ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 12, No. 4, 2017, *pp*.270-279

An Intelligent Cooperative Approach Applied to Single Machine Total Weighted Tardiness Scheduling Problem

Lamiche Chaabane [1+]

[1] *Department of Computer Science, Mohamed Boudiaf university M'sila, Algeria*

(Received July 29, 2017, accepted September 30, 2017)

Abstract. In this research work, we propose an intelligent search technique called genetic simulated annealing algorithm (GASA) to obtain an approximate solution to the single machine total weighted tardiness job scheduling problem, which is a strong *NP-hard*. The developed approach is based on two metaheuristics: genetic algorithm (GA) and simulated annealing (SA) algorithm. In this context, when GA is exploited as a global search strategy to discover solution space, SA algorithm is used as a local search technique to enhance more efficiently the visited attractive regions to improve solution quality. Numerical results using a set of benchmarks have shown the capability of the proposed method to produce better solutions compared to results given by some other recently literature works.

Keywords: genetic simulated annealing, scheduling, genetic algorithm, simulated annealing, benchmarks.

1. Introduction

The single machine total weighted tardiness (SMTWT) problem is an important special case of scheduling problem that is referred to be a strongly NP-hard problem [1][2]. It has been discussed for many years, and several effective algorithms have been presented in the literature. This problem is stated as following : There are a set of *n* jobs to be processed on exact one machine. The machine processes only one job at a time and without interruption. Each job *i* has an integer processing time p_i, a positive weight w_i and a distinct due date d_i. For a given processing order of the jobs, the earliest completion time C_i and tardiness $T_i = max(0,C_i - d_i)$ can be computed for each job. The objective is to find the sequence of *n* jobs that minimizes the total weighted tardiness (TWT) which is done as following :

$$TWT = \sum_{i=1}^{n} w_i * \max(0, C_i - d_i) \quad (1)$$

Actually, the SMTWT problem becomes an important combinatorial optimization problem that has various real life applications such as sequencing in production process, assigning the sequence of stages in a construction project, delivering the goods with the customer's priority in supply chain and so on [3].

A various number of exact methods have been developed in literature in order to resolve the SMTWT problem for a moderate number of jobs. The well-known of them is dynamic programming paradigm and Branch-and-Bound method, optimality of the solution obtained by these approaches is guaranteed, but they are mostly constrained by available memory and required computation time, especially when the number of jobs is more than 50 [4]. To overcome these limits, metaheuristic algorithms were shown to be promising. The basic idea of this methods class, is to start by an initial solution and iteratively improve it through a series of iterations until the solution doesn't become better any longer. They include polynomial-time approximation algorithm [5], tabu search [6], [7], simulated annealing [8], ant colony optimization [9], iterated dynasearch [10], genetic algorithm [11], variable neighborhood search [12] and so on.

In this research paper, we aim to present a hybrid model based on metaheuristics to solve the SMTWT problem. The proposed approach combines the strong global search ability of genetic algorithm to discover the search space using a part of the initial population with the excellent local improvement which is given by simulated annealing using the rest of the population. The reminder of this paper is organized as follows : In section 2, some previous related works are briefly cited. Section 3 summarizes principals concepts of both

[+] Lamiche Chaabane. Tel.: +00-213-774 34 34 03.
E-mail address: lamiche07@gmail.com.

GA and SA algorithms. Section 4 outlines the proposed GASA algorithm and its components for the SMTWT problem. In section 5, we present the obtained simulation results. Finally, conclusion and perspectives of our work are given in section 6.

2. Related Works

A brief review of some related works which are used to solve SMTWTP is presented in this section. In Ref. [4], the authors demonstrated that exact methods such as Dynamic Programming and Branch-and-Bound are inconsistently solve problems with more than 50 jobs. Potts and van Wassenhove [13] compared four dispatching rules: weighted shortest processing time (WSPT), earliest due date (EDD), modified cost over time (MCOVERT) and apparent urgency (AU). They found that AU performs best and WSPT is greatly inferior. Alidaee and Ramakrishnan [14] tested the COVERT-AU class of dispatching rules for problems of up to 200 jobs. As a result of their study, dispatch rules give a quick sequencing method, but have poor solution quality.

Crauwels et al. [15] proposed a single and multi-start versions of simulated annealing, tabu search (TS) and genetic algorithm implementations for the SMTWTP problem. The authors showed that an acceptable results are produced by simulated annealing and tabu search dominates the other methods. An iterated dynasearch algorithm, which is a local search technique that uses dynamic programming to generate the best moves, is introduced in [10]. An excellent solutions are also obtained in a short time by a natural permutation encoded GA and its multistart version using problems with 40 and 50 jobs [16].

A particle swarm optimization (PSO) algorithm to solve the single machine total weighted tardiness problem is proposed in [17]. Experimental results show that the PSO algorithm is able to find the optimal and best-known solutions on all instances of widely used benchmarks from the OR libary. An heuristic search applied to the SMTWTP is discussed in [18]. In Ref. [19], the authors developed a variable structure learning automata to solve SMTWT problem. Recently, the authors in [20] proposed a new genetic algorithm to produce a best approximate solutions for SMTWT problem. The developed procedure provided a good results compared to some existing dispatching rules.

3. Preliminaries

3.1. Genetic Algorithm Overview

The basic principles of GAs were introduced by Holland in 1975 [21]. GA is an intelligent random search technique which have been successfully applied to find optimal solutions of many complex problems [22]. A genetic algorithm starts with a population of potential solutions and iteratively replaces the current population by a new population. It is based on a suitable encoding for the problem and a fitness function that is used to measure each solution quality. At each generation, a selection operator is applied to choose the parents and recombines them using a crossover operator to produce offsprings that are submitted to a mutation operator in order to perturb them locally. This will continue for many generations until the termination condition is met.

3.2 Simulated Annealing Overview

Simulated annealing was first introduced by Kirkpatrick et al. [23]. It is a stochastic method that is used to find approximate solutions to very large combinatorial problems. The annealing algorithm begins with an initial feasible configuration and proceeds to generate a neighboring solution by perturbing the current solution. If the cost of the new solution is less than that of the current solution, the new solution is accepted; otherwise, it is accepted or rejected with probability $p = e^{-\Delta C/T}$. The probability of accepting solutions depending of the temperature (T), and the difference in cost between the new solution and the current solution ($-\Delta C$). Typically, SA starts with a large value of T, which means that an inferior solution has a high probability of being accepted and the algorithm works as a random search to find a promising region in the solution space. As the optimization process progresses, the temperature decreases and there is a lower probability of accepting an inferior solution. The process is repeated until the stopping criterion is reached.

4. Materials and Methods

The mean advantage of GA is its capability to discover solution space globally according to the crossover and mutation operators, but it suffers from premature convergence and it can not escape from local optimum. In other hand, SA algorithm accepts a worse solution and it has the ability of escaping from local optimal solution which allows to visit another promising regions until solving overall optimal solution. The new developed hybrid model called GASA consists in a strong cooperation of GA and SA, it takes advantages of the diversification strategy assured by GA and the local improvement allowed by SA technique.

In our SMTWT problem, the main idea of this cooperation consists of creating randomly two equitably size populations from the whole basic initial population, these ones called respectively: *GA-population* and *SA-population*. When the first half is treated by GA algorithm to guide global search, the second part is exploited using SA algorithm to obtain local best solution for each chromosome.

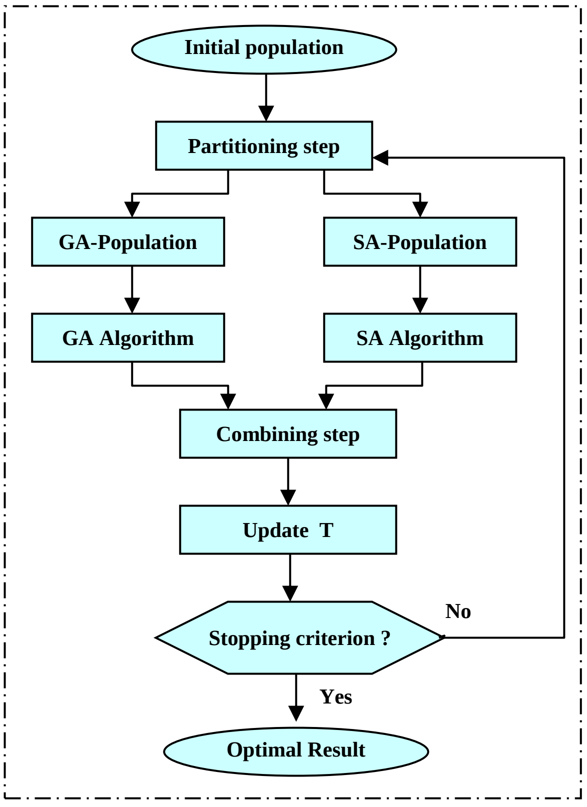

Fig. 1: Folowchart of the proposed algorithm.

The temperature T is updated after the combined population is constructed. The general flowchart of the proposed approach is presented in Fig. 1.

4.1 GA components for SMTWT problem

4.1.1 Chromosome representation

For the single machine total weighted tardiness problem, each chromosome has *n* genes and each gene of the chromosome has an allele value chosen randomly from the set {1, 2, . . . , *n*}, where *n* is the number of the jobs to be scheduled. Hence a chromosome is represented as a vector of feasible processing order of *n* jobs which represent a scheduling solution.

4.1.2 Fitness function

The fitness computation process consists of two phases. In the first phase, the total weighted tardiness is calculated for each chromosome in the population. After that, all chromosomes are ordered according to the decreasing values of $1/\sum w_i T_i$ and individuals whose are high values of $1/\sum w_i T_i$ are retained in the next generation.

4.1.3 Population initialization

Initial population is an important element for all evolutionary algorithms such as GAs. Therefore, using a better technique to create the initial population would not only improve the average performance of genetic algorithm, but also to reduce computation time dramatically. In this study, the initial population is composed of a certain number denoted as P of chromosomes which are generated randomly and corresponds of feasible solutions, after that, the mixed dispatch rule MDR [24] is applied to each chromosome in order to improve its quality.

4.1.4 Selection scheme

Selection scheme plays an essential role in improving the average quality of the population by passing better chromosomes to the genetic operators in order to construct the next generation. Here, the scheme of roulette wheel is chosen according to its simplicity and its effectiveness. In this way, the values given a better fit have higher probability to produce a child chromosomes.

4.1.5 Crossover operator

Crossover is the first basic genetic operator for GAs. This operator processes the current solutions in order to find better ones by the selection of genes from parent chromosomes and the creation of a new offspring. In this work, we retain the one cut-point crossover strategy. The proposed one-point-cut method to generate the offspring is described as follows :

Choose randomly two parents and one position to cut parents, the first child is generated using the inferior part of parent1 and it is completed by the other genes of parent2 in which the order of added genes is respected. The same way is used to generate child2 and the proposed scheme is illustrated in Fig. 2 bellow :

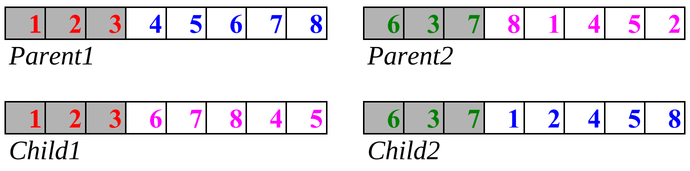

Fig. 2: Scheme of crossover operator.

We can indicate here, that this scheme is simple and generate feasible solutions for our SMTWT problem.

4.1.6 Mutation operator

The principal role of the mutation operator is to provide a diversity in a population and helps GA to keep away from local optima. For that, it alters one or more genes with a probability equal to the mutation rate. The mutation procedure used in this work can be summarized as follows :

Choose randomly one gene and one position in the chromosome to mute. After that, insert this gene in the selected position. This technique is showed in Fig. 3 bellow:

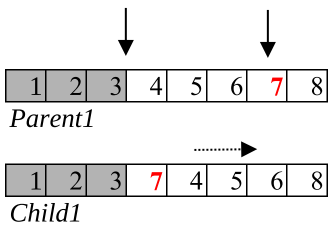

Fig. 3: Scheme of mutation operator.

4.1.7 Replacement

This step is very important to construct the next generation for the genetic algorithm. The replacement strategy used here consists of the selection of the best chromosomes of the current population and their offspring. They will form a new population to survive into the next generation.

4.1.8 Termination Criteria

The last element necessary for genetic algorithm is termination criteria. In our study, the search process can be stopped when certain number of iterations *Itmax* is completed.

The pseudo-code of our genetic algorithm can be summarized as follows :

Pseudo-code of GA Algorithm

```
Initialize $P_{size}$, $P_{cross}$ , $P_{mut}$, $Itmax$
Set $\mu = 0$
Generate $P_{size}$ sequences randomly
Apply MDR rule for each chromosome
Repeat
I=0
$\mu = 0$
While $(\mu < \frac{P_{size}}{2})$ do
    Select two parents sing selection scheme,
    Apply crossover operator to generate two childs using
    $P_{cross}$ probability value
    Muted the obtained Childs according to the $P_{mut}$ probability
    $\mu = \mu + 1$
End While
    Evaluate all chromosomes ($2P_{size}$, parents and childs) using the fitness function
    Arranged parents and childs in decreasing order using their fitness function
    Save the best $P_{size}$ chromosomes in $Best_{pop}$ and remove the rest set of chromosomes
    Replace $P_{size}$ with $Best_{pop}$
    $i=i+1$
Until $i=Itmax$
```

4.2 SA components for SMTWT problem

4.2.1 Cost Function

Each scheduling problem has its own cost function to evaluate input sequence. A cost function should be explicitly defined as a measure the quality of each sequence of jobs. In our study, a cost function used is the total weighted tardiness (*TWT*) of the *n* jobs in the considered sequence.

4.2.2 Initial Solution

The generation of an initial solution is an important step towards getting a final improved solution. In our developed method, the initial solution is constructed using the same strategy which is described in section 4.1.3 cited above.

4.2.3 Neighborhood Generation Strategy

Basically, all the move sets are related to change the jobs position in the sequences to generate a new solution. In this study, we choose to apply the interchange neighborhood strategy. This mechanism consists of all permutations that can be obtained by swapping two elements at the *i*th and *j*th position ($i \neq j$) regardless of their adjacency. The size of this neighborhood is $n.(n-1)/2$.

4.2.4 Cooling Schedule

The cooling schedule of the simulated annealing algorithm consists of three essential components: initial temperature, final temperature and the temperature decrement scheme. The initial temperature must be hot enough to allow a move to almost any neighbourhood state. For the final temperature, it is usual to let the temperature decrease until it reaches zero. However, this can make the algorithm run for a lot longer. One way to decrement the temperature is the simplest geometric decrement rule which is defined by $T_{k+1} = \alpha.T_k$. This scheme decreases the temperature value by an α factor, which does a range of $0.70 \leq \alpha < 1.0$. In our study, both constant α, initial temperature T_{initial} and final temperature T_{final} values are fixed experimentally.

The pseudo–code of our SA is given below :

Pseudo-code of SA algorithm

```
Initialize $T_{\text{initial}}$, $T_{\text{final}}$ and $\alpha$
Generate an initial solution $S_{\text{initial}}$
Create $S_{\text{current}}$ from initial solution $S_{\text{initial}}$
Calculate TWT of $S_{\text{current}}$
$S_{better} \leftarrow S_{\text{current}}$
$TWT_{better} \leftarrow TWT(S_{\text{current}})$
$T = T_{\text{initial}}$
Setting $L_{\text{cm}}$
While ($T > T_{\text{final}}$) do
n = 1
While ($L_{\text{cm}} > n$) do
   Create $S_{\text{new}}$ using interchange neighborhood strategy
   Calculate TWT of $S_{\text{new}}$
   difference $\leftarrow$(TWT($S_{\text{new}}$) - TWT($S_{\text{current}}$))
   If (difference $\Leftarrow$ 0) then
      $S_{\text{current}} = S_{\text{new}}$
      If  TWT($S_{\text{new}}$) < TWT($S_{\text{better}}$) then
        $S_{\text{better}} = S_{\text{new}}$
        $TWT_{better} \leftarrow TWT(S_{\text{new}})$
     end if
  else
   Boltzmann probability = exp(−difference/$T$)
   If (Boltzmann probability) > random(0,1) then
      $S_{\text{current}} = S_{\text{new}}$
   end if
 end if
$n = n + 1$
End while
$T = \alpha.T$
Increase $L_{\text{cm}}$
End while
Return the optimal sequence $S_{better}$ and its cost $TWT_{better}$
```

After the description of the different components of the proposed model, the pseudo-code of GASA procedure is summarized as follows :

Pseudo-code of GASA algorithm

```
1. Fix the number of generations GMAX
2. Fix the population size $P_{size}$
3. Initialize the initial population randomly;
4. g$\leftarrow$ 1
5. While (g $\leq$ GMAX ) do
   5.1. Create GA-population and SA-population randomly
      // Update chromosomes character using genetic operators.
   5.2.  Apply GA (GA-population)
      // Improve locally each chromosome
   5.3. Apply SA (SA-population)
      // Reconstruct the whole population
   5.4. Combine GA-population and SA-Population
   5.5.  g $\leftarrow$ g + 1
```

```
6. End while
// Optimal sequence and the best cost.
7. Return the chromosome corresponding to the best cost value.
```

5. Experimental Results

The developed hybrid approach GASA presented to solve the SMTWT problem is coded in JAVA language and run on an Intel Pentium IV 2.6GHz PC with 1GB memory. We tested the performance of our proposed algorithm using the benchmark set of randomly generated instances as follows :

1. Each job j has an integer processing time p_j in the uniform distribution (1,100), an integer weight w_j in the uniform distribution (1,10).

2. The range of due dates (RDD) and the tightness factor of due dates (TF) values are selected from the set {0.2, 0.4, 0.6, 0.8, 1.0} respectively.

3. The due date d_j of each job i is selected in the uniform distribution (P*(1-TF-RDD/2), P*(1-TF+RDD/2)), where:

4. The size of the considered problems is n=25, 50 and 100 jobs, five instances have been generated

$$P = \sum_{i=1}^{n} P_i$$

for each of the 25 combinations of values of RDD and TF. Thus, our used dataset is composed of 375 problems. In addition, all key parameter values of our developed GASA algorithm are reported in table 1.

TABLE 1. PARAMETER SETTINGS FOR EXPERIMENTS.

Parameter	**Value**
Population size (P_{size})	*100*
Number of iteration ($Iter_{max}$)	*1000*
Crossover rate (P_{cross})	0.70
Mutation rate (P_{mut})	0.01
Initial temperature ($T_{initial}$)	*100*
Final temperature (T_{final})	*0.0001*

To validate the performance of the proposed method GASA, a set of comparison results with the genetic algorithm presented in [20] are summarized in table 2, 3 and 4. In this tables, columns 1 and 2 present the RDD and TF values. Columns 3 and 4 provide the average values of the total weighted tardiness for the genetic algorithm (GA) and our proposed approach GASA respectively. The last column, records the percentage relative improvement (*PRI*) values. *PRI* is calculated by the following equation [25]:

$$PRI(\%) = \frac{(TWT_{GA} - TWT_{GASA})}{TWT_{GA}} * 100 \tag{2}$$

where TWT_{GA} is the total weighted tardiness obtained by GA algorithm, TWT_{GASA} is the total weighted tardiness obtained by our developed GASA algorithm.

TABLE 2. COMPARATIVE RESULTS OF 25 JOBS.

RDD	TF	GA[20]	Our GASA	PRI (%)
0.2	0.2	485	**210**	56.70
	0.4	3292	**2206**	32.99
	0.6	13454	**11502**	14.51
	0.8	26167	**21945**	16.13
	1.0	37451	**33204**	11.34
0.4	0.2	213	**157**	26.29
	0.4	2339	**2013**	13.94
	0.6	12312	**10536**	14.42
	0.8	23579	**21618**	8.32
	1.0	29654	**24806**	16.35
0.6	0.2	0	**0**	-
	0.4	909	**714**	21.45

	0.6	9801	**6312**	35.60
	0.8	15871	**11468**	27.74
	1.0	28687	**25307**	11.78
0.8	0.2	0	**0**	-
	0.4	658	**533**	19.00
	0.6	8251	**7159**	13.23
	0.8	13458	**11201**	16.77
	1.0	21790	**17805**	18.29
1.0	0.2	0	**0**	-
	0.4	0	**0**	-
	0.6	3283	**2604**	20.68
	0.8	13017	**12116**	6.92
	1.0	18423	**12508**	32.11
			Average	**20.69**

TABLE 3. COMPARATIVE RESULTS OF 50 JOBS.

RDD	TF	GA[20]	Our GASA	PRI (%)
0.2	0.2	1570	**834**	46.88
	0.4	15018	**10903**	27.40
	0.6	35878	**30216**	15.78
	0.8	58769	**47915**	18.47
	1.0	172663	**165457**	4.17
0.4	0.2	106	**82**	22.64
	0.4	7620	**3608**	52.65
	0.6	29149	**21347**	26.77
	0.8	86420	**72146**	16.52
	1.0	130146	**97547**	25.05
0.6	0.2	0	**0**	-
	0.4	3467	**1648**	52.47
	0.6	19137	**13345**	30.27
	0.8	54251	**48201**	11.15
	1.0	121932	**112809**	7.48
0.8	0.2	0	**0**	-
	0.4	2426	**1023**	57.83
	0.6	23354	**19645**	15.88
	0.8	454230	**316407**	30.34
	1.0	89779	**72015**	19.79
1.0	0.2	0	**0**	-
	0.4	0	**0**	-
	0.6	15272	**10314**	32.46
	0.8	34195	**26309**	23.06
	1.0	66424	**54618**	17.77
			Average	**26.42**

TABLE 4. COMPARATIVE RESULTS OF 100 JOBS.

RDD	TF	GA[20]	Our GASA	PRI (%)
0.2	0.2	4881	**1208**	75.25
	0.4	54041	**33142**	38.67
	0.6	141276	**118006**	16.47
	0.8	374016	**201412**	46.15
	1.0	690059	**332045**	51.88
0.4	0.2	89	**52**	41.57
	0.4	25125	**16003**	36.31
	0.6	130173	**11754**	90.97
	0.8	315322	**201623**	36.06
	1.0	525424	**301711**	42.58
0.6	0.2	0	**0**	-
	0.4	11343	**7501**	33.87
	0.6	113384	**100245**	11.59
	0.8	270245	**168204**	37.76
	1.0	407011	**203425**	50.02
0.8	0.2	0	**0**	-
	0.4	4339	**1416**	67.37

	0.6	78918	**62375**	20.96
	0.8	228601	**119205**	47.85
	1.0	347094	**101248**	70.83
1.0	0.2	0	**0**	-
	0.4	0	**0**	-
	0.6	25381	**13546**	46.63
	0.8	100770	**85326**	15.33
	1.0	266852	**124709**	53.27
			Average	**44.35**

According to the results figured in table 2, 3 and 4 above, it is evident that our GASA approach consistently outperforms the results obtained by the genetic algorithm presented in Ref. [20]. In addition, We can show clearly that the average of the percentage relative improvement increase when the problem size increase. This remark is due to the excellent way to discover locally and globally of the space search by our hybrid approach. The *PRI* values is very significant when the number of jobs attain 100 which demonstrate the superior capabilities of our approach to solve a large size problems.

6. Conclusion

In this research work, a hybrid approach based on metaheuristics is proposed to tackle the single machine total weighted tardiness problem. The developed method combines the advantages of GA algorithm ; particularly its potent to diversify the search over the solution space and those of simulated annealing ; as a local improvement approach to intensify the search process in local regions and to allow the hybrid algorithm to escape from local optima. Obtained results using different size problems show the superior capability of our hybrid model compared to others state of the art works.

As a perspective of this work, the improvement of the initial population by another better heuristics is desired. In addition, we can integrate other mechanisms to neighborhood generation step or to use another genetic operators in GA core to improve the quality of the solution. A comparison of the proposed method with some other state-of-the-art techniques such as PSO, SA, tabu search and ant colony is possible to verify its effectiveness.

7. Acknowledgements

The authors like to express their sincere appreciation to the reviewers for their careful and thorough critical appraisement of an earlier draft of the paper.

8. References

[1] E.L. Lawler, A *Pseudopolynomial Algorithm for Sequencing Jobs to Minimize Total Tardiness*, Annals of Discrete Mathematics 1(1977), 331-342.

[2] J.K. Lenstra, K. Rinnooy, P. Brucker, *Complexity of Machine Scheduling Problems*, Annals of Discrete Mathematics 1(1977), 343-362.

[3] K. Wanatchapong, *Solving Single Machine Total Weighted Tardiness Problem Using Gaussian Process Regression*, Int. Journal of Mathematical, Computational, Physical and Quantum Engineering 8 (2014), pp. 938-944.

[4] T.S. Abdul-Razaq, C.N. Potts, L.N. Van Wassenhove, *A Survey of Algorithms for the Single Machine Total Weighted Tardiness Scheduling Problem*, Discrete Applied Mathematics 26 (1990), pp. 235-253.

[5] T. C. E. Cheng, C. T. Ng, J. J. Yuan, and Z. H. Liu, *Single machine scheduling to minimize total weighted tardiness*, *Eur. J. Oper. Res*earch 165 (2005), pp. 423-443.

[6] W. Boejko, J. Grabowski, and M. Wodecki, *Block approach—tabu search algorithm for single machine total weighted tardiness problem*, Comput. Ind. Eng. 50 (2006), pp. 1-14.

[7] U. Bilge, M. Kurtulan, and F. Kýraç, *A tabu search algorithm for the single machine total weighted tardiness problem*, Eur. J. Oper. Research 176 (2007), pp. 1423-1435.

[8] A. C. Nearchou, *Solving the single machine total weighted tardiness scheduling problem using a hybrid simulated annealing algorithm*, Proc. of the 2nd IEEE Int. Conf. Industrial Informatics, pp. 513-516, 2004.

[9] O. Holthaus and C. Rajendran, *A fast ant-colony algorithm for single machine scheduling to minimize the sum of weighted tardiness of jobs*, *J. Oper. Res. Soc.* 56 (2005), pp. 947-953.

[10] R. K. Congram, C. N. Potts, and S. L. Van de Velde, *An iterated dynasearch algorithm for the single-machine total weighted tardiness scheduling problem*, *Informs J. Comput.* 14 (2002), pp. 52-67.

[11] N. Liu, M. Abdelrahman, and S. Ramaswamy, *A genetic algorithm for single machine total weighted tardiness scheduling problem*, Int. J. Intelligent Control and Systems 10 (2005), pp. 218-225.

[12] J. Wang and L. Tang, *A population-based variable neighborhood search for the single machine total weighted tardiness problem*, Comput. Oper. Research 36 (2009), pp. 2105-2110.

[13] C. N. Potts and L. N. Van Wassenhove, *Single machine tardiness sequencing heuristics*, *IIE Transactions* 23 (1991), pp. 346-354.

[14] B. Alidaee and K. R. Ramakrishnan, *A computational experiment of COVERT-AU class of rules for single machine tardiness scheduling problem*, Computers and Industrial Engineering 30 (1996), pp. 201-209.

[15] H.A.J. Crauwels, C.N. Potts, L.N. Van Wassenhove, *Local Search Heuristics for Single Machine Total Weighted Tardiness Scheduling Problem*, Informs Journal on Computing 10 (1998), pp. 341-350.

[16] M. Ana, R. Carlos, D.C.S. Silvio, *A GA Based Scheduling System for Dynamic Single Machine Problem*, Proc. of the 4th IEEE International Symposium on Assembly and Task Planning, Soft Research Park, Fukuoka, Japan, pp. 262-267, 2001.

[17] M.F. Tasgetiren, S. Mehmet and Y.L.G. Gencyilmaz, *Particle Swarm Optimization Algorithm for Single Machine Total Weighted Tardiness Problem*, Proc. of the 2004 Congress on Evolutionary Computation, pp. 1412-1419, 2004.

[18] M.j. Geiger, *On heuristic search for the single machine total weighted tardiness problem – some theoretical insights and their empirical verification*, European Journal of Operation Research 207 (2010), pp. 1235-1243.

[19] S. Sabamoniri, K. Asghari, M.J. Hosseini, *Solving Single Machine Total Weighted Tardiness Problem Using Variable Structure Learning Automata*, Int. J. Comput. Appl. 56 (2012), pp. 37-42.

[20] Y. Suppiah, KP. Shen, *Minimizing total weighted tardiness on single machine*, international Journal of Research in Science Engineering and technology 2 (2015), pp. 12-18.

[21] J. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*, The University of Michigan Press, 1975.

[22] C.-J. Hsu, Y.-J. Yang, and D.-L. Yang,, *Due-date assignment and optimal maintenance activity scheduling problem with linear deteriorating jobs*, Journal of Marine Science and Technology 19 (2011), pp. 97-100.

[23] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, *Optimization by Simulated Annealing*, Science 220 (1983) , pp. 671-680.

[24] A. Ying and J. Wang, *Mixed Dispath rule for single machine total weighted tardiness problem*, Journal of Applied sciences 13(2013), pp. 4616-4619.

[25] F. Jin, S. song and C. Wu, *A simulated Annealing algorithm for single machine scheduling problems with family setups*, Computers and Operations Research 36 (2009), pp. 2133-2138.