Skipping Strategy (SS) for Initial Population of Job-Shop Scheduling Problem

M Abdolrazzagh-Nezhad¹, E B Nababan²*, and H M Sarim³

¹Department of Computer Engineering, Faculty of Engineering, Bozorgmehr University of Qaenat, Qaen, Iran.
²Department of Information Technology, Faculty of Computer Science and Information Technology, Universitas Sumatera Utara, Medan, Indonesia
³Data Mining and Optimization Research Group, Center for Artificial Intelligence Technology, Universiti Kebangsaan Malaysia, 43600 UKM Bangi, Selangor, Malaysia

*Corresponding author: ernabrn@usu.ac.id

Abstract. Initial population in job-shop scheduling problem (JSSP) is an essential step to obtain near optimal solution. Techniques used to solve JSSP are computationally demanding. Skipping strategy (SS) is employed to acquire initial population after sequence of job on machine and sequence of operations (expressed in Plates-jobs and mPlates-jobs) are determined. The proposed technique is applied to benchmark datasets and the results are compared to that of other initialization techniques. It is shown that the initial population obtained from the SS approach could generate optimal solution.

1. Introduction

Researchers have been trying for many years to find effective algorithms for optimal feasible solutions in job-shop scheduling problem. Branch and bound algorithm [1], [2], were acceptable for small size instances. For other instances, meta-heuristic algorithms, such as tabu search [3], [4], [5] genetic algorithms [6], simulated annealing [7], ant colony optimization [8], particle swarm optimization [9], artificial immune system [10], DNA computing [12], and others[11], [13], [14], are mostly population-based algorithms that require an initial population to start exploring and exploiting the optimal schedule.

In this paper, a Skip Strategy (SS) is proposed. We first determine sequence of job on machine and sequence of operations followed by designing a set of rules to activate the mPlates-jobs. Then skipping strategy based on the rules generated on mPlates is employed to obtain initial population.

2. Previous Research

Several random techniques were employed to get initial population of job shop scheduling problem [5],[7],[9],[12],[15]. However it demands more computational time to get to near optimal solution. A new representation and schedule deduction method using a heuristic technique based on genetic algorithms (GA) to initialize the population was proposed in [17]. Additionally, integer linear programming [18] was also proposed, and the results were compared with some other proposed initialization procedures. However, there is still a need to consider different procedure to generate a population of initialization points to get a better solution.
3. Method
The job-shop scheduling problem formulated as follows: given set of jobs $J = \{J_1, J_2, ..., J_n\}$ on a set of machines $M = \{M_1, M_2, ..., M_m\}$. The operations of the jobs have to be processed on the machines. $O = \{O_{ij}, i = 1, 2, ..., n \land j = 1, 2, ..., m\}$. The objective function is:

$$\min \ c_{\text{max}} = \min \left( \max_{1 \leq i \leq n} (C(O_{im})) \right)$$

(1)

where $c_{\text{max}}$ maximum amount of time required for any particular task to be completed

$C(O_{im})$ is the completion time of $O_{im}$

$O_{im}$ is the last operation of $J_i$ and $i = 1, 2, ..., n$.

3.1 Pre-Processing
3.1.1 Determining Sequence of Job on Machine (SJM)
Matrix of a sequence of jobs on machines (SJM) is first determined to represent a solution space. Each row of the matrix represents jobs to be processed on a given machine and the machine has to be visited just once.

3.1.2 Constructing Active Schedule
We utilize an instance of six jobs six machines (F06 benchmark data set) to build an active schedule of a given SJM.

| Operation | Machine (Processing Time) |
|-----------|---------------------------|
| Job       | 1  | 2  | 3  | 4  | 5  | 6  |
| 1         | 3(1) | 1(3) | 2(6) | 4(7) | 6(3) | 5(6) |
| 2         | 2(8) | 3(5) | 5(10) | 6(10) | 1(10) | 4(4) |
| 3         | 3(5) | 4(4) | 6(8) | 1(9) | 2(1) | 5(7) |
| 4         | 2(5) | 1(5) | 3(5) | 4(3) | 5(8) | 6(9) |
| 5         | 3(9) | 2(3) | 5(5) | 6(4) | 1(3) | 4(1) |
| 6         | 2(3) | 4(3) | 6(9) | 1(10) | 5(4) | 3(1) |

The sequence of operations (SOJ) of the instance is:

$$SOJ = [3 \ 1 \ 2 \ 4 \ 6 \ 5 \ 1 \ 2 \ 3 \ 5 \ 6 \ 1 \ 4 \ 2 \ 1 \ 3 \ 4 \ 5 \ 6 \ 3 \ 4 \ 6 \ 1 \ 2 \ 5 \ 2 \ 1 \ 3 \ 4 \ 5 \ 6 \ 3 \ 2 \ 5 \ 6 \ 1 \ 4 \ 2 \ 4 \ 6 \ 1 \ 5 \ 3]$$

$$= [O_{11} \ 0_{12} \ O_{13} \ O_{14} \ O_{15} \ O_{16} \ O_{21} \ O_{22} \ O_{23} \ O_{24} \ O_{25} \ O_{26} \ O_{31} \ O_{32} \ O_{33} \ O_{34} \ O_{35} \ O_{36} \ O_{41} \ O_{42} \ O_{43} \ O_{44} \ O_{45} \ O_{46} \ O_{51} \ O_{52} \ O_{53} \ O_{54} \ O_{55} \ O_{56} \ O_{61} \ O_{62} \ O_{63} \ O_{64} \ O_{65} \ O_{66}]$$

(2)
3.2 Generating Initial Population

3.2.1 Plates-Jobs and mPlates-Jobs

After determining SOJ we then generate Plates-jobs and mPlates-Jobs respectively. The first column/operation \((O_{11})\) of SOJ has two machines \((M_2, M_3)\), the second one \((O_{12})\) has four machines \((M_1, M_2, M_3, M_4)\) and so on. For example \(O_{11}\) has 2 Plates-Jobs, a Plate-Jobs indexes with \(M_2\) has three jobs \((J_2, J_4, J_6)\) and a Plate-Jobs indexes with \(M_5\) with three other jobs \((J_1, J_3, J_5)\). Another example, \(O_{12}\) has four Plates-Jobs of which the first includes just one job \((J_1)\), the second consists of two other jobs \((J_2, J_3)\), the third has one job \(J_3\) and the last one contains two jobs, \(J_4\) and \(J_5\). The remaining operations/columns of SOJ, other Plates-Jobs is illustrated in Figure 1.

| Plates-Jobs related to \(O_{11}\) | \(M_3:\ J_1, J_3, J_5\) | \(M_2:\ J_2, J_4, J_6\) |
|---|---|---|
| Plates-Jobs related to \(O_{12}\) | \(M_1:\ J_1, J_4\) | \(M_5:\ J_2\) | \(M_4:\ J_3, J_6\) | \(M_2:\ J_5\) |
| Plates-Jobs related to \(O_{13}\) | \(M_2:\ J_1\) | \(M_5:\ J_2, J_5\) | \(M_6:\ J_3, J_6\) | \(M_3:\ J_4\) |
| Plates-Jobs related to \(O_{14}\) | \(M_4:\ J_1, J_4\) | \(M_6:\ J_2, J_5\) | \(M_1:\ J_3, J_6\) |
| Plates-Jobs related to \(O_{15}\) | \(M_6:\ J_1\) | \(M_1:\ J_2, J_5\) | \(M_2:\ J_3\) | \(M_5:\ J_4, J_6\) |
| Plates-Jobs related to \(O_{16}\) | \(M_5:\ J_1, J_3\) | \(M_4:\ J_2, J_5\) | \(M_6:\ J_4\) | \(M_3:\ J_6\) |

**Figure 1** Plates-Jobs based on SOJ

Furthermore, the Plates-Jobs of Ft06 are reorganized one-by-one, based on the numbers of the machines, to create mPlates-Jobs. For example, \(M_4\) has three Plates-Jobs related to \(O_{12} (J_1\) and \(J_4)\), \(O_{14} (J_3\) and \(J_6)\) and \(O_{15} (J_2\) and \(J_5)\) respectively. These operations are belongs to the same machine \(M_4\). The same process will be repeated to the remaining machines \(M_2\) to \(M_6\). The mPlates-Jobs is illustrated in Figure 2.

| mPlates-Jobs related to \(M_1\) | \(O_{12}: J_1, J_4\) | \(O_{14}: J_3, J_6\) | \(O_{15}: J_2, J_5\) |
|---|---|---|---|
| available processing orders | | | |
| mPlates-Jobs related to \(M_2\) | \(O_{11}: J_1, J_4, J_6\) | \(O_{12}: J_5\) | \(O_{14}: J_1\) | \(O_{15}: J_3\) |
| available processing orders | | | |
| mPlates-Jobs related to \(M_3\) | \(O_{11}: J_1, J_3, J_5\) | \(O_{12}: J_2\) | \(O_{14}: J_4\) | \(O_{16}: J_6\) |
| available processing orders | | | |
| mPlates-Jobs related to \(M_4\) | \(O_{12}: J_3, J_6\) | \(O_{14}: J_1, J_4\) | \(O_{16}: J_2, J_5\) |
| available processing orders | | | |
| mPlates-Jobs related to \(M_5\) | \(O_{13}: J_2, J_5\) | \(O_{15}: J_4, J_5\) | \(O_{16}: J_1, J_3\) |
| available processing orders | | | |
| mPlates-Jobs related to \(M_6\) | \(O_{13}: J_3, J_6\) | \(O_{14}: J_2, J_5\) | \(O_{15}: J_1\) | \(O_{16}: J_4\) |
| available processing orders | | | |

**Figure 2** mPlates-Jobs obtained after reorganizing the Plates-Jobs
Individually mPlate-Jobs is split into two parts where the top part indicates the available jobs of the mPlate-Jobs and indexed operation, and the lower part shows an available process orders of the mPlatet-Job. The number of rows and its elements on the lower part of mPlates-Jobs are similar to the number of rows and its elements on SJM. The empty mPlates-Jobs will be filled up with the available jobs on each mPlate as shown in Figure 3.

| mPlates-Jobs related to $M_1$ $\rightarrow$ | $O_{12}$: $J_1$, $J_3$ | $O_{15}$: $J_6$, $J_5$ | $O_{14}$: $J_2$, $J_4$ |
| SJM $\rightarrow$ | $J_1$ | $J_4$ | $J_3$ | $J_2$ | $J_5$ | $J_6$ |
| mPlates-Jobs related to $M_2$ $\rightarrow$ | $O_{13}$: $J_2$, $J_4$, $J_5$ | $O_{16}$: $J_5$, $J_6$ | $O_{12}$: $J_3$, $J_4$ | $O_{15}$: $J_6$, $J_5$ |
| The second row of SJM $\rightarrow$ | $J_2$ | $J_4$ | $J_3$ | $J_5$ | $J_1$ | $J_6$ |
| mPlates-Jobs related to $M_3$ $\rightarrow$ | $O_{13}$: $J_1$, $J_3$, $J_5$ | $O_{12}$: $J_2$, $J_3$ | $O_{13}$: $J_4$, $J_6$ | $O_{16}$: $J_6$, $J_5$ |
| The third row of SJM $\rightarrow$ | $J_3$ | $J_1$ | $J_5$ | $J_2$ | $J_4$ | $J_6$ |
| mPlates-Jobs related to $M_4$ $\rightarrow$ | $O_{12}$: $J_3$, $J_6$, $J_5$ | $O_{14}$: $J_1$, $J_6$ | $O_{16}$: $J_2$, $J_5$ |
| The fourth row of SJM $\rightarrow$ | $J_5$ | $J_6$ | $J_3$ | $J_1$ | $J_2$ | $J_5$ |
| mPlates-Jobs related to $M_5$ $\rightarrow$ | $O_{13}$: $J_2$, $J_5$ | $O_{16}$: $J_4$, $J_6$ | $O_{15}$: $J_1$, $J_3$ |
| The fifth row of SJM $\rightarrow$ | $J_2$ | $J_5$ | $J_4$ | $J_6$ | $J_3$ | $J_1$ |
| mPlates-Jobs related to $M_6$ $\rightarrow$ | $O_{13}$: $J_3$, $J_6$, $J_5$ | $O_{16}$: $J_2$, $J_5$, $J_3$ | $O_{15}$: $J_1$, $J_4$ |
| The sixth row of SJM $\rightarrow$ | $J_6$ | $J_3$ | $J_2$ | $J_5$ | $J_1$ | $J_4$ |

**Figure 3** mPlates-Jobs filled up with available jobs on each mPlate-Jobs

Therefore, the SJM mPlates-Jobs is:

$$
SJM = \begin{bmatrix}
M_1 = [J_1, J_3, J_6, J_3, J_2, J_5] & \begin{bmatrix} 1 & 4 & 6 & 3 & 2 & 5 \\
M_2 = [J_2, J_6, J_4, J_5, J_3, J_2] & 2 & 6 & 4 & 5 & 1 & 3 \\
M_3 = [J_3, J_1, J_5, J_2, J_4, J_6] & 3 & 1 & 5 & 2 & 4 & 6 \\
M_4 = [J_3, J_6, J_4, J_1, J_2, J_5] & 3 & 6 & 4 & 1 & 2 & 5 \\
M_5 = [J_2, J_5, J_6, J_4, J_3, J_1] & 2 & 5 & 6 & 4 & 3 & 1 \\
M_6 = [J_6, J_3, J_2, J_5, J_1, J_4] & 6 & 3 & 2 & 5 & 1 & 4
\end{bmatrix}
\end{array}
$$

Based on (4), the makespan (completion time) of the SJM obtained is 60 while the optimal makespan for FT06 is 55. It shows the ability and efficiency of the proposed classification of mPlates-Jobs.

### 3.2.2 Generating Rules

Furthermore to continue the process in producing initial population, we create rules to conform:

1. Two consecutive operations of mPlates on a given machine can join together.
2. If SJM on a given machine do not match their own mPlates or consecutive mPlates, move these jobs towards their own mPlates.
3. Sorted the jobs on their mPlates based on the maximum of the remainder processing times (RPT).
4. If two consecutive mPlates on a given machine are joined together and one job is formed, the first mPlate becomes a candidate to move to the next mPlate, and the processing order of the candidate job on the operation, should be recalculated.

### 3.2.3 Skipping Strategy (SS)

By considering the conditions and following the generated rules we then employ a skipping strategy (SS) to obtain initial population. The brief outline of SS is as follows:
Step 1: Define mPlates, the real processing orders ($r_{order}$) of jobs related to their mPlates and their current processing orders ($c_{order}$) on $M_a/row \alpha$ of the SJM.

Step 2: Consider mPlate on $M_a$, if it satisfy (4), sort its $c_{order}$ on SJM follow the third rule of mPlates-Jobs, recalculate makespan of the exchanged primal SJM.

\[
\begin{align*}
CT(o_{ij}) + PT(o_{rs}) & \leq ST(o_{ij}+1) \\
\text{and} & \\
\left( CT(o_{rs-1}) \leq ST(o_{ij}) \right) & \text{or} \\
\left( ST(o_{ij}) + PT(o_{rs}) \leq ST(o_{rs+1}) \right)
\end{align*}
\]  

(4)

Step 3: Match the $c_{order}$s of jobs on the primal SJM with their $r_{order}$s predefined on the mPlate or consecutive mPlates (the first and second rule of mPlates-Jobs), by moving the $c_{order}$s of jobs toward their $r_{order}$s one-by-one, if condition (5) or (6) is satisfied, and then recalculate the makespan of the exchanged primal SJM.

\[
\begin{align*}
\theta & \leq CT(o_{rs+1}) \\
\text{and} & \\
\max(CT(o_{ij-1}), \theta) + PT(o_{ij}) & \leq CT(o_{ab})
\end{align*}
\]  

(5)

where $\theta = \max(CT(o_{rs-1}), CT(o_{pq})) + PT(o_{rs})$. if there is no $J_a$ then $CT(o_{ab})$ equals the makespan of the primal SJM, and if $J_p$ is not available then $CT(o_{pq})$ equals zero. Conversely, if $\rho < \tau$, the jobs with $c_{order}$s $\rho - 1$, $\rho + 1$ and $\rho + 2$ are $J_p$ with operation $O_{pq}$, $J_i$ with operation $O_{ij}$ and $J_a$ with operation $O_{ab}$, respectively. If they satisfy condition (6), then swap the $c_{order}$s of $J_p$ and $J_i$, recalculate the makespan of the exchanged primal SJM, and repeat the procedure until the current order of $J_p$ reaches $\tau$.

\[
\begin{align*}
\delta & \leq CT(o_{ij+1}) \\
\text{and} & \\
\max(CT(o_{rs-1}), \delta) + PT(o_{rs}) & \leq CT(o_{ab})
\end{align*}
\]  

(6)

where $\delta = \max(CT(o_{ij-1}), CT(o_{pq})) + PT(o_{ij})$. if there is no $J_a$ then $CT(o_{ab})$ equals the makespan of the primal SJM, and if $J_p$ is not available then $CT(o_{pq})$ equals zero.

Step 4: Any job in the mPlate next to the chosen mPlate with smaller RPT than the maximum of the RPTs on the chosen mPlate can get the order of the chosen mPlate (the fourth rule of mPlates-Jobs).

Step 5: At the same time as moving jobs from their current order to a new order (Steps 2-4).

4. Results and Discussion

We implemented our proposed method to some of benchmark data and compared the result to those of some other algorithms in term of relative percentage of error (RPE) for the completion time.

\[RPE = \frac{bf-BKS}{BKS} \times 100\]  

(7)
where \(bf\) is the best quality of initial population produced, BKS is the Best Known Solution, \(bf^{Target}\) is the best quality of generated points by the target algorithm.

Result comparison of our experiment is shown in Table 1, where the deviation from the BKS of the best point in the initial population produced by SS is on average 3.6% for Ft06, Ft10, and Ft20 datasets, 4.3% for La01–40 datasets, and so on. The negative RPE for Willem’s dataset [18] by SS means that SS has succeeded in generating a point in the initial population 2.1% better than the BKS for Willem’s dataset.

| Table 1 Result comparison with BKS |
|-----------------------------------|
| Ft  | La  | Abz | Ord | Swv | Willem | Yn  |
| RPE | 3.6% | 4.3% | 7.9% | 9.6% | 23.7% | -2.1% |
| (SS) | 18.3% |

5. Conclusion
In this paper, a new strategy (SS) to generate an initial population to near optimal solution has been proposed. The SS was designed based on \(mPlates-Jobs\), with respect to \(SOJ\). Then we create rules by considering the solution space, monitoring the processing order of jobs over machines, and checking their effects on the quality of the solutions and on gaps created in their schedules.

From the result obtained it is shown that SS has the ability to produce any size of initial population compare to those of available valid methods. The important point is the observation of the best known solution among the initial population of FT06, FT10, La04 and La15.

6. Acknowledgements
We would like to thank Prof. Dr. Runtung Sitepu, Rector of Universitas Sumatera Utara, and Lembaga Penelitian Universitas Sumatera Utara, Medan, Indonesia for supporting this research and to The Ministry of Higher Education of Malaysia under the Centre for Research and Instrumentation Management, Universiti Kebangsaan Malaysia (ERGS Code: ERGS/1/2013/ICT07/UKM /02/5). We would also like to thank Prof. Dr. Salwani Abdullah of the Data Mining and Optimization Research Group Universiti Kebangsaan Malaysia for her contributions to this research.

References
[1] Brucker P, Jurisch B and Sievers B 1994 A branch and bound algorithm for the job-shop scheduling problem Discrete Applied Mathematics 49(1) 107-127
[2] Nababan E B, Abdullah S, Hamdan A R, Zakaria M S 2008 A branch and bound algorithm in optimizing job shop scheduling problems IEEE Int. Symp. on Information Technology ITSIM’08
[3] Vilcot G and Billaut J -C 2011 A tabu search algorithm for solving a multicriteria flexible job shop scheduling problem Int. Journal of Production Research 49(23) p 6963-80
[4] Nababan E B, Sitompul O S 2011 Manipulating tabu list to handle machine breakdowns in Job shop scheduling problem American Institute of Physic (AIP) Conf. 1337(1) p 224-9.
[5] Shaheed I M, Syukor S A, Nababan E B 2016 An empirical analysis of the relationship between the initialization method performance and the convergence speed of a meta-heuristic for Fuzzy Job-Shop scheduling problems Journal of Theoretical and Applied Information Technology 93(2) pp 297-311
[6] Moon I, Lee S,and Bae H 2008 Genetic algorithms for job shop scheduling problems with alternative routings Int. Journal of Production Research 46(10) p 2695-2705
[7] Zhang R, and Wu C 2011 A simulated annealing algorithm based on block properties for the job shop scheduling problem with total weighted tardiness objective *Computers & Operations Research* 38(5) p 854-867

[8] Seo M and Kim D 2010 Ant colony optimisation with parameterised search space for the job shop scheduling problem *Int. J. of Production Research* 48(4) p 1143-54

[9] Hong-Wei G, Liang S, Yan-Chun L, and Feng Q 2008 An effective PSO and AIS-based hybrid intelligent algorithm for job-shop scheduling *IEEE Transactions on Industrial Electronics* 55(5) p 2027-58.

[10] Qiu X and Lau H 2012 An AIS-based hybrid algorithm for static job shop scheduling problem *Journal of Intelligent Manufacturing* 23(4) p 743-58.

[11] Sha D Y and Lin H H 2010 A multi-objective PSO for job shop scheduling problems *Expert Systems with Applications* 37 p 1065-70.

[12] Willems T M and Brandts L W 1995 Implementing heuristics as an optimisation criterion in neural networks for job shop scheduling *Journal of Intelligent Manufacturing* 6 p 377-387.