Scheduling Single Machine Problem to Minimize Completion Time

Yasothei Suppiah, Thangavel Bhuvaneswari, Pang Shen Yee, Ng Wei Yue, Chan Mun Horng

Faculty of Engineering & Technology, Multimedia University, Melaka, Malaysia

Abstract **–In this research, a single batch machine scheduling problem with sequence dependent setup time is studied with the aim of minimizing the completion time. This problem has been proven to be NP-hard and therefore, a tabu search algorithm is developed to solve the single batch machine scheduling problem. Furthermore, a genetic algorithm and several dispatching heuristics are also developed to study the capabilities of all the algorithms in providing the average completion time. The computational result reveals that the developed tabu search algorithm is capable of producing a better outcome compared to the other algorithms.**

Keywords **– scheduling, tabu search, genetic algorithm, batch, completion time.**

1. Introduction

In most manufacturing industries, production planning and scheduling are important for decision making process which is crucial for the survival of the industries. Complex production systems in industries are usually decomposed into a series of single machine problem which is practically relevant [6].

DOI: 10.18421/TEM112-08
https://doi.org/10.18421/TEM112-08

Corresponding author: Yasothei Suppiah,
Faculty of Engineering & Technology, Multimedia University, Melaka, Malaysia.
Email: yasothei.suppiah@mmu.edu.my

Received: 07 February 2022.
Revised: 27 March 2022.
Accepted: 01 April 2022.
Published: 27 May 2022.

Besides that, the single machine scheduling problem is also theoretically relevant since its solution methods provide fundamentals for inspecting scheduling problems in other production environments, such as parallel machines and flow shops. Batch scheduling is one of the popular production environment configurations that is applicable in most industries such as CNC machine in metal fabrication industry, semiconductor industry, burner in steel working and food industries [2]. A batch machine can simultaneously process multiple jobs into batches that has a capacity limit and can only process few jobs once during same time. Batch setup time is used only when there is a change from processing a job in one batch to a job in another batch [8]. The single machine problem with the objective to minimize completion time with consideration of sequence dependent setup times is known to be NP-hard [10].

Although exact methods such as mixed integer linear programming models are known to represent many real-life scheduling problems, it is not possible to solve industrial size scheduling problems within a reasonable polynomial time despite the recent advancement of computer technology and commercial solvers [1]. Therefore, heuristics based algorithms has been the trend for solution methods these days for scheduling problems in the scheduling literature. A dispatching heuristic is a rule that derives a priority index for a job based on its attributes, and then simply schedules the job with the highest priority [4]. It is popular in most industries due to its simple nature which are easily understood and provides quick solution to the scheduling problem. Other common solution methods in scheduling literature are metaheuristics which consists of more challenging and complex intrinsic design which is able to provide better outcomes compared to the dispatching heuristics.

Tabu search, a metaheuristic which is first developed in [5], is broadly applied to tackle scheduling problem. Tabu search starts with an initial population to generate neighbourhood schedule at each iteration until a stopping criteria is met. It has special characteristics which is tabu list that enables the solutions to escape being trapped inside a local optima value. The tabu search which is initiated with the dispatching heuristic modified Apparent Tardiness Cost with Setup rules tends to provide a good solution for their single machine scheduling problem [13]. More success of tabu search combined with other heuristics can be found in [3], [12] and [9]. Another prominent metaheuristic is genetic algorithm which is based on evolutionary process [7]. New neighbourhood of schedule is generated in every iteration based on biological process, mutation and crossover. This approach has been found to quickly generate good solutions for a wide variety of scheduling problems by many researchers [11].

This paper deals with a single machine scheduling problem with the objective of minimizing completion time. The scheduling problem can be described as n independent jobs which is grouped into batches. Every job has its processing time (pj). The maximum number of jobs in each batch does not exceed the capacity of batch, B. The processing time of each batch is equal to the maximum processing time of jobs assigned to batch b. Setup time (Stj) is needed when there is a change from processing a job in one batch to a job in another batch. Only one batch can be processed at a time in the machine. A tabu search, genetic algorithm and two dispatch heuristics are developed as a solution methodology which is presented in the next section. This is followed by the experimental results and finally the conclusion.

2. Solution Methodology

Dispatch Heuristics

Two types of dispatching heuristic are developed: SPT (Shortest first job) and LPT (Longest processing time).

- SPT: Jobs are given as ascending order of processing time.
- LPT: Jobs are given as descending order of processing time.

Tabu Search

Tabu search extracts the solutions from the dispatch heuristics stated above as an input for its initial solution and its objective value which is the total completion time is recorded for every schedule. The tabu list size and the number of iterations are determined at first. Then at each iteration, the swap moves were done to generate neighbourhood and all the schedules are checked for the least completion time values and the swap moves that contributed to the best schedule added to the tabu list. Check if the solution is tabu, if not, best solution found so far was updated and included in the tabu list. If the population of entries in the tabu list is greater than tabu list size, then the oldest entry from the list was deleted. If it is tabu, aspiration criterion was applied. If the aspiration criterion is not met, continue iteration; otherwise, the best solution and tabu list was updated. The algorithm terminates once it reaches maximum iteration. The best solution with order of jobs together with the completion time is recorded. Figure 1 presents the flowchart of tabu search.

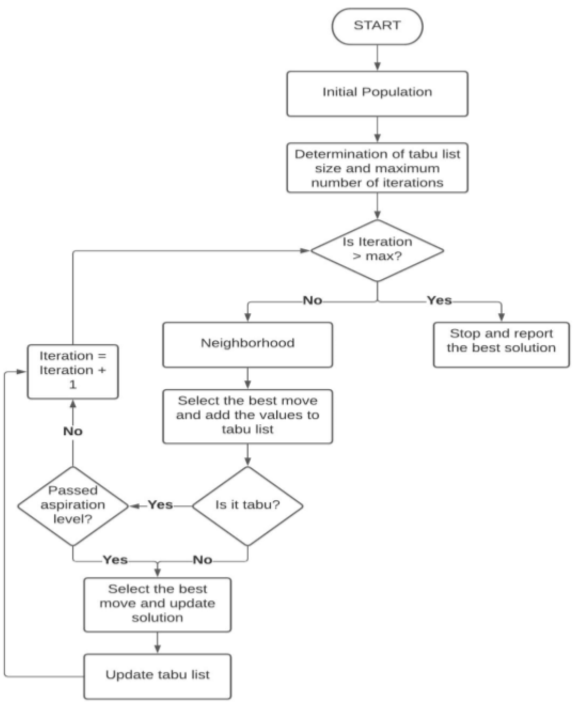
START
Initial Population
Determination of tabu list size and maximum number of iterations
Is Iteration > max?
No
Yes
Neighborhood
Stop and report the best solution
Iteration = Iteration + 1
Select the best move and add the values to tabu list
No
Passed aspiration level?
Yes
Is it tabu?
Yes
No
Select the best move and update solution
Update tabu list

Figure 1. Flowchart of tabu search

Genetic Algorithm

Similar to tabu search, the genetic algorithm begins with initial population consisting of random schedules and schedules based on SPT and LPT dispatching heuristics as described above. The fitness value which is the completion time of every schedule is recorded. The population size and maximum iterations are set in the beginning. Then, the neighbourhood schedules are generated by applying the crossover and mutation process. At each iteration, two finest chromosomes (schedules) based on their objective value is selected as parental chromosomes. In the single point crossover method, the parental segment is swapped to perform two new children. On the other hand, the mutation process selects the parental chromosome and swaps the jobs randomly to create a diversity in the schedule. The schedule that performs the worst in terms of their completion time, inclusive of the schedules in population, will be discarded and the population pool is updated on new schedules. The algorithm ends once it reaches maximum number of iterations. Figure 2 presents the flowchart of genetic algorithm.

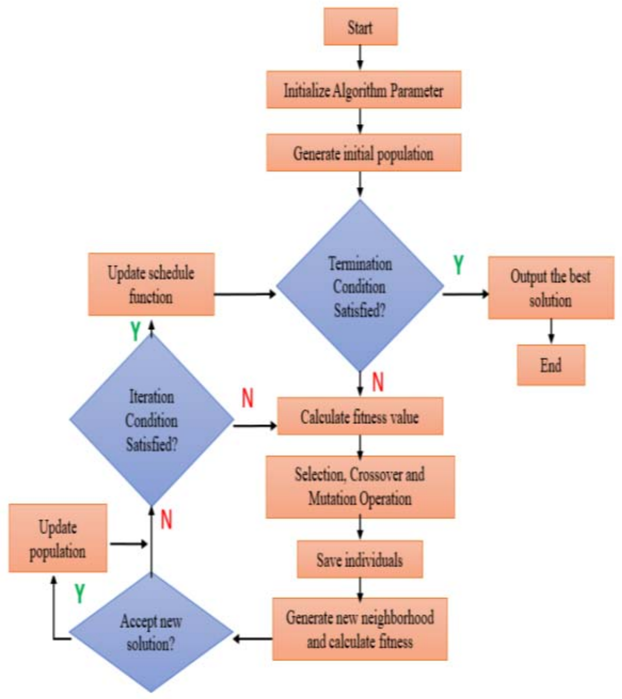
Start
Initialize Algorithm Parameter
Generate initial population
Termination Condition Satisfied?
Y
Output the best solution
End
N
Update schedule function
Y
Iteration Condition Satisfied?
N
Calculate fitness value
Selection, Crossover and Mutation Operation
Save individuals
N
Update population
Y
Accept new solution?
Generate new neighborhood and calculate fitness

Figure 2. Flowchart of genetic algorithm

3. Experimental Results

All the heuristics programming codes and data simulation are written and run by using Spyder Python (3.9.2).

The experimental data used for simulation are given in Table 1.

Table 1. Data for the experimental set

Parameter Values	
Number of Jobs in Each Batch	30, 40, 50, 60
Job Processing Time's	[10, 100]
Setup Time	[10, 100]
Number of Batches	4
Maximum Number of Runs	5
Maximum Number of Population for genetic algorithm	50

Table 2. Summary results for 30 jobs per batch

Heuristic	Average completion time	Average CPU time(s)
SPT	1289.5	0.5426
LPT	1351.4	0.4986
TS(1000)	391.4	10.9496
TS(2000)	328.5	22.4292
TS(3000)	299.1	32.2963
TS(4000)	281.8	39.4719
TS(5000)	240.6	48.5612
GA(1000)	376.8	29.1488
GA(2000)	342.1	60.0697
GA(3000)	321.4	95.7992
GA(4000)	305.3	123.5689
GA(5000)	279.5	152.3341

In Table 2, the first column provides the heuristics, whereby the number of iterations for both tabu search (TS) and genetic algorithm (GA) is written in the bracket. For example, TS (1000) represents tabu search with maximum iterations 1000, and GA (1000) represents genetic algorithm with maximum 1000 iterations. The second column gives the average completion time of each of the TS and GA heuristics from 5 runs. Having said that, both the SPT and LPT are only generated once since it provides the same value for completion time for 5 runs. The third column presents the time taken to run each of the heuristics.

Based on Table 2, it is found that TS (5000) provides the least completion time which is the best performing heuristic. When the number of iterations increases for both TS and GA, the value of the completion time improves (decreases). The time taken to complete the heuristics also increases when the number of iterations increases for both TS and GA, however, the CPU time increases drastically for GA compared to the TS. Although both the SPT and LPT performs the worst in terms of the objective function value but the computational time to find the solution are minimal since it takes less than a second to find the solution due to its simplicity features. The results of all the computational experiments are presented in Table 3, Table 4, Table 5 and Table 6 for the percentage of improvement for 30, 40, 50 and 60 jobs respectively of both tabu search and genetic algorithm from the dispatch heuristics.

Table 3 provides the percentage of improvement of TS (5000) compared to the rest of the heuristics. TS (5000) provides the percentage of improvement in the range of 14%-82%. The least improvement was seen with comparison with GA (5000).

Table 3. Percentage of improvement of TS (5000) for 30 jobs per batch

Heuristics	% of improvement of TS(5000)
SPT	81.34
LPT	82.20
TS(1000)	38.53
TS(2000)	26.76
TS(3000)	19.56
TS(4000)	14.62
GA(1000)	36.15
GA(2000)	29.67
GA(3000)	25.14
GA(4000)	21.19
GA(5000)	13.92

TS (5000) continues to perform the best heuristic among the other heuristics for the cases of 40, 50 and 60 jobs per batch. Tables 4-6 presents the percentage of improvement of TS (5000) compared to other heuristics for the cases of 40, 50 and 60 jobs.

Table 4. Percentage of improvement of TS (5000) for 40 jobs per batch

Heuristics	% of improvement of TS(5000)
SPT	69.57
LPT	70.96
TS(1000)	24.89
TS(2000)	21.63
TS(3000)	12.90
TS(4000)	6.19
GA(1000)	25.30
GA(2000)	20.02
GA(3000)	12.70
GA(4000)	11.12
GA(5000)	9.77

Table 5. Percentage of improvement of TS (5000) for 50 jobs per batch

Heuristics	% of improvement of TS(5000)
SPT	63.83
LPT	65.49
TS(1000)	22.60
TS(2000)	12.23
TS(3000)	8.69
TS(4000)	4.25
GA(1000)	20.81
GA(2000)	16.22
GA(3000)	10.64
GA(4000)	7.01
GA(5000)	2.39

Table 6. Percentage of improvement of TS (5000) for 60 jobs per batch

Heuristics	% of improvement of TS(5000)
SPT	57.24
LPT	59.2
TS(1000)	33.53
TS(2000)	21.27
TS(3000)	19.05
TS(4000)	11.34
GA(1000)	34.21
GA(2000)	25.77
GA(3000)	23.50
GA(4000)	20.93
GA(5000)	9.68

TS (5000) provides a percentage of improvement of a range of 9% -71%, 2%-66% and 9%-60% for 40, 50 and 60 jobs per batch respectively. Having said that, GA (5000) presents a fairly close results as TS (5000) as the percentage of improvement of TS (5000) is in the range of 2% - 14%. However, it is found that GA takes a rather longer computational time to find the solution compared to TS.

4. Conclusion

In this paper, a single batch machine scheduling problem to minimize the completion time has been studied. Tabu search, genetic algorithm, and two dispatch heuristic based on SPT and LPT have been developed as a solution methodology to the scheduling problem. Tabu search has outperformed the performances of the other heuristics in terms of the objective function. Another contribution is all the heuristics programming language and computational experiments were written and run using the Python software which was scarcely used in any scheduling literature. Future direction can be in considering other scheduling environments such as parallel machine, job shops, flow shops with consideration of more realistic features such as release dates.

Acknowledgements

This research work is supported by the Project IR Fund (MMUI/210022) supported by Multimedia University.

References

[1]. Altunc, A. B. C., & Keha, A. B. (2009). Interval-indexed formulation based heuristics for single machine total weighted tardiness problem. *Computers & operations research*, *36*(6), 2122-2131.

[2]. Beldar, P., & Costa, A. (2018). Single machine batch processing problem with release dates to minimize total completion time. *International Journal of Industrial Engineering Computations*, *9*(3), 331-348.

[3]. Bilge, Ü., Kurtulan, M., & Kıraç, F. (2007). A tabu search algorithm for the single machine total weighted tardiness problem. *European Journal of Operational Research*, *176*(3), 1423-1435.

[4]. Branke, J., Hildebrandt, T., & Scholz-Reiter, B. (2015). Hyper-heuristic evolution of dispatching rules: A comparison of rule representations. *Evolutionary computation*, *23*(2), 249-277.

[5]. Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers & operations research*, *13*(5), 533-549.

[6]. Homayouni, S. M., & Fontes, D. B. (2020, December). Optimization of Sustainable Single-Machine Scheduling Problem: Short Research Paper, CSCI-ISCI. In *2020 International Conference on Computational Science and Computational Intelligence (CSCI)* (pp. 1517-1520). IEEE.

[7]. Holland, J. H. (1975). Adaptation in natural and artificial systems. an introductory analysis with applications to biology, control and artificial intelligence. *Ann Arbor: University of Michigan Press*.

[8]. Monma, C. L., & Potts, C. N. (1989). On the complexity of scheduling with batch setup times. *Operations research*, *37*(5), 798-804.

[9]. Pei, J., Liu, X., Liao, B., Pardalos, P. M., & Kong, M. (2018). Single-machine scheduling with learning effect and resource-dependent processing times in the serial-batching production. *Applied Mathematical Modelling*, *58*, 245-253.

[10]. Pinedo, M. L. (2012). *Scheduling* (Vol. 29). New York: Springer.

[11]. Schaller, J. E. (2014). Minimizing total tardiness for scheduling identical parallel machines with family setups. *Computers & Industrial Engineering*, *72*, 274-281.

[12]. Shen, L. (2014). A tabu search algorithm for the job shop problem with sequence dependent setup times. *Computers & Industrial Engineering*, *78*, 95-106.

[13]. Shin, H. J., Kim, C. O., & Kim, S. S. (2002). A tabu search algorithm for single machine scheduling with release times, due dates, and sequence-dependent set-up times. *The International Journal of Advanced Manufacturing Technology*, *19*(11), 859-866.