Manufacturing Rev. 2014, **1**, 13
© M. Khalili, Published by EDP Sciences, 2014
DOI: 10.1051/mfreview/2014013

Manufacturing Review

Available online at:
http://mfr.edp-open.org

Research Article — Open Access

An electromagnetism-inspired method for a generalized flowshop problem

Majid Khalili*

Department of Industrial Engineering, Islamic Azad University, Karaj Branch, Karaj, Iran

Received 3 November 2013 / Accepted 17 August 2014

Abstract – There are two common simplifying assumptions made in the scheduling of operations. One is that machines are available continuously, and the other is that once a job completes one process it is immediately available for its next. Moreover, in flowshops, it is commonly assumed that jobs must visit all machines. In practice, however, these assumptions may not be justifiable in some applications. This paper deals with a generalized flow shop (GFS) problem with machine availability constraints, transportation times between stages, and machine skipping. The electromagnetism-like method (EM) has been successfully applied to some NP-hard problems and this has motivated us to apply and assess the effectiveness of an EM algorithm in the GFS scenarios. Simulated annealing (SA) and a number of other well-recognized heuristics to the given GFS scheduling problem that minimizes two independent objective functions, namely the total tardiness and the total completion time also has been applied. In order to evaluate the performance of the proposed EM and SA, a set of practical instances has been considered. The related results are analyzed in two parts; in term of the objective functions, and the observed effects on variables in our instances. Extensive experiments and statistical analyses demonstrate that our proposed EM is more efficient than SA and other heuristics with regards to the objective functions considered in this paper.

Key words: Generalized flowshop scheduling, Transportation times, Machine availability, Electromagnetism-like method

1. Introduction

Production scheduling is defined as: determining the process sequence by which a given set of parts are processed on a certain number of machines in order to satisfy some performance measures under the production constraints. Through scheduling, all parts in a manufacturing system can be processed by passing through each machine according to a predetermined sequence. Production scheduling develops into a very active and relevant field of research after the first methodical study by Johnson [1]. Since, many papers with a variety of practical and impractical assumptions have been published; however, there always exists a gap between the theory and practice in the literature for this field.

One of the most thoroughly studied scheduling problems is the flow shop (FS) problem. In regular FS, we have a set of n jobs $\{J_1, J_2, \ldots, J_n\}$ and a set of m machines $\{M_1, M_2, \ldots, M_m\}$. Each job has a set of m operations and each operation j can be done by one machine. All the jobs have the same processing routes, starting from machine 1 until finishing at machine m. Each job has a known and fixed processing time for operation j. P_{ij} denotes the processing time of job i on machine j. By removing the restriction that all jobs need to be processed on all machines, the regular FS converts to a more realistic problem [2]. In this problem, we still require jobs to move from the first to the last machine.

Three types of decision-making goals are prevalent in scheduling: (1) Efficient utilization of resources: minimizing completion time or makespan; (2) rapid response to demands, or minimizing the work-in-progress: mean completion time, mean flow time, or mean waiting time; and (3) close conformance to prescribed deadlines: total tardiness, maximum tardiness, and the number of tardy jobs. The tardiness of each job is equivalent to the amount of time when the job is completed after its due date. The first and second types of decision-making goals are process oriented and the third one, as we show in Figure 1, is considered as a customer-oriented objective. In generalized flow shop (GFS), minimizing the total completion time (TCT) is synonymous to maximizing the throughput.

Since the best sequence with respect to the TCT minimization may lead to a large number of jobs being completed after their due dates, we also consider the total tardiness (TT) minimization. The TT minimization is vital prominence in make-to-order or just-in-time environments [3]. The main aim of

*e-mail: Khalili.mj@gmail.com

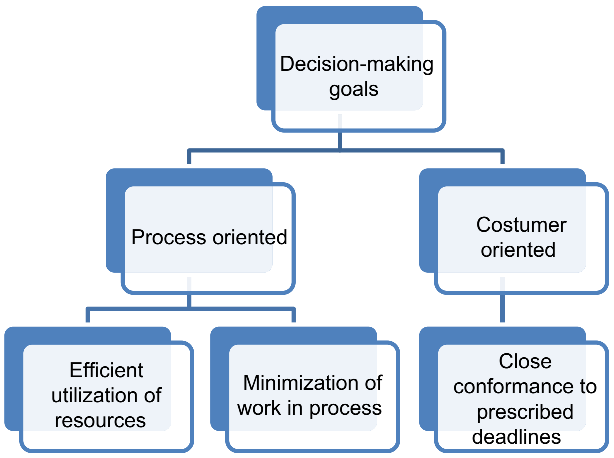
Decision-making goals
Process oriented
Costumer oriented
Efficient utilization of resources
Minimization of work in process
Close conformance to prescribed deadlines

Figure 1. Classification of decision-making goals in scheduling problems.

considering two independent objectives is to gauge the robustness of our proposed EM by two essentially different objectives and study as to whether the high performance of EM is transferable to other objective functions. In the other words, we consider independently the total tardiness and total completion time in order to analyze the efficiency and robustness of our proposed EM on each of these objectives against some existing methods.

In real-world situations machines are not continuously available due to many reasons. For example it is possible that certain important jobs may have been promised during the previous planning horizon. Another common reason for this unavailability could be breakdown (a stochastic event) or preventive maintenance (a deterministic event). A breakdown makes the shop behavior hard to predict, and thereby reduces the efficiency of the production system. Therefore, scheduling maintenance to reduce the breakdown rate has commonly been recognized by the decision makers. This paper investigates the deterministic case of machine unavailability before making decisions for job sequence starts.

In most studies on scheduling it is assumed that processing of a job i on machine j commences without delay after finishing its process on machine $j-1$. In practice, in most manufacturing and distribution systems, semi-finished jobs are transferred from one processing facility to another by transporters such as robots and conveyors, and finished jobs are delivered to warehouses or customers by vehicles such as trucks. The transportation time from machine $j-1$ to machine j is T_f and transportation time from machine j to machine $j-1$ is T_b. The time to load and unload the transporter is included in the transportation time. When transporter reaches machine j, it delivers the job to the machine if the setup time has been completed and machine is ready to receive the job. Otherwise, it places the job in a waiting line and immediately starts its return to machine $j-1$. In other words, once the transporter leaves the first machine, it always returns in time $T_f + T_b$ to take the next job. This transportation time could be either job-dependent or job in-dependent. We assume that all transportations are job-independent; and transportations between two machines are handled by one transporter. This transporter can only carry one job at a time; therefore, a job may have to wait for the transporter before it is transported.

In regular flowshop, the earliest starting time of job j processed on machine i is computed by:

$$s_{ji} = \max\{C_{ji-1}, C_{j'i}\}, \tag{1}$$

where C_{ji-1} is the completion time of job j on machine $i-1$ (i.e., $C_{ji-1} = s_{ji-1} + p_{ji-1}$) and j' is the predecessor of job j in the sequence.

In this paper, flowshop with transportation times is considered, on which the earliest starting time, s_{ji}, is calculated by:

$$s_{ji} = \max\left\{C_{ji-1} + T_{f_j}, C_{j'i-1} + 2T_{f_j} + T_{b'_j}, C_{j'i}\right\} \tag{2}$$

where T_{f_j} and T_{b_j} are transportation time from machine $i-1$ to machine i and return time of a transporter from machine i to machine $i-1$. Precisely, the term of $C_{ji-1} + T_{f_j}$ determines the s_{ji} when job j does not wait for the transporter while the term of $C_{j'i-1} + 2T_{f_j} + \ Tb_j$ is used when job j has to wait for the transporter. The term of $C_{j'i}$ shows the availability time of machine i.

Electromagnetism-inspired algorithms have recently been classified as a meta-heuristic approach to tackle complex optimization problems. The motivation behind this meta-heuristic approach has risen from the attraction-repulsion mechanism of electromagnetic theories and this basic idea that in meta-heuristics we desire to bring our search closer to a region with the superior objective function and at same time, go away from the region with the inferior objective function to move the solution gradually toward the optimality. The EM shows very high performance than other meta-heuristics in NP-hard problems [4, 5]. In our case, machine availability constraints and transportation times as well as skipping probability of some jobs from some stages are assumed. To judge the performance of the proposed EM, we go through the corresponding problem under minimizing two separate objectives, total tardiness from make-to-order environments, and total completion times from make-to-stock environments.

The GFS is categorized as a combinatorial optimization problem known to be strongly NP-hard requiring an effective metaheuristic algorithm to be utilized to overcome the complexity of the problem considered and solve it. Therefore, in this paper we have employed EM which is one of the recently introduced metaheuristic for the first time. This is the first time that the EM algorithm is being employed for the problem considered GFS. Therefore, there is no similar EM in the literature to compare against. We have compared the most relevant and effective algorithms in the literature of the problem at hand to prove the superiority of our algorithm. Meanwhile, based on your comments, we mentioned our pioneering effort to employ EM in our problem in the paper.

The purpose of this paper is to investigate a realistic flowshop case with three practical assumptions, transportation time, machine availability constraints and machine skipping. Doing so, we aim to establish a simple criterion to integrate GFS production scheduling with preventive maintenance (PM) activities to yield a high level of machine availability noting that PM operations have priority over production operations. So PM

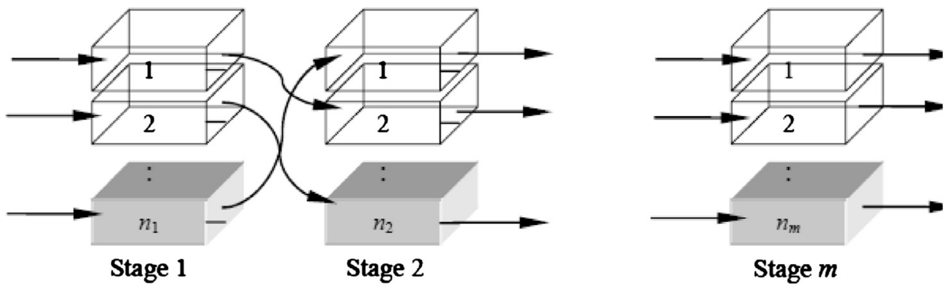

Figure 2. 3D flowshop scheduling problem.

operations must be scheduled first, and then jobs sequenced along with them. To solve such an NP-hard problem [5], we propose to apply the EM. This proposed algorithm tackles two independent objectives and we examine its robustness in different situations. Finally, we compare these results with the results of a number of other related algorithms. The following assumptions are usually characterized to GFS: each machine in each stage can process at most one job at a time, each job can be processed by at most one machine at a time in each stage, the process of a job on a machine cannot be interrupted, machines aren't available continuously and once a job completes one process it isn't immediately available for its next (Figure 2).

The problem under consideration generalizes the problem studied by the other paper. This generalization comes from assuming machine availability constraints. This paper considers total completion minimization as well as total tardiness, while the other paper minimizes makespan and total tardiness.

The rest of this paper is organized as follows: Section 2 gives literature review on the problem. We introduce the preventive maintenance and how to integrate it with production operations sequencing in Section 3. In Section 4, the proposed algorithm is presented. In Section 5 we evaluate the algorithms by a benchmark. Finally, Section 6 concludes the paper.

2. Literature review

The FS has been a very dynamic research domain in meta-heuristic and heuristic techniques, principally because of the difficulty encountered by exact methods to solve large or even medium instances.

Salvador [6] was the first who defined the FS problem. A detailed survey for the FS problem has been given by Linn and Zhang [7] and Wang [8]. Moursli and Pochet [9] presented a branch-and-bound algorithm to minimize the makespan in FS. Exact algorithm is applied to large FS problems; such an approach can take hours or days to derive a solution so Sriskandarajah and Sethi [10] proposed a heuristic algorithm for solving a special case of the FS problem. Guinet et al. [11] proposes heuristic techniques for a simplified FS makespan problem with two stages and only one machine at stage two. Guinet et al. [11] also proposed a heuristic for the makespan problem in a two-stage FS based on Johnson's rule [1]. To obtain a near optimal solution, meta-heuristic algorithms have also been proposed. Nowicki and Smutnicki [12] proposed a tabu search (TS) algorithm for the FS makespan problem. Gourgand et al. [13] and Zhang and Wu [14] presented several simulated annealing (SA)-based algorithms for the FS problem. A genetic algorithm has been widely used in many previous works for the FS makespan problem, see e.g., Reeves [15].

Tavakkoli-Moghaddam et al. [16] proposed a memetic algorithm (MA) combined with a new local search engine, namely, nested variable neighborhood search (NVNS), to solve a FS scheduling problem with processor blocking. The performance of the proposed EM was verified by a number of instances and then compared with genetic algorithms. Cheng et al. [17] addressed the earliness/tardiness scheduling problem with identical parallel machines, and they apply a genetic algorithm to solve this problem. Yang [18] was the first to use EM to train a neural network. The results show a great saving on the computation memories and time, and indicate that EM performed much better than genetic algorithm in finding the global optimum. For its merit of simple concept and economic computational cost, EM has been used in the areas of function optimization, fuzzy neural network training, project scheduling, and other combinatorial optimization fields [5] but seldom have used in scheduling problems, so we were motivated to solve our problem with this method. Khalili and Tavakkoli-Moghadam [19] proposed a new multi-objective electromagnetism algorithm for a bi-objective flowshop scheduling problem.

Because of its significance both in industrial and theory applications, the GFS has attracted the attention by many researchers. To obtain a near-optimal solution, meta-heuristic algorithms have also been proposed. Different genetic algorithms (GA) are applied by Chen and Neppalli [20], Aldowaisan and Allahverdi [21]. Among the other metaheuristics, one could refer the reader to particle swarm optimization (PSO) by Pan et al. [22], simulated annealing (SA) by Fink and Voß [23], ant colony optimization (ACO) by Shyu et al. [24] and tabu search (TS) by Grabowski and Pempera [25]. Khalili [26] proposed an iterated local search algorithm for flexible flow lines with sequence dependent setup times to minimize total weighted completion and also studied multi-objective no-wait hybrid flowshop scheduling problems to minimise both makespan and total tardiness [27].

Next, briefly review the related research on availability constraint and the traveling time between stages, which are other characteristics of the problem at hand. Traveling times between stages in most investigations have been considered as processing time. Hurink [28] assumed that unlimited buffer spaces exist between the machines, and all transportation is accomplished through a single robot. Soukhal et al. [29] analyzed two-machine FS scheduling problems with transportation and assumed that finished jobs are transported from the processing facility and delivered to customers by trucks. In the other words, both transportation capacity and transportation times were explicitly taken into account.

Scheduling problems with availability constraints have been extensively examined by Ruiz et al. [30]. For more details, a survey of existing methods for solving scheduling problems under availability constraints as well as complexity results can be found in Schmidt [31]. Lee [32] handled the preemptive FS problem with two machines and one unavailability period first imposed on machine 1 and then on machine 2 with the makespan objective. Both problems are proved to be NP-hard in the ordinary sense, and heuristics with error bounding analysis are proposed. Blazewicz [33] investigated the two-machine problem with arbitrary number of unavailability periods on one machine, and proved that the makespan minimization problem is strongly NP-hard. Breit [34] investigated the problem of scheduling n preemptable jobs in a two-machine FS where the first machine is unavailable for processing during a given time interval. A more complex hybrid FS problem is tackled in Allaoui and Artiba [35].

3. Preventive maintenance

Machine scheduling is concerned with the problem of optimally scheduling available machines. However, the majority of the scheduling literature carries a common assumption that machines are available at all times. This availability assumption may not be true in real industrial settings, since a machine may become unavailable during certain periods of time. For instance, a machine may not be available at the beginning of a planning horizon as it may continue to process the late jobs from the previous horizon. Similarly, when a machine breakdown has occurred or a preventive maintenance activity has been scheduled. A breakdown causes the shop behavior difficult to predict, and thereby reduces the efficiency of the production system. Therefore, scheduling maintenance to reduce the breakdown rate has commonly been recognized by the decision makers. It is known that maintenance plays an important role in many industries because more reliable production systems with higher serviceable performance is what factories are mostly intending to reach; on the other hand, maintenance systems are designed to make sure that production facilities are serviceable and reliable to operate to achieve target production levels [30].

Maintenance policies influence the machine availability and the machine utilization ratio. Maintenance actions usually can be classified into two major categories: corrective maintenance (CM) and preventive maintenance (PM). CM is the actions carried out when a failure has already occurred. PM is the action taken on a system while it is still in service, but is carried out in order to keep the system at the desired level of performance. In PM, activities are conducted at fixed time intervals determined for machines and facilities before a failure or breakdown occurs. Here, the main question is whether PM decisions and production sequencing should be jointly scheduled. Ruiz et al. [30] investigated this issue and concluded that the optimal decision is case dependent.

Table 1. Processing times (P_i) for single machine with n = 4.

	Jobs (i)			
	1	2	3	4
P_i	25	30	25	20

According to different criteria, various PM policies are defined. The commonly used PM policy in industry is "preventive maintenance at fixed predefined time intervals" [30]. PM operations are scheduled in advance at pre-determined time intervals without taking into account a probabilistic behavior for time to failure. The intervals between PM operations can be weekly, monthly or other periods depending on the system. In this policy, fixed time intervals (T_{PM}) are determined and PM operations are carried out exactly at those times. Our criterion works as follows: whenever a new job is to be processed on each machine, the completion time is computed. If this time exceeds the pre-defined T_{PM}, then the process of the next job is postponed and the PM is carried out first because PM operations have priority over processing. It should be pointed out that has been assumed that the processing of a job cannot be interrupted (or preemption is not allowed or the jobs are non-resumable), and availability time of any machine is after finishing the process of the last job on that machine. The following example shows how PM operations scheduling and single machine scheduling are jointly considered. Suppose a shop decides to carry out PM at every 50 time units. The durations of these PM operations (D_{PM}) for the machine are 15 time units. The processing times are shown in Table 1. The durations of these PM operations (D_{PM}) for the machine are 15 time units. The processing times (P_i) are also shown in Table 1. In fact, will been scheduled the jobs in the production horizon shown in Figure 3.

Figure 4 depicts a sequence of all the jobs as {4, 1, 3, 2}. As it can be seen in this figure, the first maintenance is carried out after processing jobs 4 and 1, the accumulated total processing time is 25 + 20 = 45 and it is not possible to process the third job of sequence (Job 3), because it has a processing time of 25 time units, which would result in an accumulated total processing time of 70 units, which is longer than T_{PM} = 50. So, machine will be idle for 5 time units, and then PM operation begins at time 50 units, and lasts for 15 time units. After carrying out PM, Job 3 is processed. Then, once more it is not possible to process the next job (Job 2) since a processing time of 90 + 30 = 120 will be accumulated. So, in the same manner, second PM operation is carried out from 100 to 115, then Job 2 is processed.

4. Proposed algorithms

4.1. Heuristics

In general, the heuristics applied to each decision making goal, process or customer oriented, are different. These two

	1	2	3	4
P_i	25	30	25	20

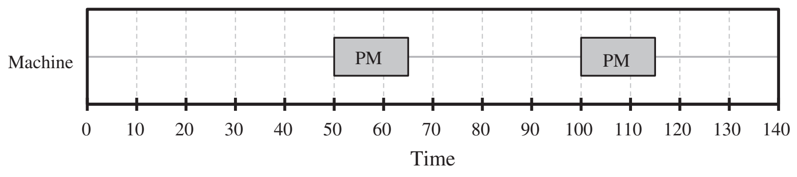

Figure 3. Gantt chart of the production horizon after applying PM.

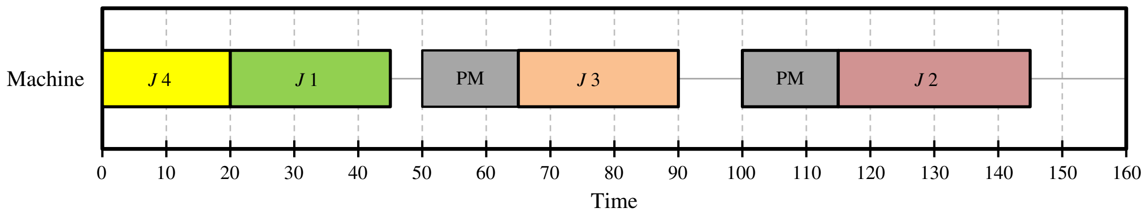

Figure 4. Gantt chart of the solution for the given example.

groups have been introduced and explain the heuristics, which we have used in this study:

1. Heuristics to solve scheduling problem with emphasis on completion times such as SPT, LPT, NEH (Nawaz, Enscore and Ham), and (*g*/2, *g*/2) Johnson' rule [1].
2. Heuristics to solve scheduling problem with emphasis on tardiness such as EDD, SLACK and NEH_EDD.

The above seven heuristics are briefly described below.

- Shortest Processing time (or SPT): jobs are processed on machine 1 in increasing order of the processing times on machine 1. That is, the process of jobs with shortest processing times is first started. At subsequent machines, jobs are sorted in earliest ready time order.
- LPT arranges jobs on machine 1 in descending order of processing times of jobs on machine 1.
- (*g*/2, *g*/2) Johnson's rule, the sum of processing time of jobs on machine 1 to [*m*/2] and the sum over machine [*m*/2] + 1 to *m* are calculated to order jobs on machine 1 [1].
- NEH, This heuristic can be divided into three simple steps:

    1. The total processing times for all the jobs on *m* machines are computed as follows:

    $$p_i = \sum_{j=1}^{m} P_{ij}; \quad i = 1, \ldots, n$$

    2. Jobs are sorted in descending order of P_i. Then, the first two jobs (those with higher P_i) are taken and the two possible schedules containing them are evaluated.
    3. Take job *i* (*i* = 3, . . ., *n*) and find the best schedule by placing it in all the possible *i* positions in the sequence of jobs that are already scheduled. For example, if *i* = 4, the already built sequence would contain the first three jobs of the sorted list calculated in step 2, then, the fourth job could be placed either in the first, second, third or the last position of the sequence. The best sequence of the four would be selected for the next iteration.

- EDD is a well known dispatching rule and orders the jobs according to imminent due dates.
- SLACK, another name of this method is the minimum slack because in this method at time *t*, the job with the minimum value of $d_j - C_j(s)$ is selected, where $C_j(s)$ will be the completion time of job $j \notin s$ if it is scheduled at the end of sequence *s*.
- In NEH_EDD we consider the due dates for defining an initial order in which the jobs are considered for insertion. The initial order in NEH_EDD is based on the earliest due date dispatching rule that arranges jobs in ascending order of their due dates.

4.2. Simulated annealing

Simulated annealing (SA) was first presented as a search algorithm for combinational optimizations (CO) problems in Cerny [36]. SA is modeled after physical annealing of solid metal. In annealing, a metal is first heated to a high temperature and then cooled down with a very slow rate to the room temperature. Sometimes, if cooling is not fast enough, quenching is done. In SA, solutions are randomly generated from a set of feasible solutions. This process accepts not only those solutions that improve the objective function, but also those solutions, which do not improve objective function on the basis of transition probability (TP). Transition probability depends on the change in objective function and the annealing temperature. The main features of SA that make this algorithm more sophisticated are perturbation annealing schedule and the

```
Procedure Simulated_annealing
t = $T_0$
x = initialization                          % Initial solution by another algorithm
$x_{best} = x$
while stopping criterion is not met do
    for iter = 1 to max do
        s = move x by an operator           % generating a neighbor solution from x
        if $f(s) < f(x)$ then               % Acceptance criterion
            x = s
            if $f(s) < f(x_{best})$ then    % check with the best solution
                $x_{best} = s$
            endif
        else
            if random < $exp\{-(f(s) - f(x))/t\}$ then
                x = s
            endif
        endif
    endfor
    $t = \alpha \cdot t$                    % temperature decrease
endwhile
```

Figure 5. General outline of a simulated annealing algorithm.

transition probability. Perturbation generates a new solution, and the annealing schedule controls the initial temperature, final temperature and the rate of cooling, while transition probability help heuristic to escape local optima. SA is commonly said to be the oldest among the meta-heuristics and one of the first algorithms that had an explicit strategy to avoid local optima.

The fundamental idea is to generate a new job sequence s by a random rule from the neighborhood of present sequence x. This new sequence is accepted or rejected by another random rule. A parameter t, called the temperature, controls the acceptance rule. The variation between objective values of two candidate solutions is computed $\Delta C = obj(s) - obj(x)$, where obj is the value of the objective function. If ($\Delta C \leq 0$, sequence s is accepted. Otherwise, sequence s is accepted with probability equal to $\exp(\Delta C/t_i)$. The algorithm proceeds by trying a fixed number of neighborhood moves (max) at temperature t_i, while temperature is gradually decreased. The procedure is repeated until a stopping criterion is met.

Moves resulting in solutions of worse quality (uphill move) than the current solution may be accepted to escape from local minimum. SA starts at a high temperature (T_0), so most of the moves are accepted at first steps of the procedure. The probability of doing such a move is decreased during the search. Figure 5 shows the general outline of SA algorithm.

The commonly used encoding scheme for FS problem is permutation of jobs which shows the job sequence on machine 1. It is known that initial solution can influence the quality of the solutions. A good initial solution can result in a better and faster subsequent result. There are some alternative choices to consider different parameters and adjust them by tuning and settings. The proposed SA algorithm checks 100 neighbors at temperature t_i (i.e., $max = 100$).

Moving operator generates a neighbor solution from current candidate solution by making a slight change in it. This operator must work in such way to avoid infeasible solutions. In this research, two different move operators have been considered:

- Swap operator (SO): the positions of two randomly selected jobs are swapped (Figure 6).
- Single point operator (SPO): the position of one randomly selected job is randomly changed (Figure 7).

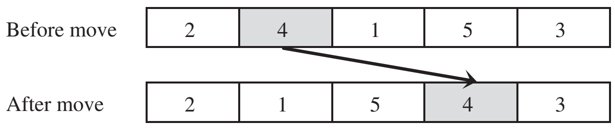

Figure 6. Single point operator.

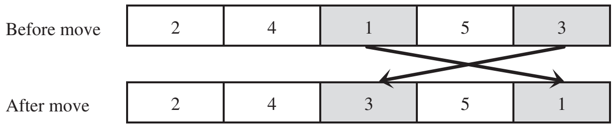

Figure 7. Swap operator.

The algorithm developed in this paper uses different operators from the other paper. As mentioned earlier, to avoid a local minimum, solutions with worse objective values are probabilistically accepted depending on the value of the temperature. When the procedure proceed, the temperature is slightly lowered under a mechanism which is called cooling schedule. Here, exponential cooling schedule has been used, $T_i = \alpha \times T_{i-1}$ (where $\alpha \in (0, 1)$ is temperature reducing rate), which is often believed to be an excellent cooling recipe.

4.3. Electromagnetism-like method

Birbil and Fang [37] proposed the electromagnetism-like method (EM), which is a flexible and effective population-based algorithm to search for the optimal solution of global optimization problems. EM originates from the electromagnetism theory of physics by considering potential solutions as electrically charged particles spread around the solution space. This meta-heuristic method utilizes an attraction-repulsion mechanism to move the particles towards optimality. Debels et al. [38] applied successfully a hybridization of the EM with scatter search for resource-constrained project scheduling problem. EM is useable for particular set of optimization problems with bounded variables.

Each candidate solution as a charged particle has been considered. The charge of each candidate solution is related to the objective function value. The size of attraction or repulsion over candidate solutions in the population is calculated by this charge. The direction of this charge for candidate solution i is determined by vectorially adding the forces from all other solutions on the candidate solution i. In this mechanism, a candidate solution with good objective function value attracts the other ones; candidate solutions with unfavourable objective function repel the other population members; and better the result of the objective function value the higher the size of attraction.

EM has four phases including *Initialization* of algorithm, computation of *total force* exerted on each particle, *movement* along the direction of the force, and the *local search*.

4.3.1. Encoding scheme and initialization

The most frequently used encoding for the flowshop problem is a simple permutation of the jobs. The relative order of the jobs in the permutation indicates the processing order of

```
Procedure Local_search
  $k = 1$
  while $k < n + 1$ do
      $v$ = Shift $x_{ik}$ to a new random position
      if $f(v) < f(x_i)$ then
          $x_i = v$
          $k = n$
      endif
      $k = k + 1$
  endwhile
```

Figure 8. Procedure of the local search.

the jobs on the first machine in the shop. It is necessary to mention that a drawback of the algorithms proposed for generalized flowshop problems is that they only order jobs according to earliest ready time of jobs at the beginning of each stage. However in GFS problems, it is very likely to have some jobs with the same ready time at beginning of each stage. For example, all the jobs which skip the first stage would have the ready time of zero at the beginning of the second stage. So we need to establish a clear criterion to order these jobs. In this paper, if some jobs have the same ready time at the beginning of stage t ($t = 2, 3, \ldots, g$), the same as their order at stage $t - 1$ has been arranged. We subject all the compared algorithms to this criterion.

Traditionally, in an EM, the initial population is generated randomly. However, it is known that the initial solutions can affect on the quality of the results obtained by the algorithms. The initialization procedure in such a complex combinatorial problem has to be made with great care, to ensure convergence to desirable objective function values in a reasonable amount of time. Because of this, initial solutions for proposed EM are generated by EDD and NEH_EDD for total tardiness objective and SPT and NEH for the completion time objective. The population size (*popsize*-2) is randomly generated. The objective function values of solutions are calculated and the best one is recorded as x_{best}.

4.3.2. Local search

The proposed EM is hybridized with a local search in order to improve the performance of the algorithm. The procedure for this local search can be described as follows: the first job in the sequence of candidate solution i (x_{i1}) is relocated to a new random position in the sequence. If this new sequence (v) results in a better makespan, the current solution (x_i) is replaced by the new sequence (v). This procedure iterates at most for all the subsequent jobs in sequence. If there is any improvement in the kth $< n$, the local search for the current solution terminates. Subsequently, the best solution is updated. Procedure for the local search is shown in Figure 8.

4.3.3. Computation of total forces

The charge of each solution candidate i is calculated in relation to its objective function by:

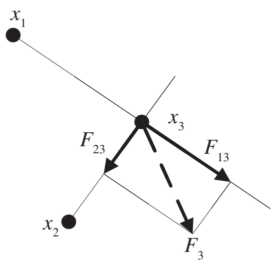

Figure 9. Example of exertion of forces.

$$q_i = \exp\left(-n \cdot \frac{f(x_i) - f(x_{\text{best}})}{\sum_{j=1}^{\text{popsize}}\left(f(x_j) - f(x_{\text{best}})\right)}\right), \quad \forall i\; i = 1, 2, \ldots, \text{popsize}. \tag{3}$$

This formula ensure that the better objective values are assigned higher charges. The *total force* F_i exerted on candidate solution i is also calculated by the following formula:

$$F_i = \begin{cases} \sum_{j \neq i}^{P} (x_j - x_i) \dfrac{q_i \cdot q_j}{\|x_j - x_i\|^2}; & f(x_j) < f(x_i), \\ \sum_{j \neq i}^{P} (x_i - x_j) \dfrac{q_i \cdot q_j}{\|x_j - x_i\|^2}; & f(x_i) < f(x_j). \end{cases} \tag{4}$$

A two-dimensional example *total force vector* F_i exerted on candidate solutions is shown in Figure 9. The force exerted by x_1 on x_3 is F_{13} (repulsion: the objective function of x_1 is worse than that of x_3) and the force exerted by x_2 on x_3 is F_{23} (attraction: the objective function of x_2 is better than that of x_3). F_3 is the *total force* exerted on x_3 by x_1 and x_2.

4.3.4. Moving by total forces

All the candidate solutions are moved with the exception of the current best solution. The move for each candidate solution is in the direction of the *total force* exerted on it by a random step length. This length is generated from a uniform distribution between [0, 1]. By selecting random length that candidate solutions have a nonzero probability to move to the unvisited solution space along this direction has been guaranteed. Moreover, by normalizing the *total force* exerted on each candidate solution, we are able to avoid producing infeasible solutions. Figure 10 shows a pseudo code of our proposed EM.

5. Experimental evaluation

In this section, we apply a benchmark to evaluate the performance of our proposed algorithms. They are implemented in MATLAB 7.0 and run on a Pentium IV PC with an Intel processor running at 3 GHz and 1 GB of RAM memory. Relative percentage deviation (*RPD*) for total completion time as a common performance measure to compare the methods has been used. The best solutions, Min_{sol}, obtained for each instance

```
Procedure Electromagnetism-like method

Initialization
While termination criterion are not satisfied  do
    for $i = 1$ to popsize do
        $x_i$ = Local search
    endfor
    Update $x_{best}$

    for $i = 1$ to popsize do                                 %Procedure computation_of_total_forces
        calculate $(q_i)$
        $F_i = 0$
    endfor
    for $i = 1$ to popsize do
        for $j = 1$ to popsize do
            if $f(x_j) < f(x_i)$  then
                $F_i = F_i + (x_j - x_i)\frac{q_i \cdot q_j}{\|x_j - x_i\|^2}$  $(attraction)$
            else
                $F_i = F_i - (x_j - x_i)\frac{q_i \cdot q_j}{\|x_j - x_i\|^2}$  $(repulsion)$
            endif
        endfor
    endfor

    for $i = 1$ to popsize do                                 %Procedure Moving_by_total_forces
        if $i \neq best$ then
            $\beta$ = random [0, 1]
            $F^i = \frac{F_i}{\|F_i\|}$
            for $k = 1$ to $n$ do
                if $F_{ik} > 0$  then
                    $x_{ik} = x_{ik} + \beta \cdot F_{ik}(1 - x_{ik})$
                else
                    $x_{ik} = x_{ik} + \beta \cdot F_{ik}(x_{ik})$
                endif
            endfor
        endif
    endfor
Endwhile
```

Figure 10. Pseudo code of the proposed EM.

are computed by any of the five algorithms. *RPD* is obtained by the following equation:

$$\mathrm{RPD} = \frac{\mathrm{Alg}_{\mathrm{sol}} - \mathrm{Min}_{\mathrm{sol}}}{\mathrm{Min}_{\mathrm{sol}}} \times 100 \tag{5}$$

where $\mathrm{Alg}_{\mathrm{sol}}$ is the objective function value obtained for a given algorithm and instance. Clearly, lower values of RPD are preferred. In the case of the total tardiness GFS problem, the best solution could be zero (and therefore optimal), then in the above equation we will have division by zero. Moreover, if the best solution is a small value, the performance measure underestimates an algorithm which obtains a solution slightly worse than the best. Therefore, a different performance ratio is usually used in tardiness cases to avoid these problems, which has been termed the relative deviation index (RDI). The RDI is obtained by:

$$\mathrm{RDI} = \frac{\mathrm{Alg}_{\mathrm{sol}} - \mathrm{Min}_{\mathrm{sol}}}{\mathrm{max}_{\mathrm{sol}} - \mathrm{Min}_{\mathrm{sol}}} \times 100. \tag{6}$$

With this measure, an index between 0 and 100 is obtained for each method such that a good solution will have an index very close to 0. Note that if the worst and the best solutions take the same value, all the methods provide the best (or same) solution and hence, the index value will be 0 (i.e., the best index value) for all methods.

Indeed, the above formulas are two different ways of normalization. They calculate the deviation from the best solution, which is always a positive number.

5.1. Data generation

Data required for the given problem consist of the number of jobs (n), number of machines (m), range of processing times (p) and transportation times (T_f and T_b), the ready times (r), skipping probability, time interval between two consecutive PM (T_{PM}) and duration of PM operations (D_{PM}). Our instances based on benchmark values [39] that are shown in Table 2 has been guaranteed.

We have n = {20, 50, 100, 200, 500} and m = {5, 10, 20}, which results in 15 combinations of $n \times m$. The processing time in Taillard's instances [39] are generated from a uniform distribution over the range [1, 99]. GFS is considered by allowing

Table 2. Factor levels.

Factor	Levels
Number of jobs	20
	50
	100
	200
	500
Number of machines (m)	5
	10
	20
Range of processing times (p)	U [1, 99]
Range of transportation times ($T_f + T_b$)	U [1, 30]
Skipping probabilities	0.1
	0.4
Range of time interval between two consecutive PM (T_{PM})	U [200, 300]
Range of duration of PM operations (T_{PM})	U [1, 50]
	U [1, 99]
	U [1, 150]

some jobs to skip some stages. The probability of skipping a stage is set at 0.1 or 0.4. The transportation times (T_f and T_b) come from a uniform distribution in the range [1, 30], where the running average will work out about 30% of the processing time. T_{PM} for each machine are obtained from a uniform distribution in the range [200, 300]. D_{PM} of each machine are distributed uniformly over three ranges [1, 50], [1, 99] and [1, 150], where the running average will be about 50%, 100% and 150% of the processing times, respectively. The different levels of factors result in $5 \times 3 \times 1 \times 1 \times 2 \times 1 \times 3 = 90$ different scenarios. Ten instances for each scenario, similar to a Taillard's benchmark have been produced. Therefore, $90 \times 10 = 900$ instances have been.

To generate due dates for all n jobs an approach similar to [40] have been used. The following steps are applied to produce the due dates [41]:

1. Compute the total processing time of each job on all g stages.

$$p_i = \sum_{t=1}^{g} p_{it}, \quad \forall i \in N. \tag{7}$$

2. Determine a due date for each job:

$$d_i = (p_i) \times (1 + \text{random} \times 3), \quad \forall i \in N, \tag{8}$$

where *random* is a random number from a uniform distribution over the range [0, 1].

5.2. Parameter tuning

The quality of algorithms is significantly influenced by the values of parameters. In this section, the behavior of different operators and parameters of SA and the proposed EM have been studied. In order to tune the algorithms, we apply a full factorial design in the design of experiment (DOE) approach [42]. Five instances for each combination of n and m, which results in a total of 75 instances have been randomly generated. The stopping criterion is $n \times m \times 0.2$ s of the computational time. This criterion allows for more time as the number of jobs or machines increases.

5.2.1. Simulated annealing

The proposed SA has three parameters, initial solution, {initial temperature, cooling rate} and the move operator. The considered levels of the parameters are:

- Initial solution (*IS*): two levels (SPT, NEH).
- {Initial temperature (T_0), cooling rate (α)}: three levels ({50, 0.985}, {100, 0.98}, {200, 0.97}).
- Move operator (*MO*): two levels (SO, SPO).

So, $2 \times 3 \times 2 = 12$ different SAs are obtained by these levels and all the 75 instances are solved by them. The results are analyzed by means of the analysis of variance (ANOVA) technique. Three main hypotheses, including normality, homogeneity of variance and independence of residuals have been examined and no bias have been found to question the validity of the experiments. The means plot and least significant differences (LSD) intervals (at the 95% confidence level) for the levels of *IS*, {T_0, α} and *MO* parameter are shown in Figures 11–13, respectively.

Figure 10 illustrates that the initial solution of NEH provides statistically better results than SPT. In Figure 11 can be seen that there is statistically significant difference between two move operators and that SPO is superior and also in Figure 12 can be seen $T_0 = 50$ and $\alpha = 0.985$ results in statistically better output than either of the other two sets. As the results of the analysis, the parameters *IS* = NEH, *MO* = SPO and (T_0, α) = (50, 0.985) have been set.

5.2.2. Electromagnetism-like method

One of the advantages of the proposed EM is that it has only one parameter, *popsize* (number of population). The considered levels for *popsize* are 2, 4, 6 and 8. All 75 instances are solved by the EM algorithm obtained by the above values of *popsize*. The results are assessed by means of Analysis of Variance (ANOVA) technique. The means plot and least significant differences (LSD) intervals for various levels of *popsize* parameter factor are shown in Figure 14. This figure reveals that the number of population of four provides statistically better results than other values of *popsize* = 2, 6, 8.

5.3. Experimental results

In this subsection, the authors intend to compare our proposed EM with other existing methods. As described before, this phase includes two subsections each of which considers one objective function. In each subsection, the proposed EM

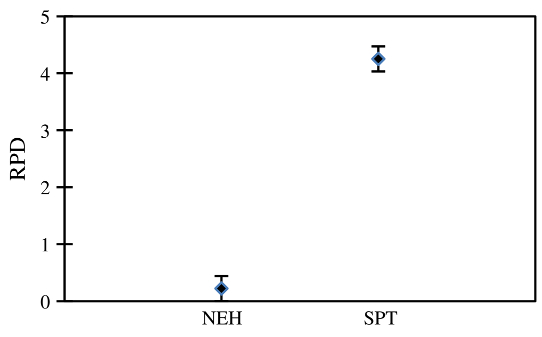

Figure 11. Means plot and LSD intervals for the initial solution of SA.

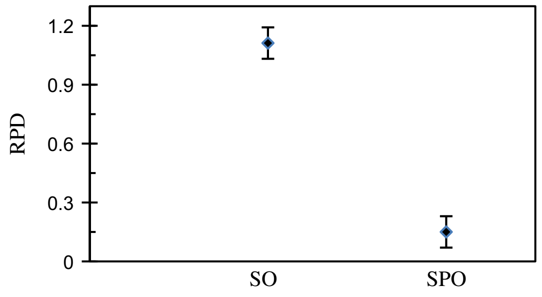

Figure 12. Means plot and LSD intervals for the move operator of SA.

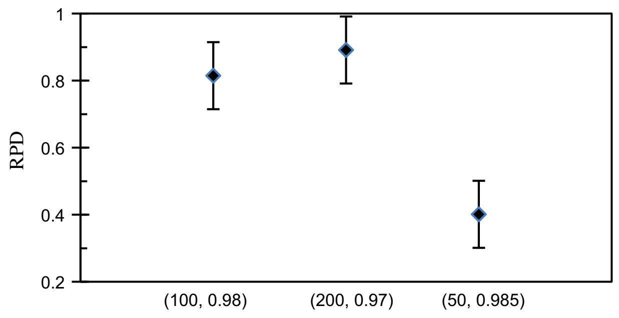

Figure 13. Means plot and LSD intervals for the initial parameters (temperature, cooling rate) in SA.

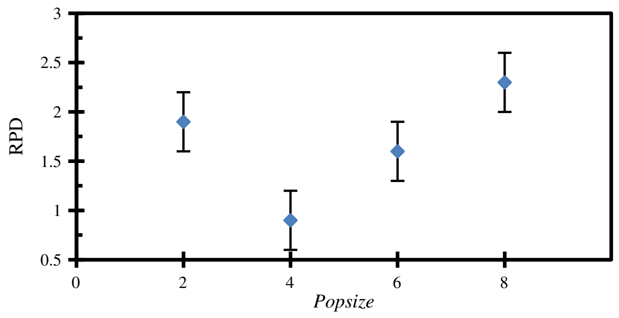

Figure 14. Means plot and LSD intervals for number of population in the proposed EM.

Table 3. Average relative percentage index ($\overline{\text{RPI}}$) for the algorithms grouped by n and m.

Instances	EDD	SLACK	NEH_EDD	SA	EM
20 × 5	97.50	93.25	36.43	6.42	**3.14**
20 × 10	80.00	76.40	29.87	3.36	**1.97**
20 × 20	60.20	66.08	26.54	4.41	**3.57**
50 × 5	91.05	96.50	39.18	18.08	**6.53**
50 × 10	89.00	92.63	36.67	17.83	**2.81**
50 × 20	91.00	95.18	19.20	15.94	**1.69**
100 × 5	90.53	94.46	48.24	26.57	**8.88**
100 × 10	95.47	96.28	46.02	28.89	**2.18**
100 × 20	92.63	98.05	35.96	19.80	**3.03**
200 × 5	90.34	92.47	38.76	17.94	**5.63**
200 × 10	93.00	98.00	26.02	18.67	**7.63**
200 × 20	92.02	98.74	25.27	18.54	**6.60**
500 × 5	93.00	98.00	24.68	15.46	**8.46**
500 × 10	92.63	98.03	18.02	14.51	**6.78**
500 × 20	91.36	97.79	13.29	11.40	**9.31**
Average	89.32	92.79	30.94	15.85	**5.21**

with some well-known heuristics as well as SA for associated objective functions has been compared. In the first subsection, the EM is compared with SA, EDD, SLACK and NEH_EDD, and in the next stage it is compared with SA, SPT, (g/2, g/2) Johnson's rule, and NEH [1]. In each subsection, the effects of variables such as problem size and numbers of machines on the performance of the algorithms are investigated. The stopping criterion is $n \times m$ 0.2-s computation time.

As mentioned earlier, the total completion time (TCT) belonging to the process oriented goals, and the total tardiness (TT) belonging to the costumer oriented goals independently to analyze the efficiency and robustness of our proposed EM on each of these objectives against some existing methods has been considered.

5.3.1. Analysis of the total tardiness

In this section, the authors compare our proposed EM algorithm with SA, NEH_EDD, SLACK and EDD with the aim of minimizing the total tardiness. The results of the experiments, averaged for each combination of n and m are shown in Table 3. As expected, the meta-heuristics perform better than the heuristics, and the worst performing algorithms in almost all the instances are EDD and SLACK with RDI of 89.32% and 92.79%, respectively. The proposed EM provides best results among these algorithms with the RDI of 5.21%.

For further analysis, the ANOVA has been carried out. The means plot for the different algorithms with the least significant difference (LSD) intervals are shown in Figure 15. As it is seen, the proposed EM provides statistically better results than the other methods. NEH_EDD supersedes the traditional EDD and SLACK. EDD and SLACK are statistically the same.

The results also show that EM statistically outperforms the other algorithms. It is interesting to see the performance of NEH in comparison with two other heuristics. NEH_EDD significantly supersedes EDD and SLACK. In addition, we carry out a two-way ANOVA for each variable to find the interaction between them and performance of the algorithms. Figure 16 shows the trend of the quality of the methods evaluated as

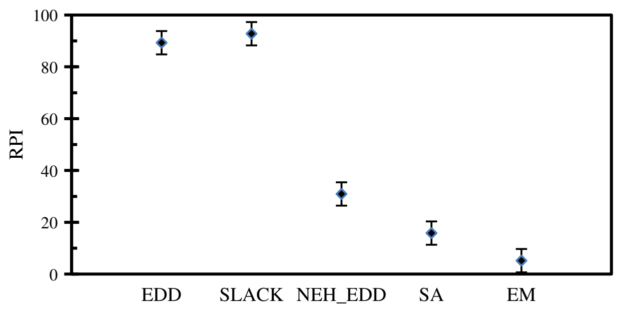

Figure 15. Means plot and LSD intervals (at 95% confidence level) for the type of the algorithm factor.

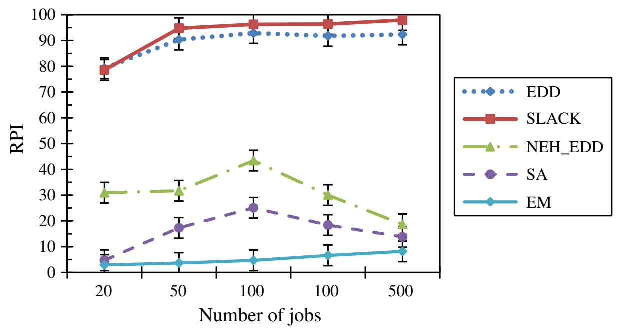

Figure 16. Means plot and LSD intervals for the interaction between the algorithm type and the number of jobs.

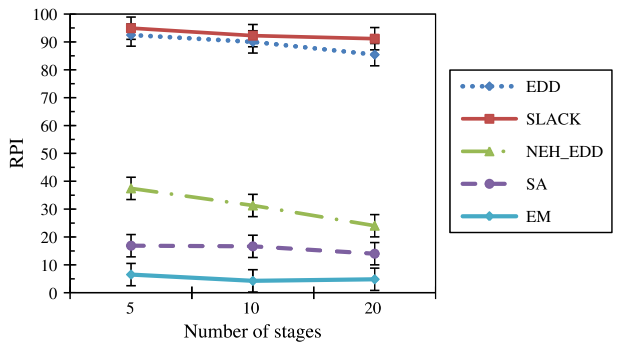

Figure 17. Means plot and LSD intervals for the interaction between the algorithm type and the number of stages.

the number of jobs increases. EM and SA are the same in the case of $n = 200$ and the greatest difference between SA and EM is shown when $n = 100$ and again the same in the case of $n = 500$. As illustrated in Figure 17, the higher the number of stages, the better the heuristics performs, while there is no change of the metrics for EM and SA.

Figure 16 depicts the means plot and LSD intervals for the interaction between the various algorithm as a functions of the number of stages. It is interesting to see in this figure that the proposed EM shows a robust performance in almost all cases against the number of stages.

Table 4. Average relative percentage deviation ($\overline{\text{RPI}}$) for the algorithms grouped by n and m.

Instances	SPT	Johnson's	NEH	SA	EM
20 × 5	21.57	22.33	3.25	**0.81**	0.83
20 × 10	22.51	23.18	4.63	0.75	**1.68**
20 × 20	14.13	14.76	2.58	0.92	**0.90**
50 × 5	25.44	29.70	7.44	5.12	**0.01**
50 × 10	22.01	24.26	6.19	3.34	**0.28**
50 × 20	16.82	16.01	3.85	1.07	**0.89**
100 × 5	17.66	24.33	6.74	4.11	**0.23**
100 × 10	20.07	23.07	5.83	3.94	**0.08**
100 × 20	15.51	16.60	4.53	2.71	**0.34**
200 × 5	19.45	29.28	6.42	4.42	**0.27**
200 × 10	19.22	24.02	6.75	5.84	**0.07**
200 × 20	19.35	20.11	6.45	5.70	**0.01**
500 × 5	12.82	27.92	4.56	3.35	**1.57**
500 × 10	11.55	25.07	5.01	3.68	**1.84**
500 × 20	11.80	20.45	4.28	3.07	**1.21**
Average	17.99	22.74	5.23	3.26	**0.68**

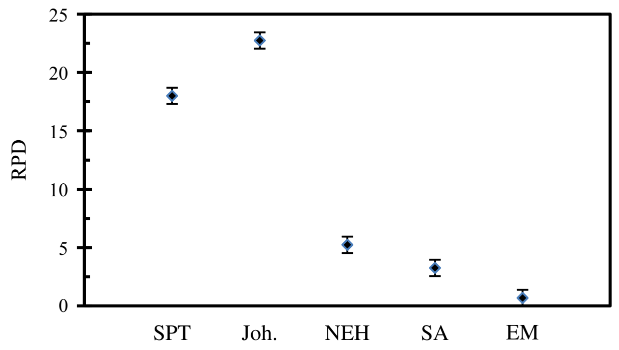

Figure 18. Means plot and LSD intervals (at 95% confidence level) for the type of the algorithm factor.

5.3.2. Analysis of the total completion times

In the total completion time objective, the authors compare the results of SA, NEH, and EM. First, the algorithms in terms of the selected objective function (i.e., total completion time) have been analyzed, and then effects of variable, such as problem size and numbers of machines, on the performance of the algorithms are examined. RPD measure to compare the algorithms has been used. The stopping criterion is again $n \times m \times 0.2$ s of the computational time. EM outperforms the other algorithms again, which support its robustness (Table 4).

Here an analysis similar to the previous section has been applied. The means plot for the different algorithms with the LSD intervals are shown in Figure 18. The Johnson's rule and SPT have a poor performance while the proposed EM gives excellent results than the heuristics and the SA.

Figure 19 shows the quality trend of the evaluated methods as the number of jobs increases. As seen, there is almost no effect on the performance of all five algorithms with regards to the problem size. Only SPT in comparison with the (g/2, g/2) Johnson's rule improves when the number of jobs increases. As shown in Figure 20, increasing the number of

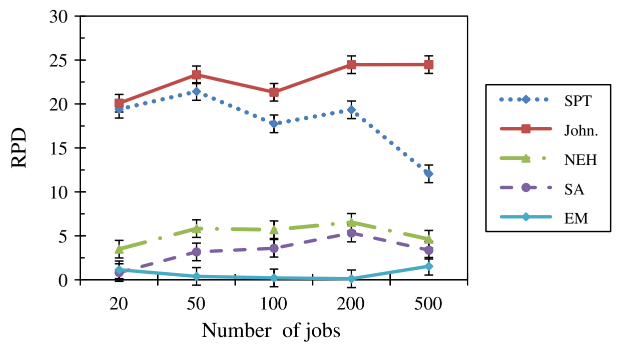

Figure 19. Means plot and LSD intervals for the interaction between the algorithm type and the number of jobs.

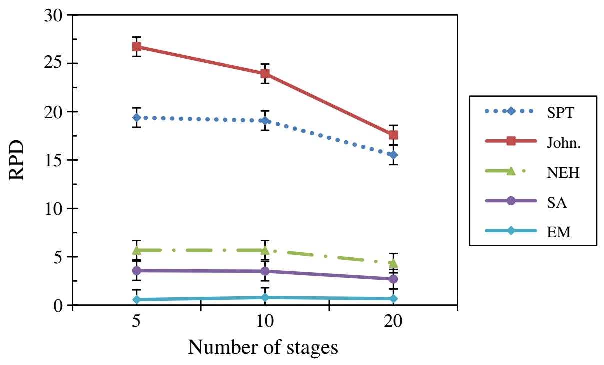

Figure 20. Means plot and LSD intervals for the interaction between the algorithm type and the number of stages.

stages results in better performance for the heuristics. It is noteworthy to see that EM again shows a robust performance in almost all cases with the number of jobs as the variable.

It is concluded that the algorithms perform differently in various situations. In particular, the number of jobs, as a factor, has noticeable impact on the performance quality of the algorithms.

6. Conclusion

In this paper, an EM for a generalized flow shop (GFS) problem with machine availability constraints and transportation times between stages to minimize two independent objectives has been proposed. EM and also SA as well as a number of other well-known heuristics to minimize total tardiness and total completion times, independently. The results were analyzed in terms of objective functions and the effects of variable factors on the instances considered in this paper. The results show that the proposed EM performs very well. This is the first time that an EM has been applied to GFS problems, and the computational results show it holds a good promise. In fact, the overall performance of our proposed method can be regarded as a very good meta-heuristic algorithm since it does not make the use of problem specific knowledge such as the critical paths concept or extensive speed-ups as used in other high-performing algorithms.

In modern manufacturing systems such as agile and flexible manufacturing systems, there is an increasing demand for customized products, which are produced in smaller lot sizes than before. Therefore, there appear to be an increased focus on finding new methods that have the most of the strategic advantages of a GFS but also can provide some of the operational advantages of an assembly line without flexibility limitations of Cellular Manufacturing Systems. Virtual Cellular Manufacturing Systems (VCMSs) are a new manufacturing technology generated from the changing and dynamic marketing environment, which has gained momentum during the last decade. Therefore, extensions of the proposed EM to VCMS considering other objectives or features, such as sequence-dependent setup times, are possible. For further insight, it will be interesting to work on other iterative algorithms or on hybridization of the proposed EM with other algorithms to achieve even better results.

References

1. S.M. Johnson, Optimal two and three-stage production schedules with set up times included, Naval Research Logistics Quarterly 1 (1954) 61–68.
2. W.H. Yang, A study on the intelligent neural network training using the electromagnetism algorithm, Unpublished Master Thesis, Dept. of Industrial Engineering and Management, I-Shou University, Kaohsiung County, Taiwan, 2002.
3. R. Ruiz, T. Stützle, An Iterated Greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives, European Journal of Operational Research 187, 3 (2008) 1143–1159.
4. M. Khalili, M.J. Tarokh, B. Naderi, Using electromagnetism algorithm for determining the number of kanbans in a multi-stage supply chain system, Journal of Industrial Engineering 6 (2010) 63–72.
5. P. Wu, W.-H. Yang, N.-C. Wei, An electromagnetism algorithm of neural network analysis – an application to textile retail operation, Journal of the Chinese Institute of Industrial Engineers 21 (2004) 59–67.
6. M.S. Salvador, A solution to a special case of flow shop scheduling problems, in: S.E. Elmaghraby (Ed.), Symposium of the Theory of Scheduling and its Applications, Springer, New York, 1973, pp. 83–91.
7. R. Linn, W. Zhang, Hybrid flow shop scheduling: a survey, Computers & Industrial Engineering 37, 1–2 (1999) 57–61.
8. H. Wang, Flexible flowshop scheduling: optimum, heuristics, and artificial intelligence solutions, Expert Systems 22, 2 (2005) 78–85.
9. O. Moursli, Y. Pochet, A branch-and-bound algorithm for the hybrid flowshop, International Journal of Production Economics 64, 1–3 (2000) 113–125.
10. C. Sriskandarajah, S.P. Sethi, Scheduling algorithms for flexible flowshops: worst and average case performance, European Journal of Operational Research 43, 2 (1989) 143–160.
11. A. Guinet, M.M. Solomon, P.K. Kedia, A. Dussauchoy, A computational study of heuristics for two-stage flexible flowshops, International Journal of Production Research 34, 5 (1996) 1399–1415.
12. E. Nowicki, C. Smutnicki, The flow shop with parallel machines: a tabu search approach, European Journal of Operational Research 106, 2–3 (1998) 226–253.

13. M. Gourgand, N. Grangeon, S. Norre, Metaheuristics for the deterministic hybrid flow shop problem, Proceeding of the International Conference on Industrial Engineering and Production Management (IEPM'99), Glasgow, United Kingdom, July 12–15 (1999), pp. 136–145.
14. R. Zhang, C. Wu, A simulated annealing algorithm based on block properties for the job shop scheduling problem with total weighted tardiness objective, Computers & Operations Research 385 (2011) 854–867.
15. C.R. Reeves, A genetic algorithm for flowshop sequencing, Computers & Operations Research 22, 1 (1995) 5–13.
16. R. Tavakkoli-Moghaddam, N. Safaei, F. Sassani, A memetic algorithm for the flexible flow line scheduling problem with processor blocking, Computers & Operations Research 36 (2009) 402–414.
17. R. Cheng, M. Gen, M. Tozawa, Minmax earliness/tardiness scheduling in identical parallel machine system using genetic algorithms, Computers & Industrial Engineering 29, 1–4 (1995) 513–517.
18. J. Yang, Minimizing total completion time in two-stage hybrid flow shop with dedicated machines, Computers & Operations Research 38, 7 (2011) 1045–1053.
19. M. Khalili, R. Tavakoli-Moghadam, A multi-objective electromagnetism algorithm for a bi-objective flowshop scheduling problem, Journal of Manufacturing Systems 31 (2012) 232–239.
20. C. Chen, R. Neppalli, Genetic algorithms applied to the continuous flow shop problem, Computers & Industrial Engineering 30, 4 (1996) 919–929.
21. T. Aldowaisan, A. Allahverdi, New heuristics for m-machine no-wait flowshop to minimize total completion time, Omega 32, 5 (2004) 345–352.
22. Q.K. Pan, M.F. Tasgetiren, Y.C. Liang, A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem, Computers & Operations Research 35 (2008) 2807–2839.
23. A. Fink, S. Voß, Solving the continuous flow-shop scheduling problem by metaheuristics, European Journal of Operational Research 151 (2003) 400–414.
24. S.J. Shyu, B.M.T. Lin, P.Y. Yin, Application of ant colony optimization for no-wait flowshop scheduling problem to minimize the total completion time, Computers & Industrial Engineering 47 (2004) 181–193.
25. J. Grabowski, J. Pempera, Some local search algorithms for no-wait flow-shop problem with makespan criterion, Computers & Operations Research 32 (2005) 2197–2212.
26. M. Khalili, An iterated local search algorithm for flexible flow lines with sequence dependent setup times to minimize total weighted completion, International Journal of Management Science and Engineering Management 7, 1 (2012) 63–66.
27. M. Khalili, Multi-objective no-wait hybrid flowshop scheduling problem with transportation times, Journal International Journal of Computational Science and Engineering 7, 2 (2012) 147–153.
28. J. Hurink, S. Knust, Makespan minimization for flow-shop problems with transportation times and a single robot, Discrete Applied Mathematics 112 (2001) 199–216.
29. A. Soukhal, A. Oulamara, P. Martineau, Complexity of flow shop scheduling problems with transportation constraints, European Journal of Operational Research 161 (2005) 32–41.
30. R. Ruiz, C.J. Garica-Diaz, C. Maroto, Considering scheduling and preventive maintenance in the flowshop sequencing problem, Computers & Operations Research 34 (2007) 3314–3330.
31. G. Schmidt, Scheduling with limited machine availability, European Journal of Operational Research 121 (2000) 1–15.
32. C.Y. Lee, Minimizing the makespan in the two-machine flowshop scheduling problem with an availability constraint, Operations Research Letters 20 (1997) 129–139.
33. J. Blazewicz, J. Breit, P. Formanowicz, W. Kubiak, G. Schmidt, Heuristic algorithms for the two-machine flowshop problem with limited machine availability, Omega 29 (2001) 599–608.
34. J. Breit, A polynomial-time approximation scheme for the two-machine flow shop scheduling problem with an availability constraint, Computers & Operations Research 33 (2006) 2143–2153.
35. H. Allaoui, A. Artiba, Integrating simulation and optimization to scheduling a hybrid flow shop with maintenance constraints, Computers & Industrial Engineering 47 (2004) 431–450.
36. V. Cerny, Thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm, JOTA 45 (1985) 41–51.
37. I. Birbil, S.C. Fang, An electromagnetism-like mechanism for global optimization, Journal of Global Optimization 25 (2003) 263–282.
38. D. Debels, B.D. Reyck, R. Leus, M. Vanhoucke, A hybrid scatter search/electromagnetism meta-heuristic for project scheduling, European Journal of Operational Research 169 (2006) 638–653.
39. E. Taillard, Benchmarks for basic scheduling problems, European Journal of Operational Research 64, 2 (1993) 278–285.
40. R. Ruiz, A. Allahverdi, Some effective heuristics for no-wait flowshops with setup times to minimize total completion time, Annals of Operation Research 156 (2007) 143–171.
41. B. Naderi, M. Mousakhani, M. Khalili, Scheduling multi-objective open shop scheduling using a hybrid immune algorithm, The International Journal of Advanced Manufacturing Technology 66, 5–8 (2013) 895–905.
42. D.C. Montgomery, Design and Analysis of Experiments, Fifth edition, John Wiley & Sons, 2000.

Cite this article as: Khalili M: An electromagnetism-inspired method for a generalized flowshop problem. Manufacturing Rev. 2014, **1**, 13.