Journal of Theoretical and Applied Information Technology
15th November 2016. Vol.93. No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ELECTROMAGNETIC-LIKE MECHANISM FOR JOB-SHOP SCHEDULING BY NOVEL HEURISTIC INITIALIZATION

[1]MAJID ABDOLRAZZAGH-NEZHAD, [2]ERNA BUDHIARTI NABABAN, [3]NAJMEH SADAT JADDI, [4]HAFIZ MOHD SARIM

[1] Department of Computer Engineering, Bozorgmehr University of Qaenat, Qaen, Iran.
[2] Department of Information Technology, Universitas Sumatera Utara, Medan, Indonesia
[3,4] Data Mining and Optimization Research Group, Centre for Artificial Intelligence Technology
Universiti Kebangsaan Malaysia, Bangi, Selangor, Malaysia
E-mail: [1]abdolrazzagh@buqaen.ac.ir, [2]ernabrn@usu.ac.id, [3]najmehjaddi@gmail.com, [4]hms@ukm.edu.my

ABSTRACT

This paper describes a novel method to enhance the performance of population-based algorithms in solving the job-shop scheduling problem. A novel heuristic initialization technique that is based on the concept of head and tail paths is applied to produce a new initial population. The proposed method is based on an intelligent skip from the primal point of the solution space to a better one, which is achieved by shortening the maximum head and tail paths (SMHT) of all jobs on the given machine. Also in this method, the electromagnetic-like mechanism is applied as an improvement algorithm as it is the state-of-the-art choice to improve the produced initial populations. The experimental results show that the quality of the initial population produced by SMHT is better than that produced by some state-of-the-art techniques. Moreover, the experimental results for the Electromagnetic-like Mechanism part of the method show that SMHT makes a significant contribution to accelerating the convergence speed of the improvement algorithm to optimality and improves the obtained output results.

Keywords: *Heuristic, Initialization, Job-Shop Scheduling Problem, Electromagnetic-Like Mechanism, Head And Tail Path.*

1. INTRODUCTION

The job-shop scheduling problem (JSSP) has extensive applications in industry, management, transportation, and so on. It is a non-deterministic polynomial hard (NP-hard) optimization problem [1]. Current efforts to solve the JSSP using metaheuristic algorithms involve the use of population-based algorithms such as the genetic algorithm (GA) [2], memetic algorithm (MA) [3], tabu search (TS) [4], simulated annealing (SA)[5], ant colony (AC) [6], bee colony (BC) [7], immune algorithm (IA) [8], and particle swarm optimization (PSO) [9]. To implement a metaheuristic for JSSP, three steps need to be followed: preprocessing, initialization, and improvement. Preprocessing involves indicating the solution space and aiming a decoder algorithm to construct the active schedule matching to the encoded solution. There are nine representations for the JSSP and their definitions are described in [10]. Next, an initial population has to be constructed in the initialization step [11, 12]. Although the production of an initial population has attracted less attention than the other steps of the metaheuristic algorithm, a high-quality initial population is important as it speeds up these algorithms. At the end, the initial solutions are enhanced by heuristic algorithms that apply exploration and exploitation mechanism on search space of the JSSP.

The JSSP has been constructed by a variety of techniques such as priority dispatching rules, random methods, and heuristic algorithms. Regarding to the literature, most of the researchers have employed random techniques for example random keys to construct the initial population [3, 5, 8, 9, 13-17]. However, random techniques have three main disadvantages. First, there is a high probability that infeasible solutions will be generated. Second, the quality of the solutions produced is extremely below that of the optimal solution and thirdly, the algorithms that employ random initialization step take longer to attain the optimal solution compared to algorithms that use heuristic techniques for initialization step. Priority dispatching rules [2, 4, 7, 18-20] are ranked second among initialization techniques based on its number of applications in the literature.

Journal of Theoretical and Applied Information Technology
15th November 2016. Vol.93. No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Some heuristic initialization algorithms have also been proposed for the JSSP [11, 12, 21-23]. These algorithms have complicated structures and a slightly longer calculation time in comparison to random techniques and priority rules so they have attracted less interest. However, the significant advantage of heuristic initialization procedures is their ability to generate an initial population that is close to the optimal solution. This advantage speeds up the improvement algorithms enabling them to reach the optimal solution sooner and compensates for the abovementioned small amount of extra time consuming on the initialization step. However, there are as yet few initialization algorithms that have ability to construct initial solutions near to the optimum solution, which is indicative of a significant research gap in terms of solving the JSSP.

Nevertheless, a review of the methodologies for the JSSP in the available literature [24, 25] shows that, recently, metaheuristic algorithms have clearly become the most popular method and have demonstrated many improved performances when applied to the JSSP. Undoubtedly, these metaheuristic approaches have already achieved a certain amount of success in solving scheduling problems and many other combinatorial optimization problems [2], but there is more work to be done.

With regard to further improving the efficacy of metaheuristics when applied to the JSSP, the electromagnetic-like mechanism (EM) has three special advantages: it possesses memory, where the characteristics of the good solutions are retained by all particles/solutions, it enables constructive cooperation among the particles, where all electrically charged particles share their information, and it moves the points of the population toward global optimality by using an attraction-repulsion mechanism. The EM is a state-of-the-art algorithm based on the attraction-repulsion mechanism of electromagnetic theory and has benn implemented by [14] for the JSSP with a sequence-dependent setup time. The EM has also been used in [13], where it was hybridized with SA for periodic JSSP and in [15], where it was hybridized with SA for the JSSP with machine availability and sequence-dependent setup times.

In this paper the combination of a novel heuristic technique to construct the initial population and a modified version of the EM as an improvement algorithm are proposed to solve the JSSP. The proposed initial population construction technique is intended to improve existing population-based algorithms by constructing an initial population near to the optimal solution in a well enough small computation time. To attain this aim, a new initialization procedure is proposed, which is based on the concept of head and tail paths and involves skipping from the initial solution to an enhanced solution using the intelligent ideas first proposed in [1].

2. RESEARCH METHOD

Typically, a JSSP can be described as follows: assume that there are n jobs and m machines, where the jobs have to be developed on the machines based on a number of sets of hard and soft constraints. There are three kinds of constraints for the JSSP: precedence constraints, capacity constraints, and release and due date constraints.

Three precedence constraints are presented: every job must be procedure by the machines in a fixed order (sequence of operations for jobs, SOJ), the order of machines among the different jobs is free, and there are no precedence constraints for the actions needed to complete special jobs. There are also three capacity constraints: machines can handle only one process at a time independently, every job can be progressed once on a specified machine, so all the jobs must be processed independently from one other. Finally, there are three release and due date constraints there exist no negative starting times, the processing times of the actions have specified lengths, and the processing of each function cannot be suspended. Therefore, in order to attain the objective of the JSSP and to deal with these constraints, the preliminary processing time of operations is assumed to be the JSSP decision variable.

A schedule is formed by assigning time slots to the operations of jobs on machines while satisfying the problem's constraints and discovering a sequence of jobs on the machines (SJM). As the typical objective function of the JSSP [1, 17] the SJM schedule minimizes the maximum achievement period of the terminated operation (makespan/ C_max). Thus the target of the problem is to find a SJM whose schedule can simultaneously ensure that all soft and hard constraints are met and that the makespan is minimized.

2.1. Pre-processing

An edited version of preference list-based representation proposed in [10] where matrix form of the encoding scheme for the initialization part of the proposed method SJM is assumed to be the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

encoding scheme and to stand for the points in the solution space [12]. The rows in SJM matrix correspond to a variation in the sequence of the jobs to process on a given machine while the factors of the SJM are the job's number. The EM was intended for continuous optimization problems with bounded variables. With regard to the structure of the EM, the operation-based representation, where a Sequence of Jobs with the length of nm, called SJ, is considered in a vector design as the encoding scheme for the improvement part of the method. To change the discrete structure of operation-based representation to a continuous structure, a series of real values between $(0,1)$, called sorted real random numbers (RSJ), is the corresponding relationship between RSJ and SJ.

In order to calculate the RSJ, the length of $(0,1)$ is separated into nm equal parts. A random value is then generated from each part and allocated to the job corresponding to the part. Since these real values alter during the application of the EM, the new exchanged RSJ is sorted increasingly and the sequence of jobs in SJ is altered based on the new sort of its related real values in RSJ. Since a type of preference list-based representation is used for initialization, a convertor is mapped to convert a specified SJM to its related SJ. The process of converting SJM to SJ is based on placing the job with the minimum of starting time in the initial location in the SJ and removing that job from the SJM. The procedure undertaken by this convertor is presented in Figure 1.

```
Convertor SJM to SJ
Set all operations of jobs in SJM as Ω
1 → i
while (Ω ≠ ∅)
    min_{O_rs∈Ω}(ST(O_rs)) = ST(O_ab)
    J_a → SJ(1,i)
    Remove O_ab from Ω
    ++i
end while
```

Figure 1 Procedure of Converting SJM to SJ.

The procedure of converting SJ to SJM is explained in Figure 2. In this paper, the switching function, which is presented in [12], is assumed to be a decoding algorithm for constructing a feasible schedule corresponding to a specified SJM.

```
Convertor SJ to SJM
Set the repeat number of jobs with the length n as Repeat_jobs
Set the number of jobs to settle in machines with the length m as NJ_mach
1 → Repeat_jobs(i) : i = 1,…,n
1 → NJ_mach(j) : j = 1,…,m
for i = 1 : nm
    SJ(i) → J_k
    Repeat_jobs(k) → r
    SOJ(k,r) → m //     SOJ(O_kr)
    NJ_mach(m) → p
    SJ(i) → SJM(m,p)
    ++Repeat_jobs(k)
    ++NJ_mach(m)
end for
```

Figure 2 Procedure of Converting SJ to SJM.

2.2. Initialization Using SMHT

In order to speed up metaheuristic algorithms and to increase the quality of the results generated by these algorithms, a novel heuristic initialization procedure is investigated in order to construct the initial population close to the optimal solution for the JSSP. This algorithm works based on moving from a initial solution in the search space to a better solution by Shortening the Maximum Head and Tail paths (SMHT) of jobs on the given machine. Before discussing how SMHT is applied in the proposed method, the concept of SMHT is explained.

2.2.1. The concept of head and tail paths

The concept of head and tail paths was first proposed on insertion techniques [26] for solving the JSSP. It was presented in the form of a disjunctive graph-based representation. To explain this concept, a kind of presentation of SJM such as SJM^2 is described and the relations among the jobs of each machine (M_1, M_2, and M_3) are shown by bold arrows in Figure 3.

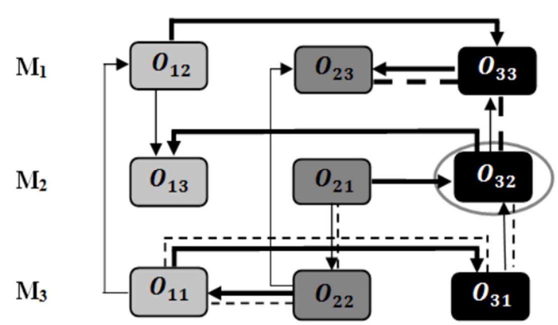

Figure 3 Head and Tail for O_{32} Based On SJM^2

The weight of the bold arrows is equal to the processing time of the origin operation. For example, the weight of the bold arrow between

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$O_{12}$and O_{33} is the processing time of O_{12} and the weight of normal directed line between $O_{22}$and O_{23} is the processing time O_{22}. The sources are the primary processed operation of jobs with the least order on each machine. In Figure 3, there are no sources for $M_1$and M_3, but O_{21} is the source for M_2. The sinks are the most recent processed operation of jobs with the maximum order on each machine. O_{23} is the sink for M_1, O_{13} is the sink for M_2, and there is no sink for M_3. The length of the path between two operations O_{ij} and O_{rk} is equal to the sum of the weights of the directed lines that connect O_{ij} to O_{rk}. For example, from O_{21} to O_{13} in the instance, there are three paths, as shown in Eq. 1:

$$\begin{cases} O_{21} \xrightarrow{3} O_{32} \xrightarrow{4} O_{13} \\ O_{21} \xrightarrow{3} O_{22} \xrightarrow{5} O_{11} \xrightarrow{1} O_{12} \xrightarrow{3} O_{13} \\ O_{21} \xrightarrow{3} O_{22} \xrightarrow{5} O_{11} \xrightarrow{1} O_{31} \xrightarrow{5} O_{32} \xrightarrow{4} O_{13} \end{cases} \quad (1)$$

where the lengths of these paths are 7, 12, and 18, respectively, and the longest path from O_{21} to O_{13} is 18. Therefore the head for a given operation O_{ij} is equal to the longest path from one of the sources to O_{ij} and the tail for a given O_{ij} is the length of the longest path from O_{ij} to one of sinks.

The head and the tail for O_{32} based on SJM^2 are shown with normal dashed lines ($O_{21} \xrightarrow{3} O_{22} \xrightarrow{5} O_{11} \xrightarrow{1} O_{31} \xrightarrow{5} O_{32}$) and bold dashed lines ($O_{32} \xrightarrow{4} O_{33} \xrightarrow{3} O_{23}$), respectively. If $SJM^2$is exchanged, the directions of the bold arrows will change and different values will be obtained for the head and tail for given operations. A similar procedure is used to produce head and tail paths for another given operation.

The flowcharts for generating the head path are given in Figure 4, where $PSJM$ and $PSOJ$ have been generated based on SJM and SOJ, respectively. $PSJM$ presents the sequence orders of jobs on machines and $PSOJ$ shows the sequence of machines that the jobs should pass through

A comparison of SJM with PSJM and SOJ with PSOJ is presented in Eq. 2 and 3, respectively:

$$\begin{cases} SJM(M_k, Order) = J_i \\ PSJM(M_k, J_i) = Order \end{cases} \quad (2)$$

$$\begin{cases} SOJ(O_{ij}) = M_k \\ PSOJ(J_i, M_k) = O_{ij} \end{cases} \quad (3)$$

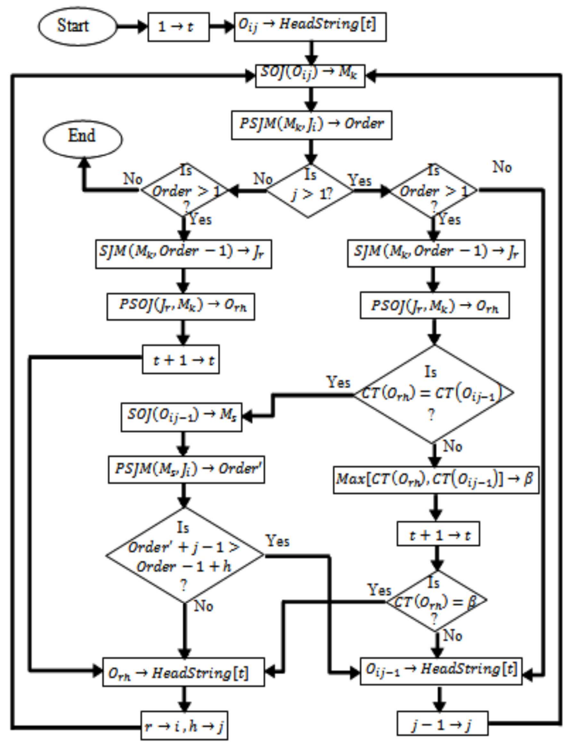

Figure 4 Procedure for Generating The Head Path of O_{ij}

2.2.2. The Heuristic Skipping Strategy (SMHT) for the proposed method

In the SMHT strategy, an exchanged initial SJM is constructed by calculating the head and tail paths for all jobs on the given machine and trying to shorten the maximum of the head and tail paths. At the end, the sequenced of the moved jobs on the other machine are maintained. A summary outline of the skipping strategy is given below:

Step 1. Calculate the head and tail of all jobs on machine α (M_α).
Step 2. Focus on the job or jobs that have the maximum length of head and tail.
Step 3. Swap the processing order of the job or jobs on M_α randomly by taking account of Condition 1:
Condition 1: Let J_i with operation O_{ij} and J_r with operation O_{rs} that should be processed on M_α and also these jobs with operations $O_{ij'}$ and $O_{rs'}$ respectively, that should be processed on M_β on a given SJM. **(1)** If the order of O_{ij} to process on M_α is smaller than the order of O_{rs} on M_α and ($j' < j$) and ($s' > s$) then the order of $O_{ij'}$ to process on M_β cannot be bigger than the order of $O_{rs'}$ on M_β. **(2)** If the order of O_{ij} to process on M_α is bigger than the order of O_{rs} on M_α and ($j' > j$)

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

and ($s' < s$) then the order of $O_{ij'}$ to process on M_β cannot be smaller than the order of $O_{rs'}$ on M_β.

Step 4. Recalculate the head and tail of all jobs and find their new maximum lengths.

Step 5. If the new maximum length of the head and tail paths is shorter than the previous one, replace the new exchanged sequence of jobs on M_α in the corresponding row of the primal SJM and go to Step 6, else terminate the procedure of the skipping strategy.

Step 6. In the new sequence of jobs on M_α, find any job (for example, J_r) which has obtained a better processing order than its previous order, and then modify the processing order of jobs on other machines if Condition 2 and Eq. 4 are satisfied:

Condition 2: Consider the jobs which belong to orders $\rho - 2$, $\rho - 1$, ρ, and $\rho + 1$ on a given machine. These jobs equal J_p with operation O_{pq}, J_i with operation O_{ij}, J_r with operation O_{rs}, and J_a with operation O_{ab}, respectively. If they meet Eq. 5, then swap the order of J_r and J_i and recalculate the makespan of the exchanged SJM.

$$\begin{cases} max\left(CT(O_{rs-1}), CT\left(O_{pq}\right)\right) + PT(O_{rs}) \leq CT(O_{rs+1}) \\ \quad and \\ max\left(CT(O_{ij-1}), \theta\right) + PT\left(O_{ij}\right) \leq CT(O_{ab}) \end{cases} \quad (4)$$

where $\theta = max\left(CT(O_{rs-1}), CT\left(O_{pq}\right)\right) + PT(O_{rs})$. If J_a is not available then $CT(O_{ab})$ is equal to the makespan of the initial SJM and if J_p is not existing then $CT\left(O_{pq}\right)$ equals to zero.

2.2.3. Framework to generate the initial population

In the proposed method, a novel heuristic strategy incorporating the above described SMHT is used to construct the initial population and consists of two major parts. In the first part, by employing the priority rules one initial SJM has been produced. Note that the existing literature on generating initial schedules employs the priority rules in a straight line, but in the current study the rules have been employed for generating the primal SJM^1. The rule shortest remaining time first (SRTF) is employed to intend SJM^1. The second part includes creating the remaining SJM^i, where $i = 2, \dots, pop\_size$. In this part, to compute the remaining $pop\_size - 1$ of SJMs, the SJM^{i-1} produced before is assumed to be the initial SJM, then the SMHT is applied to find a better SJM on two machines of the initial SJM successively. Before the operation of SMHT, a machine (a row of SJM^{i-1}) called α is chosen at random from among all of the machines by generating a random number from $[1 \dots m]$. The SMHT is calculated based on identifying operations with the maximum distance between their heads and tails on M_α and moving these operations on M_α to shorten the maximum distance between their heads and tails by taking account of Condition 1 and Condition 2. The aim of the SMHT is to discover a new SJM with improved quality. So the SMHT zooms on M_α, that is placed in row α of SJM^{i-1}, and next modifies the sequences of jobs on M_α and the other machines. Then, the SJM^i is reconsidered as the initial SJM to construct additional SJM by choosing another machine randomly, apart from M_α, and the SMHT is performed on the SJM^i.

In the procedure using the SMHT on M_α, if a better SJM is not found subsequent to R_{expect} times of repeats to find a new better SJM that R_{expect} corresponds to the number of jobs, and then a random reselection of value of α from among $[1 \dots m]$, omitting the present value of α. Next, a re-run of ISS is considered derived from the new value of α. The Reselection of the value of α is applicable up to the number of machines. This reselection is indicated by R_{repeat}. When the process has been unsuccessful R_{repeat} times in discovering a new enhanced SJM^i by reselection of the value of α, SJM^{i-2} is considered as the primal SJM to generate SJM^i. If i equals 2 ($i = 2$), then the initial SJM is constructed by the precedence rule that has not yet been concerned in the generation process. For instance, if SJM^1 is generated by SRTF and there are unsuccessful R_{repeat} times in discovering a new enhanced SJM^2 by reselection of value of α, then a new initial SJM is constructed by the first due date. This process is finished to generate SJM^i because the new order of jobs is discovered on two rows of the initial SJM

2.3. Electromagnetic-Like Mechanism (EM)

The EM works based on the attraction-repulsion mechanism of electromagnetic theory. In this procedure, each candidate solution is considered as an electrically charged particle. The objective function value of each candidate solution assigns the charge of each particle and calculates the magnitude of attraction of the particles over the population. Each sample particle is moved based on a vector reflecting the charge of the corresponding particle relative to the other particles in the population. The direction of the objective function value for particle i is calculated by addition of forces from other particles on particle i. In this

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

strategy, particles (solutions) with lower charges (better objective function values) attract others, while those with higher charges (worse objective function values) will repel each other. There are four phases in the EM, as shown in Figure 5.

```
Procedure Electromagnetism-like mechanism
Initialization
While termination criterion are not satisfied do
    Local search
    Computation of total forces
    Movement by total forces
end while
```

Figure 5 Procedure of The EM

In this paper, the EM is performed by the proposed heuristic initialization (SMHT) and random initialization. Random generated initial points are obtained by producing random keys for operations and sorted them by [15]. In this initialization, the operations are listed first. For example, the list of operations belonging to the problem presented in Table 1 is (1,1,1,2,2,2,3,3,3).

Furthermore, a random number from a uniform distribution between (0, 1) is generated for each operation. Then, these random numbers are sorted to find the relative order of operations. The sorted real random numbers are denoted as RSJ and the related operations list is denoted as SJ. In order to decode the generated SJ, first the generated SJ should be converted into its corresponding SJM by running the convertor of SJ to SJM (see Figure. 2). The procedure of this random initialization is presented in Figure 6.

Random Initialization Procedure

for $i = 1$: *pop_size*
  Set the operations list of the problem as SJ^1
  Generate a random number from (0,1) related to each component of SJ^1 as RSJ^1
  Sorted the components of $RSJ^1 \rightarrow RSJ^1$
  Sort the components of SJ^1 based on the new order of the related components in RSJ^1
  Convert SJ^1 to SJM^1 by running the convertor Figure 6.2
  Run Switching Function for SJM^1 to decode SJ^1 and calculate its objective function
end for

Figure 6 Procedure of Random Initialization

Table 1. Comparison of The Experimental Results

Instance	**Size**	**BKS**	**SMHT**	**T**	**SMHT+EM**	**T**	**Random+EM**	**T**
FT06	6×6	55	71-55	0.7	**55**	0.7	**55**	1.3
FT10	10×10	930	1509-1046	3.1	**930**	4.7	968	5.4
Ft20	20×5	1165	1739-1218	4.9	**1165**	6.8	1211	8.3
Willem	10×10	95	109-95	2.1	**91**	2.8	**95**	2.9
Abz5	10×10	1234	1527-1303	2.6	**1234**	4.1	1269	4.7
Abz6	10×10	943	1344-998	2.8	**943**	3.9	**943**	4.6
Abz7	20×15	656	968-764	6.5	661	8.8	673	10.3
Abz8	20×15	665	1023-797	6.7	**665**	8.5	689	10.0
Abz9	20×15	679	1026-842	6.4	697	8.2	754	10.2
La05	10×5	593	838-593	1.5	**593**	1.5	595	2.3
La08	15×5	863	1175-863	2.7	**863**	2.7	**863**	4.1
La14	20×5	1292	1420-1292	3.2	**1292**	3.2	**1292**	5.0
La27	20×10	1235	1806-1412	5.3	**1235**	6.5	1301	8.1
La28	20×10	1216	1928-1373	4.8	**1216**	6.2	1276	7.4
La29	20×10	1157	1934-1381	5.1	**1157**	6.7	1187	7.3
La30	20×10	1355	2110-1542	5.4	**1355**	6.5	1369	7.8
La31	30×10	1784	2641-1784	7.3	**1784**	7.3	1785	10.6
La32	30×10	1850	2567-1869	7.8	**1850**	9.1	**1850**	12.1
La33	30×10	1719	2416-1731	7.7	**1719**	9.9	**1719**	12.7
La34	30×10	1721	2789-1876	8.0	**1721**	10.8	1733	13.6
La37	15×15	1397	2120-1588	7.2	**1397**	8.9	1507	10.1
La38	15×15	1196	1830-1398	7.0	1204	8.5	1381	10.6
La39	15×15	1233	1767-1429	6.9	**1233**	8.7	1312	10.3
La40	15×15	1222	1844-1415	7.4	**1222**	8.9	1345	10.5
Orb07	10×10	397	1344-998	2.4	**397**	3.1	402	3.9
Swv05	20×10	1678	2929-2134	5.5	**1678**	8.8	1715	10.1
Yn1	20×20	885	1401-1113	9.8	902	13.6	1101	18.8
Yn2	20×20	909	1362-1106	9.5	914	12.9	1086	17.5
Yn3	20×20	892	1253-1098	9.7	910	13.4	1077	18.3
Yn4	20×20	969	1450-1224	10.1	971	14.0	1123	18.9

Journal of Theoretical and Applied Information Technology
15th November 2016. Vol.93. No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

2.4. Local Search

The purpose of local search (LS) is to explore the search space for a better quality of solution. For the EM, random LS is utilized and repeated Max_iter times in the neighbourhood of a given point in the population to find a better new solution than the current solution. If by the maximum number repetitions LS has failed to find a better solution, another point in the population is selected on which to perform the neighbourhood search. This selection is also be repeated up to Max_Iter times. The neighbourhood search is also performed on a given point in the population that is a candidate solution SJ^k by selecting a component of SJ^k such as $SJ^k(i)$. Next, a new random value from a uniform distribution between (0,1) is generated and assigned as $RSJ^k(i)$. Re-sorting RSJ^k and SJ^k based on the newly sorted RSJ^k are the next steps of the neighbourhood search. To calculate the quality of the exchanged version of SJ^k, it has to be converted into its corresponding SJM and then the quality of SJM is calculated by performing the switching function on SJM related to the exchanged version of SJ^k. The procedure of random LS is presented in Fig. 7.

```
Random LS
Set the maximum iteration on different solutions of population → Max_Iter
Set the maximum iteration on a given solution → Max_iter
1 → Iter
while (Iter ≤ Max_Iter)
    Select one of solutions from population → SJ^k
    1 → iter
    while (iter ≤ Max_iter)
        Select a component of SJ^k randomly → SJ^k(i)
        Regenerate a random value from (0,1) → RSJ^k(i)
        Resort RSJ^k
        Resort SJ^k based on RSJ^k
        Convert SJ^k to SJM^k by running the convertor Figure 6.2
        Run Switching Function for SJM^k to decode SJ^k
        Calculate the quality of the exchanged version of SJ^k
        if (the quality of the exchanged SJ^k better than the quality of SJ^k)
            exchanged SJ^i → SJ^i
            Max_iter → iter
            Max_Iter → Iter
        end if
        ++iter
    end while
    ++Iter
end while
```

Figure 7 Pseudocode of Random LS.

2.5. Total Force Computation

To calculate the force among two particles, first a charge q_i is allocated to each particle SJ^i. In order to calculate the charge of the particle the relative quality of the objective function values in the present population is used as in Eq. 5:

$$q_i = exp\left(-n \times \frac{f(SJ^i)-f(SJ^{Best})}{\sum_{j=1}^{pop\_size}\left(f(SJ^j)-f(SJ^{Best})\right)}\right) \quad (5)$$

where $i = 1, \dots, pop\_size$ and SJ^{Best} is the particle with the best objective function between the particles at the present iteration. Thus, the particles with superior objective function values have superior charges. As noted above, SJ^i, as the representation scheme of the solution space of the JSSP applied to the EM, has a discrete structure and its related real value matrix (RSJ^i) has a continuous structure. As also noted earlier, the EM is designed to minimize continuous optimization problems with bounded variables. Therefore RSJ^i is considered as the candidate particle to calculate the total force F_i, exerted on the candidate particle SJ^i by Eq. 6:

$$F_i = \sum_{\substack{j=1 \\ i\neq j}}^{\substack{pop \\ size}} \begin{cases} (RSJ^j - RSJ^i)\dfrac{q_i q_j}{\|RSJ^j - RSJ^i\|^2} & iff(SJ^j) < f(SJ^i) \\ (RSJ^i - RSJ^j)\dfrac{q_i q_j}{\|RSJ^j - RSJ^i\|^2} & if \;\; f(SJ^j) \geq f(SJ^i) \end{cases} \quad \dots (6)$$

2.6. Movement by Total Force

The calculation of total force used on each particle by all the other particles is performed based on Eq. 5 and 6and the procedure shown in Figure 8. All real sequences (RSJ^i), which are related to the solutions (SJ^i), are moved with the omission of the present best solution. The movement of each RSJ^i is done in the direction of F_i, and F_i is exerted on RSJ^i by a random step length generated from a uniform distribution between (0, 1). The moved RSJ^i is then re-sorted and based on this newly sorted RSJ^i the order of jobs is sorted in SJ^i and the corresponding moved SJ^i is obtained. Note that the feasibility of the moved particles can be guaranteed and the candidate particles can be moved to the unvisited solution with a nonzero probability.

```
Procedure of Calculation of Total Force
for i = 1: pop_size
    Calculate q_i based on Eq. 6.1
end for
for i = 1: pop_size
    for j = 1: pop_size
        if (f(SJ^j) < f(SJ^i))
            (RSJ^j − RSJ^i) q_i q_j / ||RSJ^j − RSJ^i||^2 → F_i    // attraction
        else
            (RSJ^i − RSJ^j) q_i q_j / ||RSJ^j − RSJ^i||^2 → F_i    // repulsion
        end if
    end for
end for
```

Figure 8 Procedure for Calculation of The Total Force.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3. COMPUTATIONAL RESULT

To validate the effectiveness and efficiency of the EM and demonstrate the influence of the SMHT on the metaheuristic algorithm, computational experiments were conducted on 30 benchmark datasets available from the OR-library (http://people.brunel.ac.uk/~mastjjb/jeb/orlib/jobshopinfo.html), including FT, LA, ORB, ABZ, SWV, and YN, and a benchmark dataset presented in [27] called Willem. These experiments were programmed by executing Matlab9 on an Intel Core2 Duo P8600 2.4 GHz RAM, 4 GB processor. The parameters of the SMHT and EM were tuned by running multiple simulations on some datasets. The size of initial population produced for each benchmark was twice the number of jobs on that. Random initialization and the SMHT were run 10 times on the datasets and the best initial population produced by each initialization procedure was saved as the output result of that procedure.

During the implementation of the SMHT, the value of R_{expect} for each dataset was 80% of the number of its jobs and the value of R_{repeat} was the number of machines available in the dataset. The stopping criterion of the EM was set at 300 iterations and the Max_Iter and Max_iter of the LS parameters were set at $\lfloor n/3 \rfloor$ and $\lfloor nm/10 \rfloor$, respectively. The measures considered in the experiments were the makespan (the maximum completion times on the machines), the quality of the points generated, and the CPU time (as the computational time) taken to produce points. The experimental results of the SMHT were compared with the results of some previous state-of-the-art construction techniques found in [19], [28] (see Table 2) and in [29] and [30] (see Table 3).

Table 2. Comparison of Our Experimental Results with [19], [28], and BKS

Instance	Size	BKS	FIFO	LIFC	HVF	LVF	FM	SM	YTPM	CSANN	SMHT	
											Ran. C_{max}	T
FT06	6×6	55	61	69	68	69	85.2	129.1	67.4	55	71–58	0.7
FT10	10×10	930	1184	1283	1240	1370	–	–	–	–	1509–1046	3.1
Willem	10×10	95	–	–	–	–	142	256	107	95	109–**95**	2.1
FT20	20×5	1165	1645	1291	1656	1336	–	–	–	–	1739–1218	4.9

Table 3. Comparison of Our Experimental Results with [29], [30], and BKS

Instance	Size	BKS	JRSD	ORSD	PRSD	G&T	SMHT	
							Ran. C_{max}	T
La05	10×5	593	–	–	–	658	838–**593**	1.5
La08	15×5	863	–	–	–	1035	1175–**863**	2.7
La14	20×5	1292	–	–	–	1292	1420–**1292**	3.2
Abz5	10×10	1234	1411	1461	1361	1774	1527–1303	2.6
Abz6	10×10	943	1212	1251	1076	1389	1344–998	2.8
Orb07	10×10	397	487	493	499	542	513–461	2.4
La37	15×15	1397	1807	2034	1883	1723	2120–1588	7.2
La38	15×15	1196	1561	1622	1521	1530	1830–1398	7.0
La39	15×15	1233	1720	1701	1560	1719	1767–1429	6.9
La40	15×15	1222	1642	1797	1596	1813	1844–1415	7.4
La27	20×10	1235	1730	1671	1626	1729	1806–1412	5.3
La28	20×10	1216	1612	2050	1478	1686	1928–1373	4.8
La29	20×10	1157	1594	1958	1551	1885	1934–1381	5.1
La30	20×10	1355	1682	1963	1605	1633	2110–1542	5.4
Abz7	20×15	656	909	1031	846	822	968–764	6.5
Abz8	20×15	665	925	1197	886	905	1023–797	6.7
Abz9	20×15	679	934	1080	950	977	1026–842	6.4
Swv05	20×10	1678	2272	2582	2564	2589	2929–2134	5.5
Yn1	20×20	885	1286	1349	1123	1203	1401–1113	9.8
Yn2	20×20	909	1299	1548	1144	1285	1362–1106	9.5
Yn3	20×20	892	1278	1585	1220	1274	1253–1098	9.7
Yn4	20×20	969	1388	1637	1334	1247	1450–1224	10.1
La31	30×10	1784	2160	2410	1902	2168	2641–**1784**	7.3
La32	30×10	1850	2309	2439	2142	2147	2567–1869	7.8
La33	30×10	1719	2145	2251	1951	2055	2416–1731	7.7
La34	30×10	1721	2209	2341	1961	2052	2789–1876	8.0

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Furthermore, to compare the results of the SMHT with the results of the state-of-the-art algorithms presented in Tables 2 and 3, the relative percentage of error (RPE) and the relative percentage of improvement (RPI) were calculated based on benchmark-by-benchmark. The RPE or the percentage of the deviation regarding to the BKS is a standard index to show the success of an algorithm in reaching the optimal points based on the quality of the points generated. In some publications such as [31], RPE is shown by RD%. It should be noted that the proposed algorithm has been designed to construct an initial population, so the value of RPE cannot be expected to equal zero, although in the implementations of the proposed method the value of RPE equals zero for some datasets. The RPI is a numerical index to compare different algorithms in terms of the quality of generated points, and its value represents the percentage improvement of a given algorithm relative to the target algorithm. The RPE and RPI were considered in Eq. 7 and 8 to compare and analyse the results based on the following

$$RPE = \frac{bf - BKS}{BKS} \times 100 \quad (7)$$

$$RPI = \frac{bf^{Target} - bf}{bf^{Target}} \times 100 \quad (8)$$

where bf is the best quality of initial population produced, BKS is the best-known solution of Eq. 7, bf^{Target} is the best quality of points generated by the target algorithm, and bf is equal to the best quality of points generated by the algorithm that is to be compared with the target algorithm in Eq. 8.

Table 4 shows the RPE for BKS and the RPI for the results of [19], [28], and [29] and simulated results of [30], as an average. From Table 4 it can be seen that although the deviation from the BKS of the best point in the initial population produced by SMHT is 12.3% on average, it has succeeded in attaining an average improvement of 15.7% compared to the results published in [19], 25.3% compared to the results published in [21], 14.7% compared to the results published in [20], and 14.0% compared to the simulated results published in [18].

Table 4. Comparison of SMHT with BKS, the results in [19], [28], and [29] and The Simulated Results Obtained by [30]

RPE/ RPI	SMHT				
	BKS	The results in			
		[19]	[28]	[29]	[30]
	12.3%	15.7%	25.3%	14.7%	14.0%

4. CONCLUSION

In this paper, a novel metaheuristic construction technique to construct an initial population was proposed derived from a strategy of skipping from the initial point to an improved point. The skipping strategy is intended to shorten the maximum of the head and tail paths (SMHT) of all jobs on a given machine and to modify the processing sequence of the jobs progressed to other machines. Also, the SMHT has the ability to create any dimension of initial an initial population that is close to the optimal solution in an acceptably less time consuming. The important point which can prove the advantage of the proposed construction technique is observation population. A comparison of the results with state-of-the-art methods showed that SMHT is able to generate of the better performance among the initial population of Willem, La05, La08, La14, and La31.

The proposed algorithm has been designed to construct an initial population, so the value of RPE cannot be expected to equal zero, however in the implementations of the proposed method the value of RPE equals zero for some datasets. Therefore in future works, we will aim to design more effective and intelligent procedures for the construction of the initial population in order to generate a better initial population in a shorter computation time than SMHT, based on analyzing the solution space, monitoring the location of a job's operations on various machines, checking their effects on the quality of solutions and on the gaps created in schedules, and discovering the causes of gaps and idle time on the right side of operations.

REFERENCES

[1] Applegate D, Cook W. "A Computational Study of The Job Shop Scheduling Problem". *ORSA Journal on Computing*. 1991, Vol (3):149-56.
[2] Zhang R, Wu C. "Bottleneck machine identification method based on constraint

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

transformation for job shop scheduling with genetic algorithm". *Information Sciences*. 2012; 188(0): 236-52.

[3] Qian B, Wang L, Huang DX, Wang X. "Scheduling Multi-Objective Job Shop Using a Memetic Algorithm Based on Differential Evolution". *The International Journal of Advanced Manufacturing Technology*. 2008; 35: 1014-27.

[4] Geyik F, Cedimoglu I. "The strategies and parameters of tabu search for job-shop scheduling". *Journal of Intelligent Manufacturing*. 2004 2004/08/01; 15(4): 439-48.

[5] Laarhoven P, Aarts E, Lenstra J. "Job Shop Scheduling by Simulated Annealing". *Oper Res.* 1992; 40: 113-25.

[6] Huang KL, Liao CJ. "Ant colony optimization combined with taboo search for the job shop scheduling problem". *Computers & Operations Research*. 2008; 35(4): 1030-46.

[7] Zhang R. "An Artificial Bee Colony Algorithm Based on Problem Data Properties for Scheduling Job Shops". *Procedia Engineering*. 2011; 23(0): 131-6.

[8] Luh GC, Chueh CH. "A Multi-Modal Immune Algorithm for The Job-Shop Scheduling Problem". *Information Sciences*. 2009; 179: 1516-32.

[9] Sha DY, Lin HH. "A Multi-Objective PSO for Job Shop Scheduling Problems". *Int J of Expert System with Application*. 2010; 37: 1065-70.

[10] Cheng R, Gen M, Tsujimura Y. "A tutorial survey of job-shop scheduling problems using genetic algorithms—I. Representation". *Computers and Industrial Engineering*. 1996; 30(4): 983-97.

[11] Abdolrazzagh-Nezhad M, Abdullah S, editors. "Robust start for population-based algorithms solving job-shop scheduling problems". *Proceeding on 3rd Conference on Data Mining and Optimization (DMO)*- IEEE, 2011

[12] Abdolrazzagh-Nezhad M, Abdullah S. "A Robust Intelligent Construction Procedure for Job-Shop Scheduling". *Information Technology and Control*. 2014; 43(3): 217-29.

[13] Jamili A, Shafia M, Tavakkoli-Moghaddam R. "A hybridization of simulated annealing and electromagnetism-like mechanism for a periodic job shop scheduling problem". *Expert Systems with Applications*. 2011; 38(5): 5895-901.

[14] Roshanaei V, Balagh AKG, Esfahani MMS, Vahdani B. "A mixed-integer linear programming model along with an electromagnetism-like algorithm for scheduling job shop production system with sequence-dependent set-up times". *The International Journal of Advanced Manufacturing Technology*. 2010; 47(5): 783-93.

[15] Tavakkoli-Moghaddam R, Khalili M, Naderi B. "A hybridization of simulated annealing and electromagnetic-like mechanism for job shop problems with machine availability and sequence-dependent setup times to minimize total weighted tardiness". *Soft Computing-A Fusion of Foundations, Methodologies and Applications*. 2009; 13(10): 995-1006.

[16] Park BJ, Choi HR, Kim HS. A hybrid Genetic Algorithm for the Job Shop Scheduling Problems. *Computers and Industrial eEngineering*. 2003; 45(4): 597-613.

[17] Zhang CY, Li PG, Rao YQ, Guan Z. "A very fast TS/SA Algorithm for The Job-Shop Scheduling Problem". *Computer and Operation Research*. 2008; 35: 282-94.

[18] Holthaus O, Rajendran C. "Efficient Dispatching Rules for Scheduling in a Job Shop". Int J Production Economics. 1997; 48: 87-105.

[19] Moghaddam RT, Daneshmand-Mehr M. "A Computer Simulation Model for Job Shop Scheduling Problems Minimizing Makespan". *Computer and Industrial Engineering*. 2005; 48: 811-23.

[20] Sels V, Craeymeersch K, Vanhoucke M. "A Hybrid Single and Dual Population Search Procedure for the Job Shop Scheduling Problem". *European Journal of Operational Research*. 2011; 215(3): 512-23.

[21] Kuczapski AM, Micea MV, Maniu LA, Cretu VI. "Efficient Generation of Near Optimal Initial Populations to Enhance Genetic Algorithms for Job-Shop Scheduling". *Information Technology and Control*. 2010; 39: 32-7.

[22] Bülbül K. "A hybrid shifting bottleneck-tabu search heuristic for the job shop total weighted tardiness problem". *Computers and Operations Research*. 2011; 38(6): 967-83.

[23] Yahyaoui A, Fnaiech N, Fnaiech F. "A Suitable Initialization Procedure for Speeding a Neural Network Job-Shop Scheduling". *IEEE Transactions on Industrial Electronics*. 2011; 58(3): 1052 - 60.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[24] Zobolas G, Tarantilis C, Ioannou G. Exact, "Heuristic and Meta-heuristic Algorithms for Solving Shop Scheduling Problems". *Metaheuristics for Scheduling in Industrial and Manufacturing Applications*. 2008:1-40.

[25] Biskup D. "A State-of-The-Art Review on Scheduling with Learning Effects". *European Journal of Operational Research*. 2008; 188(2): 315-29.

[26] Werner F, Winkler A. "Insertion Techniques for The Heuristic Solution of The Job Shop Problem". *Discrete Applied Math*. 1995; 58: 191-211.

[27] Willems TM, Brandts LW. "Implementing Heuristics as an Optimation Criterion in Neural Networks for Job Shop Scheduling". *International Journal of Manufufacturing*. 1995;6:377-87.

[28] Yahyaoui A, Fnaiech N, Fnaiech F. "A Suitable Initialization Procedure for Speeding a Neural Network Job-Shop Scheduling". *IEEE Transactions on Industrial Electronics*, 2011; 58(3): 1052 - 60.

[29] Amirthagadeswaran KS, Arunachalam VP. "Improved Solutions for Job Shop Scheduling Problems through Genetic Algorithm with a Different Method of Schedule Deduction". *International Journal of Advance Manufacturing Technology*. 2006; 28: 532-40.

[30] Giffler J, Thompson GL. "Algorithm for Solving Production Scheduling Problems". *Oper Research*. 1960; 8: 487-503.

[31] Ge HW, Sun L, Liang YC, Qian F. "An Effective PSO and AIS-Based Hybrid Intelligent Algorithm for Job-Shop Scheduling". *Systems, Man and Cybernetics, Part A: IEEE Transactions on Systems and Human,* 2008; 38(2): 358-68.