic parallel machine scheduling problem can now be formulated as follows.

$$P : \min_{b_{ij}} J = \sum_i w_i T_i^2, \tag{13}$$

subject to capacity constraints

$$\sum_i \delta_{ijk} \leq M_{kh} \quad (k = 1, 2, \ldots, K; h = 1, 2, H), \tag{14}$$

processing time constraints

$$c_{ij} - b_{ij} + 1 = t_{ij} \quad (i = 1, 2, \ldots, N; j = 1, 2, \ldots, N_i), \tag{15}$$

and precedence constraints

$$c_{ij} + 1 \leq b_{i(j+1)} \quad (i = 1, 2, \ldots, N; j = 1, 2 \ldots, N_i - 1). \tag{16}$$

The complexity of the above scheduling problem motivates a decomposition approach. Hoitomt *et al* (1993) propose a Gauss–Seidel method based on quadratic penalty terms for precedence constraints (16), since relaxing those constraints with Lagrangian multipliers alone would cause oscillations in the values of the multipliers relaxing them and therefore the beginning times b_{ij}. The oscillation phenomenon is due to nondecomposability of J with respect to the operations of each job.

3. Multiclass jobs on parallel identical machines

In a multiclass production system, *switchover times* or *setup times* can have a significant effect on the way parts are scheduled. The jobs of a given part type need not be processed together. It is desired to find a schedule that minimizes the sum of the weighted tardiness and switchover costs.

Several complications arise with the introduction of switchover times. The Lagrangian relaxation technique of Luh *et al* (1990) cannot be directly applied because

- For every job j, we now need to evaluate L_{ij}^*, and b_{ij}^* where i is the part type of the job that was processed immediately before j $(j = 1, 2 \ldots, N)$; $(i = 1, 2 \ldots, P)$ where P is the total number of part types.
- Designing an effective greedy heuristic to arrive at near-optimum feasible schedule at the termination of the subgradient algorithm is not easy.

To circumvent this, a hybrid approach that makes use of simulated annealing (Van Laarhoven *et al* 1992) to arrive at a near optimal sequence of setup operations and Lagrangian relaxation to arrive at the schedule of jobs of a part type on the machines

is developed. The assumption here is that once the machines are set up for a part type, all jobs belonging to the part type are processed. The following simplifying assumptions are made regarding switchover times and costs.

(1) The switchover times are the same for all classes.

(2) The setup costs depend only on the setup times and further, are a monotonically increasing function of the setup times.

Defining the state of a machine at time t to be the type of part it is processing at t, the extra data necessary are the initial states of the machines and the time instants at which each machine first becomes available.

First, we describe a method to arrive at the schedule of parts of a particular type on the machines. Let Q denote the total number of machines. An upper bound on the planning horizon K for scheduling jobs belonging to class l is given by $\sum_{i \in l} t_i$ (if all jobs are scheduled on a single machine). Let v_i denote the time instant at which machine i ($i = 1, 2, \ldots, Q$) first becomes available (after necessary setup operations). Let the permutation $(S_1, S_2, \ldots, S_Q)$ denote the sequence of machines such that $v_{S_1} \leq v_{S_2} \ldots \leq v_{S_Q}$. Determine $q = j$ such that $\max_j [v_{S_j} \leq v_{S_1} + K]$; machines $S_{q+1}, \ldots, S_Q$ cannot process any jobs belonging to the part type under consideration. For $k = 1, 2 \ldots, K$, form M_k based on $v_{S_1}, \ldots, v_{S_n}$, where $n = 1, 2, \ldots, Q$. It is here that the second assumption regarding setup costs becomes important. If two or more machines become available at the same time, any machine can be chosen for processing the jobs belonging to the part type thus preventing unnecessary enumeration at this stage. Use Lagrangian relaxation to arrive at the schedule of jobs and cost for each n. Each of these tasks is parallelizable. The schedule for which the sum of the setup cost and tardiness cost is minimum is chosen and the availability of the machines and states of the machines are accordingly updated.

To determine the order of the part types, higher level simulated annealing optimization is carried out. The simulated annealing process will give us the order in which to process the part types, taking into account the setup times and setup costs. Having obtained the order of part types, the schedule on each machine and cost is computed using the method discussed in the previous paragraph. It can easily be shown that in the global optimum schedule, jobs belonging to the same part type and having the same processing times and due dates have to be processed in the decreasing order of their weights (Srigopal 1994). These can be reordered to yield a lower cost at the termination of the algorithm.

3.1 *Numerical results*

The examples discussed here are the multiclass versions of the ones appearing in Hoitomt *et al* (1990, 1993) and Luh *et al* (1990).

3.1a *Example 1* There are 12 jobs belonging to 4 part types (call the part types A, B, C, and D), shown in table 1. They are to be scheduled on 2 machines that are available from time instant 1. Initial state of M_1 is given to be A and that of M_2 to be B. The setup times and setup costs are shown in table 2.

Table 1. Job data for example 1.

i	w_i	t_i	D_i	Class	i	w_i	t_i	D_i	class
1	2	4	41	A	2	2	4	41	A
3	2	4	61	A	4	2	2	71	B
5	2	2	46	B	6	2	3	31	C
7	2	3	31	C	8	2	3	31	C
9	2	3	36	C	10	2	3	56	C
11	2	1	26	D	12	2	1	61	D

Table 3 shows the schedule obtained after 16 iterations. The cost of the above schedule is 274 units, out of which setup costs account for 200 units and tardiness costs equal 74 units.

3.1b *Example 2* There are 25 jobs belonging to 7 part types (call these A, B, C, D, E, F, and G), shown in table 4. Table 5 shows the setup times and setup costs. The jobs are to be scheduled on 4 machines that are available from time instant 1. The initial state of M_1 is given to be A, M_2 is B, M_3 is C and M_4 is D. The schedule obtained (after 49 iterations) is shown in table 6. The cost of the (suboptimal) schedule is 930 units. Tardiness cost is 130 units and the rest are setup costs.

3.1c *Example 3* Eighty-nine jobs belonging to 15 part types are to be scheduled on 10 machines (see table 7). The first five machines are available from the beginning of the planning horizon and the next five are available from time instant 10. Initial states of the machines 1, 2, . . ., 10 are A, B, . . ., J respectively. Table 8 shows the setup times and the setup costs. The detailed schedule, obtained after 225 iterations, is shown in table 9. The cost of the schedule is 4749 units out of which 1199 units are tardiness costs and the rest are setup costs.

Table 2. Setup times and setup costs.

Job class	A	B	C	D
Setup time	40	20	30	10
Setup cost	200	100	150	50

Table 3. Schedule obtained for example 1.

M_1	1	2	3	11	12		
M_2	5	4	6	7	8	9	10

Table 4. Job data for example 2.

i	w_i	t_i	D_i	Class	i	w_i	t_i	D_i	Class
1	6	4	21	A	2	2	4	21	A
3	5	4	101	A	4	2	5	61	B
5	8	5	101	B	6	2	8	61	C
7	2	8	41	C	8	5	8	76	C
9	2	8	126	C	10	1	2	61	D
11	2	2	20	D	12	2	2	66	D
13	1	2	101	D	14	6	2	126	D
15	2	6	126	E	16	2	7	61	F
17	2	7	126	F	18	2	7	176	F
19	2	7	76	F	20	2	7	101	F
21	2	7	101	F	22	2	7	151	F
23	2	7	151	F	24	2	3	176	G
25	2	3	76	G					

Table 5. Setup times and setup costs for example 2.

Job class	A	B	C	D	E	F	G
Setup time	40	50	80	20	60	70	30
Setup cost	200	250	400	100	300	350	150

4. Scheduling an FMS with centralized material handling

4.1 *Architecture of the FMS*

The FMS under study consists of three identical numerically controlled machines (NCs), a rail-guided vehicle (RGV), a pallet pool (PP), a pallet preparation area (PPA), and a tool preparation area (TPA). Each fixture is first prepared in the PPA (mount operation). It is then taken to the NCs for machining that requires a predetermined amount of time. The fixture is then unmounted in the PPA and the pallet is released back into the PP. Assume the NCs and the PPA to have infinite buffer capacity. Periodically, the RGV feeds the tool

Table 6. Schedule obtained for example 2.

M_1	1	2	3	15			
M_2	4	5	24	25			
M_3	7	6	8	9			
M_4	10	11	12	13	14	16	19
	17	20	21	18	22	23	

magazine of the NCs with tools prepared in the TPA and is unavailable for the transit operations. The travel times between the PP and PPA, the PPA and the NC centres, and the TPA and the NCs are all given. However, these times are applicable only if the RGV is loaded. In other words, if the RGV is free, it takes negligible amount of time to reach any of the facilities.

Every job (pallet) therefore, undergoes seven operations.

(1) Travel from the PP to the PPA.

(2) The mount operation in the PPA.

(3) Travel from the PPA to the NC centres.

Table 7. Job data for example 3

i	w_i	t_i	D_i	Class	i	w_i	t_i	D_i	Class
1	1	1	1	A	2	9	1	121	A
3	1	1	131	A	4	5	1	11	A
5	1	1	11	A	6	1	1	11	A
7	1	1	−29	A	8	1	1	71	A
9	1	1	71	A	10	1	1	71	A
11	1	1	71	A	12	1	1	71	A
13	1	1	81	A	14	1	1	81	A
15	6	1	21	A	16	1	1	21	A
17	1	1	21	A	18	1	1	51	A
19	9	1	−9	A	20	1	1	−9	A
21	1	1	101	A	22	1	1	101	A
23	1	1	101	A	24	1	1	91	A
25	1	2	−19	B	26	1	2	151	B
27	1	2	151	B	28	1	2	1	B
29	1	2	1	B	30	1	2	131	B
31	1	2	−9	B	32	1	2	−9	B
33	1	2	−9	B	34	1	2	81	B
35	1	2	71	B	36	1	2	71	B
37	1	2	71	B	38	1	2	41	B
39	1	2	11	B	40	1	2	11	B
41	1	2	11	B	42	1	3	251	C
43	1	3	241	C	44	1	3	201	C
45	1	3	21	C	46	1	3	21	C
47	1	3	21	C	48	1	3	21	C
49	1	3	21	C	50	1	3	21	C
51	1	3	21	C	52	1	3	71	C
53	1	3	71	C	54	1	3	91	C
55	1	3	91	C	56	1	3	91	C

Continued

Table 7. *Continued.*

i	w_i	t_i	D_i	Class	i	w_i	t_i	D_i	Class
57	1	3	51	C	58	1	3	11	C
59	6	3	121	C	60	1	3	101	C
61	1	3	1	C	62	1	3	1	C
63	1	3	31	C	64	1	4	181	D
65	1	4	1	D	66	9	4	11	D
67	1	4	91	D	68	1	4	41	D
69	1	4	151	D	70	6	5	191	E
71	1	5	51	E	72	1	5	51	E
73	16	5	21	E	74	6	5	101	E
75	16	6	71	F	76	1	7	31	G
77	6	8	601	H	78	1	8	111	H
79	1	8	101	H	80	1	9	81	I
81	9	10	91	J	82	1	10	201	J
83	1	10	201	J	84	1	11	111	K
85	6	12	91	L	86	1	15	421	M
87	1	16	241	N	88	1	16	171	N
89	1	20	241	O					

(4) Undergo machining in one of the NC machines.

(5) Travel back to the PPA for the unmount operation.

(6) The unmount operation at the PPA.

(7) Travel back to the PP.

For the sake of convenience, we shall re-label the RGV to be facility 1, PPA to be facility 2, and the three NCs to be facilities 3, 4, 5 respectively. We also find it useful to label the RGV as machine type 1, PPA as machine type 2, and the three NC machines as machine type 3.

Table 8. Setup times and setup costs for example 3.

Class	A	B	C	D	E	F	G	H
Time	10	20	30	40	50	60	70	80
Cost	50	100	150	200	250	300	350	400
Class	I	J	K	L	M	N	O	
Time	90	100	110	120	150	160	200	
Cost	450	500	550	600	750	800	1000	

Table 9. Schedule obtained for example 3.

M_1	84										
M_2	25	28	29	31	32	33	39	40	41	35	36
	38	37	34	27	30	26					
M_3	61	62	58	48	49	50	45	51	46	47	63
	53	54	55	56	57	43	60	44	42	52	59
M_4	65	66	68	67	69	64	87	88			
M_5	71	72	74	73	70	89					
M_6	75	19	15	4	5	6	20	17	7	1	18
	16	24	8	22	3	13	10	12	9	2	21
	11	23	14								
M_7	76	85									
M_8	78	77	79								
M_9	80	86									
M_{10}	82	81	83								

4.2 *A new formulation*

The job shop scheduling methodology of Hoitomt *et al* (1990, 1993) was implemented for the above problem and an oscillation phenomenon was found to persist despite adding quadratic penalty terms. Therefore, a slightly different problem formulation as shown below is employed. Define the following additional variables.

ω_{ijk} integer variable, equals 1 for every time unit $k \leq c_{ij}$ and 0 otherwise.

σ_{ijk} integer variable, equals 1 for every time unit $k \geq b_{ij}$ and 0 otherwise.

With these new variables, the precedence constraints (16) can be replaced by the following:

$$\omega_{ijk} + \sigma_{i(j+1)k} \leq 1 \quad (i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, 6;\ k = 1, 2, \ldots, K). \tag{17}$$

With (17) in place of (16), the problem formulation is still valid and standard Lagrangian relaxation employing subgradient optimization can be employed.

Now, we relax the capacity constraints (14) using Lagrangian multipliers $\pi_{kh}(k = 1, 2, \ldots, K;\ h = 1, 2, 3)$ and relax the precedence constraints (17) using Lagrangian multipliers $\mu_{ijk}(i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, 6;\ k = 1, 2, \ldots, K)$ to form the relaxed problem,

$$R : \min_{b_{ij}} V, \tag{18}$$

with V given by:

$$\sum_i \left[w_i T_i^2 + \sum_{j,k} \mu_{ijk} \left(\omega_{ijk} + \sigma_{i(j+1)k} - 1 \right) + \sum_j \sum_{k=b_{ij}}^{c_{ij}} \pi_{kh_j} \right] - \sum_{kh} \pi_{kh} M_{kh}, \tag{19}$$

subject to (15). Then the dual problem is

$$D : \max_{\pi,\mu} L, \tag{20}$$

with L given by

$$\min_{b_{ij}} \sum_i \left[w_i T_i^2 + \sum_{j,k} \mu_{ijk} \left(\omega_{ijk} + \sigma_{i(j+1)k} - 1 \right) + \sum_j \sum_{k=b_{ij}}^{c_{ij}} \pi_{kh_j} \right] - \sum_{kh} \pi_{kh} M_{kh}, \tag{21}$$

subject to

$$\pi \geq 0, \mu \geq 0; \tag{22}$$

This leads to the following decomposed subproblems for each job i (given π, μ)

$$D_i : \max_{\mu_i,\pi} L_i(\mu_i, \pi), \tag{23}$$

with

$$D_i \equiv - \sum_{j,k} \mu_{ijk} + \min_{b_{ij}} \left[w_i T_i^2 + \sum_j \left(\sum_{k=1}^{k=c_{ij}} \mu_{ijk} + \sum_{k=b_{i(j-1)}}^{K} \mu_{i(j-1)k} + \sum_{b_{ij}}^{c_{ij}} \pi_{kh_j} \right) \right], \tag{24}$$

subject to (15). b_{ij} is not a linear function of the multipliers (μ_{ijk}) and therefore the oscillation phenomenon disappears.

The various steps to obtaining a near optimum solution (Hoitomt *et al* 1990, 1993; Luh *et al* 1990) are

(1) solving the subproblems,

(2) solving the dual problem,

(3) constructing a feasible solution and

(4) finding a (sub) optimal solution.

4.2a *Scheduling individual operations* The scheduling of (i, j) becomes the selection of optimal beginning times b_{ij}'s. To do this L_i is computed for each possible value of b_{ij} and of these b_{ij}^* is the one yielding the lowest value of L_i's. This selection again is decomposable by operations. Determination of b_{ij}^* can be parallelized.

4.2b *Solving the dual problem* A subgradient method is used to solve the dual problem. The multipliers π and μ are updated according to

$$\pi^{n+1} = \pi^n + \alpha_1^n g(\pi^n), \tag{25}$$

$$\mu^{n+1} = \mu^n + \alpha_2^n g(\mu^n), \tag{26}$$

where n is the iteration index. $g_{kh_j}(\pi^n)$ is the kh_jth component of the subgradient of D_i with respect to π and equals $\sum_{ij} \delta_{ijk} - M_{kh_j}$ and α_1^n is the step size at the nth iteration. Similarly, $g_{ijk}(\mu^n) = \mu_{ijk} + \sigma_{i(j+1)k} - 1$. We have

$$\alpha_1^n = \lambda(\bar{L} - L^n)/ \mid g(\pi^n) \mid^2, \tag{27}$$

$$\alpha_2^n = \lambda(\bar{L} - L^n)/ \mid g(\mu^n) \mid^2, \tag{28}$$

where $\bar{L}$ is an estimate of the optimal solution and L^n is the value of the optimal solution at the nth iteration. The method converges at the rate of geometric progression. λ is halved whenever L^n fails to increase in some fixed number of iterations. The subgradient algorithm terminates when the step size α_1^n and α_2^n remains small for a fixed number of iterations while L^n is not increasing.

4.2c *Construction of a feasible schedule* Because of the stopping criterion used, the solution in the dual space is generally associated with an infeasible solution, viz. some of the capacity constraints (14) and precedence constraints (17) may be violated. The processing time constraints are always satisfied. In the optimal dual solution, each operation is uniquely associated with a beginning time b_{ij}^*.

The dual solution is first modified to ensure that the precedence constraints are satisfied. This is done by pushing all b_{ij} which violate precedence constraints forward in time, starting with the second operation of each job. A list U is then created by arranging operations of all jobs in the ascending order of the modified operation beginning times. Operations are scheduled on the required machines as they become available. If the capacity constraint for a particular machine type is violated, a greedy heuristic based on the incremental change in J determines which operations should begin at that time and which ones are to be delayed by one time unit.

Operations in U are ordered in such a way that if (i, j) is before another operation (u, v), then

(1) $b_{ij} \leq b_{uv}$,

(2) if $b_{ij} = b_{uv}$, then $f(i, j) \geq f(u, v)$.

The incremental cost function for job i is defined as

$$f(i, j) = w_i[(T_i + 1)^2 - T_i^2]. \tag{29}$$

Additionally, let n be a time index tracking machine availability, and let E be a set of unscheduled operations which cannot be scheduled between time n and their respective beginning times b_{ij}. Given the sequence U and $\{M_{kh}\}$, the greedy algorithm works as follows.

Step 0: Set $E = \emptyset$ and go to step 1.

Step 1: Determine the job and operation indices i and j of the first operation in U. Determine h_j; Set $b = b_{ij}$.

Step 2: Determine the first time l such that $M_{lh_j} > 0$. Set n=l, and go to step 3.

Step 3: If $M_{lh_j} \neq 0$ for $l = n, n+1, \ldots, n+t_{ij}-1$, go to step 4. Otherwise go to step 5.

Table 10. Job data for the FMS.

i	w_i	t_i	D_i	i	w_i	t_i	D_i
1	1	6	11	2	1	10	21
3	2	12	1	4	1	48	41
5	2	72	51	6	3	12	101
7	3	48	51	8	3	48	61
9	3	48	71	10	1	48	1

Step 4: If the precedence constraints related to preceding operations are not violated, set $M_{lh_j} = M_{lh_j} - 1$ for $l = n, n+1, \ldots, n + t_{ij} - 1$ and go to step 8. Otherwise go to step 5.

Step 5: Set $n = n + 1$. If $n > b$, go to step 6; Otherwise go to step 3.

Step 6: If operation 2 on list U has beginning time b, then set set sequence $U = U - \{(i, j)\}$, re-index the sequence, and set $E = E \cup \{(i, j)\}$, and go to step 1; otherwise go to step 7.

Step 7: For any unscheduled operations $(i, j) \in E$ such that $b_{ij} \leq b$, modify $b_{ij} = b + 1$; check all subsequent operations of job i and reset those beginning times which violate precedence constraints; set $U = U \cup E$; and reform sequence U; set $E = \emptyset$ and go to step 1.

Step 8: Set $U = U - \{(i, j)\}$; if $U = \emptyset$, stop; otherwise go to step 1.

4.2d *Performance evaluation* Once a feasible schedule is obtained, the corresponding value of the objective function J is an upper bound on the optimal objective function J^*. The value of the dual function L^*, on the other hand is a lower bound on J^*. An upper bound of the duality gap is thus provided by $J - L^*$ which is a measure of the suboptimality of the feasible schedule with respect to the optimal schedule.

4.3 *Numerical results*

Ten jobs had to be scheduled on the system under consideration (table 10). The travel times are given to be: 2 units from PP to PPA and vice–versa; 1 unit from PPA to NCs and vice–versa. Table 11 shows the detailed schedule obtained. The cost of the schedule is 50435 units. The best lower bound that could be obtained (by tuning the various parameters) was 31084 units. Thus, the schedule is at most 60% more than the optimum.

5. Conclusions

In the first part of this work, a new hybrid scheduling algorithm that uses simulated annealing and Lagrangian relaxation has been proposed and tested for multiclass production systems consisting of identical parallel machines. The technique is found to work very well for many examples. However, the two key issues

(1) performance evaluation, and

(2) schedule reconfiguration in the event of dynamic changes,

have not been answered. Future work should concentrate on

(1) answering questions regarding performance evaluation and schedule reconfiguration in the event of dynamic changes for multiclass production systems and,

(2) extending the hybrid technique to job shop scheduling.

In the second part of this work, a Lagrangian relaxation algorithm for job shops was extended to the scheduling of a real-world flexible manufacturing system. An interesting feature of the algorithm is that it is quite sensitive to the initial value of λ. Also, it is

Table 11. Schedule obtained for the FMS.

i	j	b_{ij}	c_{ij}	m_{ij}	i	j	b_{ij}	c_{ij}	m_{ij}
1	1	12	13	1	1	2	15	17	2
1	3	18	18	1	1	4	24	29	4
1	5	34	34	1	1	6	36	36	2
1	7	37	38	1	2	1	14	15	1
2	2	18	20	2	2	3	22	22	1
2	4	103	112	3	2	5	113	113	1
2	6	114	114	2	2	7	115	116	1
3	1	3	4	1	3	2	6	8	2
3	3	11	11	1	3	4	12	23	4
3	5	24	24	1	3	6	25	25	2
3	7	26	27	1	4	1	39	40	1
4	2	41	43	2	4	3	44	45	1
4	4	89	136	5	4	5	137	137	1
4	6	138	138	2	4	7	139	139	1
5	1	7	8	1	5	2	9	11	2
5	3	16	16	1	5	4	17	88	5
5	5	89	89	1	5	6	90	90	2
5	7	91	92	1	6	1	32	33	1
6	2	37	39	2	6	3	59	59	1
6	4	113	124	3	6	5	127	127	1
6	6	128	128	2	6	7	129	130	1
7	1	9	10	1	7	2	12	14	2
7	3	17	17	1	7	4	30	77	4
7	5	78	78	1	7	6	79	79	2
7	7	81	82	1	8	1	30	31	1
8	2	33	35	2	8	3	36	36	1
8	4	55	102	3	8	5	103	103	1

Continued

Table 11. (*Continued.*)

i	j	b_{ij}	c_{ij}	m_{ij}	i	j	b_{ij}	c_{ij}	m_{ij}
8	6	104	104	2	8	7	106	107	1
9	1	28	29	1	9	2	30	32	2
9	3	35	35	1	9	4	78	125	4
9	5	128	128	1	9	6	129	129	2
9	7	131	132	1	10	1	1	2	1
10	2	3	5	2	10	3	6	6	1
10	4	7	54	3	10	5	55	55	1
10	6	56	56	2	10	7	57	58	1

not monotonic. In other words, a higher value of the dual of the objective function at the termination of the subgradient optimization algorithm, does not necessarily yield a schedule that is better in quality. This is mainly attributable to the termination condition and the heuristic that is applied to arrive at a feasible schedule. Future work should concentrate on designing a monotonic Lagrangian relaxation algorithm that is insensitive to the initial values of λ. At least it should provide an empirical rule for arriving at good lower bounds and better quality schedules.

This research was supported in part by the Office of Naval Research and the Department of Science and Technology grant N00014-93-1017. We would also like to acknowledge the excellent facilities at the Intelligent Systems Laboratory, Department of Computer Science and Automation, Indian Institute of Science. We also thank the authorities of a private industry in Bangalore for helping us with the data for the FMS example.

References

Aarts E H L, Van Laarhoven P J M 1985 Statistical cooling – A general approach to solve combinatorial optimization problems. *Philips J. Res.* 5: 193–226

Fischer M L 1973 Optimal solution of scheduling problems using Lagrange multipliers. *Oper. Res.* 21: 1114–1127

Fischer M L 1976 A dual algorithm for the one machine scheduling problem. *Math. Program.* 11: 229–251

Fischer M L 1981 Lagrangian relaxation methods for solving integer programming problems. *Manage. Sci.* 27: 1–18

Fleishmann B 1990 The discrete lot sizing and scheduling problem. *Eur. J. Oper. Res.* 44: 337–348

Hoitomt D J, Luh P B, Max E, Pattipati K R 1990 Scheduling jobs with simple precedence constraints on parallel machines. *IEEE Control Syst. Mag.* 10: 34–40

Hoitomt D J, Luh P B, Pattipati K R 1993 A practical approach to job shop scheduling problems. *IEEE Trans. Robot. Automat.* 9: 1–13

Kirkpatrick S, Gelatt S D Jr, Vecchi M P 1983 Optimization by simulated annealing. *Science* 220: 671–680

Lawler E L, Lenstra J K, Rinnoy Kan A H G, Shmoys D B 1989 Sequencing and scheduling – algorithms and complexity. *Logistics of production and inventory* (ed.) S C Graves, A H G Rinnoy Kan, P Zipkin, vol. 4

Luenberger D G 1984 *Linear and nonlinear programming* (Reading, MA: Addison Wesley)

Luh P B, Hoitomt D J, Pattipati K R, Max E 1990 Schedule generation and reconfiguration for parallel machines. *IEEE Trans. Robot. Automat.* 6: 687–696

Srigopal R 1994 *Scheduling algorithms for multiclass manufacturing systems.* M E Project Report, Department of Computer Science and Automation, Indian Institute of Science, Bangalore

Van Laarhoven P J M, Aarts E H L, Lenstra J K 1992 Job shop scheduling by simulated annealing. *Oper. Res.* 40: 113–125