Journal of Applied Sciences 8 (20): 3621-3629, 2008
ISSN 1812-5654

Hybrid Electromagnetism-Like Algorithm for the Flowshop Scheduling with Sequence-Dependent Setup Times

[1]M. Mirabi, [1]S.M.T. Fatemi Ghomi, [2]F. Jolai and [3]M. Zandieh
[1]Department of Industrial Engineering, Amirkabir University of Technology, Hafez Ave., No. 424, 15916-34311, Tehran, Iran
[2]Department of Industrial Engineering, Faculty of Engineering, University of Tehran, Tehran, Iran
[3]Department of Industrial Management, Faculty of Management and Accounting, Shahid Beheshti University, Tehran, Iran

Abstract: This study investigate the permutation flowshop scheduling problem in which there are sequence dependent setup times on each machine, commonly known as the SDST flowshop. The optimization criteria considered is the minimization of the makespan or C_{max}. Many heuristics and meta-heuristics have been successfully applied to this kind of problem before like genetic algorithm, tabu search and greedy algorithm and the objective of this study is to assess their effectiveness in a more realistic and complex environment. We present a hybrid electromagnetism-like (HEM) algorithm for the permutation flowshop scheduling with sequence dependent setup times that have shown superior performance against other meta-heuristics when applied to proposed problem. The proposed HEM algorithm benefits of a new concept named priority assigning to calculate electrostatic force and also it implements a new formulation for solution charge. Using a good approach for acquiring the initial solutions and also some effective local searches to finding neighborhood solutions are other novelties of the HEM. For evaluating the proposed algorithm we have coded several well-known algorithms for SDST flowshop. All methods including HEM are tested on the randomly instances and results indicate that HEM is very competitive with the existing best-performing algorithms.

Key words: Scheduling, flowshop, sequence-dependent setup times, hybrid meta-heuristics, electromagnetism-like algorithm

INTRODUCTION

The flowshop scheduling problems (FSP) have been studied for over five decades. The classical flowshop problem with the makespan minimization criterion has always attracted the attention of researchers because of its applications in practice. The flowshop problem is easy to describe and formulate, yet computationally it is rather challenging. Therefore, this problem has inspired the development of a number of solution procedures (Ekşioğlu *et al*., 2008).

In an m machine flowshop, there are m stages in series, where there exist one or more machines at each stage. Each job has to be processed in each of the m stages in the same order. That is, each job has to be processed first in stage 1, then in stage 2 and so on. Processing times for each job in different stages may be different.

In the flowshop literature, one can find an overwhelming number of papers for the regular flowshop problem with the objective of minimizing the maximum completion time across all jobs. However, the sequence-dependent setup time flowshop problem (SDST flowshop problem in short) has attracted much less attention. Setup is sequence-dependent if its duration depends on both the current and the immediately preceding job and is sequence-independent if its duration depends only on the current job to be processed.

The objective in flowshop scheduling problems is to find a sequence for processing the jobs on the machines so that a given criterion is optimized. This yields a total of n! possible orderings of the operations on each machine and a total of $(n!)^m$ possible processing sequences. In flowshop scheduling research usually only so called permutation sequences are considered, where the processing order of operations is the same for all machines. Here, we also adopt this restriction. In this study, we consider sequence dependent flowshop scheduling problem with the makespan minimization criterion.

Corresponding Author: S.M.T. Fatemi Ghomi, Department of Industrial Engineering, Amirkabir University of Technology, Hafez Ave., No. 424, 15916-34311, Tehran, Iran
Tel: +982166413034 Fax: +982166413025

Regarding the computational complexity, the SDST flowshop with the C_{max} objective has been shown to be NP-hard even when m = 1 and also when m = 2 and setups are present only on the first or second machine (Ruiz and Stutzle, 2008). For m = 1, the SDST flowshop is known to be a special case of the traveling salesman problem (TSP) that is also well known to be NP-hard which means that an efficient algorithm for solving the problem to optimality is unavailable.

Compared to the regular flowshop, on which hundreds of papers have been published, the literature on the SDST counterpart is scarce. In the pioneering work of Johnson, he author proposed a simple rule to obtain optimal sequences for the permutation flowshop problem (PFSP) with two machines (Ruiz *et al*., 2005). This study raised significant interest in the PFSP and was followed by several attempts for solving the PFSP with more than two machines. Due to the NP-completeness of the PFSP, researchers have mainly focused on the development of effective heuristics and meta-heuristics. For a recent review and evaluation of PFSP heuristics and metaheuristics, the reader can refer to Ruiz and Concepcion (2005). Stafford and Tseng (1990) reported minor corrections to the SG/SDST model and showed that the revised model (SGST/SDST) was robust with regard to the triangular inequality relationship of setup times. Rios-Mercado and Bard (1988) showed that the SGST/SDST model performed better than their new MILP model for the SDST flowshop. Tseng and Stafford (2001) extended the Stafford MILP model to solve both the SDST and the SDST/NIQ flowshop problems.

Ríos-Mercado and Bard (1999a) addressed the sequence dependent flowshop problem with makespan criterion (denoted as $F_m/ST_{sd}/C_{max}$). In the first study (Ríos-Mercado and Bard, 1999a), they presented a branch and bound algorithm, incorporating lower and upper bounds and dominance elimination criterion, to solve the problem. They provided test results for a wide range of problem instances. In the second study (Ríos-Mercado and Bard, 1999b), they proposed a heuristic for the same problem, which transforms an instance of the problem into an instance of the traveling salesman problem by introducing a cost function that penalizes both large setup times and bad fitness of a given schedule. Ruiz *et al*. (2005) proposed two genetic algorithms for the same problem and showed that their heuristics outperform that of Ríos-Mercado and Bard (1999b) and others. Ruiz and Stutzle (2008) presented two simple local search based iterated greedy algorithms and showed that their algorithms perform better than those of Ruiz *et al*. (2005). Ríos-Mercado and Bard (2003) studied the polyhedral structure of two different MIP formulations for the same problem. One is related to the asymmetric traveling salesman problem and the other is derived from an earlier proposed model. The two approaches were evaluated using a branch and cut algorithm, which indicated that the approach related to the asymmetric traveling salesman problem was inferior in terms of the computational time. Stafford and Tseng (2002) also proposed two MILP models, which are based on the study of Tseng and Stafford (2001), for the same problem. The MIP models proposed by Ríos-Mercado and Bard (2003) and Stafford and Tseng (2002) were independently developed and hence, remain to be compared to each other. Tseng *et al*. (2005) developed a penalty-based heuristic algorithm for the same problem and compared their heuristic with an existing index heuristic algorithm. The $F_m/ST_{sd}/C_{max}$ problem was studied by Norman (1999) where there exist buffers with finite capacity between machines. He proposed a tabu search based heuristic and compared it with some other methods. Computational experiments showed the effectiveness of the tabu search approach.

Sun and Hwang (2001) addressed a related problem of $F_2/ST_{sd}/C_{max}$, where the setup times are present only on the second machine and the setup time of a job depends on k (k>1) immediately preceding jobs. They proposed a dynamic programming formulation and a genetic algorithm for the problem.

Allahverdi *et al*. (2008) have put together a much more updated and comprehensive review of scheduling research with setup times in which other relevant papers related to the SDST flowshop can be found.

Recently, EM algorithm has been used for optimization problems like scheduling. The approach starts with a randomly selected point from the feasible region for a given optimization problem. EM employs an attraction-repulsion mechanism to move points (particles) towards the optimal solution. Each point (particle) is treated as a solution and has a charge. A better solution contains a stronger charge. The charge of each point relates to the objective function to be optimized. A theoretical study of this EM analysis and a modification for convergence to the optimal solution are presented in Birbil *et al*. (Chang *et al*., 2007). However, this study only deals with continuous optimization problems. Debels *et al*. (2006) integrated a scatter search with EM for the solution of resource constraint project scheduling problems. This is the first study that includes an EM type methodology for the combinatorial optimization problem. Their experimental results show that the hybrid method of incorporating EM type analysis outperforms the current best solution available in literature.

Though EM algorithm is designed for solving continuous problems, the algorithm can be extended to solve scheduling problems. Chang *et al.* (2007) applied the random-key approach to represent a schedule incorporated with the EM methodology to solve a single machine scheduling problem and the objective is to minimize the total sum of earliness and tardiness penalties. In this study we apply priority assigning idea to incorporate EM algorithm for solving flowshop scheduling with sequence dependent setups and makespan criterion.

FORMULATION

We present a MILP model for this SDST flowshop scheduling problem that has been derived from the scheduling literature (Das and Canel, 2005).

Notation

n : The number of jobs
m : The number of machines
i : 1, 2, … , n, subscript of the ith job
j : 1, 2, … , m, subscript of the jth machine
p_{ij} : Processing time of job i on machine j
s_{ikj} : Setup time from job i to job k on machine j (i=0 indicates setup time of the job scheduled first)
c_{ij} : The completion time of job i on machine j
x_{ik} : 1, if job k follows job i; 0, otherwise

Objective function: The objective is to minimize the makespan which is given by the completion time of the last job on the last machine.
Minimize makespan = c_{nm}

Constraint sets:

$$c_{i1} \geq p_{i1},\ i = 1, 2, \ldots, n \tag{1}$$

Constraint (1) is required to ensure that all jobs are scheduled and that the completion times of any job i on machine 1 is at least as great as the processing time for that job on the machine.

$$c_{ij} \geq c_{i[j-1]} + p_{ij},\ i = 1, 2, \ldots, n;\ j = 2, 3, \ldots, m. \tag{2}$$

Processing of job i cannot begin on machine j, unless its processing is completed on the previous machine (j-1). Constraint (2) ensures that the completion time of job i on machine j, must be greater than the completion time of the same job on the previous machine (j-1), by at least the processing time required for job i on machine j.

$$c_{ij} - c_{kj} + Mx_{ik} \geq s_{kij} + p_{ij} \tag{3}$$

$$c_{kj} - c_{ij} + M[1 - x_{ik}] \geq s_{ikj} + p_{kj} \tag{4}$$

where, k>i≥l,

$$i = 1, 2, \ldots, n;\ k = 1, 2, \ldots, n;\ j = 1, 2, \ldots, m;$$

and M is a very large No.

Constraints (3) and (4) ensure that for any given sequence of n jobs, only one of the constraints is binding. It assures the precedence relationship between jobs.

$$\sum_{i=1}^{n} x_{ik} = 1,\ k = 1, 2, \ldots, n \text{ for } i \neq k \tag{5}$$

$$\sum_{k=1}^{n} x_{ik} = 1,\ i = 1, 2, \ldots, n \text{ for } i \neq k \tag{6}$$

Constraints (5) and (6) ensure that only one job can follow a job in the schedule.

ELECTROMAGNETISM-LIKE ALGORITHM

Introduction of electromagnetism-like algorithm: Birbil and Fang propose a so-called electromagnetism-like (EM) optimization heuristic for unconstrained global optimization problems, i.e. the minimization of non-linear functions (Chang *et al.*, 2007). In a multi-dimensional solution space where each point represents a solution, a charge is associated with each point. This charge is related to the objective function value associated with the solution. As in evolutionary search algorithms, a population, or set of solutions, is created, in which each solution point will exert attraction or repulsion on other points, the magnitude of which is proportional to the product of the charges and inversely proportional to the distance between the points (Coulomb's Law). The principle behind the algorithm is that inferior solution points will prevent a move in their direction by repelling other points in the population and that attractive points will facilitate moves in their direction. This can be seen as a form of local search in Euclidian space in a population-based framework. The main difference with existing methods is that the moves are governed by forces that obey the rules of electromagnetism (Debels *et al.*, 2006).

EM simulates the attraction-repulsion mechanism of electromagnetism theory which is based on Coulomb's law. Each particle represents a solution and the charge of each particle relates to its solution quality. The higher charge the particle has the better solution quality of the particle. Moreover, the electrostatic force between two

point charges is directly proportional to the magnitude of each charge and inversely proportional to the square of the distance between the charges. The fixed charge of particle i is shown as follows:

$$q^i = \exp\left(-n\frac{f(x^i)-f(x^{best})}{\sum_{k=1}^{m}\left(f(x^k)-f(x^{best})\right)}\right), \forall i \qquad (7)$$

where, q^i is the charge of particle i, $f(x^i)$,$f(x^{best})$ and $f(x^k)$ denote the objective values of particle i, the best solution and particle k, respectively. Finally, m is the population size. The solution quality or charge of each particle determines the magnitude of an attraction and repulsion effect in the population. A better solution encourages other particles to converge to attractive valleys while a bad solution discourages particles to move toward this region. These particles move along with the total force and so diversified solutions are generated. The following formulation is the force of particle i:

$$F^i = \begin{cases} (x^j - x^i)\dfrac{q^i q^j}{\left\|x^j - x^i\right\|^2} & \text{if } f(x^j) < f(x^i) \\ (x^i - x^j)\dfrac{q^i q^j}{\left\|x^j - x^i\right\|^2} & \text{Else } f(x^j) \geq f(x^i) \end{cases}, \forall i. \qquad (8)$$

The fundamental procedure of EM includes initialization, local search, calculating total force and moving the particles. The generic pseudo-code for the EM is as follows (Chang *et al*., 2007):

Algorithm 1: EM ()
1 : Initialize ()
2 : While (has not met stop criterion) do
3 : Local Search ()
4 : Calculate the total force F ()
5 : Move the particle by F ()
6 : Evaluate the particles ()
7 : End while

Proposed hybrid electromagnetism-like algorithm: Flowshop scheduling problem can be regarded as a hard optimization problem. A simple EM may not perform well in this situation. Therefore, the EM developed in this study benefits of a new approach for the initial solutions, acceptance criteria and a local search. EM hybridizes with the modified NEH heuristic proposed by Ruiz *et al*. (2005). Besides the modified NEH, it also hybridizes with the local optimizer. We use three different search neighborhoods as pairwise interchange neighborhood, forward insertion neighborhood and backward insertion neighborhood (Gupta and Smith, 2006). One step in the local search is to decide whether the new sequence is accepted or not as the incumbent solution for the next iteration. A pure descent criterion would be to accept solutions with better objective function values. However, this acceptance criterion is prone to stagnation. As an alternative, we consider an acceptance criterion that is frequently used in simulated annealing (SA) algorithms. The hybrid system starts from determining whether a new solution obtained from one of initial solution using local search is accepted by SA or moved by EM.

The algorithm 2 is the pseudo code of the main procedure of the hybrid framework.

Algorithm 2: HEM ()
1 : Initialize ()
2 : Priority assignment of initial solutions
3 : While (has not met stop criterion) do
4 : Initialize Max-iterations, Temp-start
5 : Set Count = 1, T = Temp-start
6 : B ← calculates the average makespan of all solutions ()
7 : x^c ← solution related to the worst makespan among all solutions ()
8 : x^{New} ← solution related to the best makespan among all solutions ()
9 : LocalSearch()
Let the neighboring sequence be called x^{Nei}
10 : Priority assignment of x^{Nei}
11 : Compute $C_{max}(x^{Nei})$
12 : If $C_{max}(x^{Nei}) \leq B$
13 : $x^c \leftarrow x^{Nei}$, go to 26
14 : Else If $C_{max}(x^{Nei}) > B$
15 : Set T= Temp-start/log (1+Count)
16 : With probability $e^{-\Delta/T}$set $x^c \leftarrow x^{Nei}$, go to 26
17 : With probability $1\text{-}e^{-\Delta/T}$
18 : Move x^{Nei} by EM () and let the new solution be called x^{New}
19 : Priority assignment of x^{New}
20 : Compute $C_{max}(x^{New})$
21 : If $C_{max}(x^{New}) \leq B$
22 : $x^c \leftarrow x^{new}$, go to 26
23 : Else if $C_{max}(x^{New}) > B$, go to 9
24 : End if
25 : End if
26 : Increment Count by 1
27 : If Count < Max-iterations, go to step 6
28 : End while
29 : Output the best sequence or x^{best}

By line 9 we mean randomly generate a neighboring solution of x^{New} using either the interchange neighborhood, forward insertion neighborhood or backward insertion neighborhood. According to the algorithm 2 (algorithm 2, line 1), we initiate the solutions in the population. Then, the neighborhood search procedure is implemented before the EM procedure (algorithm 2, line 9). To determine which solution is good or inferior one, an average objective value B is calculated. Then, if the solution is not worse than B, it is accepted and substituted with the worst solution in the population (algorithm 2, lines 12-13). Otherwise, this solution is accepted with probability of $e^{-\Delta/T}$ and substituted with the worst solution (algorithm 2, line 17) or moved by modified EM algorithm with probability of $1-e^{\Delta/T}$ (algorithm 2, line 19). After solutions are obtained, their makespan can be calculated. The best makespan is final solution. Finally, the initialization, priority assigning, solution charge, calculated total force and move are modified. Following discusses these topics in details.

Initialization: The initial solution for EM is ideally generated by a high performance constructive heuristic. For the sequence dependent flowshop scheduling with makespan criterion, we use the NEHT-RMB heuristic and a modified NEHT-RMB heuristic proposed by Ruiz *et al.* (2005) for the initialization of the population. Recall that NEH is an insertion heuristic, where at each step the next unscheduled job is tentatively inserted in each possible position of some partial solution. The job is then finally inserted into the position where the objective function takes the lowest value. For executing such an insertion heuristic, the jobs need to be ordered in some way. For more details how this is done in NEHT-RMB and modified NEHT_RMB (Ruiz *et al.*, 2005). We obtain m initial solutions based on this method.

Priority assigning: In this step we assign one random variable x^i_k between 0 and 1 to each job k in each solution i. For example consider one problem with 4 jobs numbered 1 to 4. Assume the second initial solution is represented by (1, 4, 3, 2). It means job 1 is the first job in the sequence, job 4 is second, job 3 is third and job 2 is the last. We assign one random variable between 0.75 and 1 to job 1, one between 0.5 and 0.75 to job 4, one between 0.25 and 0.5 to job 3 and finally one between 0 and 0.25 to job 2. One of the results can be shown as follows:

$$x^2_1 = 0.89,\ x^2_4 = 0.54,\ x^2_3 = 0.48,\ x^2_2 = 0.11$$

Therefore $x^2 = (0.89, 0.54, 0.48, 0.11)$. Also if there are n jobs in each sequence, one random variable between (n-1)/n and n is assigned to the first job, one between (n-2)/n and (n-1)/n to the second and so on. Finally random variable of the last job is between 0 and 1/n. Hence if there are m initial solutions, there are m random variables for each job i (i = 1,…,n).

Solution charges, electrostatic force and move: In the previous section it was described that each solution i has one vector of random variable denoted as x^i including n random variables from x^i_1 to x^i_n. Therefore $C_{max}(x^i)$ is equivalent to C_{max}(solution(i)). Let the force exerted on neighborhood solution (denoted as x^{Nei} in algorithm 2, line 9) by current solution i use the fixed charge of q_i (related to the fixed charge of solution i indicated in 7). We have:

$$q^i = \frac{B - C_{max}(x^i)}{\sum_{k=1}^{m}(B - C_{max}(x^i))},\ \forall i = 1,...,m \qquad (9)$$

where B is the average makespan of all solutions i (i = 1,…,m). It is clear that

$$\sum_{i=1}^{m} q^i = 0$$

After the q_i is obtained, we calculate the force on x^{Nei} by other solutions i. To calculate the electrostatic forces imposed by all solution for x^{Nei}, we obtain electrostatic forces imposed to each particle of x^{Nei} (particle means x_1^{Nei}, x_2^{Nei},…, x_n^{Nei}) as follows (related to the force of particle i indicated in 8):

$$F_k^{Nei} = \sum_{i=1}^{n} F_k^i = \sum_{i=1}^{n} (x_k^i \times q^i),\ \forall k = 1,...,n \qquad (10)$$

Therefore,

$$x_k^{New} = x_k^{Nei} + F_k^{Nei},\ \forall k = 1,...,n \qquad (11)$$

We can set upper and lower bounds for x_k^{New}. If x_k^{New} is greater than upper bound or less than lower bound, it is substituted with the value of bounds. Hence we have one x^{New} with new particles. We sort all jobs in x^{New} based on its x_k^{New} in decreasing order and obtain a new sequence of jobs corresponding x^{New}. Thus solution x^{Nei} moves to $x^{Nei}+F_k^{New}$. For example if the solution related to x^{Nei} is represented by (2, 1, 4, 3) and new particles of x^{New} are (0.22, 0.52, 0.43, 0.85), the new solution will be (3, 1, 4, 2). Therefore to obtain x^{New} (algorithm 2, line 19), we follow algorithm 3.

Algorithm 3: Move neighborhood solution by EM ()

1 : For i = 1 to m

2 : $q^i = \frac{B - C_{max}(x^i)}{\sum_{k=1}^{m}(B - C_{max}(x^i))}$

3 : End for
4 : For k = 1 to n
5 : $F_k^{Nei} = \sum_{i=1}^{n} F_k^i = \sum_{i=1}^{n} (x_k^i \times q^i)$
6 : $x_k^{New} = x_k^{Nei} + F_k^{Nei}$
7 : End for
8 : Output x^{New}
9 : Output new sequence based on x^{New}

Finally, in order to maintain the feasibility of each solution, we check the boundary feasibility by the algorithm 4.

Algorithm 4: Check boundary ()
1 : For i = 1 to m do
2 : For k = 1 to n do
3 : If x_k^{New} > upper bound then
4 : x_k^{New} ← upper bound
5 : Else if x_k^{New} < lower bound then
6 : x_k^{New} ← lower bound
7 : End if
8 : End for
9 : End for

Stopping criterion: The stopping criterion of the EM could be a maximum number of iterative cycles, specified CPU time limit, or maximum number of cycles between two improvements of the global best solution. In this study, we use a given number of iterative cycles as the stopping criterion. Therefore, in our experiment setting, the algorithm will terminate when a given number of cycles has been executed. This loop is executed for Itemax = 2500 iterations.

COMPUTATIONAL EXPERIMENTS

Here, the proposed hybrid electromagnetism (referred to as HEM) with the other metaheuristics is compared. These are: iterated greedy heuristic by Ruiz and Stutzle (2008) that will be denoted as IG, hybrid genetic algorithm by Ruiz *et al.* (2005) that will be denoted as HGA and the tabu search algorithm by Ekşioğlu *et al.* (2008), which will be referred to as TS. For evaluating the different algorithms we used the performance measure stated as:

$$PM = \frac{Heu_{sol} - Best_{sol}}{Best_{sol}} \times 100 \quad (12)$$

where, Heu_{sol} is the makespan obtained by a given algorithm and $Best_{sol}$ is the makespan of the best solution obtained by all algorithms. The platform of our experiments is a personal computer with a Pentium-III 1.2 Hz CPU and 256 MB RAM. The programs are coded in MATLAB. All algorithms are compared using different problem sizes (n = 10, 20, 30, 40, 50, 100, 200 and m = 5, 10, 15, 20). For each class of the problem defined by given (n, m), 10 instances of problem are randomly generated. Thus we obtain a total of 280 problem instances. Processing time and setup time are given from Uniform random U(1, 99) and U(1, 9) discrete distributions, respectively. The numerical results are averaged through each ten instances.

The average, minimum and maximum PM values for all algorithms are shown in Table 1. The 'Min' labeled columns show, in subscript, the number of instances for which the algorithm solution was equal to the corresponding $Best_{sol}$. In 'Average' column we show two sub columns including average PM and average time to solve all 10 instances.

With respect to the solutions gained, Table 1 demonstrates that the algorithms have the rank of: 1. HEM, 2. IG, 3. HGA and 4. TS. Figure 1 shows the difference of Heu_{sol}-$Best_{sol}$ for each heuristics in each instance.

It is noticeable that the fastest algorithm is HGA (but gained solutions are not so good) and HEM is a little slower than IG.

Now for more detailed comparison, two algorithms of HEM and IG are considered. In this step, it is desired to stop both algorithms at the same CPU time. The value of this CPU time has been taken the minimum CPU time between two algorithms in Table 1. For example the common CPU time for the first class of problem (n = 10, m = 5) is min (3.5, 2.08) = 2.08. We test the hypothesis that the population corresponding to the differences has mean μ zero. Specifically, we test the (null) hypothesis μ = 0 against the alternative μ > 0. We assume that the

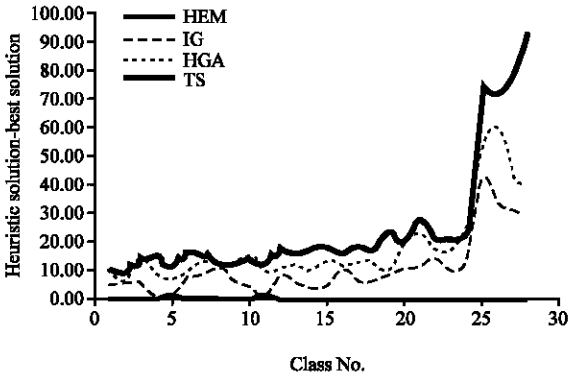

Fig. 1: Differences between heuristic solution and the best solution among all heuristic solutions for each algorithm in each instance

Table 1: PM values for comparison studies between algorithms (times are in second)

			HEM				IG				HGA				TS			
				Average				Average				Average				Average		
Class of			Min	----------		Max	Min	----------		Max	Min	----------		Max	Min	----------		Max
problem	n	M	PM	PM	Time	PM	PM	PM	Time	PM	PM	PM	Time	PM	PM	PM	Time	PM
1	10	5	0_9	0.084	3.50	0.844	0_1	0.656	2.08	1.207	0.230	1.379	1.61	2.579	0.141	1.504	0.40	2.907
2	10	10	0_9	0.051	3.82	0.512	0_1	0.495	2.38	0.864	0.076	0.526	1.67	1.533	0.204	0.860	6.61	1.732
3	10	15	0_{10}	0.000	4.60	0.000	0.122	0.425	3.17	0.744	0.354	1.062	2.64	1.480	0.161	1.030	29.51	1.876
4	10	20	0_7	0.146	6.69	0.604	0_3	0.132	5.10	0.345	0.058	0.648	3.72	1.334	0.040	1.063	92.89	1.753
5	20	5	0_7	0.223	3.53	0.911	0_3	0.199	2.36	0.632	0.045	0.683	2.36	1.218	0.439	1.064	0.39	1.862
6	20	10	0_9	0.071	3.91	0.711	0_1	0.561	2.63	0.860	0.015	0.702	3.03	1.564	0.081	1.056	6.37	1.784
7	20	15	0_{10}	0.000	4.37	0.000	0.055	0.432	2.85	0.787	0.062	0.683	3.23	1.203	0.111	0.833	34.68	1.456
8	20	20	0_{10}	0.000	5.71	0.000	0.120	0.502	4.37	0.697	0.132	0.590	4.53	1.159	0.051	0.588	97.04	1.296
9	30	5	0_8	0.039	4.23	0.296	0_1	0.373	2.90	0.766	0_1	0.653	2.99	1.224	0.055	0.714	0.41	1.809
10	30	10	0_9	0.027	5.22	0.271	0_1	0.237	3.79	0.563	0.303	0.743	3.21	1.129	0.051	0.685	8.92	0.994
11	30	15	0_7	0.145	8.34	0.498	0_3	0.109	5.70	0.347	0.040	0.489	4.02	0.999	0.094	0.626	51.45	1.307
12	30	20	0_9	0.063	8.82	0.627	0_1	0.366	7.01	0.714	0.000	0.466	6.86	1.068	0.135	0.680	119.16	1.247
13	40	5	0_8	0.057	5.64	0.518	0_2	0.278	3.78	0.823	0.047	0.543	3.84	0.912	0.036	0.724	0.43	1.398
14	40	10	0_7	0.108	6.18	0.627	0_2	0.227	4.30	0.636	0_1	0.478	4.92	0.809	0.091	0.745	11.11	1.253
15	40	15	0_8	0.145	8.65	0.976	0_2	0.277	6.59	0.646	0.098	0.614	5.77	0.999	0.171	0.766	54.29	1.353
16	40	20	0_{10}	0.000	13.09	0.000	0.104	0.324	9.81	0.487	0.025	0.343	7.73	0.645	0.024	0.480	135.21	1.011
17	50	5	0_8	0.091	6.79	0.460	0_2	0.283	4.69	0.670	0.149	0.515	3.57	0.809	0.038	0.726	0.39	1.007
18	50	10	0_8	0.073	7.44	0.394	0_1	0.252	5.15	0.516	0_1	0.465	4.63	0.865	0.013	0.610	12.57	1.317
19	50	15	0_7	0.016	11.30	0.095	0_3	0.248	8.48	0.710	0.001	0.280	7.21	0.721	0.029	0.681	69.05	1.408
20	50	20	0_9	0.020	16.04	0.200	0_1	0.292	13.11	0.768	0.042	0.518	10.51	0.850	0.242	0.538	204.54	0.907
21	100	5	0_8	0.009	9.46	0.092	0_1	0.187	7.04	0.587	0_1	0.409	6.69	0.649	0.004	0.491	0.94	0.751
22	100	10	0_7	0.036	18.61	0.259	0_3	0.270	14.79	0.577	0.161	0.333	12.34	0.476	0.139	0.387	17.46	0.774
23	100	15	0_6	0.062	26.64	0.315	0_3	0.212	21.20	0.528	0_1	0.318	20.17	0.632	0.029	0.412	96.55	1.156
24	100	20	0_{10}	0.000	41.87	0.000	0.011	0.186	32.33	0.359	0.101	0.373	25.07	0.586	0.084	0.346	328.44	0.640
25	200	5	0_8	0.024	17.16	0.209	0_2	0.396	14.91	0.876	0.015	0.491	14.24	1.051	0.174	0.670	4.72	1.011
26	200	10	0_8	0.063	34.44	0.321	0_2	0.344	28.31	0.779	0.110	0.582	24.87	0.944	0.183	0.673	30.16	1.465
27	200	15	0_8	0.066	49.51	0.388	0_2	0.326	39.61	0.701	0.090	0.428	31.93	0.927	0.102	0.718	233.56	1.160
28	200	20	0_9	0.016	77.08	0.161	0_1	0.234	68.06	0.572	0.023	0.328	60.85	0.779	0.238	0.770	739.95	1.120

Table 2: Comparison study of performance between HEM and IG

			Ave. MS or ($\overline{X}$)		Ave. SD or (S)					
Class of			----------		----------					
problem	n	m	IG	HEM	IG	HEM	T	υ	t	Sig.
1	10	5	762.70	763.20	2.85	3.81	-0.33	17	1.74	No
2	10	10	1107.06	1109.10	1.77	2.92	-1.89	15	1.75	Yes
3	10	15	1295.43	1290.57	3.01	2.48	3.94	17	1.74	Yes
4	10	20	1613.04	1607.27	5.43	4.06	2.69	17	1.74	Yes
5	20	5	1328.12	1331.91	2.64	3.70	-2.64	16	1.75	Yes
6	20	10	1586.97	1588.08	4.30	4.61	-0.56	18	1.73	No
7	20	15	1874.37	1871.63	3.29	1.30	2.45	12	1.78	Yes
8	20	20	2147.92	2143.46	4.24	4.68	2.23	18	1.73	Yes
9	30	5	1861.46	1855.80	5.41	4.24	2.6	17	1.74	Yes
10	30	10	2168.34	2163.40	5.46	3.61	2.39	16	1.75	Yes
11	30	15	2451.97	2454.12	4.45	2.79	-1.29	15	1.75	No
12	30	20	2713.15	2709.81	6.02	6.77	1.17	18	1.73	No
13	40	5	2426.54	2429.88	2.35	4.47	-2.09	14	1.76	Yes
14	40	10	2702.39	2696.18	6.38	3.91	2.62	15	1.75	Yes
15	40	15	2978.92	2974.50	5.75	5.97	1.69	18	1.73	No
16	40	20	3255.00	3246.19	7.20	2.62	3.64	11	1.8	Yes
17	50	5	2961.38	2954.02	5.55	5.19	3.06	18	1.73	Yes
18	50	10	3273.70	3268.55	7.46	6.75	1.62	18	1.73	No
19	50	15	3519.88	3511.87	7.95	6.06	2.53	17	1.74	Yes
20	50	20	3813.90	3807.91	6.84	4.70	2.28	16	1.75	Yes
21	100	5	5759.07	5745.99	12.89	10.70	2.47	17	1.74	Yes
22	100	10	6035.38	6023.19	14.66	10.93	2.11	17	1.74	Yes
23	100	15	6253.90	6265.47	4.22	13.07	-2.66	11	1.8	Yes
24	100	20	6623.50	6608.14	9.10	14.61	2.82	15	1.75	Yes
25	200	5	11263.60	11246.70	26.41	25.33	1.46	18	1.73	No
26	200	10	11655.90	11634.30	29.77	15.09	2.05	13	1.77	Yes
27	200	15	11868.20	11833.20	28.35	28.82	2.74	18	1.73	Yes
28	200	20	12331.10	12298.30	28.00	23.74	2.83	18	1.73	Yes

Ave: Average, MS: Makespan, SD: Standard deviation, Sig: Significant, Each class of problem contains 10 independent instances

makespan difference is a normal variable and choose the significance level $\alpha = 0.05$. If the hypothesis is true, the random variable:

$$T = (\bar{X}_1 - \bar{X}_2)/\sqrt{(S_1^2/n_1) + (S_2^2/n_2)}$$

has a t distribution with:

$$\upsilon = (S_1^2/n_1 + S_2^2/n_2)^2/(\frac{(S_1^2/n_1)^2}{n_1 - 1} + \frac{(S_2^2/n_2)^2}{n_2 - 1})$$

degrees of freedom. The critical value of c is obtained from the relation Prob ($T > c$) = $\alpha = 0.05$. For example, the first entry in Table 2 corresponds to the sample size = $n_1 = n_2 = 10$, $\mu_0 = 0$, sample mean for IG and HEM are $\bar{X}_1 =$ 762.70 and $\bar{X}_2 =$ 763.20, respectively. Sample standard deviation for IG and HEM are $S_1 = 2.85$ and $S_2 = 3.81$, respectively. Since $t = 1.74 > T = 0.33$, we conclude that the difference is not statistically significant. Table 2 displays HEM outperforms IG in all cases except seven classes of problem (classes 1, 2, 5, 6, 11, 13 and 23) and four of differences are significant (classes 2, 5, 13 and 23). Also in cases that HEM yields better results, all differences are significant except four classes (12, 15, 18 and 25). In fact HEM outperforms IG in 75% of cases and significantly outperforms in 61%.

CONCLUSIONS

By using priority assigning the EM algorithm is able to be applied in solving the scheduling problem. To improve the performance of the EM algorithm, a hybrid method is developed in this research which, the EM benefits of a new approach for the initial solutions, acceptance criteria and a local search. The purpose of this hybrid method is to take advantage of the EM algorithm, SA algorithm and local search.

We study the flowshop scheduling problem in sequence dependent condition to challenge a large number of real world problems. FSP is a hard optimization problem. To solve this problem, HEM is used. Computational results demonstrate the performance of our algorithm compared to some of the strong algorithms recently developed. It is noticeable when the differences between HEM and GA (strongest algorithm between all previous algorithms) are concerned, most of them are also significant in the level $\alpha = 0.05$. It demonstrates the significant strength of HEM to solve scheduling problems compared to previous developed algorithm.

REFERENCES

Allahverdi, A., C.T. Ng, T.C.E. Cheng and Y.M. Kovalyov, 2008. A survey of scheduling problems with setup times or costs. Eur. J. Operat. Res., 187: 985-1032.

Chang, P.C., S.S. Chen and C.Y. Fan, 2007. A hybrid electromagnetism-like algorithm for single machine scheduling problem, Expert Sys. Applied, 10.1016/j.eswa.2007.11.050

Das, S.R. and C. Canel, 2005. An algorithm for scheduling batches of parts in a multi-cell flexible manufacturing system. Int. J. Prod. Econ., 97: 247-262.

Debels, D., B. De Reyck, R. Leus and M. Vanhoucke, 2006. A hybrid scatter search/electromagnetism meta-heuristic for project scheduling. Eur. J. Oper. Res., 169: 638-653.

Ekşioğlu, B., S.D. Ekþioðlu and P. Jain, 2008. A tabu search algorithm for the flowshop scheduling problem with changing neighborhoods. Comput. Industrial Eng., 54: 1-11.

Gupta, S.R. and J.S. Smith, 2006. Algorithms for single machine total tardiness scheduling with sequence dependent setups. Eur. J. Operat. Res., 175: 722-739.

Norman, B.A., 1999. Scheduling flowshops with finite buffers and sequence-dependent setup times. Comput. Ind. Eng., 36: 163-177.

Rios-Mercado, R.Z. and J.F. Bard, 1988. Computational experience with a branch-and-cut algorithm for flowshop scheduling with setups. Comput. Operat. Res., 25: 351-366.

Ríos-Mercado, R.Z. and J.F. Bard, 1999a. A branch-and-bound algorithm for permutation flowshops with sequence-dependent setup times. IIE Trans., 31: 721-731.

Ríos-Mercado, R.Z. and J.F. Bard, 1999b. An enhanced TSP-based heuristic for makespan minimization in a flowshop with setup times. J. Heurist., 5: 53-70.

Ríos-Mercado, R.Z. and J.F. Bard, 2003. The flowshop scheduling polyhedron with setup times. J. Comb. Optim., 7: 291-318.

Ruiz, R. and M. Concepcion, 2005. A comprehensive review and evaluation of permutation flowshop heuristics. Eur. J. Operat. Res., 165: 479-494.

Ruiz, R., C. Maroto and J. Alcaraz, 2005. Solving the flowshop scheduling problem with sequence dependent setup times using advanced metaheuristics. Eur. J. Operat. Res., 165: 34-54.

Ruiz, R. and T. Stützle, 2008. An Iterated Greedy heuristic for the sequence-dependent setup time flowshop problem with makespan and weighted tardiness objectives. Eur. J. Operat. Res., 187: 1143-1159.

Stafford, Jr. E.E. and F.T. Tseng, 1990. On the Srikar-Ghosh MILP model for the N×M SDST flowshop problem. Int. J. Prod. Res., 28: 1817-1830.

Stafford, E. F. and F.T. Tseng, 2002. Two models for a family of flowshop sequencing problems. Eur. J. Operat. Res., 142: 282-293.

Sun, J.U. and H. Hwang, 2001. Scheduling problem in a two machine flow line with the N-step prior-job-dependent set-up times. Int. J. Syst. Sci., 32: 375-385.

Tseng, F.T. and E.E. Stafford Jr., 2001. Two MILP models for the N×M SDST flowshop sequencing problem. Int. J. Prod. Res., 39: 1777-1809.

Tseng, F.T., J.N.D. Gupta and E.F. Stafford, 2005. A penalty-based heuristic algorithm for the permutation flowshop scheduling problem with sequence-dependent set-up times. J. Opl. Res. Soc., 57: 541-551.