Development of Simulated Annealing Based Scheduling Algorithm for Two Machines Flow Shop Problem

M. Muziana

University of Malaysia Pahang
Pahang, Malaysia
{muzianamusa@gmail.com}

Kamal Z. Zamli

University of Malaysia Pahang
Pahang, Malaysia
{kamalz @gmail.com}

Abstract

Flow shop scheduling problems relates the generation of appropriate sequencing for processing of N machines in compliance with given processing sequence orders. Due to costing requirements and time delivery constraints, continuous flow of processing tasks is desired within completion time and with minimum of due date time. Viewing as optimization problem and focusing on two machines, this research aims to develop a new Simulated Annealing based scheduling algorithm, called SA2M, for flow shop problem. The developed algorithm will be compared to the existing Johnson's algorithm in term of its costs. It is expected that the developed algorithm will perform well if not a par with Johnson's algorithm.

Keyword :flowshop, Simulated Annealing, makespan

I. INTRODUCTION

In flow shop scheduling, it is generally assumed that the jobs must be processed on the machines in the same technological or machine order [1]. The commonly methods can be divided into gradient methods and heuristic methods. Simulated annealing algorithm is one of heuristic methods, which starts from a initial solution and then improves the quality of solution by searching the current solution's neighborhood constantly and finding the new solution to replace the old one. The idea of simulated annealing algorithm is firstly given out, and it was successfully applied for the combinational optimization area [2]. Because the simple and effective strategies of searching the optimal solution, simulated annealing reduces the high computational complexity of the numerical algorithm and avoids the disadvantage of the local convergence of the gradient algorithm. In the flow shop scheduling problem, *n* jobs are to be processed on *two* machines. Here, the main assumption is that a machine processes one job at a time and a job is processed on one machine at a time without preemption. For *n* jobs, the search space for total flow time minimization or makespan minimization consists of *n factorial* possible job sequences [1].

II. LITERATURE

Flow shop sequencing problem

A permutation flow shop scheduling is a production planning process consisting of a set of *n* jobs, $J=\{J_1, J_2,.. J_n\}$ to be executed in a set of two machines. In this process every job J_n will through two machine named sequence of operation. Every job sequence must be only executed on machine 1 and then machine 2. A machine cannot execute more than one operation at one time. The diagram of flow shop scheduling for SA2M is shown in Figure 1.

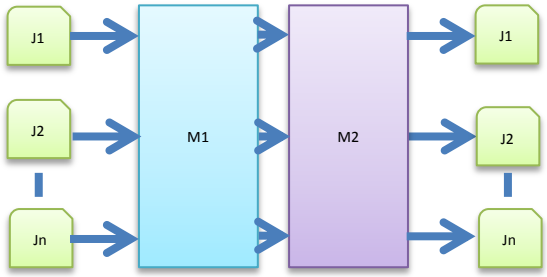


Figure 1: Flow shop diagram

Bodlaender found that parallel machine scheduling with incompatible jobs is to minimize the jobs makespan [4]. This is because two incompatible jobs cannot be processed by the same machine in a same time period and will affected the completion time. Total completion time is one of the most important performance measures, because, in practice, it can lead to stable utilization of resources, rapid turn-around of jobs, and minimization of work-in-process inventory costs [5]. So, to minimize the last completion time, called a standard flow shop scheduling problem, researcher proposed several approximation optimization algorithms and mentioned as future research other combination of well-known combinatorial optimization problem [6]. The completion time of a sequence of operation O_n denoted by C_{max}. The criterion of optimality in a flow shop sequencing problem is usually specified as minimization of make-span that is defined as the total time to ensure that all jobs are completed on all machines

as shown in Figure 2, where O_n is correspond to the processing of job J on two machines, $M = \{M1, M2\}$. If there are no release times for the jobs then the total completion time equals the total flow time. In some cases for calculating the completion times specific constraints are assumed [7].

The makespan can be calculated from the following formula:

$$C_{max} = \sum_i \delta t^i \quad (1)$$

Where δt^i is the time step by which increase the makespan iteration *i*;

$$\delta t^i = \min_{k=1,2} r_k^i \quad (2)$$

Where r_k^i is the remaining time of processing job on the machine *k* in iteration *i*.

The next remaining time of processing on machine *k* in iteration i+1 is calculated as follows:

$$r_k^{i+1} = \begin{cases} r_k^i - \delta t^i & \text{if } r_k^i > 0 \\ P_{c_k^i k} & \text{if } r_k^i = 0 \ \wedge q_k^i \neq \{0\}, \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

Where $\delta t^0 = 0$, $r_k^0 = 0$, $c_k^0 = \{0\}$, k = 1,2 and c_k^i is the job processed on machine k in iteration *i*.

The equation 4 is for the job processed on machine *k* in iteration *i*+1.

$$c_k^{i+1} = \begin{cases} c_k^i & \text{if } r_k^i > 0 \\ q_k^i(1) & \text{if } r_k^i = 0 \ \wedge q_k^i \neq \{0\}, \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Where $q_k^i(1)$ is the first element in job queue, k = 1,2.

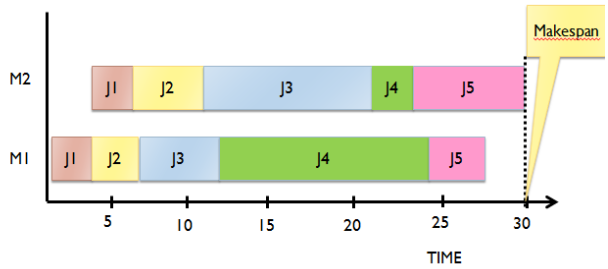


Figure 2 : Example of gantt chart for flow shop operation On sequence J1-J2-J3-J4-J5

Johnson Algorithm

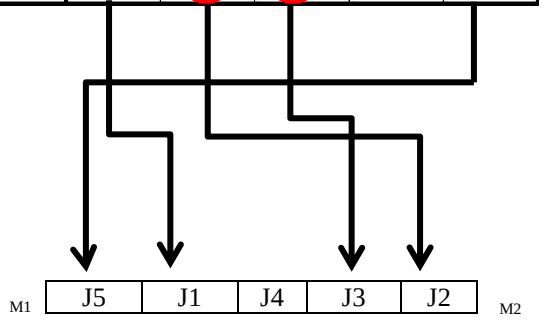
Zuzana Čičková and Stanislav Števo wrote that flow shop scheduling processing systems with two machines, where the aim is to minimize the makespan, can be solved by a Johnsons algorithm, but there is no polynomial algorithm for solving the problem for three and more machines [8][9][10]. A heuristic is used for deciding the number of blocks, Johnson's and NEH algorithm for sequencing the parts and finally Genetic Algorithm and Simulated Annealing for sizing the blocks. Four algorithms are presented by combination of this method. Three lower bounds presented and improved to evaluate the performance of algorithms [11]. The algorithm of Johnson is a classic method which solves to optimum the problem of sequencing n jobs on two machines, in a polynomial time. Assume that there are n jobs on three machines, then the problems become NP-complete (which is cannot be solved optimally in polynomial time) and the Johnson's algorithm can be applied only for some kind of cases that obey some primary conditions [12].

The Johnson's Algorithm method is done by allocates the jobs from the first and from the last position of the schedule considering them in ascending order of production times:

i. If the considered production time occurs on the first machine, allocate the job to the start of schedule (after the already scheduled ones).
ii. If the considered production time occurs on the second machine, allocate the job to the end of schedule (before the already scheduled ones).
iii. If the production time on considered job are the same on both machines, the decision is up to to the machine operator or the machine controller itself (whether to perform for machine 1 or 2).

Example:

JOB	1	2	3	4	5
M1	4	4	10	6	2
M2	5	1	4	10	3



Best sequence : J5,J1,J4,J3,J2 OR J5,J1,J3,J4,J2

Simulated Annealing

The Simulated Annealing (SA) was first introduced by Kirkpatrick in 1983, is a stochastic optimization method rooted in the principles of statistical physics. SA is a generic probabilistic met heuristic for the global optimization problem of locating a good approximation to the global optimum of a given function in a large search space. This algorithm has turned out to be a powerful tool in the field of flow shop scheduling. Researchers have been studying the application of the SA algorithm in various fields of optimization problems, but more importantly, it was shown that SA can be applied to sequencing problems [13]. Indeed, the Simulated Annealing algorithm is an exact method applied in order to obtain the optimal solution to a wide class of combinatorial optimization and scheduling problems.

The process of SA can be described as follows. First, an initial solution must be specified as a starting point. Then, repeatedly, a neighbour solution is randomly chosen from the neighbourhood of the current solution. If the newly generated solution is better than the current one, it is accepted and becomes the new current solution. Otherwise, it still has a chance to be accepted with, so called, acceptance probability. This probability is determined by the difference between objective function of the current and the neighbour solution, and depends on a control parameter, called temperature, taken from the thermodynamics.

$$P(\Delta E) = e^{\left[-\frac{\Delta E}{kT}\right]} \quad (5)$$

Where T is the temperature, ΔE is the differential of energy between current and the neighbour:

$$\Delta E = E_{neighbour} - E_{current} \quad (6)$$

and *k* is the Boltzmann constant found by:

$$k = \frac{\delta^0}{\log \frac{p^0}{T^0}} \quad (7)$$

where δ^0 is an estimated minimal difference between objective function of two solutions, p^0 is the initial value of the acceptance probability and T^0 is the initial temperature.

After a number of iterations the temperature is decreased and the process continues as described above. The annealing process is stopped either after a maximum number of iterations or when a minimum temperature is reached. The best solution that is found during the process is considered a final result [14].

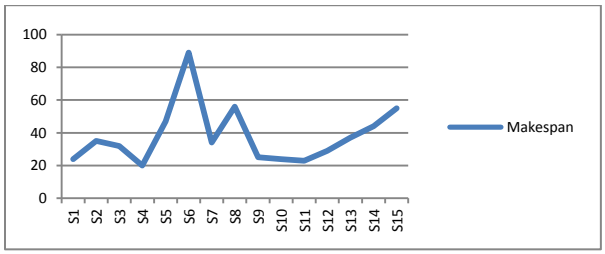


Figure 3 : The example of simulated annealing

III. METHOD

There exists much kind of methods to solve combinatorial optimization problems on flow shop scheduling depending on the complexity of the problem to solve. In this paper, we are interested on finding the minimum completion time of the last job on the last machine of the flow shop problem, using the Simulated Annealing based scheduling algorithm SA2M method. Figure 4 shown the suggested step to execute SA2M.

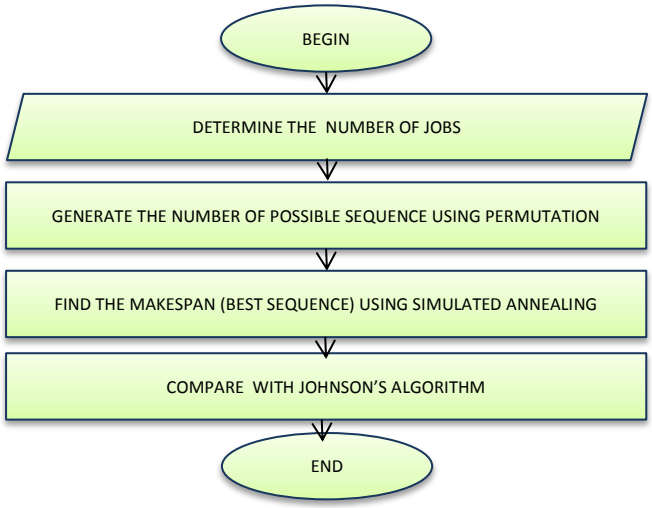


Figure 4: The steps to achieve expected result

Step1 : Determine the number of job *n* for the system and insert the time of job processed in machine *k*, t_k^i as shown in Table 1.

	M1	M2
J1	t_1^1	t_2^1
J2	t_1^2	t_2^2
J5	t_1^3	t_2^3
…	…	…
Jn	t_1^n	t_2^n

Table 1 : Time for each *n* job in machine 1 and machine 2

Step2 : Randomly generate the permutation population of correspond processing of job, O_p : calculate the possible sequence by using factorial formula, p = n!. The set of possible processing job $O=\{O_1, O_2, O_3, \ldots, O_p\}$.

- Example: O_1 ={J1-J2-J5-…Jn}.

Step 3: Assign the time of processing job, r_k^i by mapping the sequence of O, that generate in step 3 and t_k^i .

- Example:

		M1	M2
1	J1	$r_1^1 = t_1^1$	$r_2^1 = t_2^1$
2	J2	$r_1^2 = t_1^2$	$r_2^2 = t_2^2$
3	J5	$r_1^3 = t_1^5$	$r_2^3 = t_2^5$
…	…	…	…
n	Jn	$r_1^n = t_1^n$	$r_2^n = t_2^n$

Table 2 : Time for each *n* job in machine 1 and machine 2 after mapping with O_p iteration.

Step 4: Calculate makespan using equation (1).

Step 5: Find the best sequence base on simulated annealing (minimum value).

- Repeat step 2 until step 4 to get neighbor value.
- Compare the current value with neighbor value from step 4. If $\Delta E \leq 0$; then accept the neighbor solution. Else accept with probality.
- Update the cooling temperature.

Step 6: Repeat step 5 until stoping criteria satisfied, that freezing temperature.

Step 7: Compare the result with Johnson's algorithm.

IV. RESULT AND DISCUSSION

The algorithm is coded in C++ Builder. The parameters used in SA2M algorithm are setting as Table 3.

Number of Job, n		5
Initial Temperature		300
Temperature cooling rate		0.9
Stop/freezing Temperature		2
Time processed, t_k^i		
	M1	M2
J1	2	4
J2	6	10
J3	5	4
J4	6	12
J5	9	9

Table 3 : Parameter used in the algorithm

The processing of searching results for the SA2M is shown in Table 4.

T	SCurrent	CT_c	SNeighbour	CT_n	Seq_best
300	J2-J4-J5-J1-J3	45	J3-J5-J1-J4-J2	44	J3-J5-J1-J4-J2
285	J3-J5-J1-J4-J2	44	J5-J2-J4-J3-J1	48	J3-J5-J1-J4-J2
270	J3-J5-J1-J4-J2	44	J4-J2-J5-J3-J1	45	J4-J2-J5-J3-J1
257	J4-J2-J5-J3-J1	45	J1-J2-J4-J5-J3	43	J1-J2-J4-J5-J3
244	J1-J2-J4-J5-J3	43	J1-J2-J4-J5-J3	43	J1-J2-J4-J5-J3
…	…	…	…	…	…
1.97	J1-J2-J4-J5-J3	43	J5-J4-J3-J2-J1	48	J1-J2-J4-J5-J3

Table 4 : Test case for simulated annealing

Where

T = temperature
SCurrent = Current Sequence
SNeighbour = Neighbour Sequence
Seq_best = Best Sequence
CT_c = completion time for current sequence
CT_n = completion time for neighbour sequence

By using simulated annealing, we are using the completion time value of SCurrent and SNeighbour to get the minimum makespan. If the difference of completion time is less than or equal than 0, we accept the SNeighbour as a best sequence. In some condition, it has to decide whether or not to accept a SNeighbour, that will be using the acceptance probability to search new iteration. Table 4 shown that the minimum makespan is 43 for sequence J1-J2-J4-J5-J3.

Simulated Annealing	Johnson
J1-J2-J4-J5-J3	J1-J2-J4-J5-J3

Table 5: The comparison answer from Simulated Annealing and Johnson

The expected result is shown as Table 5 as a proof that Simulated Annealing can find the solution as Johnson's algorithm.

V. CONCLUSION

Summing up, this paper has discussed the Simulated Annealing based algorithm for flow shop problem. Our contention is that Simulated Annealing can help to find optimal solutions for the flow shop scheduling problem

REFERENCES

1. Uday K. Chakraborty, 2009, Minimizing Total Flow Time in Permutation Flow Shop Scheduling with Improved Simulated Annealing. . IEEE Congress on Evolutionary Computation 2009: pp 158-165

2. S. Kirkpatrick, C. D. Gellat and M. P. Vecchi(1983), "Optimization by Simulated Annealing", Science, vol. 220, pp. 671–680.

3. Ivan Lazar (2012), Approaches to Solving of Permutation Flow Shop Scheduling Problem : An Exploration Study.317, 2003

4. Bodlaender HL, Jansen K, Woeginger GJ (1994) Scheduling with incompatible jobs. Disc Appl Math, 55:219–232

5. J. M. Framinan and R. Leisten,(2003) An efficient Constructive Heuristic for Flowtime Minimisation in Permutation Flow Shops, *OMEGA, International Journal of Management Science*, vol.31, no.4, pp.311-317,

6. Wang Z, HongW, HeD(2013) Combination of Parallel Machine Scheduling and Covering problem.Working paper, Tsinghua University

7. Vladimír Modrák, R. Sudhakara Pandian (2010), Flow Shop Scheduling Algorithm to Minimize Completion Time For n- jobs m –machines Problem

8. Zuzana Čičková, Stanislav Števo (2010), Flow Shop Scheduling using Differential Evolution. Vol. 5 , No. 2, pp. 008-013

9. Brezina, I., Čičková, Z., Gežik, P., & Pekár, J. (2009). Modelovanie reverznej logistiky –optimalizácia procesov recyklácie a likvidácie odpadu. Bratislava: Ekonóm.

10. Paluch, S., & Peško, Š. (2006). Kvantitatívne metódy v logistike. Žilina: EDIS vydavateľstvo ŽU. Technical Gazette 17, pp 273-278.

11. Parviz Fattahi, Seyed Mohammad Hassan Hosseini and Fariborz Jolai (2013). Some heuristics for the hybrid Flow Shop Scheduling Problem with Setup and Assembly Operations. International Journal of Industrial Engineering Computations. Vol 4 issue 3 pp. 393-416 .

12. Mircea ANCĂU (2012). On Solving Flowshop Scheduling Problems, Proceeding of the Romanian Academy, *Series A, Volume 13, Number 1/2012, pp. 71–79*

13. C. Koulamas, S. R. Antony, and R. Jaen (1994), "A Survey of Simulated Annealing Applications to operations research problems", Omega, vol. 22, no. 1, pp. 41–56.

14. Jarosław Hurkała and Adam Hurkała (2012), Effective Design of the Simulated Annealing Algorithm for the Flowshop Problem with Minimum Makespan Criterion. Journal of Telecommunications and Information Technology (JTIT). Vol 2/2012 pp 92-98