Dual-objective Job Shop Scheduling Problem with Skilled Workers

C. Sem[1], R. Sirovetnukul[1]
[1]Department of Industrial Engineering, Faculty of Engineering, Mahidol University, Nakhon Pathom, Thailand 73170
(semchantha814@gmail.com, ronnachai.sir@mahidol.ac.th)

2021 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM) | 978-1-6654-3771-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/IEEM50564.2021.9672839

Abstract - **The Job Shop Scheduling Problem (JSSP) is work allocation on workstations to produce items that have a goal to optimize a few objectives. This research aims to develop the JSSP's model that deals with machines and job sequences by dual objective functions under the new gainful constraints involving types of skilled workers according to the case study of a steel mill. The proposed methodology of this research is applied with the Memetic algorithm (Genetic algorithm and Local Search technique) and the Pareto optimization. The result generated from the proposed idea can help the manager to decide on the assignment of the right worker to operate the right machine with the right job to achieve the objective values of minimum makespan and maximum average utilization of workers.**

Keywords - **Job shop scheduling problem, Skilled workers, Makespan, Average utilization of workers, Memetic algorithm**

I. INTRODUCTION

Most of the industries and manufacturers are trying to take the challenges of increasing customer's satisfaction, improving utilization of resources, and cutting down the operation cost. Among tasks in production management, job shop scheduling is one of the most applicable methods that help manufacturers to achieve those above goals. In this research, the proposed model will work on the problem coming up with dual objectives by using the hybrid algorithm (Memetic algorithm).

The rest of this paper is structured as follows. Section II briefly reviews the related literature. The research methodology is described in Section III. Then, numerical experiment and results are detailed in Section IV and the conclusion of the research is given in Section V.

II. LITERATURE REVIEW

A. Overview of Job Shop Scheduling Problem

The Job Shop Scheduling Problem (JSSP) is one of the well-known problems in terms of production management and combinatorial optimization problem. If there are n-jobs and m-machines, the number of possible solutions will be equal to $m \times (n!)$ [1]. JSSP needs to describe an operation with a triplet (i, k, j), job, operation, machine, respectively [2]. The important factors and key performance indicators (KPIs) in JSSP are a number of machines and jobs, processing time, machine sequence, skilled worker' s types, makespan and utilization of resource [3]. For the most convenient way, most of the studies used a Gantt chart to visually represent the solution of JSSP.

B. Solution Approaches to JSSP

JSSP is viewed as an NP-complete problem and needs to be solved by a group of algorithm or heuristic methods [4] that included searching techniques such as Genetic Algorithm (GA) and Tabu search. The memetic algorithm is the development of the classical genetic algorithm on local search features which take the advanced work of local search technique (LS) to cut the search space and had been proved as a high efficient algorithm in solving JSSP [5]. Pareto optimization (Multi-objective optimization or Pareto frontier) is the one type of optimization method that effort to obtain the solution of problem that has two or more objectives [6]. In this study, the Memetic algorithm, consisting of Local Search named Hill-Climbing, is selected to be the main tool for solving the proposed problem by coding in Python. First, the code of the single objective of makespan is tested to validate with another scheduling software (Lekin Optimization) before taking to solve two objectives of this study. On the other hand, the Pareto frontier is used to find the nondominated solution of dual-objective functions (makespan and average utilization of workers) of each group dataset in the proposed problem.

C. Early Studies for Job Shop

Meeran and Morshed used a hybrid genetic algorithm to minimize makespan [7]. Dai *et al.* studied JSSP to find optimization of multiple objectives for the energy-efficient problem by using a genetic algorithm [8]. Kassu *et al.* used the shifting bottleneck algorithm to minimize makespan [9]. Wang *et al.* use the search economic algorithm as a high performance search algorithm for solving JSSP [10]. Zhen *et al.* carried out JSSP in Make-to-stock (time-based criteria) and Make-to-order (Due date-based criteria) industries by fixed dimension particle swarm optimization [11]. Beemsterboer *et al.* noted that several job shops in practice produce some standard products to stock inventory with low demand [12]. Based on the above literature review, the following work deals with time-based criteria. To the best of our knowledge, the JSSP of Make-to-stock policy has never been solved by the Memetic algorithm with the constraints involving types of skilled workers. The work proposed in this paper is tried to fulfil the knowledge gap.

III. MATHEMATICAL MODEL OF JOB SHOP SCHEDULING WITH SKILLED WORKERS AND MEMETIC ALGORITHM

A. Assumptions

Based on [13], there are some essential assumptions which are used in this research as follows:

- All jobs and machines are available to process at time 0.
- The proposed problem is considered in terms of sequence-independent setup time.
- No preemption job is allowed and no delay schedule.

In addition, main assumptions have been made the problem more practical.

- The utilization of workers is similar to the utilization of the machine because of machine and worker working at the same time.
- The precedence constraint of the same job is defined between any pair of operations only and each machine can process with only one operation and one worker.
- The levels (types) of skilled workers are divided into two levels such as high level (H or h) and low level (L or l).
- The information of each scenario such as the processing time of each operation by types of workers is known. The availability of each type of worker in accordance with jobs and machines is predefined and there are enough workers (both levels) in all cases.

B. Mathematical Modeling

In the proposed job shop scheduling problem, there is a set of m-jobs $M = \{J1, J2, J3, ..., Jm\}$ that operate on a set of n-machine $N = \{M1, M2, ..., Mn\}$. Each job has to operate with a determined machine sequence that corresponds to a set of k-operations by the given processing time (p_{ikj}). Each machine can only operate one job at any time completely before operating with the next job. In the system, there are two types of skilled workers (High skill and low skill) who operate the job on the machine by the given processing time (t_{ikjw}). The objective is to determine the starting time (S_{ikj}) of each operation and how to assign the worker (h or l) to the operation can complete the last job at the minimum completion time with maximum average utilization of workers. To present the problem in this research, there are some notations and descriptions as follows:

Indices

i	Index of jobs, $i = 1, 2, 3, \ldots, m$
j	Index of machines, $j = 1, 2, 3, \ldots, n$
k	Index of operation orders of job i on machine j, $k = 1, 2, 3, \ldots, k$
w	Index of skilled worker's types, $w = 1,2$, where "1" represents the high-skilled worker assigned to the operation and "2" represents the low-skilled worker assigned to the operation.

Sets

M	Set of jobs, $M = \{J1, J2, J3, \ldots, Jm\}$
N	Set of machines, $N = \{M1, M2, M3, \ldots, Mn\}$
O	Set of operations of job i that operates on machine j, *where* $O = \{O_{1,1,1}, O_{1,2,2}, O_{1,3,3}, \ldots, O_{m,k,n}\}$. The notation of each operation can be represented by $O_{m,k,n}$, i.e., job m operates on machine n in operation k
W	Set of skilled worker's types, $W = \{1,2\}$

Parameters

p_{ikj}	Processing time of job i on machine j in operation k
t_{ikjw}	Processing time that depends on the type of skilled workers
S_{ikj}	Starting time of job i on machine j in operation k
C_{ikj}	Completion time of job i on machine j in operation k, $C_{ikj} = S_{ikj} + p_{ikj}$
L	Large number to ensure the correctness of constraints

Key performance indicators

$C_{\max}$	Makespan
U_j	Utilization of workers in machine j
$\bar{U}$	Average utilization of workers

Decision variable

x_{ilkj} Decision variable when job i processes before job l on machine j in operation k. $x_{ilkj} = 1$, if job i processes before job l on machine j in operation k. Otherwise, $x_{ilkj} = 0$ [14].

In this research, the processing time is determined by the dataset of the scenario that will be chosen to solve the problem. The processing time is defined in (1).

$$\boldsymbol{p_{ikj} = t_{ikjw}}\ ; \forall i \in M, \forall j \in N, \forall k \in O, \forall w \in W \tag{1}$$

- The makespan is defined in (2).

$$\boldsymbol{C_{\max} = max\ (C_{ikj})}\ ; \forall i \in M, \forall j \in N, \forall k \in O \tag{2}$$

- The utilization of workers is defined in (3).

$$U_j = \frac{1}{C_{\max}} \times \left(\sum_{k=1}^{k} \sum_{i=1}^{m} p_{ikj} \right) \ ; \forall i \in M, \forall j \in N, \forall k \in O \tag{3}$$

- The average utilization of workers is defined in (4).

$$\bar{U} = 100 \times \left(\frac{1}{n} \times \sum_{j=1}^{n} U_j \right) \ ; \forall j \in N \tag{4}$$

The objective functions are minimizing makespan in (5) and maximizing average utilization of workers in (6).

$$\boldsymbol{f_1 = min\ C_{\max}} \tag{5}$$

$$\boldsymbol{f_2 = max\ (\bar{U})} \tag{6}$$

Subject to

$$S_{lkj} \geq S_{ikj} + p_{ikj} - L \,.\, x_{iljk} \ ; \forall i, l \in M, i \neq l, \forall j \in N, \forall k \in O \tag{7}$$

$$S_{ikj} \geq S_{lkj} + p_{lkj} - L\ (1 - x_{ilkj}) \ ; \forall i, l \in M, i \neq l, \forall j \in N, \forall k \in O \tag{8}$$

$$S_{ikj} \geq S_{ik-1j} + p_{ik-1j} \ ; \forall i \in M, \forall j \in N, \forall k \in O \tag{9}$$

$$C_{\max} \geq S_{ikj} + p_{ikj} \ ; \forall j \in M, \forall i \in N, \forall k \in O \tag{10}$$

$$C_{\max} \geq S_{mkn} + p_{mkn} \ ; \forall j \in M, n \in N, \forall k \in O \tag{11}$$

$$t_{ikjw} \geq 0 \; ; \forall i \in M, \forall j \in N, \forall k \in O, \forall w \in W \quad (12)$$

$$p_{ikj} \geq 0 \; ; \forall i \in M, \forall j \in N, \forall k \in O \quad (13)$$

$$S_{ikj} \geq 0 \; ; \forall i \in M, \forall j \in N, \forall k \in O \quad (14)$$

$$C_{\max} > 0 \; ; \forall i \in M, \forall j \in N, \forall k \in O \quad (15)$$

$$x_{ilkj} = \{0, 1\} \; ; \forall i, l \in M, i \neq l, \forall j \in N, \forall k \in O \quad (16)$$

Constraints (7&8) are disjunctive constraints to ensure that each machine can process with one job at any time only. Constraint (9) is a precedence constraint to ensure that a job can start to process on the next machine after the process was completed in the previous machine. S_{ikj} is the starting time of operation k when job i process on machine j. S_{ik-1j} is the starting time of operation $k-1$ when job i process on machine j. p_{ik-1j} is the processing time of operation $k-1$ when job i process on machine j. $k-1$ is the operation that has been done before O_k for every job i. Constraints (10,11,15) ensure that the makespan is determined after all operations in the system are completed and be the positive number greater than 0. S_{mkn} is the starting time of the last operation in the system. p_{mkn} is the processing time of the last operation in the system. Constraints (12-14) ensure that the starting time of each operation and processing time of each operation must not less than 0. Constraint (16) is the binary decision variable.

C. Challenge on the Memetic Algorithm

1. Generate the skilled worker's processing time: Generate the processing time for each operation by types of skilled workers. **2. Randomize datasets for the problem:** Randomize the possible datasets that combine all two types of skilled workers in each job. The randomization technique is processed by Python code. There are two small steps in this process. First, the parameters and values are randomized separately in each job shown in Table II. Then, randomized datasets mix all jobs. For instance, h12 represents the generated processing time that is operated by the high-skilled workers for job 1 on machine 2. Table III shows the sample of a dataset represented for all whole generated datasets that mix all jobs. There are a total of 512 datasets. **3. Select the dataset for the problem:** Choose a dataset to solve in Python code based on the scenario of the problem. **4. Initial setting of parameters:** Input all values of the important parameters in Python code and run the program. **5. Generate Initial Population:** Randomly generate an initial source population of parent's chromosomes. **6. Evaluating the fitness of each chromosome:** Evaluate the fitness values which satisfy all constraints in the model. **7. Perform crossover:** Do crossover operation on every two selected chromosomes by determining crossover rate. **8. Perform mutation:** Swap the position of the pair of genes in the chromosome by determining the mutation probability.
9. Perform offspring or reproduction into population: Generate the offspring to make the child chromosome in the next generation. **10. Perform Hill-Climbing selecting technique:** Perform a local search on each offspring, evaluating the fitness of each new chromosome. After the makespan is obtained, the average utilization of workers is calculated, respectively. The next steps are done by Pareto frontier including Calculation of crowding distance and Non-dominated sorting operation by objective values. From two objective functions in the previous model, the best solution defines at any point that Cmax decreases and the average utilization of workers increases. However, there are feasible solutions greater than one and the Pareto optimal frontier illustrates the points that are not dominated by any other points. **11. Calculation of Crowding distance:** Maintain diversity by the Pareto front function on selective chromosomes. **12. Non-dominated sorting operation:** Compare every pair of selective chromosomes. After selective chromosomes have been selected by the nondominated sorting operation [15] and the process is terminated when fixed number of generation reached and solutions met minimum criteria. Otherwise, go to Step 9.

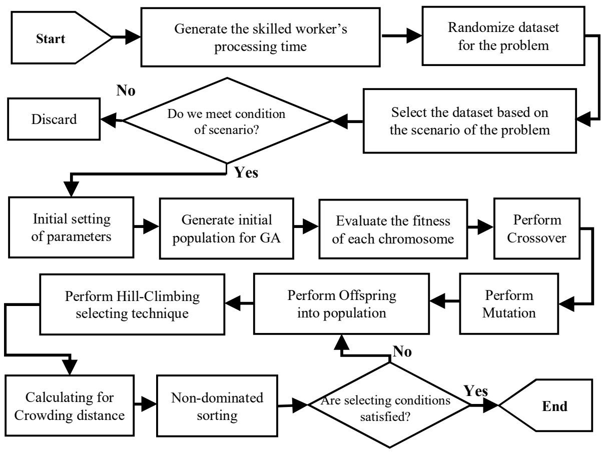
Start
Generate the skilled worker's processing time
Randomize dataset for the problem
Discard
No
Do we meet condition of scenario?
Select the dataset based on the scenario of the problem
Yes
Initial setting of parameters
Generate initial population for GA
Evaluate the fitness of each chromosome
Perform Crossover
Perform Hill-Climbing selecting technique
Perform Offspring into population
Perform Mutation
No
Calculating for Crowding distance
Non-dominated sorting
Are selecting conditions satisfied?
Yes
End

Fig. 1. Flowchart of Memetic Algorithm.

IV. NUMERICAL EXPERIMENT AND RESULTS

A. Numerical experiment

The numerical experiment is given as the illustration problem and has been solved by encoding Python code in Visual Studio. The illustration problem is considered in terms of a typical workshop of a steel mill that was mentioned in [7]. This steel mill is not a flow shop production system. There are three 3 jobs and 3 machines with 18 available workers. As the previous problem is not considered with the types of the skilled workers, the parameters and values of processing times are generated as Table I. The manager would like to assign the job to the right worker with the appropriate job sequence that can finish the last operation at the minimum total duration of the system.

Fig. 2 shows the conceptual model of the network in which 3 jobs and 3 machines are placed to the schedule. $p_{2,1,3}$ or p_{213} represents processing time of job 2 on machine 3 in the operation 1.

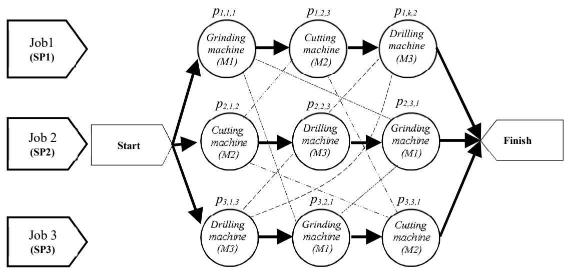
Job1 (SP1)
Job 2 (SP2)
Job 3 (SP3)
Start
Finish
$p_{1,1,1}$ Grinding machine (M1)
$p_{1,2,3}$ Cutting machine (M2)
$p_{1,k,2}$ Drilling machine (M3)
$p_{2,1,2}$ Cutting machine (M2)
$p_{2,2,3}$ Drilling machine (M3)
$p_{2,3,1}$ Grinding machine (M1)
$p_{3,1,3}$ Drilling machine (M3)
$p_{3,2,1}$ Grinding machine (M1)
$p_{3,3,1}$ Cutting machine (M2)

Fig. 2. Illustrated conceptual model of 3 jobs x 3 machines.

TABLE I
GENERATING THE PROCESSING TIMES BY TWO TYPES OF SKILLED WORKERS IN THE 3 JOBS x 3 MACHINES

Jobs	Operations	Machines	**Processing time (t_{ikjw}) Unit time in minutes** Processing time by types of skilled workers (Parameters)		Processing time by types of skilled workers (Values)	
			H	**L**	**H**	**L**
J1	$O_{1,1,1}$	1	h11	l11	12	14
	$O_{1,2,2}$	2	h12	l12	8	12
	$O_{1,3,3}$	3	h13	l13	4	14
J2	$O_{2,1,2}$	2	h22	l22	9	11
	$O_{2,2,3}$	3	h23	l23	4	13
	$O_{2,3,1}$	1	h21	l21	6	12
J3	$O_{3,1,3}$	3	h33	l33	6	10
	$O_{3,2,1}$	1	h31	l31	5	13
	$O_{3,3,2}$	2	h32	l32	8	10

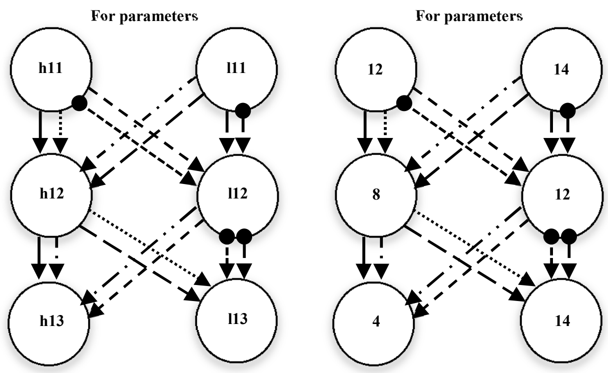

Fig. 3. The concept of randomization datasets in each job.

TABLE II
RANDOMIZED DATASETS OF JOB 1

Parameters			**Values**		
O1	**O2**	**O3**	**O1**	**O2**	**O3**
h11	h12	h13	12	8	4
h11	h12	l13	12	8	14
h11	l12	h13	12	12	4
h11	l12	l13	12	12	14
l11	h12	h13	14	8	4
l11	h12	l13	14	8	14
l11	l12	h13	14	12	4
l11	l12	l13	14	12	14

TABLE III
A SAMPLE OF DATASETS THAT MIX ALL JOBS

Parameters& Values / **Datasets**	**Parameters**			**Values**		
	O1	**O2**	**O3**	**O1**	**O2**	**O3**
Dataset_256	h11	l12	l13	12	12	14
	l22	l23	l21	11	13	12
	l33	l31	l32	10	13	10

B. Results

From the randomization process, there are 512 datasets were generated by the Python code and classified by the number of high-skilled workers (H) and the number of low-skilled workers (L) in Table IV. The exemplified result of Pareto optimization of group dataset H2L7 has been drawn in Fig.4. The star-marker in the graph represents the good solution of group dataset H2L7. The Pareto front-dataset is at Dataset_256 that has the optimal sequence [3. 1. 2. 1. 3. 2. 1. 2. 3.] at the makespan f_1 = min (Cmax) = 38 minutes with the average utilization of workers f_2 = max ($\bar{U}$) = 93.86%. The solution for the Dataset_256 is illustrated as Gantt chart in Fig.5 and interpreted in Table V. The summary result of all group datasets is shown in Table VI that provides the result of the Pareto front dataset(s) such as optimal sequence and both objective values.

TABLE IV
LIST OF GROUP DATASET BY THE NUMBER OF H AND L

No.	Names of Group datasets	Number of high-skilled workers	Number of low-skilled workers	Number of datasets in each group
1	H0L9	0	9	1
2	H1L8	1	8	9
3	H2L7	2	7	36
4	H3L6	3	6	84
5	H4L5	4	5	126
6	H5L4	5	4	126
7	H6L3	6	3	84
8	H7L2	7	2	36
9	H8L1	8	1	9
10	H9L0	9	0	1
	Total			**512**

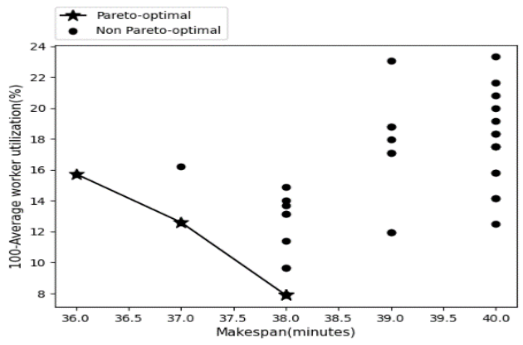

Fig. 4. A sample of Pareto front for group dataset H2L7.

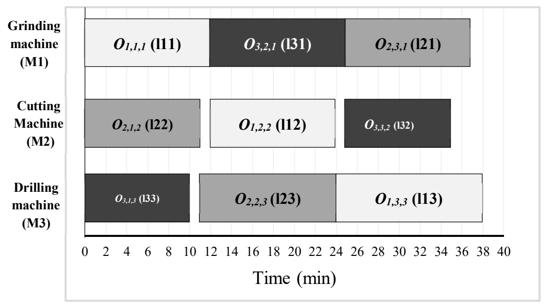

Fig. 5. Gantt chart of the dataset_256.

TABLE V
INTERPRETATION OF GANTT CHART FOR DATASET_256

Jobs	**Starting time**	**Completion time**	**Machines**	**Operations**	**Worker ID**
Job 1 **(SP1)**	0:00:00	0:00:12	Grinding machine (M1)	$O_{1,1,1}$	h11
	0:00:12	0:00:24	Cutting machine (M2)	$O_{1,2,2}$	112
	0:00:24	0:00:38	Drilling machine (M3)	$O_{1,3,3}$	113
Job 2 **(SP2)**	0:00:00	0:00:11	Cutting machine (M2)	$O_{2,1,2}$	122
	0:00:11	0:00:24	Drilling machine (M3)	$O_{2,2,3}$	123
	0:00:25	0:00:37	Grinding machine (M1)	$O_{2,3,1}$	121
Job 3 **(SP3)**	0:00:00	0:00:10	Drilling machine (M3)	$O_{3,1,3}$	133
	0:00:12	0:00:25	Grinding machine (M1)	$O_{3,2,1}$	131
	0:00:25	0:00:35	Cutting machine (M2)	$O_{3,3,2}$	132

TABLE VI
SUMMARY RESULT OF ALL PARETO FRONT DATASETS

No.	**Dataset Name**	**Group Dataset**	**Optimal Sequence**	f_1 ***(min)***	f_2 ***(%)***
1	Dataset_512	H0L9	[3. 2. 1. 3. 2. 1. 1. 3. 2.]	40	90.83
2	Dataset_256	H1L8	[3. 1. 2. 1. 3. 2. 1. 2. 3.]	38	93.86
3	Dataset_192	H2L7	[3. 2. 1. 1. 3. 2. 1. 3. 2.]	37	87.39
4	Dataset_224		[2. 3. 2. 1. 3. 1. 3. 1. 2.]	38	92.11
5	Dataset_255		[1. 3. 3. 2. 1. 3. 2. 1. 2.]	38	92.11
6	Dataset_446		[2. 3. 2. 1. 1. 3. 2. 1. 3.]	36	84.26
7	Dataset_96	H3L6	[2. 1. 3. 2. 1. 3. 3. 2. 1.]	37	90.99
8	Dataset_184		[3. 2. 1. 2. 3. 1. 3. 1. 2.]	35	86.67
9	Dataset_439		[1. 3. 2. 3. 1. 2. 2. 3. 1.]	35	86.67
10	Dataset_183	H4L5	[2. 1. 1. 3. 3. 2. 2. 1. 3.]	33	89.90
11	Dataset_310		[1. 3. 2. 1. 3. 2. 3. 1. 2.]	32	84.38
12	Dataset_54	H5L4	[1. 3. 2. 2. 1. 3. 1. 3. 2.]	30	87.78
13	Dataset_151		[1. 2. 3. 1. 3. 2. 3. 2. 1.]	33	87.88
14	Dataset_309		[2. 3. 1. 3. 1. 2. 3. 2. 1.]	30	87.78
15	Dataset_22	H6L3	[3. 2. 1. 3. 2. 1. 1. 2. 3.]	30	85.56
16	Dataset_45		[1. 2. 3. 2. 1. 3. 2. 1. 3.]	29	85.06
17	Dataset_53		[1. 2. 1. 3. 2. 1. 3. 2. 3.]	30	85.56
18	Dataset_277		[3. 1. 2. 1. 3. 2. 1. 3. 2.]	30	85.56
19	Dataset_21	H7L2	[3. 1. 2. 1. 2. 3. 2. 3. 1.]	29	86.21
20	Dataset_35		[2. 1. 3. 3. 1. 3. 2. 1. 2.]	33	72.73
21	Dataset_17	H8L1	[2. 1. 3. 1. 3. 2. 1. 2. 3.]	28	84.52
22	Dataset_1	H9L0	[3. 2. 2. 1. 3. 2. 1. 1. 3.]	28	73.81

V. CONCLUSION

This research addresses the dual-objective Job Shop Scheduling Problem (JSSP) under the Make-to-stock policy with time-based criteria, which focused on minimum makespan (Cmax) and maximum average utilization of workers. This study uses secondary data and focused on creating the new mathematical model to be solved by the Memetic algorithm (GA&LS). The main contribution is to put the new constraints (types of skilled workers) which can provide the concept for manufacturers to create an effective and reliable schedule as well as work assignment to complete a new gap on the previous studies of JSSP. The results from the proposed algorithm can output good objective values for each group dataset. This research is studied under some limitations such as no splitting job in any operating machine, no blocking, overlapping and recirculation of job, no preemption of job and no cancellation of job. However, the relaxation of them should be taken into account in further research to modify the result of medium and large problems with other parameters such as stochastic processing time in another heuristic or metaheuristics.

ACKNOWLEDGMENT

This thesis is partially supported by the Faculty of Graduate Studies and Graduate Studies of Mahidol University Alumni Association.

REFERENCES

[1] P. Kumar, P. Kumar, R. Bhool, and V. Upneja, "An efficient genetic algorithm approach for minimising the makespan of job shop scheduling problems," *Int. J. Sci. Eng. Technol. Res.,* vol. 5, no. 5, pp. 1439-1447, 2016.

[2] K. R. Baker, *Introduction to Sequencing and Scheduling*. Wiley, 1974.

[3] M. Abbas , A. Abbas, and W. A. Khan, "Scheduling job shop - A case study," in *IOP Conf. Series: Materials Science and Engineering,* vol. 146, 2016.

[4] R. Braune and G. Zäpfel, "Shifting bottleneck scheduling for total weighted tardiness minimization - A computational evaluation of subproblem and re-optimization heuristics," *Comput. Oper. Res.,* vol. 66, pp. 130-140, 2016.

[5] H. de. Haan, "Job Shop Scheduling with Metaheuristics for car workshops," Master Thesis, Artif. Intell., Faculty of Mathematics and Natural Sciences, University of Groningen, 2016.

[6] I. Giagkiozis and P. J. Fleming, "Pareto front estimation for decision making," *Evol. Comput.,* vol. 22, no. 4, pp. 651-678, 2014.

[7] S. Meeran and M. Morshed, "A hybrid genetic tabu search algorithm for solving job shop scheduling problems: a case study," *J. Intell. Manuf.,* vol. 23, no. 4, pp. 1063-1078, 2012.

[8] M. Dai, D. Tang, A. Giret, and M. A. Salido, "Multi-objective optimization for energy-efficient flexible job shop scheduling problem with transportation constraints," *Robot. Comput. Integr. Manuf.,* vol. 59, pp. 143-157, 2019.

[9] J. Kassu and B. Eshetie, "Job Shop Scheduling Problem for Machine Shop with Shifting Heuristic Bottleneck," *Global J. Res. Eng.,* vol. 15, no. 1, 2015.

[10] S.-J. Wang, C.-W. Tsai, and M.-C. Chiang, "A High Performance Search Algorithm for Job-Shop Scheduling Problem," *Procedia Comput. Sci.,* vol. 141, pp. 119-126, 2018.

[11] Z. Wang, Y. Qi, H. Cui, and J. Zhang, "A hybrid algorithm for order acceptance and scheduling problem in make-to-stock/make-to-order industries," *Comput. Ind. Eng.,* vol. 127, pp. 841-852, 2019.

[12] B. Beemsterboer, M. Land, R. Teunter, and J. Bokhorst, "Integrating make-to-order and make-to-stock in job shop control," *Int. J. Prod. Econ.,* vol. 185, pp. 1-10, 2017.

[13] M. Al-Ashhab, S. Munshi, M. Oreijah, and H. A. Ghulman, "Job Shop Scheduling Using Mixed Integer Programming," *Int. J. Mod. Eng. Res.,* vol. 7, no. 3, pp. 23-29, 2017.

[14] N. Al-Hinai and S. Piya, "Job shop scheduling for skill-dependent Make-to-order system," in *2015 Int. Conf. on Industrial Engineering and Operations Management* IEEE, *IEOM* 2015, pp. 1-5.

[15] K. Deb, "Multi-objective optimisation using evolutionary algorithms: An introduction," in *Multi-objective evolutionary optimisation for product design and manufacturing*: Springer, 2011, pp. 3-34.