A Dispatching Rules-based (DR-b) Algorithm for Single Machine Bi-criterion Scheduling Problem with Two Agents

M. M. MABKHOT[1], IBRAHIM M. ALHARKAN[1]
[1]Department of Industrial Engineering
King Saud University
Riyadh
SAUI ARABIA
mmabkhot@ksu.edu.sa; imalhark@ksu.edu.sa

Abstract: - In real-life application, it can be often found multiple agents compute on the usage of a single processing resource. These kinds of multi agent scheduling problem have received a lot of research attention lately. However, it is focused in finding an optimal solution of small number of jobs or resulted insufficient solution of large number of jobs. Based on this observation, we attempt in this paper to get a very near optimal solution of problems with large number of jobs. We studied two-agent single-machine scheduling problem where the objective is to minimize the linear combination of the total completion time and maximum tardiness of jobs from the first agent with restriction of no tardy jobs from the second agent are allowed. We developed a dispatching rule-based algorithm to search for near-optimal solution of large number of jobs. The results are compared with the simulated annealing algorithm developed in previous research. The developed algorithm can solve problems of up to 52 jobs in a reasonable amount of time. The comparison shows that the performance of the developed algorithm is better than the SA algorithm and can get a very near optimal solution.

Key-Words: - Single machine scheduling, Two agents.

1 Introduction

In a classical scheduling, it is often assumed that all jobs belong to a single agent that wishes to find a schedule of its jobs to optimize the same objective function/s. In multi-agent scheduling problems; the jobs belong to two or more agents that compete to perform their perspective jobs on shared processing resources. Each agent pursues to optimize its objective function, which depend on its completion time of its jobs [1]. This kind of problems can be occurs in several different application environments, such as industrial management, project scheduling, telecommunication service, etc. [3]. Also it can be occurred in different methodological fields, such as artificial intelligence, decision theory, operation research, etc. [4].

Since the multi-agent problems was brought into scheduling field by [1]&[2], many researcher have expanded an abundance of effort on this new scheduling research topic. The researches focused in minimizing the objective function from one agent, subjected to the objective function from the second agent doesn't exceed a given limit, such as in [3-7], or they minimized the two objective functions from both agents, such as in [8-13]. [2] Pointed out that the two-agent problem is NP-hard if both agents have the sum-type objectives. In [6] they considered a single machine problem with two agents where the objective is to minimize the linear combination of the total completion time and maximum tardiness of jobs from the first agent with restriction of no tardy jobs from the second agent are allowed. They developed a branch and bound and six simulated annealing (SA) algorithms to find optimal and near-optimal solutions. The branch and bound solved problems up to 24 jobs and the SA algorithms solved problems with an average percentage of error around 0.5%. However it is worthy to find a very near-optimal solution for large number of jobs. In this paper we developed a Dispatching Rule-based (DR-b) algorithm for the same problem in order to solve number of jobs larger than jobs that solved by the branch and bound with better performance of the SA algorithms.

The remainder of this paper is organized as follows. Section 2 shows the problem definition and the compared algorithm parameters. Section 3 presents the DR-b algorithm. In section 4, the computational experiments to evaluate the performance of the DR-b algorithm and the benchmarking against the simulated annealing are conducted. Section 5 concludes the paper and suggests topics for future research.

2 Problem Definition

2.1 Problem description

The problem description is given as follows. There are n jobs ready to be processed on a single machine. Each job belongs to either one of the two agents AG_0 or AG_1. For each job j, there is a processing time p_j, a due date d_j, and an agent code AG_0 & AG_1 represent the job from the first and second agent respectively. Let $C_j(S)$ be the completion time of the job j, $T_j(S) = max\{0, C_j(S) - d_j\}$ be the tardiness of the job j, $T_{max}(S) = max\, T_{j\in AG_0}(S)$ be the maximum tardiness of jobs from agent AG_0 under schedule S, and $U_{j\in AG_1}(s) = 1$ and zero otherwise. The objective is to find a schedule that minimizes the weight combination of the total completion time and the maximum tardiness of jobs from AG_0 with the restriction that no tardy jobs from AG_1 are allowed. Using the conventional three fields notation, this problem is denoted as: $1/\sum U_{j\in AG_1}=0/\sum C_{j\in AG_0} + T_{max,j\in AG_0}$.

2.2 Previous work for comparison

The same problem solved by [6] where they developed a branch and bound and six SA algorithms. These six SA algorithms are considered depending on the initial sequence and the neighborhood generation movements. From this six SA algorithm, we selected the one that achieved the best results in order to compare it with our developed algorithm. In the following sub-section, we show this SA algorithm.

2.2.1 SA Algorithm

The essential elements of the selected SA algorithm included:

- *Initial sequence:* The first initial sequence, jobs from agent AG_1 are first placed according to the EDD rule, followed by jobs from agent AG_0 according to the SPT first rule, denoted as $SA_{EDD+SPT}$ in later analysis.
- *Neighborhood generation:* The neighborhood generation is based on pairwise interchange. Depending on the initial sequences and the neighborhood generation movements, where it can be represented as $SA^{PI}_{EDD+SPT}$.
- *Acceptance probability:* The probability of acceptance is generated from an exponential distribution,

$$P(accept) = exp(-k \times \frac{\Delta TC}{\theta})$$

Where ΔTC is the change in the objective function, and θ was chosen equal to 5000. If the weight combination of the total completion time and the maximum tardiness of jobs from agent AG_0 increases, the new sequence is accepted with probability r, where r is a uniform random number between 0 and 1.

- *Stopping condition:* The procedure is stopped after $300n$ iterations, where n is the number of jobs.

3 DR-b Algorithm

It is noted that the problem under consideration is computationally intractable especially when the number of jobs from the second agent AG_1 increased. Thus to so solve the problem in reasonable time, the near-optimal solution was tried to reach through a large reduction of nodes number. Based on a dispatching rules and dominance steps, the DR-b algorithm was developed. The restriction of tardy jobs from the second agent should be satisfied at the beginning. As a consequence, the maximum tardiness from the first agent should be minimized. The jobs are ordered by EDD, and ordered the jobs from the second agent first. The jobs were shifted from both agents with shortest processing time earlier than the jobs from second agent without making a tardy job from the second agent. Fig. 1 depicted the DR-b algorithm flow chart. In the following, we will explain the notation and the DR-b algorithm steps.

Notation:

S	:	Sequence of jobs n.
i and j	:	Two jobs in S and job i is before j.
AG_1	:	Jobs from the second agent
AG_0	:	Jobs from the first agent
AG_1opt	:	Optimal sequence part contains all AG_1 jobs and may be part of AG_0 jobs.
AG_0seq	:	A sequence part where AG_0seq = S - AG_1opt.
AG_0opt	:	Optimal sequence of $j \in AG_0seq$.
$Sopt$	:	Optimal sequence of jobs n.

DR-b Algorithm

Step 1	:	Order job in S according to EDD for jobs from AG_1 first.
Step 2	:	If S has a $T_{i\in AG_1} \neq 0$ then assign AG_1opt and go to step 4; Else go to step 3;
Step 3	:	for S if $P_{i\in AG_1} > P_j$ & $P_i =< E_{j\in AG_1}$ Then j should be before i in the AG_1opt and go to step 4;

		Else assign AG_1opt and go to step 4;
Step 4	:	For AG_0seq =S-AG_1opt If P_i>P_j go to step 5; Else $AG_0opt = AG_0seq$ and go to step 7;
Step 5	:	find AG_0seq' with j before i and go to step 6;
Step 6	:	If $AG_0seq'(T_{maxj\in AG_0} + \sum C_{j\in AG_0}) <$ $AG_0seq(T_{maxj\in AG_0} + \sum C_{j\in AG_0})$ Then $AG_0opt = AG_0seq'$ and go to step 7; Else assign $AG_0opt = AG_0seq$ and go to step 7;
Step 7	:	Stop and Sopt= AG_1opt + AG_0opt.

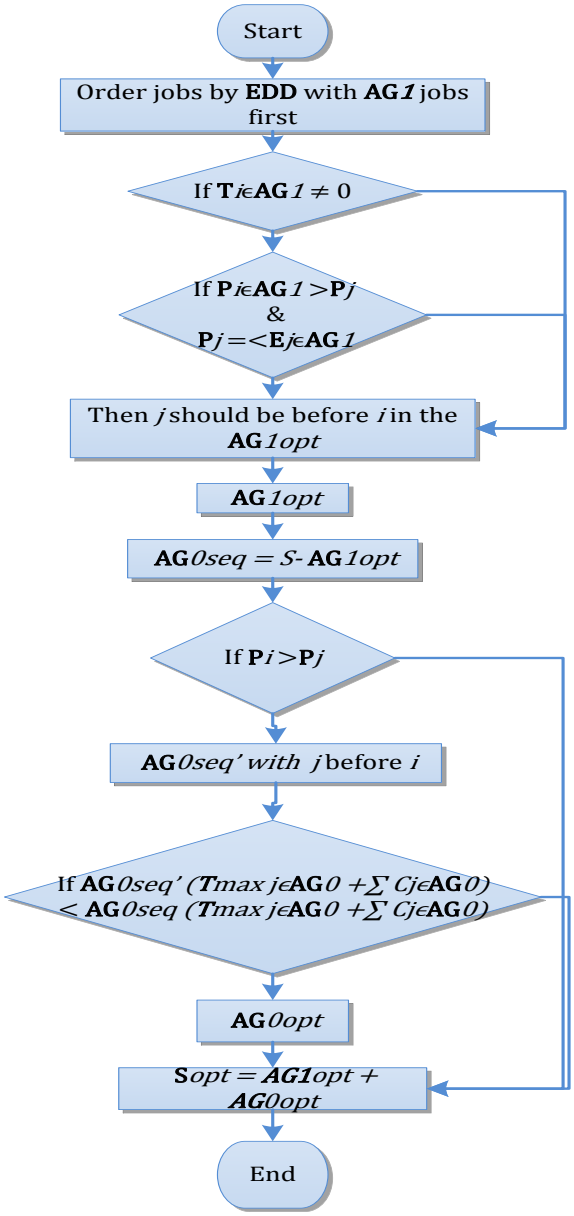

Fig. 1: DR-b algorithm flow chart.

4 Computational Experiments

In this section, the computation experiments were conducted to evaluate the performance of the DR-b algorithm and to benchmark it against the SA algorithm. They were coded in MATLAB R2011b and run it on a personal computer with 3.20 GHz Intel(R) Core (TM) 2 Duo CPU T5800 2.00 GHz and 4.096 GB RAM under Windows 7.

4.1 Instances Generation

The same instance generation parameters that used in the compared paper were used [6]. The job processing times were generated from a uniform distribution over the integers 1–100, and the due dates of jobs were generated from another uniform distribution over the integers between $T(1-t-R/2)$ and $T(1-t+R/2)$, where R is the due date range factor, τ is the tardiness factor, and T is the total processing times of all the jobs. Two values of τ $(0.25, 0.5)$ and three values of R $(0.25, 0.5, 0.7)$ were tested. The assignment of the two agents was selected according the P factor percentage from the total number of jobs n, where three value of P $(0.25, 0.5, 0.75)$ were used.

4.2 DR-b Algorithm Performance

The main purpose of this part of the experiment was conducted to study the impact of the number of jobs to the performance of the DR-b algorithm. A total of 450 instances (90 instances and 5 replicate for each instances parameters) was computed for the new algorithm. Five different job sizes (n=20, 28, 36, 44 and 52) were tested. Since problems are harder to solve if the due date factor τ is smaller or R is larger, and the proportion of jobs from the second agent P. Also we found that the problems are harder to solve when the percentage of jobs from the second gent P is increased. The mean, standard deviation of the tardy jobs number for the second agent, the sum of linear combination of completions time and max tardiness for the first agent, the number of nodes, and the CPU time (in seconds) for the new algorithm were recorded in Table 1.We noticed that the DR-b algorithm can solve problem in reasonable amount of time with average time of 0.06 sec when (τ, R, P) = (0.5, 0.75, 0.25) and with ST seams to equal zero for most instances. It was also observed that the number of nodes increased and the execution time grows when n is increased.

4.2 SA Algorithm Performance

In order to compare DR-b algorithm and SA algorithm, the same instances that evaluated in the

Table 1 .The performance of the DR-b algorithm.

n	*τ*	*R*	*P*	CPU Time		number of Nodes		*n*	*τ*	*R*	*P*	CPU Time		number of Nodes	
				Mean	ST	Mean	ST					Mean	ST	Mean	ST
20	0.25	0.25	0.25	0.04	0.00	24.20	1.17	***28***	0.25	0.25	0.25	0.04	0.00	34.00	0.63
			0.5	0.04	0.00	22.80	1.17				0.5	0.04	0.01	31.80	1.47
			0.75	0.04	0.00	22.40	0.80				0.75	0.04	0.00	29.80	0.75
		0.5	0.25	0.04	0.00	24.80	1.47			0.5	0.25	0.04	0.00	34.60	1.36
			0.5	0.04	0.00	23.60	1.62				0.5	0.04	0.00	31.00	1.10
			0.75	0.04	0.00	22.40	0.49				0.75	0.04	0.01	29.80	0.75
		0.75	0.25	0.03	0.01	24.80	1.72			0.75	0.25	0.04	0.00	36.60	1.36
			0.5	0.06	0.05	25.20	0.75				0.5	0.04	0.01	31.60	2.94
			0.75	0.03	0.00	22.20	1.60				0.75	0.04	0.00	29.60	0.49
	0.5	0.25	0.25	0.04	0.00	26.60	2.94		0.5	0.25	0.25	0.04	0.00	34.80	1.60
			0.5	0.03	0.00	21.60	0.80				0.5	0.04	0.00	31.40	1.62
			0.75	0.03	0.00	26.00	0.00				0.75	0.03	0.00	36.00	0.00
		0.5	0.25	0.04	0.00	26.20	0.98			0.5	0.25	0.04	0.00	36.80	1.83
			0.5	0.03	0.01	25.60	2.87				0.5	0.04	0.00	32.80	2.14
			0.75	0.31	0.53	25.00	2.00				0.75	0.04	0.00	35.40	1.20
		0.75	0.25	0.04	0.01	27.00	2.19			0.75	0.25	0.04	0.00	34.20	2.32
			0.5	0.04	0.00	25.00	1.26				0.5	0.04	0.00	33.00	1.79
			0.75	0.03	0.01	23.20	1.72				0.75	0.04	0.00	32.40	3.14
36	0.25	0.25	0.25	0.04	0.00	40.60	1.85	***44***	0.25	0.25	0.25	0.04	0.00	51.20	1.94
			0.5	0.04	0.00	39.60	1.36				0.5	0.04	0.01	47.40	2.06
			0.75	0.04	0.00	37.60	0.80				0.75	0.04	0.01	46.80	1.47
		0.5	0.25	0.04	0.00	44.00	0.89			0.5	0.25	0.05	0.00	51.40	2.65
			0.5	0.05	0.00	41.20	1.72				0.5	0.05	0.00	49.80	1.47
			0.75	0.04	0.00	38.40	1.02				0.75	0.04	0.00	47.20	1.47
		0.75	0.25	0.04	0.00	42.20	1.94			0.75	0.25	0.04	0.00	50.20	3.66
			0.5	0.04	0.00	41.00	3.35				0.5	0.04	0.00	48.00	2.00
			0.75	0.04	0.00	37.80	1.17				0.75	0.04	0.00	45.20	0.40
	0.5	0.25	0.25	0.04	0.00	48.00	3.29		0.5	0.25	0.25	0.05	0.00	57.20	2.64
			0.5	0.04	0.00	40.80	2.86				0.5	0.05	0.00	49.20	1.17
			0.75	0.04	0.00	46.00	0.00				0.75	0.04	0.00	56.00	0.00
		0.5	0.25	0.04	0.01	46.20	4.35			0.5	0.25	0.05	0.00	55.20	1.60
			0.5	0.05	0.00	41.60	2.33				0.5	0.04	0.00	47.00	2.53
			0.75	0.04	0.00	44.20	3.60				0.75	0.04	0.00	54.00	4.00
		0.75	0.25	0.04	0.00	45.80	3.25			0.75	0.25	0.06	0.02	53.40	0.80
			0.5	0.05	0.01	41.20	1.33				0.5	0.04	0.00	48.00	2.10
			0.75	0.05	0.02	42.60	4.18				0.75	0.04	0.00	49.80	5.08
52	0.25	0.25	0.25	0.05	0.00	61.60	1.85	***52***	0.5	0.25	0.25	0.05	0.00	63.60	5.50
			0.5	0.05	0.00	57.80	1.94				0.5	0.05	0.00	55.60	2.58
			0.75	0.05	0.00	55.00	1.41				0.75	0.04	0.00	66.00	0.00
		0.5	0.25	0.05	0.00	62.00	3.90			0.5	0.25	0.05	0.00	62.20	5.15
			0.5	0.05	0.00	57.40	2.06				0.5	0.05	0.00	58.80	2.56
			0.75	0.05	0.00	55.20	1.17				0.75	0.04	0.00	66.00	0.00
		0.75	0.25	0.05	0.00	60.20	3.60			0.75	0.25	0.05	0.00	61.80	1.60
			0.5	0.05	0.00	59.40	2.15				0.5	0.04	0.01	56.40	2.87
			0.75	0.04	0.00	53.60	0.80				0.75	0.04	0.01	59.60	5.39

Table 2. The SA algorithm percentage of error from the BR-b algorithm.

n	*τ*	*R*	*P*	*Mean*	*ST*	*n*	*τ*	*R*	*P*	*Mean*	*ST*
20	0.25	0.25	0.25	0.13	0.04	***28***	0.25	0.25	0.25	0.23	0.07
			0.5	0.14	0.02				0.5	0.22	0.04
			0.75	0.26	0.02				0.75	0.27	0.04
		0.5	0.25	0.14	0.02			0.5	0.25	0.22	0.02
			0.5	0.19	0.04				0.5	0.24	0.06
			0.75	0.29	0.25				0.75	0.25	0.07
		0.75	0.25	0.20	0.12			0.75	0.25	0.23	0.04
			0.5	0.18	0.04				0.5	0.22	0.04
			0.75	0.12	0.03				0.75	0.24	0.07
	0.5	0.25	0.25	0.19	0.06		0.5	0.25	0.25	0.30	0.14
			0.5	0.17	0.04				0.5	0.25	0.06

ISBN: 978-1-61804-347-4

			0.75	0.04	0.04				0.75	0.04	0.03
n	***τ***	***R***	***P***	***Mean***	***ST***	***n***	***τ***	***R***	***P***	***Mean***	***ST***
		0.5	0.25	0.19	0.07			0.5	0.25	0.30	0.11
			0.5	0.21	0.12				0.5	0.23	0.07
			0.75	0.09	0.04				0.75	0.18	0.05
		0.75	0.25	0.17	0.04			0.75	0.25	0.26	0.03
			0.5	0.18	0.05				0.5	0.21	0.04
			0.75	0.11	0.03				0.75	0.20	0.08
36	0.25	0.25	0.25	0.28	0.05	***44***	0.25	0.25	0.25	0.32	0.07
			0.5	0.20	0.04				0.5	0.29	0.08
			0.75	0.33	0.07				0.75	0.32	0.05
		0.5	0.25	0.26	0.02			0.5	0.25	0.27	0.03
			0.5	0.35	0.06				0.5	0.28	0.06
			0.75	0.31	0.01				0.75	0.33	0.10
		0.75	0.25	0.24	0.01			0.75	0.25	0.28	0.03
			0.5	0.28	0.03				0.5	0.28	0.03
			0.75	0.27	0.06				0.75	0.27	0.04
	0.5	0.25	0.25	0.43	0.02		0.5	0.25	0.25	0.50	0.11
			0.5	0.33	0.06				0.5	0.27	0.02
			0.75	0.04	0.03				0.75	0.03	0.02
		0.5	0.25	0.39	0.06			0.5	0.25	0.32	0.03
			0.5	0.22	0.03				0.5	0.24	0.07
			0.75	0.08	0.07				0.75	0.12	0.08
		0.75	0.25	0.34	0.10			0.75	0.25	0.29	0.04
			0.5	0.28	0.03				0.5	0.27	0.04
			0.75	0.17	0.04				0.75	0.21	0.12
52	0.25	0.25	0.25	0.36	0.03	***52***	0.5	0.25	0.25	0.36	0.05
			0.5	0.29	0.05				0.5	0.30	0.06
			0.75	0.38	0.05				0.75	0.03	0.04
		0.5	0.25	0.31	0.03			0.5	0.25	0.34	0.04
			0.5	0.30	0.04				0.5	0.27	0.03
			0.75	0.36	0.07				0.75	0.07	0.05
		0.75	0.25	0.30	0.04			0.75	0.25	0.42	0.07
			0.5	0.38	0.05				0.5	0.27	0.04
			0.75	0.34	0.05				0.75	0.19	0.11

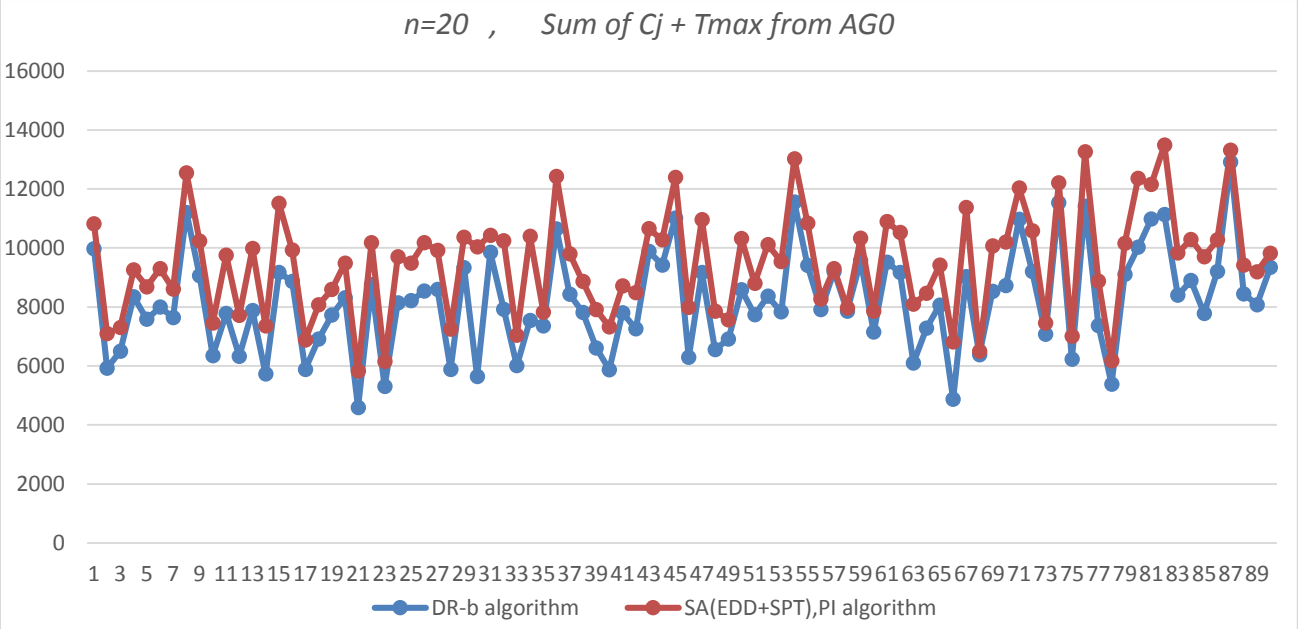

Fig.2: The comparison between DR-b algorithm and the SA algorithm for $n = 20$.

previous sub-section were conducted. A total of 2250 instances were conducted (90 instances and 5 replicate for each instances parameter and 5 replicate to estimate SA performance). Since only the mean and standard deviation of error percentages of SA algorithms were recorded Table 2. The execution

times of SA algorithm weren't recorded because they are finished within a second. We selected a sample of n=20 to show the performance of both algorithm in Fig.2. From this Figure, we clearly see for all instances the linear combination of total completion time and maximum tardiness of jobs from the first agent resulted from DR-b algorithm are less than SA algorithm. The computational experiment of all cases shows that the SA algorithm average percentage of error from DR-b algorithm is around 0.3%. This percentage is deviated by 0.2% from 0.5% that resulted from the same SA algorithm comparing with the branch and bound in the compared paper. According to this, we conclude with strong evidence that the DR-b algorithm can solve problems of up to 52 jobs in a reasonable amount of time, and its results are very near to optimal solution with an average percentage of error around 0.2%.

5 Conclusion

In this paper, we studied a two-agent single-machine scheduling problem. The objective is to minimize the weighted combination of the completion time and maximum tardiness of the jobs of the first agent, given that no tardy jobs are allowed for the second agent. We developed a DR-b algorithm to solve the problem, and we compared it with SA Algorithm that developed in previous work to find very near-optimal solution. The computational experiments showed that the DR-b algorithm can solve problems up to 52 in a reasonable amount of time. In addition, the comparison with SA algorithm shows that for all instances the DR-b algorithm results are better than the SA algorithm, and can get a solution very near to optimal. For a future work, it is interesting to consider a scheduling problem with more than two agents, or proposing more sophisticated heuristics.

References:

[1] K. R. Baker and J. C. Smith, A multiple-criterion model for machine scheduling, *Journal of Scheduling*, Vol. 6, No. 1,2003, pp. 7–16.

[2] A. P. Allesandro Agnetis, Pitu B. Mirchandani, Dario Pacciarelli, Scheduling Problems with Two Competing Agents, *Opeationr Research*, Vol. 52, No. 2, 2004, pp. 229–242.

[3] Q. Feng, B. Fan, S. Li, and W. Shang, Two-agent scheduling with rejection on a single machine, *Applied Mathematical Modelling*, Vol. 39, No. 3–4, 2015, pp. 1183–1193.

[4] A. Agnetis, D. Pacciarelli, and A. Pacifici, Multi-agent single machine scheduling, *Annals of Operations Research*, Vol. 150, No. 1, 2007, pp. 3–15.

[5] E. Gerstl and G. Mosheiov, Scheduling problems with two competing agents to minimized weighted earliness-tardiness, *Computers and Operations Research*. Vol. 40, No. 1, 2013, pp. 109–116.

[6] W. C. Lee, Y. H. Chung, and Z. R. Huang, A single-machine bi-criterion scheduling problem with two agents, *Applied Mathematics and Computation*, Vol. 219, No. 23, 2013, pp. 10831–10841.

[7] C. T. Ng, T. C. E. Cheng, and J. J. Yuan, A note on the complexity of the problem of two-agent scheduling on a single machine, *Journal of Combinatorial Optimization*, Vol. 12, No. 4, 2006, pp. 386–393.

[8] T.C. E. Cheng, C.-Y. Liu, W.-C. Lee, and M. Ji, Two-agent single-machine scheduling to minimize the weighted sum of the agents' objective functions, *Computers & Industrial Engineering*, Vol. 78, 2014, pp. 66–73.

[9] Y. Yin, T. C. E. Cheng, L. Wan, C.-C. Wu, and J. Liu, Two-agent single-machine scheduling with deteriorating jobs, *Computers & Industrial Engineering*, Vol. 81, 2015, pp. 177–185.

[10] V. Kaplanoğlu, "Multi-agent based approach for single machine scheduling with sequence-dependent setup times and machine maintenance, *Applied Soft Computing*, Vol. 23, 2014, pp. 165–179.

[11] T. C. E. Cheng, C. T. Ng, and J. J. Yuan, Multi-agent scheduling on a single machine with max-form criteria, *European Journal of Operational Research*, Vol. 188, No. 2, 2008, pp. 603–609.

[12] C. E. Cheng, C. T. Ng, and J. J. Yuan, Multi-agent scheduling on a single machine to minimize total weighted number of tardy jobs, *Theoretical Computer Science*., Vol. 362, No. 1–3, 2006, pp. 273–281.

[13] J. J. Yuan, W. P. Shang, and Q. Feng, A note on the scheduling with two families of jobs, Journal of Scheduling, Vol. 8, No. 6, 2005, pp. 537–542.