IEMS Vol. 7, No. 2, pp. 84-92, September 2008.

A Comparative Study of Two-phase Heuristic Approaches to General Job Shop Scheduling Problem

Ji Ung Sun†
School of Industrial and Managment Engineering
Hankuk University of Foreign Studies Kyongki-do 449-791, KOREA
Tel: +82-31-330-4191, E-mail: jusun@hufs.ac.kr

Received Date, October 2007; Accepted Date, May 2008

Abstract. Scheduling is one of the most important issues in the planning and operation of production systems. This paper investigates a general job shop scheduling problem with reentrant work flows and sequence dependent setup times. The disjunctive graph representation is used to capture the interactions between machines in job shop. Based on this representation, four two-phase heuristic procedures are proposed to obtain near optimal solutions for this problem. The obtained solutions in the first phase are substantially improved by reversing the direction of some critical disjunctive arcs of the graph in the second phase. A comparative study is conducted to examine the performance of these proposed algorithms.

Keywords: Two-phase Heuristics, Job Shop Scheduling, Reentrant Work Flows, Sequence Dependent Setup Times.

1. INTRODUCTION

Job shop scheduling has attracted the attention of many researchers in the fields of both production management and combinatorial optimization. Due to the difficulties inherent in job shop scheduling, many assumptions and simplifications were made in most of the existing studies. For example, it is commonly assumed that the setup time of a job is either small enough to be ignored or sequence independent, therefore it can be combined with the processing time. However, in many real-life situations such as chemical, printing, pharmaceutical and manufacturing processes, these assumptions hardly apply. Setup operations such as cleaning up or changing tools are not only often required between jobs but they are also strongly dependent on the immediately preceding operation on the same machine (Conway, 1967). Hence more researchers have begun to focus on job shop scheduling problem with sequence dependent setup times. However, the majority of existing studies have focused only on single machine, parallel machine or flowshop environments because of the difficulty involved in a more general environment. This paper studies a general job shop scheduling problem (GJSP) with reentrant work flows and sequence dependent setup times. The fact that these problems are commonly encountered in today's manufacturing environment makes the work of developing effective scheduling methodologies valuable for the well being of these industries.

Many research articles dealt with the scheduling problems with sequence dependent setup times (Zhu and Wilhelm, 2006). Corwin and Esogbue (1974) presented a dynamic programming approach to solve a two machine scheduling problem with the objective of minimizing the makespan. Uskup and Smith (1975) presented a branch and bound algorithm to determine the minimum total setup times satisfying due-date constraints for the two machine scheduling problem. Gupta (1986) discussed a flow shop problem with separable and sequence dependent setup times. Ovacik and Uzsoy (1995) presented a family of rolling horizon heuristics for minimizing maximum lateness on parallel identical machines in the presence of sequence dependent setup times and dynamic job arrivals. Hwang and Sun (1998) proposed a dynamic programming algorithm and a genetic search based procedure for a two machine sequencing problem with reentrant work flows and two-step prior-job-dependent setup times. Also Sun and Hwang (2001) extended the research to the case of a two machine sequencing problem with n-step prior-job-dependent setup times where n is a positive integer. Sun *et al*. (1999) developed a Lagrangian relaxation based approach to solve a single machine scheduling problem with release dates, due dates and sequence dependent setup times where the objective

† : Corresponding Author

is to minimize the weighted sum of squared tardiness. Asano and Ohta (1999) suggested a branch and bound algorithm for a single machine scheduling with release dates, due dates, shutdown constraints on the machines and sequence dependent setup times so as to minimize the maximum tardiness. Bianco *et al*. (1999) proposed a mathematical formulation for a flow shop scheduling problem with release dates and sequence dependent setup times to minimize the makespan. Norman (1999) presented a tabu search based solution procedure for a flow shop scheduling problem where there are finite buffers and sequence dependent setup times. Radhakrishnan and Ventura (2000) addressed the parallel machine earliness-tardiness non-common due date sequence dependent setup time scheduling problem for jobs with varying processing times.

Review of the past studies shows that there has not been a significant amount of research done on the scheduling procedure for a job shop scheduling problem with sequence dependent setup times. The following researches consider the scheduling problem of job shop in the presence of sequence dependent setup times. Wilbrecht and Precott (1969) developed and tested a setup oriented rule. The rule gives the highest priority to a job with the shortest setup times. Gupta (1982) presented a branch and bound algorithm for minimizing total setup times in the *n*-job *m*-machine job shop problem. Flynn (1987) applied the repetitive lots procedure to sequence dependent setup times. The procedure capitalizes on the sequence dependent setup times by scanning a queue of waiting jobs and seeking to find a job identical to the job that was just processed in the machine. Kim and Bobrowski (1994) categorized the research articles on the sequence dependent setup times in terms of facility capacity (number of machines) and job arrival pattern. They conducted a simulation study to illustrate the impact of sequence dependent setup times on shop performance and concluded that setup times must be considered explicitly in any scheduling strategy when setup times are significant compared with processing times. Ovacik and Uzsoy (1997) descried an application of decomposition procedure to an environment with sequence dependent setup times to minimize the maximum lateness. Artigues and Roubellat (2002) proposed an efficient algorithm for operation insertion in a job shop scheduling problem with multi-resource requirements and sequence dependent setup times with the objective of minimizing maximum lateness.

However, the previous studies considering sequence dependent setup times in a job shop have focused mainly on developing solution methods based on dispatching rules and a few study considered the reentrant work flows which could be found frequently in real-life job shop environment. Zoghby *et al*. (2005) investigated the feasibility conditions for meta-heuristic searches when incorporating setups and reentrancy in the disjunctive graph representation of the job shop scheduling problem. This paper investigates a GJSP that are complicated by sequence dependent setup times and reentrant work flows. The problem is clearly NP-hard and optimal solutions to such problems are highly intractable. Therefore, we only consider heuristic solution methods to obtain near optimal solutions within a reasonable time.

The paper is organized as follows. In section 2 we shall represent the problem using the disjunctive graph with the objective of minimum makespan to capture the interactions between machines. Based on this representation, in section 3 we propose four two-phase heuristic algorithms where the initial solution obtained in the first phase is improved by critical exchange heuristic in the second phase. In section 4 we conduct a computational experiments to examine the performance of these proposed algorithms. In section 5 we make some concluding remarks.

2. PROBLEM DESCRIPTION USING DISJUNCTIVE GRAPH

The following notations are introduced to describe our problem.

j_i	job i $(i = 1, 2, \cdots, n)$
n_i	the number of operations required for j_i
ik	the kth operation of j_i $(k = 1, 2, \cdots, n_i)$
$S_{pq,ik}$	setup time of operation ik whose immediate predecessor is operation pq
P_{ik}	processing time of operation ik
t_{ik}	start time of operation ik

The manufacturing system under study consists of several machines. A set of jobs j_i, i = 1, 2, $\cdots$, n, is available for processing at time zero. Job j_i requires n_i number of operations with ik and P_{ik}, k = 1, 2, $\cdots$, n_i, known. Some jobs require processing on certain machines more than once in their operation sequences with reentrant work flows. Some machines require a setup prior to processing each job where setup times are sequence dependent.

Let N denote the set of operations, M be the set of machines, A be the set of pairs of operations constrained by precedence relations, and E_m the set of pairs of operations to be performed on machine m. Then the problem (P) can be stated as

$$\min t_{0*}$$
$$t_{ir} - t_{ik} \geq S_{pq,ik} + P_{ik}, \quad (ik,\ ir) \in A$$
$$t_{jr} - t_{ik} \geq S_{pq,ik} + P_{ik} \vee t_{ik} - t_{jr} \geq S_{pq,jr} + P_{jr},$$
$$(ik,\ jr) \in E_m,\ m \in M$$
$$t_{ik} \geq 0, \quad ik \in N.$$

Any feasible solution to (P) is called a schedule. It is useful to represent this problem on a disjunctive graph $F(N, A, E)$, with node set N, conjunctive arc set A, and

disjunctive arc set *E*. The nodes of *N* correspond to operations, the directed arcs of *A* to precedence constraints among the operations of the same job, and the pairs of disjunctive arcs of *E* to pairs of operations to be performed on the same machine (Balas 1969).

A selection S_m in E_m contains exactly one of each disjunctive arc pair of E_m. A selection is acyclic if it contains no directed cycle. Each acyclic selection S_m corresponds to a unique sequence of the operations with respect to machine *m*, and vice-versa. Thus sequencing a machine *m* means choosing an acyclic selection in E_m. A complete selection *S* consists of the union of selections S_m, one in each E_m, $m \in M$. Picking a complete selection *S*, i.e. replacing the disjunctive arc set *E* by the ordinary (conjunctive) arc set *S*, give rise to the ordinary graph $G = (N, A \cup S)$. A complete selection *S* is acyclic if the ordinary graph *G* is acyclic. Further, the makespan of a schedule for *S* is equal to the length of a longest path in *G*. Thus, our goal is to derive from the disjunctive graph an ordinary graph having a critical path whose length is minimal.

The GJSP under study can be represented by using the disjunctive graph model presented in previous studies (Ovacik and Uzsoy 1997). Now, we explain more details of the disjunctive graph representation (DGR) for the GJSP. An example of the DGR of a problem with two machines and three jobs is shown in figure 1. Operations 11, 12, 21 and 31 are processed on the first machine while 13, 22 and 32 are processed on the second machine. Node that source node *0* and sink node *0** denotes the beginning of processing in the system and the completion of the last job, respectively.

Note that when a certain subset of the machines has been sequenced, some constraints are imposed on the scheduling problems for the remaining machines. It is important to estimate when each operation of each job will be available for processing at a particular machine, we call it operation ready time (ORT) and by which time it must be completed so that the remaining operations on that job can be finished by the completion time of the last job, we call it operation due date (ODD), given a particular state of the system.

Both ORT and ODD are affected by scheduling decisions made at other machines. To estimate the ORT and ODD for operations, the arc costs need to be updated as the state of the system changes since actual setup times incurred depend on the sequence in which the operations are processed. We define the events corresponding to the nodes of the graph to be the beginning of the setup for the operation represented by the node. Define the cost of any arc (*ik*, *jr*), $w_{ik} = S_{pq,\,ik} + P_{ik}$, where *pq* is the operation scheduled to be processed immediately before *ik* on its machine if that machine has been scheduled, and the operation immediately preceding it within its job, that is operation (*i*, *k*-1) otherwise. We define the cost of any arc directed from the source node to be zero. If we denote by $L(ik,\ jr)$ the length of a longest path from *ik* to *jr* in the directed graph, ORT, i.e., earliest start time, of operation *ik* is given by $r_{ik} = L(0,\ ik)$. And let q_{ik} be the time to be spent in the system after the end of operation *ik* then it can be represented by $q_{ik} = L(ik,\ 0^*) - w_{ik}$. Then the ODD, i.e., latest finish time, of operation *ik* is given by $d_{ik} = L(0,\ 0^*) - q_{ik}$. By means of these calculations, the disjunctive graph representation is used to capture interactions between the different machines. Each time a machine is sequenced, the ORT and ODD of operations on other machines are updated in order to include the constraints imposed on the entire system by the sequencing of that machine.

In the proposed solution methodology, we successively solve the one machine sequencing problem. To be more specific, $M_0 \subset M$ be the set of machines that have already been sequenced by choosing selection S_m, $m \in M_0$, and for $l \in M \setminus M_0$ let $P(l,\ M_0)$ be the problem obtained from *P* by (i) replacing each disjunctive arc set E_m, $m \in M_0$, by the corresponding selection S_m, and (ii) deleting *each disjunctive arc set* E_m, $m \in M \setminus M_0$, $m \neq l$. *Then, the problem* $P(l,\ M_0)$ *can be* represented as

$$\min\ t_{0^*}$$

$$t_{ir} - t_{ik} \geq w_{ik}, \quad (ik,\ ir) \in \cup(S_m;\ m \in M_0) \cup A$$

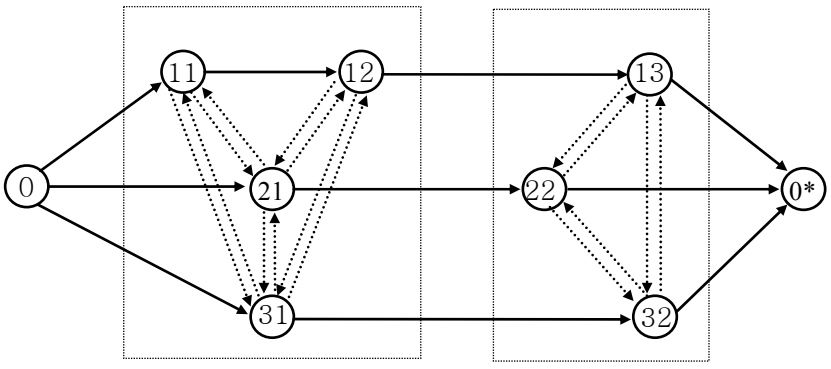

Figure 1. DGR with three jobs and two machines.

$$t_{jr} - t_{ik} \geq w_{ik} \vee t_{ik} - t_{jr} \geq w_{jr}, \quad (ik, jr) \in E_l$$
$$t_{ik} \geq 0, \qquad ik \in N.$$

Problem $P(l, M_0)$ is equivalent to that of finding schedule for machine l that minimizes the maximum lateness, given that each operation ik to be performed on machine l has a ORT r_{ik} and a ODD d_{ik} (Adams *et al.*, 1988). This problem can be viewed as a minimum makespan problem where each job has to be processed in order by three machines, of which the first and the third machine have infinite capacity, while the second machine (corresponding to machine l) processes one job at a time, and where the processing time of job ik is r_{ik} on the first machine, $S_{pq,ik} + P_{ik}$ on the second, and q_{ik} on the third machine.

3. TWO-PHASE HEURISTICS FOR THE GJSP

3.1 Single-path Heuristics

Single path heuristic simply builds up a feasible complete solution by fixing one operation at a time based on certain priority rules. For each machine, the operation that is to be scheduled next is selected from the set of operations requiring processing on the same machine. In this study, we propose two single-path heuristics, nearest setup heuristic and least penalty heuristic, which are extensions of the previous researches developed for the single machine problem (Karg and Thompson 1964, Ahn 1995). We do some modifications in the two heuristics to make them applicable to the GJSP.

In the nearest setup (NS) heuristic, it build up a complete sequence by selecting an operation with the shortest setup time while expand the partial sequence. Let $R_m = \{O_{1m}, O_{2m}, \cdots, O_{N_m, m}\}$ be the set of operations required to be processed on machine $m \in M$. Then, the NS heuristic can be stated as the following.

Step 0. (Initialization)
Set $m = 0$.

Step 1. Increase m by one. Set $k = 0$ and min $= \infty$.

Step 2. Increase k by one. Construct $\rho_m = \phi$ where ρ_m is the partial sequence of operations on machine m. Set $f(\rho_m) = l(\rho_m) = O_{km}$ where $f(\rho_m)$ and $l(\rho_m)$ is the first and last operation of the partial sequence ρ_m, respectively.

Step 3. Given ρ_m, select

$$x = \arg\min_{i,j} \{S_{ij};\ i \notin \rho_m,\ i \in R_m,\ j = f(\rho_m) \text{ or } i = l(\rho_m),\ j \notin \rho_m,\ j \in R_m\}.$$

Note that operation (i, j) must satisfy the precedence constraint. Set $\rho_m = \rho_m \cup \{x\}$ and update $f(\rho_m)$ and $l(\rho_m)$.

Step 4. If $|\rho_m| < |N_m|$, then go to step 3. Otherwise, if $v(m, M_0 = \phi) <$ min, then set min $= T(\rho_m)$ and update $\rho'_m = \rho_m$ where $v(m, M_0 = \phi)$ is the value of a solution to $P(m, M_0 = \phi)$.

Step 5. If $k < |N_m|$, then go to step 2. Otherwise, if $m < |M|$, then go to step 1.

Step 6. Construct $\rho = \cup \rho'_m$ and check the feasibility of the resulting schedule. If the schedule is infeasible, make it feasible utilizing the proposed repairing method.

In step 6, we may obtain an infeasible schedule. When this case occurred, the following repairing method is invoked. Consider an example with three jobs and three machines given in table 1.

Table 1. Operation sequence of an example with three jobs and three machines.

Job	Operation		
	1	2	3
1	MC1	MC2	MC1
2	MC1	MC3	MC2
3	MC2	MC3	MC3

Firstly we convert the resulting operation sequence into sequence of job numbers for each machine. Suppose a converted sequence of job numbers be [(1, 2, 1) (2, 1, 3) (3, 3, 2)]. The first substring (1, 2, 1) is for machine 1, (2, 1, 3) for machine 2 and (3, 3, 2) for machine 3. From these substrings we can know that the first preferential operations are job 1 on machine 1, job 2 on machine 2, and job 3 on machine 3. According to the given precedence constraints, only job 1 on machine 1is schedulable. So operation 1 which belongs to job 1 is scheduled on machine 1 firstly. Then the next position of each substring is examined whether there is schedulable operation or not. Now the current candidate operations are job 2 on machine 1, job 1 on machine 2, and job 3 on machine 3. And the next schedulable operations are job 2 on machine 1 and job 1 on machine 2. Thus the first operation of job 2 is scheduled on machine 1 and the second operation of job 1 is scheduled on machine 2. Repeat the procedure until all the operations are scheduled. In this way, we can produce a feasible schedule from an infeasible schedule.

On the other hand, LP heuristic is similar to the NS heuristic but the opportunity cost of setup time rather than the shortest setup time is used when the operation next to be sequenced is selected. Thus only step 3 of least penalty heuristic is different from that of NS heuristic. The step 3 of LP heuristic can be stated as follows.

Step 3. Given ρ_m, select

$$x = \arg\min_{i,j|S_{ij}^1} \{S_{ij}^2 - S_{ij}^1;\ i \notin \rho_m,\ i \in R_m,$$

$$j = f(\rho_m) \text{ or } i = l(\rho_m),\ j \notin \rho_m,\ j \in R_m\}$$

where S_{ij}^1 and S_{ij}^2 is the minimum setup time and the second minimum setup time, respectively. Note that operation (i, j) must satisfy the precedence constraint. Set $\rho_m = \rho_m \cup \{x\}$ and update $f(\rho_m)$ and $l(\rho_m)$.

3.2 Shifting Bottleneck Heuristics

The success of the shifting bottleneck (SB) approach developed by Adams *et al*. (1988) for the classical job shop scheduling problem (CJSP) indicates that similar types of methods hold great promise for more complicated scheduling problems such as the GJSP considered in this study. It decomposes the job shop problem into a number of single machine sub-problems, using the DGR of the problem to capture interactions between the subproblems. Computational experience to date (Adams *et al*., 1988, Balas *et al*., 1995, Uzsoy and Wang 2000) suggests that problems of realistic size can be solved with high quality schedules in very reasonable CPU times. However, this methodology must be generalized to accommodate the special features present in the GJSP of interest to us.

The methodology proceeds by scheduling one machine at a time. The machine which is to be scheduled is selected based on the idea of giving priority to bottleneck machines. We use as a measure of the bottleneck quality of machine l the value of solution to the one machine scheduling problem, $P(l, M_0)$ on machine l. Machine p is then called the bottleneck among the machines if $v(p, M_0) = \max\{v(l, M_0), l \in M \setminus M_0\}$, where $v(l, M_0)$ is the value of a solution to $P(l, M_0)$. The SB procedure is as follows.

Step 1. Represent the job shop using by disjunctive graph and set $M_0 = \varnothing$.
Step 2. Identify a bottleneck machine p among the machines $l \in M \setminus M_0$ and sequence it by a heuristic algorithm. Set $M_0 = M_0 \cup \{p\}$.
Step 3. Update the interactions between the machines and already scheduled and those not yet scheduled.
Step 4. If $M_0 \neq M$, go to step 2. Otherwise, check the feasibility of the resulting schedule. If the schedule is infeasible, make it feasible utilizing the proposed repairing method in previous section.

In step 2, we should solve one machine scheduling problem with ORT and ODD for the bottleneck machine. For this purpose, we propose two heuristics, earliest operation due date (EODD) and pairwise exchange (PX). The EODD rule is quite obvious. And PX heuristic is developed based on the fact that feasibility is maintained by pairwise exchange between operations which are adjacent and have no precedence constraints in a feasible sequence. For each machine with n operations to be performed, we have at most n-1 exchanges to consider at any stage of the search. In order to determine whether an exchange will lead to an improvement, consider two adjacent operations ik and jr in the current sequence with pq be the operation preceding ik and st the operation succeeding jr. Let W_{st} be the earliest start time of processing the operation st before exchange and W'_{st} the earliest start time after exchange. When $W_{st} > W'_{st}$, the exchange will result in the latest job being completed earlier, thus reducing its lateness. And it is clear that lateness of jr improves because jr start earlier after the exchange. However, the lateness of ik after the exchange, L'_{ik}, may increase. In order to ensure that $L_{\max}$ does not increase, L'_{ik} must be less than $L_{\max}$.

It is important to note that if the operation with maximum lateness exists before ik in the sequence, the exchange cannot improve $L_{\max}$. Thus we only consider the operations in the sequence up to the latest operation. Based on the above facts, the detailed procedure of PX heuristic can be stated as follows.

Step 1. Generate an initial feasible solution by sequencing all operations in ascending order of ODD. We randomly select one to break ties among the operations with the same value of ODD. Set current best objective function value (*C_OBJ*) to the value of $L_{\max}$ for this sequence. Set this sequence as the current best sequence (*C_BEST*).
Step 2. For the current best sequence, *C_BEST*, examine pairwise exchanges which satisfy the conditions of $W_{st} > W'_{st}$ and $W'_{st} - S_{ik,st} - d_{ik} < L_{\max}$. If the exchange leads to an infeasible solution due to precedence constraints or a poorer solution, it is ignored.
Step 3. If the best one among the generated sequences has a smaller $L_{\max}$ value than *C_OBJ*, set the sequence to *C_BEST* and update *C_OBJ*.
Step 4. If there are no exchanges leading to improvement, stop. Otherwise, go to step 2.

Consequently, we have two variants of SB heuristic, we call them SB/EODD and SB/PX.

3.3 Critical Exchange Heuristic

It is important to note that the critical exchange (CX) heuristic proposed by Chu *et al*. (1998) for the CJSP can be directly applied to the GJSP by setting an appropriate arc cost in the DGR. In the ordinary graph representing a

schedule, the directed path with the largest weighted length joining node 0 to node *ik* is called the critical path related to node *ik*. If node *ik* is $0,^*$ we call this path the *critical path of the ordinary graph*. We refer to the disjunctive arcs belonging to the critical paths as the *critical disjunctive arcs* (*CDAs*), the other ones being the *non-critical disjunctive arcs*.

It is clear that the length of the critical path can be reduced by appropriately reversing the direction of some disjunctive arcs. It is important to point out that the approach proposed by Chu *et al*. (1998) also guarantees for the GJSP that the critical path is improved at each iteration, except if the disjunctive arc to be reversed does not belong to all the critical paths. In this case, at least as many iterations as the number of critical paths are necessary.

As mentioned above, reversing the direction of a disjunctive arc leads to a new feasible schedule if and only if the resulting graph is acyclic. Note that reversing the direction of a non-critical disjunctive arc cannot reduce the makespan because the critical path of the initial graph is still a path from node 0 to node 0^* in the resulting graph. Thus, only the *CDAs* need to be considered to improve a schedule. We introduce a theorem that is a basis of the critical exchange heuristic.

THEOREM 1. The ordinary graph obtained by reversing a *CDA* in an acyclic ordinary graph is still acyclic. (It is quite clear, so the proof is omitted)

Theorem 1 shows that reversing the direction of a *CDA* in an acyclic graph leads to a graph which is still acyclic, and thus represents another schedule. In other words, theorem 1 guarantees that we can reverse the direction of any *CDA* without checking the cyclicity, assuming that the initial ordinary graph is acyclic. Let C be the makespan of the current schedule. And $C(ik, jr)$ be the makespan of the schedule after reversing a critical disjunctive arc (*ik*, *jr*). Then the CX heuristic can be summarized as follows.

Step 1. Generated an initial schedule and derive its ordinary graph.

Step 2. Compute a critical path in this ordinary graph and identify the *CDAs*.

Step 3. For each *CDA* (*ik*, *jr*), compute the makespan for the schedule after reversing the *CDA* and get the following value.

$$C^* = \min_{(ik,jr)} \{C(ik, jr),\ (ik, jr) \in E \text{ be a } CDA\}$$

$$(ik, jr)^* = \arg\min_{(ik,jr)} \{C(ik, jr),\ (ik, jr) \in E \text{ be a } CDA\}$$

Table 2. Performance of the algorithms for the 6 groups of problems.

Problem size	Set of data	NS-CX		LP-CX		SB/EODD-CX		SB/PX-CX	
		MS	CPU time	MS	CPU time	MS	CPU time	MS	CPU time
10x10x10	1	1581	0.22	1330	0.22	1230	0.44	1072*	0.44
	2	1331	0.38	1243*	0.17	1450	0.38	1549	0.71
	3	1177	0.27	1176	0.11	1096*	0.16	1118	0.16
	4	1152*	0.33	1271	0.44	1214	0.17	1185	0.22
10x15x15	5	1790	0.55	1811	0.44	1505*	0.61	1664	0.33
	6	2016	0.44	1889	0.60	2136	0.17	1717*	0.38
	7	2679	0.55	1452*	0.49	1581	0.33	1466	0.28
	8	1503	0.55	2016	0.55	2012	0.22	1382*	0.38
10x20x20	9	2250	0.71	2527	0.65	1859*	1.64	1947	1.04
	10	2445	0.77	2698	0.55	2914	0.33	2059*	0.65
	11	3153	1.10	3469	0.55	2333	0.60	2300*	0.44
	12	2040	0.77	2788	0.44	2031*	1.70	2151	0.33
30x10x10	13	2932	6.20	2590*	7.80	2888	1.26	2949	0.66
	14	2801	5.39	2579*	8.07	2607	2.53	2899	1.65
	15	3326	4.95	2555	8.19	2438*	1.54	2502	1.42
	16	2651	6.92	2649	8.18	2827	1.26	2437*	1.92
30x15x15	17	3366	17.69	3616	23.45	3228*	0.82	3580	1.64
	18	3834	16.09	3600	22.96	3374	2.91	3119*	2.20
	19	3987	16.32	3186*	23.02	3446	1.92	3794	2.48
	20	3605	16.81	3202*	22.57	3281	2.63	3453	1.43
30x20x20	21	5952	30.76	3843*	52.40	3997	2.86	4114	2.58
	22	4834	30.87	4304*	51.41	4388	6.05	5193	2.97
	23	4182	32.79	4075	52.67	3921*	1.48	3966	2.19
	24	5381	32.02	3879	52.62	3559*	2.75	4240	1.93

Step 4. If $C^* < C$, then reverse arc $(ik, jr)^*$, update the value of C and go to Step 2, otherwise stop the procedure.

4. COMPUTATIONAL EXPERIENCES

This section evaluates the effectiveness of the proposed algorithms through a comparative study. For this purpose, the proposed algorithms are programmed in C language and run on an IBM PC Pentium III with 600 MHz microprocessor running Windows 2000. The four two-phase heuristics are first tested on some numerical problems randomly generated. The results of the numerical computations are given in table 2. Columns 'MS' give the value of the makespan corresponding to the resulting schedule obtained by applying each algorithm. Columns 'CPU time' contain the computational time required to get the initial schedule and to improve the makespan of the initial schedule by applying the second phase. Problem size referred to as 10x10x10 means 10 jobs, 10 operations per job and 10 machines. The sequence dependent setup times and processing times of each operation are generated randomly from a uniform distribution in the range of [5, 50]. In the table, asterisk indicates the best one among the solutions obtained by each of the four algorithms. From the table 2, It can be observed that any algorithm can not outperform the others in all problem instances.

Another set of tests are carried out using 12 groups of data where the setup time and processing time of the operations are generated at random following a uniform distribution on [5, 50]. Each group of data is composed of 20 examples. Four initial schedules are generated for each problem using respectively proposed heuristics, i.e. NS, LP, SB-EODD and SB-PX. The average results for each group are presented in table 3. In table 3, problem size, Initial makespan (Init. MS), improved makespan after applying CX heuristic (After CX), and CPU time (in seconds) are given. The figures in parentheses denote the improvements by the second phase of each heuristic which is measured in terms of relative improvement ratio over the solutions obtained in the first phase and are expressed in percentages rounded to two decimal places. From the table 3, the solutions obtained by applying the first phase of each algorithm are significantly improved by applying the CX heuristic in most of the cases. It can be observed that the best final solutions are not always the ones derived from the best initial solution.

5. CONCLUSIONS

In this paper, a general job shop scheduling problem characterized by sequence dependent setup times and

Table 3. Performance of the algorithms for the 12 groups of problems.

	NS-CX			LP-CX			SB/EODD-CX			SB/PX-CX		
Problem size	Init. MS	After CX	CPU time	Init. MS	After CX	CPU time	Init. MS	After CX	CPU time	Init. MS	After CX	CPU time
10x10x10	1844	1374 (25.5)	0.34	1663	1250 (24.8)	0.29	1772	1193 (32.7)	0.30	1631	1194 (26.8)	0.28
10x15x15	2832	1988 (29.8)	0.56	2830	1898 (32.9)	0.52	2416	1724 (28.6)	0.53	2200	1565 (28.9)	0.47
10x20x20	3873	2560 (33.9)	0.85	3725	2606 (30.0)	0.55	2932	2228 (24.0)	0.69	2756	2008 (27.1)	0.53
20x10x10	2639	2023 (23.3)	2.07	2172	1801 (17.1)	2.52	2540	1927 (24.1)	0.75	2729	2010 (26.3)	1.05
20x15x15	3958	2837 (28.3)	4.24	3058	2447 (20.0)	4.91	3607	2644 (26.7)	1.75	3662	2703 (26.2)	1.80
20x20x20	5711	3968 (30.5)	4.74	3932	3126 (20.5)	3.88	4376	3286 (24.9)	1.86	4481	3215 (28.3)	2.78
30x10x10	3643	2835 (22.2)	5.83	3076	2509 (18.4)	8.33	3328	2536 (23.8)	1.81	3225	2594 (19.6)	1.25
30x15x15	5175	3681 (28.9)	16.07	4083	3296 (19.3)	23.71	4156	3339 (19.7)	1.97	4087	3386 (17.2)	1.96
30x20x20	6989	4941 (29.3)	32.14	4806	3801 (20.9)	52.40	5078	4030 (20.6)	2.92	5438	4379 (19.5)	3.38
40x10x10	4751	3584 (24.6)	12.52	4108	3555 (13.5)	16.78	3973	3229 (18.7)	1.99	3960	3300 (16.7)	1.98
40x15x15	6790	5020 (26.1)	31.93	5320	4574 (14.0)	50.50	5091	4188 (17.7)	3.32	4718	3999 (15.2)	2.96
40x20x20	8936	6085 (31.9)	68.22	6388	5140 (19.5)	113.01	6358	5211 (18.0)	5.55	6175	5182 (16.1)	5.05

reentrant work flows is studied. With the disjunctive graph model, scheduling a general job shop is equivalent to selecting directions from the disjunctive arcs. The makespan of a schedule equals the length of the critical path on the corresponding graph.

We present four kinds of solution methods which are composed of two phases. The obtained solutions in the first phase are substantially improved by reversing the direction of some critical disjunctive arcs of the graph in the second phase. The results of computational experiments indicate that the proposed two-phase heuristic approaches give good solutions within a reasonable computational time.

ACKNOWLEDGMENT

This work was supported by Hankuk University of Foreign Studies Research Fund of 2008.

REFERENCES

Adams, J., Balas, E., and Zawack, D. (1988), The shifting bottleneck procedure for job shop scheduling, *Management Science*, **3**, 391-401.

Ahn, S. H. (1995), A study on algorithm to minimize makespan of sequenc-dependent jobs, *Journal of the Korean Institute of Management Science*, **12**, 77-87.

Artigues, C. and Roubellat, F. (2002), An efficient algorithm for operation insertion in a multi-resource jobshop schedule with sequence-dependent setup times, *Production Planning and Control*, **13**, 175-186.

Asano, M. and Ohta, H. (1999), Scheduling with shutdowns and sequence dependent set-up times, *International Journal of Production Research*, **37**, 1661-1676.

Balas, E. (1969), Machine sequencing via disjunctive graphs: an implicit enumeration approach, *Operations Research*, **17**, 941-957.

Balas, E., Lenstra, J. K., and Vazacopoulos, E. (1995), The one machine problem with delayed precedence constraints and its use in job shop scheduling, *Management Science*, **41**, 94-109.

Blanco, L., Dell'olmo, P., and Giordani, S. (1999), Flow shop no-wait scheduling with sequence dependent set-up times and release dates, *INFOR Journal*, **37**, 3-11.

Chu, C., Proth, J. M., and Wang, C. (1998), Improving job-shop schedules through critical pairwise exchanges, *International Journal of Production Research*, **36**, 683-694.

Conway, R. W., Maxwell W. L., and Miller L. W. (1967), *Theory of Scheduling*, Addison-Wesley, Reading, MA.

Corwin, B. D. and Esogbue, A. O. (1974), Two machine flow shop scheduling problems with sequence dependent setup times : A dynamic programming approach, *Naval Research Logistics Quarterly*, **21**, 515-524.

Flynn, B. B. (1987), Repetitive lots: the use of a sequence-dependent set-up time scheduling procedure in group technology and traditional shops, *Journal of Operations Management*, **7**, 203-216.

Gupta, J. N. D. (2002), An excursion in scheduling theory: an overview of scheduling research in the twentieth century, *Production Planning and Control*, **13**, 105-116.

Gupta, J. N. D. (1986), The two machine sequence dependent flow shop scheduling problem, *European Journal of Operations Research*, **24**, 439-446.

Gupta, S. K. (1982), *N* jobs and *m* machines job-shop problems with sequence dependent setup times, *International Journal of Production Research*, **20**, 643-656.

Hwang, H. and Sun, J. U. (1998), Production sequencing problem with re-entrant work flows and sequence dependent setup times, *International Journal of Production Research*, **36**, 2435-2450.

Kagr, L. L. and Thompson, G. L. (1964), A heuristic approach to the traveling salesman, *Management Science*, **11**, 225-248.

Kim, S. C. and Bobrowski, P. M. (1994), Impact of sequence-dependent setup time on job shop scheduling performance, *International Journal of Production Research*, **32**, 1503-1520.

Norman, B. A. (1999), Scheduling flowshops with finite buffers and sequence-dependent setup times, *Computers and Industrial Engineering*, **36**, 163-177.

Ovacik, I. M. and Uzsoy, R. (1995), Rolling horizon procedures for dynamic parallel machine scheduling with sequence-dependent setup times, *International Journal of Production Research*, **33**, 3173-3192.

Ovacik, I. M. and Uzsoy, R. (1997), *Decomposition Methods for Complex Factory Scheduling Problems*, Norwell, MA: Kluwer Academic.

Radhakrishnan, S. and Ventura, J. A. (2000), Simulated annealing for parallel machine scheduling with earliness-tardiness penalties and sequence-dependent set-up times, *International Journal of Production Research*, **38**, 2233-2252.

Sun, J. U. and Hwang, H. (2001), Scheduling problem in a two-machine flow line with the *N*-step prior-job-dependent set-up times, *International Journal of Systems Science*, **32**, 375-385.

Sun, X., Noble, J. S., and Klein, C. M. (1999), Single-machine scheduling with sequence dependent setup to minimize total weighted squared tardiness, *IIE Transactions*, **31**, 113-124.

Uskup, E. and Smith, S. B. (1975), A branch-and-bound algorithm for two-stage production-sequencing problems, *Operations Research*, **23**, 118-136.

Uzsoy, R. and Wang, C. S. (2000), Performance of decomposition procedures for job shop scheduling problems with bottleneck machines, *International Journal of Production Research*, **38**, 1271-1286.

Wilbrecht, J. K. and Prescott, E. (1969), The influence of set-up time on job shop performance, *Management Science*, **16**, 274-280.

Zhu, X. and Wilhelm, W. (2006), Scheduling and lot sizing with sequence-dependent setup: A literature review. *IIE Transactions*, **38**, 987-1007.

Zoghby, J., Barnes, J. W., and Hasenbein, J. J. (2005), Modeling the reentrant job shop scheduling problem with setups for metaheuristic searches. *European Journal of Operational Research*, **167**, 336-348.