EFFICIENT DISPATCHING RULES BASED ON DATA MINING FOR THE SINGLE MACHINE SCHEDULING PROBLEM

Mohamed Habib Zahmani[1], Baghdad Atmani[2] and Abdelghani Bekrar[3]

[1]Laboratory of Pure and Applied Mathematics,
University of Mostaganem, Mostaganem, Algeria
habib.zahmani@univ-mosta.dz
[2]Laboratoire d'Informatique d'Oran,
University of Oran 1, Oran, Algeria
atmani.baghdad@univ-oran.dz
[3]Laboratory of Industrial and Human Automation control,
Mechanical engineering and Computer Science,
University of Valenciennes and Hainaut-Cambresis, Valenciennes, France
abdelghani.bekrar@univ-valenciennes.fr

ABSTRACT

In manufacturing the solutions found for scheduling problems and the human expert's experience are very important. They can be transformed using Artificial Intelligence techniques into knowledge and this knowledge could be used to solve new scheduling problems. In this paper we use Decision Trees for the generation of new Dispatching Rules for a Single Machine shop solved using a Genetic Algorithm. Two heuristics are proposed to use the new Dispatching Rules and a comparative study with other Dispatching Rules from the literature is presented.

KEYWORDS

Data Mining, Decision Trees, Dispatching Rules, Single Machine, Scheduling, Genetic Algorithm

1. INTRODUCTION

In today's business environment, competition has become very fierce and the customers have become very demanding in terms of quality, cost and time. In this context of an increasingly globalized world, and in order to guarantee the survival of the enterprise it is necessary to enhance the manufacturing process. By doing so, the company secures a place in this highly-competitive environment. The manufacturing process goes hand in hand with the scheduling problem. The scheduling problem is NP-Hard due to the exponential number of solutions [1].

In this paper we focus on the Single Machine (SM) Scheduling Problem. Many methods have been proposed to solve this problem including its many variations, namely, static and dynamic, with or without perturbations and a large set of objective functions (sum of tardiness, length of schedule, etc.) [1]–[4]. Still no proposed approach can solve it for all its variations.

One of the most popular methods used to solve scheduling problem are the Dispatching Rules (DR). These techniques are widely used [5]–[7] owing to their efficiency and their ability to quickly define a priority for each job waiting on a machine queue. This reactivity increases the systems responsivity and its fault tolerance in case of dynamic or perturbation scenarios.

An evolution of Dispatching Rules introduced in literature is Data Mining. DM is used in order to create new rules using previous problems and experiments. In this way, the previously acquired knowledge is transformed to new DR to be used for new scheduling problems. Among these approaches we notice some in particular Koonce [8], Shahzad [9] or Aissani [10]. In all these works DM is used and more often Decision Trees for the extraction of Dispatching Rules to be used to schedule jobs in a new problem or reschedule in case of perturbation. In these papers authors focus mainly on multiple-machines shops such as Flow Shop or Job Shop for the generation of new DR.

In this paper Genetic Algorithms (GA) are employed in pretreatment process by solving the Single Machine problem, and Decision Trees are called upon to extract hidden knowledge in the form of Dispatching Rules. Also are present two heuristics for the setup and use of the new Dispatching Rules since they are different from classic ones. Finally, a comparative study with other well-known DR is performed.

2. STATE OF THE ART

In planning and scheduling, Aytug & al. [11] distinguish two ways of using Data Mining. The first for decision support, where DM helps to identify the best DR since no one rule outperforms all others such as in the paper of Metan & al. [12]. In this approach the state of the system is continuously monitored and the Decision Support System changes the Dispatching Rules if need be to optimize the objective function.

As for the second way, DM is applied to face perturbations scenarios for example a machine breakdown or new jobs arrival. A recent study of Said & al. [13] where the DR is dynamically changed in order to minimize the impact of the perturbation.

A third trend initially introduced by Li [14] where Decision Trees are employed to generate new Dispatching Rules capable of mimicking a metaheuristic or even an exact method for the resolution of a scheduling problem. This approach is proposed for a Single Machine problem using DR. Authors use the LPT rule (Longest Processing Time) where the job having the longest processing time is processed first. Then DT algorithm is applied for the generation of the new DR. One drawback in this paper is the use of the LPT rule since it is a heuristic it is not capable of finding the best solution. Therefore as an alternative, we propose the use of a Genetic Algorithm for the problems resolution.

Other works based on the same idea use different solving methods and for other scheduling problems. For instance in [9], a Job Shop problem is addressed using Tabu Search. The TS algorithm is used at first to find a feasible solution, afterwards a data pretreatment of the solution is done. Finally, Decision Trees, based on the pretreated data, generate a new DR.

Balasundaram & al. [15] perform also a DR generation using DT in a Flow Shop environment. At first a simple heuristic is used to solve the problem comprising 5 jobs and 2 machines. For the scheduling, the makespan (end date of all jobs) is estimated by scheduling a job i before job j and vice versa. The combination minimizing the makespan is used. Then, DT are applied on the scheduling solution to find a new DR. Another idea of Khademi Zare & al. [16] using a hybrid algorithm combining Genetic Algorithm, Data Mining, fuzzy sets, similarity algorithm and attribute-driven deduction algorithm. DM is used to extract rules to help GA to boost its speed to reach the optimal solution.

After a thorough analysis of these approaches, two problems arise. The first resides in the fact that the new Dispatching Rule in a "if-then" form is only compared with the solving methods used in the first step, a heuristic in [15], Tabu Search [9] or Shortest Processing Time for Li & al. [14]. This comparison shows that the new DR performs nearly as efficiently as the heuristic/metaheuristic used for the resolution of the problem, with a certain degree of error. This difference is due to the bad decisions taken by the decision (inverse some jobs). Also, the problem of jobs order inversion is not addressed in any of the quoted approaches, which may have heavy consequences on the system's performance. To illustrate the problem we propose the following example.

Suppose that the new obtained DR is constructed using 3 jobs, J_1, J_2 and J_3. The sequence returned using a solving method is for example: J_2, J_3 and finally J_1. The new DR, contrary to a classic DR, can only compare jobs one by one (see Figure 1):

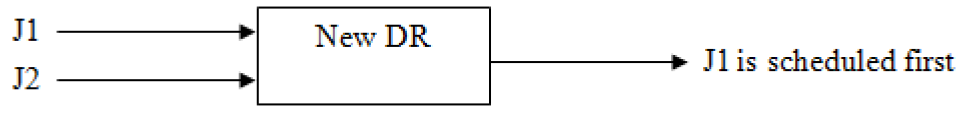

Figure 1. Defining sequence using the new DR

For example, while comparing the jobs the new DR will return the following results:

J_2 vs J_3: J_2 is scheduled first (correct decision)

J_3 vs J_1: J_3 is scheduled first (correct decision)

J_2 vs J_1: J_1 is scheduled first (wrong decision)

In case of a wrong decision it becomes impossible to construct a scheduling. According to the two first decisions, job 1 should be processed last. But with the third decision, being incorrect, there is an ambiguity. Should job 1 be processed last taking into consideration only the two first decisions, or must it be processed first, ignoring the two first decisions and taking only the third one into consideration.

A second problem is that no details are provided as to how to use the new DR in particular the approaches of Li and Shahzad [9], [14]. In those two papers the generated decision tree (DR) have a small size (2 nodes) because of the number of jobs used (5 jobs). But in a real complex problem the jobs number is much higher and consequently the tree size will be larger and more complex. Also the authors never compare the new DR with other rules from literature in order to evaluate its performance.

In this paper we propose two heuristics to take into consideration the bad decision problem and also a comparative study with some Dispatching Rules from the literature is performed.

3. PROPOSED APPROACH

To generate Dispatching Rules based on Decision Trees it is necessary to use an exact or approximate method to solve the problem. In this paper we use a Genetic Algorithm to solve the Single Machine Scheduling Problem, the solution is used by the Decision Tree to create a DR. The choice of GA is justified by its ability to quickly explore the research space, its proven results and wide use for such a problem [1], [17]–[19]. And as done by Li & Shahzad [9], [14] Decision Trees will be used for the DR generation.

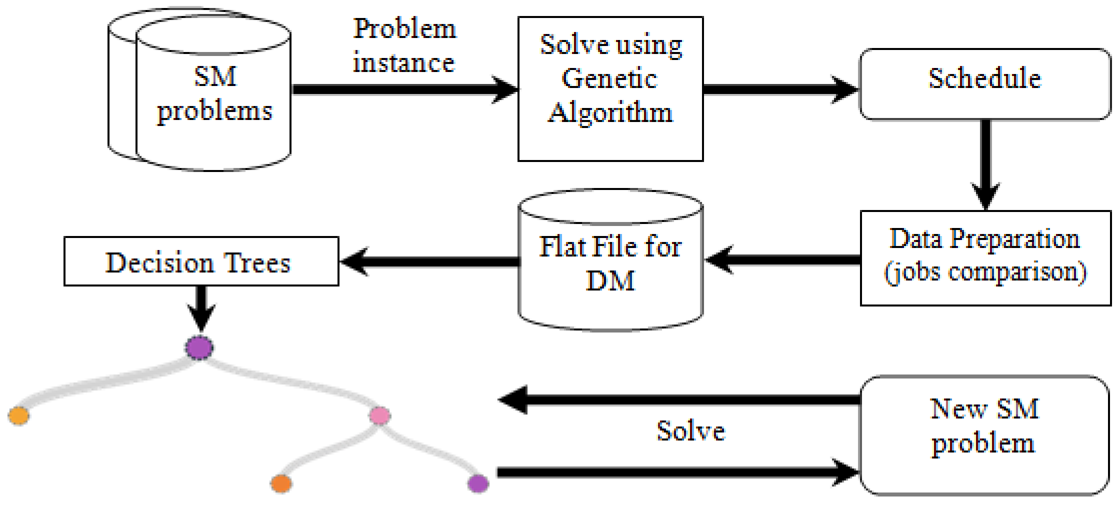

Figure 2. Proposed approach

3.1. Single Machine

The Single Machine Scheduling Problem with Total Weighted Tardiness (SMTWT) can be defined as follows. A set of n jobs $J = \{J_1, J_2, \dots, J_n\}$ to be processed on a machine. Each job J_i consists of a single operation having a processing time $PT_i > 0$, $i = 1, \dots, n$. The importance of a job is expressed using a positive weight $w_i > 0$, $i = 1, \dots, n$. The machine can process only one job at a time and a job's execution cannot be suspended. Each job is supposed to be finished before its deadline d_i. If not a penalty "tardiness" is then calculated where tardiness $T_i = \max\{S_i + PT_i - d_i; 0\}$. S_i denotes the J_i start time. The scheduling objective is to minimize the Total Weighted Tardiness:

$$\min TWT = \sum_{i=1}^{n} w_i T_i$$

3.2. Genetic Algorithm

Genetic Algorithms are used to explore a solution space by mimicking the biological process. They have been successfully applied in literature for several problems including Single Machine problem [18], [19]. The main components of GA are as follows:

- Solution encoding: a representation of solutions
- Initial population: generation of an initial population
- Fitness: measurement function for a given solution, total weighted tardiness in this case
- Selection: selection process for chromosomes to generate a new population
- Genetic operators: a genetic operator such as crossover and mutation are applied on the selected chromosomes in order to create new ones
- Replacement: natural selection of the members of population who will survive

In this paper, the implemented GA is mainly inspired from the one proposed by Armentano [18] with a modified initial population generation process. We suggest that the reader consult the paper for more details.

3.3. Decision Tree

For the generation of Dispatching Rules, we use the method of Li [14] which is adapted to our problem since the used attributes are not the same. Release time, start time, processing time and completion time in the case of [14]; and processing time, weight and due date in this paper. To explain the process we propose the following example:

Suppose there is 4 jobs, and the sequence returned by the GA is |3|0|2|1|

Table 1. Jobs attributes

Job N°	Processing Time	Weight	Due Date
0	15	3	80
1	20	2	30
2	35	6	50
3	7	8	60

A comparative table is constructed (see Table 2) based on Table 1 where jobs are compared one to another. So, job J_1 is compared with $N-1$, job J_2 with $N-2$ and so on. The size of the new table is equal to $\sum_{j=1}^{N-1}(n-j)$. This new table will be used as an entry by the Decision Tree to generate a new Dispatching Rule.

Table 2. Jobs comparison

Job1 N°	**PT1**	**w1**	**d1**	**Job2 N°**	**PT2**	**w2**	**d2**	**Job1 1er**
0	15	3	80	1	20	2	30	Yes
0	15	3	80	2	35	6	50	Yes
0	15	3	80	3	7	8	60	No
1	20	2	30	2	35	6	50	No
1	20	2	30	3	7	8	60	No
2	35	6	50	3	7	8	60	No

The C4.5 Decision Tree algorithm is the applied on the data of Table 2 to create the new DR. The new rules is an if-then form as follows:

$$\boldsymbol{if}\ w_2 \leq 6\ \boldsymbol{then}\ J_1\ is\ processed\ first$$

$$\boldsymbol{if}\ w_2 > 6\ \boldsymbol{then}\ J_2\ is\ processed\ first$$

3.4. New Dispatching Rule

In order to apply the new DR (DT), we propose two heuristics in order to take into consideration the bad decisions.

Proposed Heuristic 1 sort by number of accumulated of "yes"

```
1 : Initialize vector NumberYes[N] to 0
2 : For each Job i to N
3 :     For each Job j to N
4 :         Decision = JobiScheduledFirst(i, j) ;
5 :         If Decision = Yes
6 :             NumberYes [i] = NumberYes [i] + 1 ;
7 :         End if
8 :     End for
9 : End for
10 : Sort NumberYes in decreasing order
```

Proposed Heuristic 2 quick sort

```
1 : While Iteration <= IterationsNumber
2 : For each Job i to N
3 :     For each Job j to N
4 :         Decision = JobiScheduledFirst(i, j) ;
5 :         If Decision = Yes
6 :             Swap(i, j);
7 :         End if
8 :     End for
9 : End for
10 : End while
```

4. Experiments and Results

In order evaluate the performances of the GA and the new DR, we focus on the Weighted Tardiness benchmark available in OR-Library (see http://people.brunel.ac.uk/~mastjjb/jeb/orlib/ wtinfo.html for more details). Where for each problem the best Total Weighted Tardiness is known allowing us by the same way to compare the two heuristics and other DR from literature. Experiments are performed on 125 problems with 40 jobs.

The population size of the GA I is set to 100 chromosomes and the size of the selected population for crossover/mutation is $I/2$ i.e. 50 chromosomes. While the number of iterations is set to 150 and if the best known solution is reached before the GA stops. The modified GA finds the best known solution for 32 problems, that is 25.6%.

We also conducted a comparison between the GA results and the best results. To do so, a gap using the following formula is calculated (1):

$$Average\ Difference = \left[\sum_{i=1}^{Problems\ Number} \frac{GA(i) - ORLibrary(i)}{ORLibrayr(i)}\right] / ProblemsNumber \quad (1)$$

$GA(i)$ is the score found using the Genetic Algorithm for a problem i.

$ORLibrary(i)$ is the best known score in OR-Library for a problem i.

It is worth mentioning that for some problems the best known score in the OR-Library is equal to 0. Thus it becomes impossible to calculate the gap, consequently, those problems are ignored. This reduces the number of problems to 107 (instead of 125 initially). On these 107 problems GA has 9.76% average difference with the OR-Library.

Once the problems are solved using GA, Decision Trees are applied to generate a new Dispatching Rule as explained in the proposed approach section. All the data of all the problems is gathered in one file for the learning process. Then, in order to apply the new DR the two heuristics PH1 and PH2 are used and a comparison is performed using the following formula (2) (see Table 3). The best heuristic in then applied for the comparison with literature Dispatching Rules.

$$Average\ Difference = \left[\sum_{i=1}^{Problems\ Number} \frac{H(i) - ORLibrary(i)}{ORLibrayr(i)}\right] / ProblemsNumber \quad (2)$$

$H(i)$ is a score found using a heuristic H for a problem i.

Table 3. Comparison of PH1 and PH2

PH1		PH2	
Aver. Gap (%)	NP	Aver. Gap (%)	NP
912	1	115,32*	106*

NP is the number of times where the heuristic finds the best results.

From these results we conclude that the second heuristic PH2 is much better than the first one. So, for the following experiments only this one is used for the comparison with the other DR.

The Dispatching Rules used for the comparison (see Table 4) study are:

- Shortest Processing Time (SPT)
- Longest Processing Time (LPT)
- Weighted Shortest Processing Time (WSPT), [2] this rule is best for the SMTWT problem.
- Earliest Due Date (EDD)
- Critical Ratio (CR)
- Mixed Dispatching Rule (MDR) propose dans [20]

Table 4: Comparison of PH2 with other DR

	PH2	**SPT**	**LPT**	**WSPT**	**EDD**	**CR**	**MDR**
Gap (%)	**115.32***	1203.43	4929.03	681.08	162.58	1828.13	162.58
NP	**81***	0	0	16	13	0	13

Based on this results, it is clear that the new Dispatching Rule returns better results in terms of average gap compared to the best known results of the OR-Library. Also in regards to time where it finds the smallest Total Weighted Tardiness value.

In Figure 3 we compare the results of PH2 and MDR being the best two DR. When the objective value is equal tp 0 it means that the optimal solution is reached by one of the two heuristics, if not the difference in terms of Total Weighted Tardiness is shown.

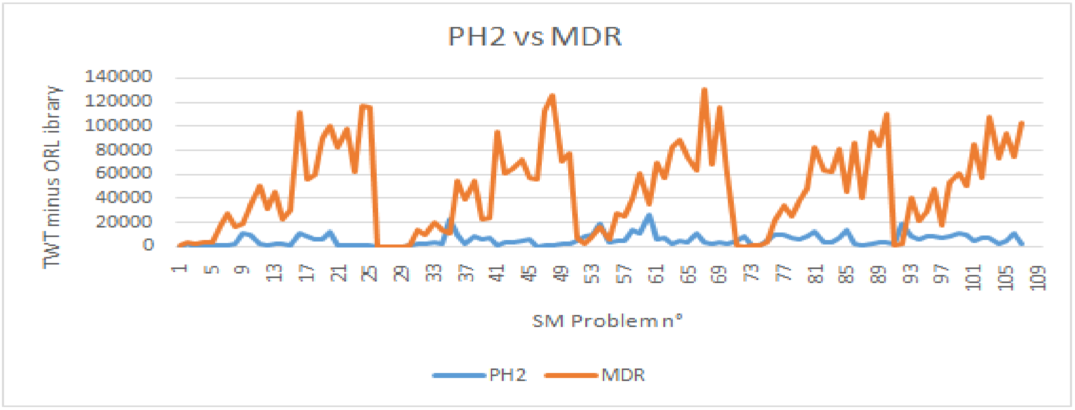

Figure 3. PH2 vs MDR

Also and in order to accurately measure the new DR the set of 107 problems is split in two parts 92/15. The 92 problems will be used for the learning process to create a new DR and other 15 for tests. The aim is to evaluate PH2 for new scheduling problems, results are shown in Table 5.

Table 5: PH2 vs classic DR for new problems

	PH2	**SPT**	**LPT**	**WSPT**	**EDD**	**CR**	**MDR**
Gap (%)	**63.59***	126.11	360.30	68.07	123.86	274.85	123.86
NP	**7***	0	0	**7***	1	0	1

In case of an entirely new Single Machine Scheduling Problem, the proposed heuristic PH2 has an average difference of 63.59% compared to the best known results. In terms of number of times where the best score is reached it performs as well as the WSPT rule. In Figure 4 we compare the results of PH2 and WSPT as done in Figure 3.

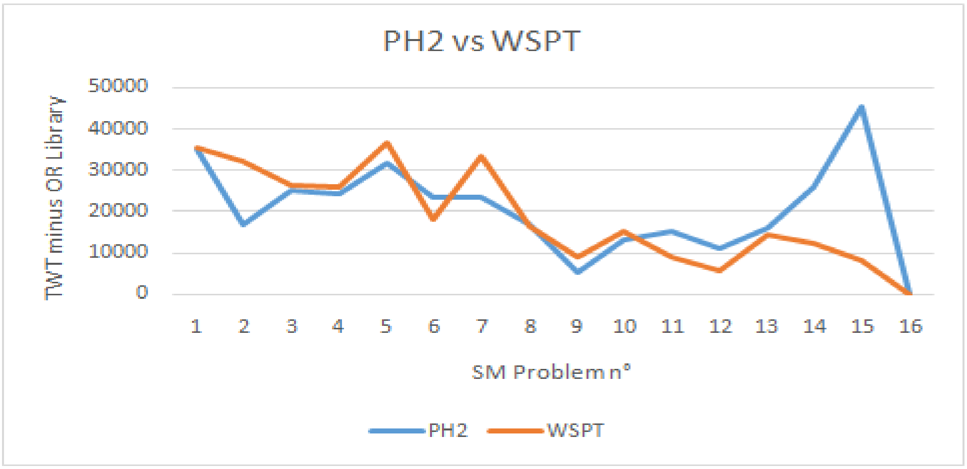

Figure 4. PH2 vs WSPT

From this experiments we prove the superiority of the proposed heuristic for the use of the Decision Tree as Dispatching Rule, while taking into consideration bad decisions. The proposed heuristic also outperforms WMDD [21], H2 et H3 [22] for the same set of data.

5. CONCLUSIONS

In this paper two heuristics for the use of Decision Trees as Dispatching Rule based on a Genetic Algorithm are proposed. The approach was tested for Single Machine problem with Total Weighted Tardiness objective. Experiments show the superiority of the proposed approach compared to some well-known DR for problems used in learning or completely new ones.

In perspective, an improvement of the proposed heuristic is possible. Also, knowing that OR-Library includes SM problems with 50 and 100 jobs, it is interesting to test the new heuristic for such problems. Finally, it might be interesting to consider more complex problems with multiple machines such as the Job Shop Scheduling Problem.

REFERENCES

[1] N. Liu, M. Abdelrahman, and S. Ramaswamy, "A Genetic Algorithm for Single Machine Total Weighted Tardiness Scheduling Problem," Int. J. Intell. Control Syst., vol. 10, no. 3, pp. 218–225, 2005.

[2] E. J. Anderson and C. N. Potts, "Online scheduling of a single machine to minimize total weighted completion time," Math. Oper. Res., vol. 29, no. 3, pp. 686–697, 2004.

[3] H. Zhu and H. Zhou, "Predictive Scheduling for a Single Machine with Random Machine Breakdowns," in LISS 2013, Springer, 2015, pp. 753–758.

[4] J. Tian, R. Fu, and J. Yuan, "A best on-line algorithm for single machine scheduling with small delivery times," Theor. Comput. Sci., vol. 393, no. 1–3, pp. 287–293, 2008.

[5] S. Nguyen, M. Zhang, M. Johnston, and K. C. Tan, "A Computational Study of Representations in Genetic Programming to Evolve Dispatching Rules for the Job Shop Scheduling Problem," IEEE Trans. Evol. Comput., vol. 17, no. 5, pp. 621–639, Oct. 2013.

[6] T. Hildebrandt, D. Goswami, and M. Freitag, "Large-scale simulation-based optimization of semiconductor dispatching rules," in WSC '14 Proceedings of the 2014 Winter Simulation Conference, 2014, pp. 2580–2590.

[7] T. Hildebrandt, J. Heger, and B. Scholz-Reiter, "Towards improved dispatching rules for complex shop floor scenarios: a genetic programming approach," in GECCO '10: Proceedings of the 12th annual conference on Genetic and evolutionary computation, 2010, pp. 257–264.

[8] D. Koonce and S. Tsai, "Using data mining to find patterns in genetic algorithm solutions to a job shop schedule," Comput. Ind. Eng., vol. 38, no. 3, pp. 361–374, Oct. 2000.

[9] A. Shahzad and N. Mebarki, "Discovering Dispatching Rules for Job Shop," in 8th International Conference of Modeling and Simulation - MOSIM'10, 2010.

[10] N. Aissani, B. Atmani, D. Trentesaux, and B. Beldjilali, "Extraction of Priority Rules for Boolean Induction in Distributed Manufacturing Control," Serv. Orientat. Holonic Multi-Agent Manuf. Robot. Stud. Comput. Intell., vol. 544, pp. 127–143, 2014.

[11] H. Aytug, S. Bhattacharyya, G. J. Koehler, and J. L. Snowdon, "A review of machine learning in scheduling," IEEE Trans. Eng. Manag., vol. 41, no. 2, pp. 165–171, May 1994.

[12] G. Metan, I. Sabuncuoglu, and H. Pierreval, "Real time selection of scheduling rules and knowledge extraction via dynamically controlled data mining," Int. J. Prod. Res., vol. 48, no. 23, pp. 6909–6938, Dec. 2010.

[13] N. Said, W. Mouelhi, and K. Ghedira, "Classification Rules for the Job Shop Scheduling Problem with Machine Breakdowns," Int. J. Inf. Electron. Eng., vol. 5, no. 4, p. 300, 2015.

[14] X. Li and S. Olafsson, "Discovering Dispatching Rules Using Data Mining," J. Sched., vol. 8, no. 6, pp. 515–527, Dec. 2005.

[15] R. Balasundaram, N. Baskar, and R. S. Sankar, "A New Approach to Generate Dispatching Rules for Two Machine Flow Shop Scheduling Using Data Mining," Procedia Eng., vol. 38, pp. 238–245, 2012.

[16] H. Khademi Zare and M. B. Fakhrzad, "Solving flexible flow-shop problem with a hybrid genetic algorithm and data mining: A fuzzy approach," Expert Syst. Appl., vol. 38, no. 6, pp. 7609–7615, Jun. 2011.

[17] F. Der Chou, P. C. Chang, and H. M. Wang, "A hybrid genetic algorithm to minimize makespan for the single batch machine dynamic scheduling problem," Int. J. Adv. Manuf. Technol., vol. 31, no. 3–4, pp. 350–359, 2006.

[18] V. a. Armentano and R. Mazzini, "A genetic algorithm for scheduling on a single machine with set-up times and due dates," Prod. Plan. Control, vol. 11, no. 7, pp. 713–720, Jan. 2000.

[19] M. Sevaux and K. Sörensen, "A genetic algorithm for robust schedules in a one-machine environment with ready times and due dates," 4OR, vol. 2, no. 2, pp. 129–147, Jul. 2004.

[20] A. Yin and J. Wang, "Mixed Dispatch Rule for Single Machine Total Weighted Tardiness Problem," J. Appl. Sci., vol. 13, no. 21, pp. 4616–4619, 2013.

[21] J. J. Kanet and X. Li, "A weighted modified due date rule for sequencing to minimize weighted tardiness," J. Sched., vol. 7, no. 4, pp. 261–276, 2004.

[22] S. H. Yoon and I. S. Lee, "New constructive heuristics for the total weighted tardiness problem," J. Oper. Res. Soc., vol. 62, no. 1, pp. 232–237, 2011.