International Journal of Industrial Engineering & Production Research (2010) pp. 89-95

September 2010, Volume 21, Number 2

International Journal of Industrial Engineering & Production Research

ISSN: 2008-4889

http://IJIEPR.iust.ac.ir/

Minimizing Total Weighted Tardiness with Drop Dead Dates in Single Machine Scheduling Problem

Mohammad Mahdavi Mazdeh*, Ali Khan Nakhjavani & Abalfazl Zareei

M. Mahdavi Mazdeh, *Assistant professor of Industrial Eng. Department, Iran University of Science and Technology.*
A. Nakhjavani, *BSC student of Industrial Eng. Department, Iran University of Science and Technology.*
A. Zareei, *BSC student of Industrial Eng. Department, Iran University of Science and Technology*

KEYWORDS

Total Weighted Tardiness,
Single machines,
due date,
heuristic algorithms

ABSTRACT

This paper deals with minimization of tardiness in single machine scheduling problem when each job has two different due-dates i.e. ordinary due-date and drop dead date. The drop dead date is the date in which jobs' weights rise sharply or the customer cancels the order. A linear programming formulation is developed for the problem and since the problem is known to be NP-hard, three heuristic algorithms are designed for the problem based on Tabu search mechanism. Extensive numerical experiments were conducted to observe and compare the behavior of the algorithms in solving the problem..

1. Introduction

The Problem of minimizing total weighted tardiness of jobs on a single machine is as follows: there are *n* jobs to be processed; each job has a processing time (P_i), a weight (w_i) and a due date (d_i). The objective is to minimize the total weighted tardiness of jobs ($\sum w_i T_i$). The problem belongs to the class of NP-hard problems. The total tardiness problem with ordinary due-date on a single machine has received considerable attention recently and has been shown to be NP-complete. Many researches have been done to solve the static problem. Based on Emmons [2], the shortest processing time (SPT) sequence is optimal if all jobs have positive tardiness and the earliest due date (EDD) sequence is optimal if no more than one job has positive tardiness. In Lawler [3] a pseudo-polynomial algorithm for solving the problem is proposed under the assumption of agreeability of the weighting of jobs with the processing time.

Corresponding author. Mohammad Mahdavi Mazdeh
Email: *mazdeh@iust.ac.ir*
Paper first received Jan. 17. 2010, and in revised form Oct. 27. 2010.

In Koulamas [5], the latest theoretical developments for single-machine total tardiness are reviewed and some extensions to the most recent works have been proposed.

Potts and Van Wassenhove [4] have developed an efficient algorithm that can solve problems up to 100 jobs based on decomposing the problem into sub-problems that are solvable by dynamic programming because they are sufficiently small.

Grosso *et al.*[7] developed an enhanced dynasearch neighborhood obtained by the generalized pairwise interchange (GPI) operators which provides a fast search procedure using also elimination criteria for the single-machine total weighted tardiness scheduling problem.

Chou [8] proposed a jigsaw algorithm to solve the single-machine total weighted tardiness problem with sequence dependent setups. A jigsaw intelligent genetic algorithm (JIGA) uses a jigsaw heuristic as an immigration mechanism in the evolution of a genetic algorithm.

Angel and Bampis [12] considered a single machine total weighted tardiness problem where processing times of jobs are a function of their starting times. They proposed a multi-start dynamic search algorithm to solve the problem.

Anghinolfi and Paolucci [13] considered the single machine total weighted tardiness with sequence-dependent setup times and they proposed a new particle swarm optimization approach to solve the problem.

Akturk and Ozdemir [14] proposed a new dominance rule for the problem of minimizing total weighted tardiness with unequal release dates. They proposed an algorithm based on dominance rule and showed that the proposed algorithm dominates the competing algorithms in all runs.

Tasgetiren *et al.* [15] developed a discrete differential evolution algorithm for the single machine total weighted tardiness problem with sequence dependent setup times. They also enhanced its performance by employing different population initialization schemes.

Kolliopoulos and Steiner [16] designed a pseudo-polynomial algorithm for the weighted problem of minimizing tardiness on a single machine with a fixed number of due dates They provide a quasi-polynomial algorithm For the case with an arbitrary number of due dates and polynomially bounded processing times. They also investigated the performance of algorithms for minimizing the related total weighted number of late jobs.

Bozejko [17] proposed a tabu search algorithm with a specific neighborhood which employs blocks of jobs and a compound-moves technique for the problem. They defined compound-move as a move consisting of several moves performed simultaneously in a single iteration of algorithm which helps the acceleration of the convergence to good solutions.

Bigle *et al.*[18] proposed a deterministic TS algorithm with a hybrid neighborhood and dynamic tenure structure. The proposed TS attains the best performance by employing a candidate list strategy based on problem context, a dynamic tenure structure and an intensification phase around elite solutions after the short-term memory.

Kellerer and Strusevich [19] developed a fully polynomial-time approximation scheme (FPTAS) for minimizing the weighted total tardiness on a single machine with common due-dates. Zhou *et al.* [6] proposed a hybrid genetic algorithm for solving total tardiness problem in job shop environment.

Wang and Tang [20] proposed a population-based variable neighborhood search (PVNS) algorithm to solve the problem of single machine total weighted tardiness problem. Their algorithm outperforms the basic variable neighborhood search (VNS) in the way that, in each iteration, a population of solutions is used to simultaneously generate multiple trial solutions in a neighborhood and also the PVNS adopts a combination of path-relinking, variable depth search and tabu search to act as the local search procedure so as to improve the search intensification.

Chou [21] proposed an experienced learning genetic algorithm (ELGA) to solve the single machine total weighted tardiness problem. In the proposed ELGA, an updatable position–job and a job–job matrix is used to build the relationships between jobs and positions according to information on the genes of chromosomes in the generation. They used an experienced learning (EL) heuristic to produce chromosomes for the Genetic Algorithm.

In recent years, researchers in scheduling are moving toward solving problems closer to real-world circumstances. Tardiness minimization problem focuses on minimizing the positive or absolute value of the difference of completion time and ordinary due-date of jobs ($|C_i - d_i|$). The producer pays the penalty caused by this type of tardiness. The question here is to what extend will the customer wait for completion of the jobs? In real-world situation, the customer waits a predetermined time after ordinary due-date and afterwards, he or she will cancel the order or increase the tardiness penalty.

In this paper, two different due-dates are considered in single machine scheduling problem *i.e.* ordinary due-date and drop dead date. After ordinary due-date (d_{1j}-type due date), the producer has to pay delay penalty. After drop dead date (d_{2j}-type due date), the job is dead; this means that the customer will cancel the order or the delay penalty will increase. If the customer cancels the order, the producer should pay for inventory holding until he can find a new customer. It is assumed that the product is not perishable and also the delay cost after drop dead date is more than the delay cost between ordinary due-date and drop dead date *i.e.* $w_{2j} > w_{1j}$. The total tardiness cost of job j is shown in Fig 1. Z_j is the total weighted tardiness. As presented in Fig.1, the total weighted tardiness of job j increases more sharply after drop dead date (d_{2j}).

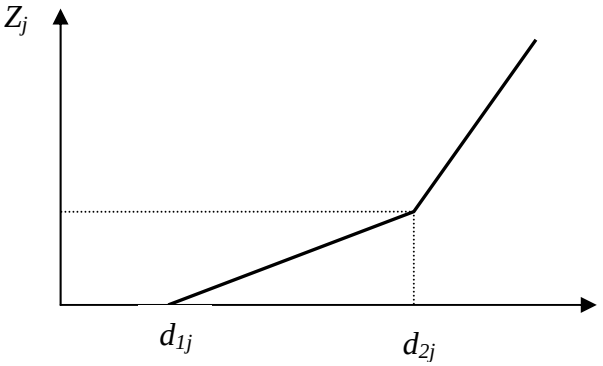


Fig.1. Total tardiness of job *j*

The problem presented in this paper is a generalization of weighted tardiness scheduling problem which is known to be NP-hard [1]. In order to solve the problem, three different algorithm based on Tabu search mechanism are designed. The difference between these algorithms is in their neighborhood generation mechanisms. Extensive numerical cases

have been conducted to present and compare the performance of the algorithms.
The paper is organized as follows. In section 2, the mathematical model for the problem is presented. In section 3, the algorithms based on Tabu search mechanism are presented. In section 4, the numerical analysis is presented and finally through section 5, conclusion and future research are presented.

2. Mathematical Model

The problem is to schedule N jobs (j =1, 2,…, N) on a single machine. Two due dates are considered for each job *i.e.* d_{1j}, d_{2j}. Job j is considered tardy after d_{1j} and the penalty (w_{1j}) is paid for tardiness. d_{2j} is the date the customer will cancel the order and w_{2j} is the cost paid for inventory holding, finding a new customer or in the case that the product is perishable and no new customer is found, it would be the manufacturing cost. The assumptions for the problem are as follows.

Assumptions:

The jobs are available at time zero
The jobs are independent of each other
Each job is processed only once on the machine
No job preemption is allowed
Job processing times and due dates are known at time t = 0.

Parameters:

N total number of jobs to be scheduled,
P_j processing time of job j,
C_j completion time of job j
d_{1j} ordinary due-date
d_{2j} drop dead date

Decision Variables:

$x_{ij} = \begin{cases} 1 & \text{if job j is assigned to } kth \text{ position,} \\ 0 & \text{otherwise,} \end{cases}$
T_{1j} tardiness related to ordinary due-date
T_{2j} tardiness related to dead-due date

The model:

$$Min\ Z = \sum_{j=1}^{N} w_{1j} T_{1j} + \sum_{i=1}^{N} w_{2j} T_{2j} \quad (1)$$

S.t.

$$\sum_{i=0}^{N+1} x_{ij} = 1 \qquad j = 1, 2, \ldots, N+1 \ \ and \ \ i \neq j \quad (2)$$

$$\sum_{j=1}^{N+1} x_{i-1\,j} = 1 \qquad i = 1, 2, \ldots, N+1 \ \ and \ \ i \neq j + \quad (3)$$

$$\sum_{i=1}^{N+1} x_{i0} = 0 \quad (4)$$

$$C_j + M(1 - x_{ij}) \geq C_i + P_j \quad (5)$$

$$T_{1j} = C_j - d_{1j} \qquad j = 1, 2, \ldots, N \quad (6)$$

$$T_{2j} = C_j - d_{2j} \qquad j = 1, 2, \ldots, N \quad (7)$$

$$T_{1j} \geq 0 \quad (8)$$

$$T_{2j} \geq 0 \quad (9)$$

$$x_{ij} \in \{0, 1\} \quad (10)$$

$$p_0 = 0,\ p_{n+1} = 0,\ C(0) = 0 \quad (11)$$

$$d_{1(1)} = d_{2(1)} = M \quad (12)$$

$$w_{1(1)} = w_{2(1)} = M \quad (13)$$

Eq.(1) consists of two terms. The first term represents the total weighted tardiness related to d_{1j}-type due date and the second term represents the total weighted tardiness related to d_{2j}-type due date. Eq. (2) ensures that always one job, job i, is assigned before job j. Eq. (3) states that each job (except the last dummy job), immediately followed by one job, job j. Eq. (4) ensures that a dummy (i=0), job is positioned in the first sequence position. Eq.(5) determines the completion time of job j. The tardiness values of d_{1j}-type and d_{2j}-type due-dates are acquired by Eq.(6) and (7), respectively. Eq. (8) and (9) assure the non-negativity of tardiness values. Eq.(10) impose the binary constraint. Finally, in Eq. (11), (12) and (13) the basic assumptions of the model are presented.

2.1. An Example

In this section, an example is solved using optimization software LINGO ® [19]. It demonstrates that the optimal solution of the model leads to the job scheduling scheme that minimizes total weighted tardiness. The example data values are given in Table 1.

Tab.1. Example data values

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
d_{1j}	0	3	4	5	2	10	11	12	8	9	10	10000
d_{2j}	0	7	7	8	6	12	14	15	11	13	15	10000
P_j	0	3	4	9	6	7	9	7	4	10	14	0
w_{1j}	1000	4	3	5	2	9	3	5	6	5	4	0
w_{2j}	1000	14	11	10	15	20	9	10	12	13	11	0

Solving the example will lead to a total weighted tardiness value (Z) equal to 148 and optimal job sequence as shown in Fig.2. Note that job 1 and 12 are dummy jobs.

1	6	9	7	2	10	11	4	3	5	8	12

Fig.2. Optimal sequence of jobs

The above problem which is a generalization of weighted tardiness scheduling problem is NP-hard since weighted tardiness scheduling problem is known to be NP-hard (Lenstra et al.[1]). In order to solve the problem three different algorithms based on Tabu search mechanism are developed and presented in section 3.

3. Tabu Search

Tabu search (TS) is one of the most effective meta-heuristics used to find near optimal solutions by overcoming the issue of local optimality [9, 10, 11]. TS employs different techniques in achieving near optimal solutions. One is constructing neighborhoods which play an important role in TS. Another one is Tabu list which is a list of prohibited moves during TS run in order to escape from local optimum. Determining moves in TS in order to generate neighborhoods is of great importance in TS. In TS, an arc indicates the neighborhood relationship before and after a move on a sequence. The definition of arcs is important in TS since different arcs can be achieved from a move and selecting a compatible arc's definition will result in an efficient TS.

A feasible solution of the presented problem is a selection of jobs in string-representation, as shown in Fig.3.

J[1]	J[2]	J[3]	.....	J[N]

Fig. 3. A feasible Solution

Where J[i] represents the job in *i*th position.

Starting solution: The starting solution is the EDD order of jobs based on d_2-type due date. w_{2j} is greater than w_{1j}, therefore the EDD order of jobs based on d_2-type due dates will usually result in less objective value.

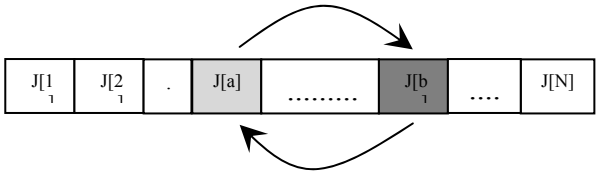
Neighborhood generation: In order to generate the neighborhood of current solution, two different moves are considered: (1) Swamp move and (2) Job insertion. In swamp move the positions of two selected jobs are changed, as shown in Fig.4. In job insertion, a job is selected and moved to a selected position (Fig.5).

Arcs Definition: The arcs definition is of great importance in TS. Two different definitions for arcs are chosen in this paper. An arc in single machine scheduling problem is defined as (*i, j*) in which based on the move used in TS, *i* and *j* have different meanings.

Job swamp: *i* and *j* are the *i*th and *j*th jobs.

Job insertion: *i* is the *i*th job and *j* is the aimed position.

Job swamp and job insertion: Since both moves are considered in TS two independent Tabu lists are used, one for the job swamp move and the other for the job insertion move. The resulted arc of a move will be checked with its Tabu list to see whether it is in the list or not.



Befor Move

After Move

Fig.4. Job swamp

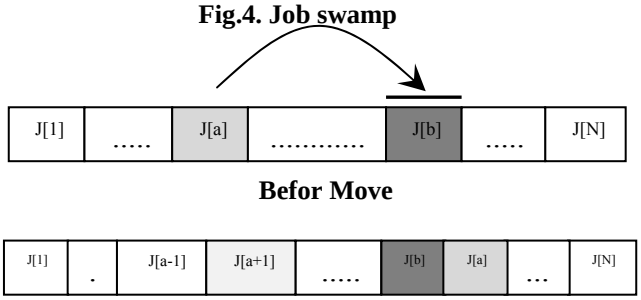


Fig.5. Job insertion

Aspiration Criterion: To measure the level of aspiration, the objective value Z(x) is used. If a move is in the Tabu list (Tabu active) while its corresponding objective value passes the given criterion, then the move will be performed.

Tabu tenure modification rule. Tabu tenure for an arc is the number of iterations through which the arc remains in Tabu list. The number of iterations in which an arc stays in Tabu list is shown by TL. Note that the arc may leave the Tabu list before TL iterations if that arc meets the aspiration criterion.

Stopping Criterion: The Tabu search procedure stops after a predetermined number of iterations. This number is normally decided before starting the search process with reference to the problem size.

Based on the above description of Tabu search concepts, three different algorithms are designed which differ in their neighborhood generation methodology. They are based on using different moves in Tabu search.

TS-Swamp: Job-swamp move is used in this Tabu search.

TS-Insertion: Job-insertion move is used in this Tabu search.

TS- General: Both moves, Job-swamp and job insertion, are applied to this Tabu search.

In the following, the short term procedure of Tabu search is described.

Step1. Initialization

Generate starting solution (EDD order of jobs based on d_2 -type due date)

Set Tabu list (or Tabu lists for General-TS) empty

Let Z_1 and S_1 calculated from initial solution be the best objective value and sequence, respectively, as below:

$$Z^{best} = Z_1 \quad \text{and} \quad S^{best} = S_1$$

Step2. Main procedure

Let S^i be the generated neighborhood in i^{th} iteration with objective value Z^i, and Z^{best} be the best objective value found until $(i-1)^{th}$ iteration.

If the move $S^{best} \rightarrow S^i$ is not Tabu and $Z^i < Z^{best}$,
or the move $S^{best} \rightarrow S^i$ is Tabu and Z^i meets the aspiration criterion ($Z^i < Z^{best}$),
then let $S^{best} = S^i$ and $Z^{best} = Z^i$.

Else if the move $S^{best} \rightarrow S^i$ is tabu and $Z^i > Z^{best}$ then add the move's arc to the Tabu list.

The Tabu lists follow the Tabu modification tenure modification rule.

Step3. Termination

If the stopping criterion is reached, stop.

The above algorithm was coded using VBA. In the next section, several numerical cases are provided and solved using the presented algorithm.

4. Numerical Analysis

In order to observe the behavior of TS for solving the problem, extensive numerical experiments were conducted. The parameter values (i.e. processing times, d_1-type due dates, d_2-type due dates and their weights w_1 and w_2) for each experiments were generated randomly using the probabilistic patterns presented in Table 2.

The probabilistic pattern of d_2-type due date is designed in the way to be definitely greater than d_1-type due-date. The value of w_{2j} is assumed to be equal to $3 \times w_{1j}$ which shows the importance of the d_2-type due-date compared to d_1-type due-date.

Based on the number of jobs, three different sizes of problems are considered for the experiments. They are small size (20 jobs), medium size (50,100 jobs) and large size (200, 250 jobs). Since the selection process is probability-based, each example is solved 30 times and the average values are considered for comparison.

Tab. 2. Probabilistic patterns for parameters

Parameter	Probabilistic patterns
P_i	Uniform [1,a]
d_1 -type due-date	$\sum_{j=1}^{n} p_j / 2 + \text{Uniform}\left[-\sum_{j=1}^{n} p_j / 4, \sum_{j=1}^{n} p_j / 4\right]$
d_2 -type due-date	$3(\sum_{j=1}^{n} p_j)/4 + \text{Uniform}\left[0, \sum_{j=1}^{n} p_j / 2\right]$
w_{1j}	Uniform[1,b]
w_{2j}	$3 \times w_{1j}$

The result of applying Tabu search to different generated problems is presented in Table 3. The Tabu search solution is run 10 times for each instance and the results are presented in Table 3.

Tab. 3. The average values of algorithms results

No. Jobs	TS-Swamp	TS-Insertion	TS-General
20 (1000 iter)	1010.6	1000.4	**964.5**
50 (3000 Iter)	44549.4	44319.3	**44110.3**
100 (5000 iter)	376873.5	375866.9	**375238.7**
200 (6000 iter)	1946050.8	1933538.7	**1929320.9**
250 (8000 iter)	2289096.3	2285938.5	**2280358.7**

Fig.6 shows the best objective value in different iterations for TS-Swamp in solving the 20-jobs problem among the 10 times the program is run. The TS-Swamp finds its objective value (1000.4) in the 781th iteration and stays in this value up to the 1000th iteration. Fig.6. briefly illustrates the step by step attempt of TS-Swamp towards achieving the near optimal solution.

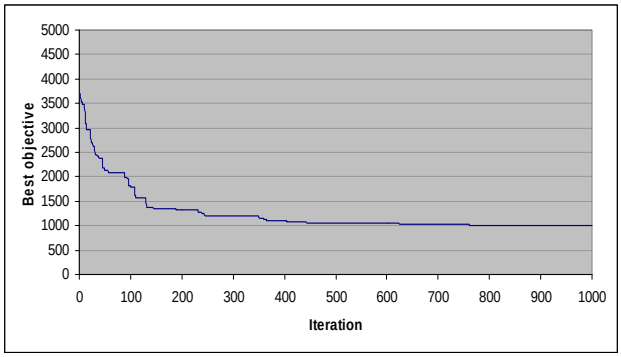


Fig. 6. TS-Swamp (1000 iter)

In Fig.7, the best objective value in different iterations for TS-Insertion in solving the 20-jobs problem among the 10 times the program is run is presented. The TS-Insertion finds its objective value (1010.6) in 435th iteration and stays in this value up to 1000th iteration.

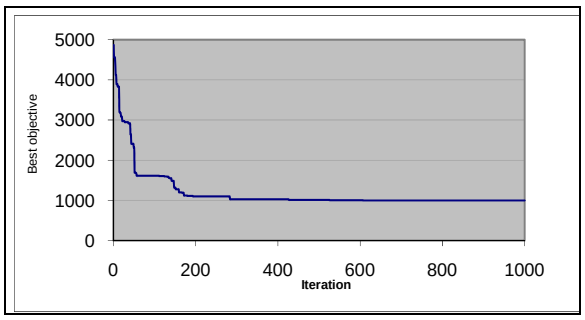


Fig.7. TS-Insertion (1000 iter)

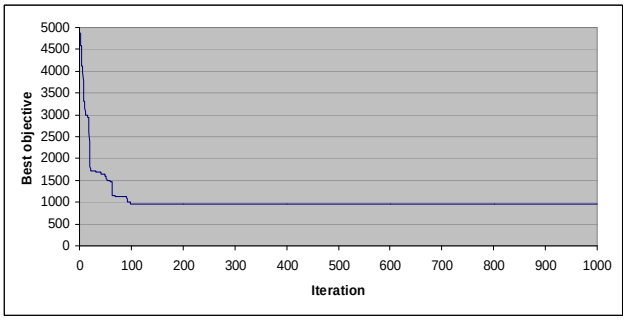


Fig. 8. TS-General (1000 iter)

In Fig.8, the best objective value in different iterations for TS-General in solving the 20-jobs problem among the 10 times the program is run is presented. It is obvious that TS-General outperforms the other algorithms in finding the best objective value in fewer iterations and TS-General finds its best objective (964.4) in the 101th iteration.

Result discussion:

Fig.9 illustrates the final objective values gained by algorithms. TS-General outperforms the other two algorithms in reaching the best objective value. TS-Insertion outperforms the TS-Swamp.

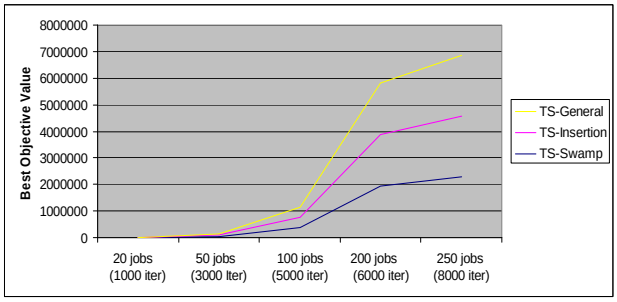


Fig. 9. Comparison on best objectives in final iterations

5. Conclusion and Future Research

This paper discusses the problem of single machine total weighted tardiness. This objective is known as a measure of conformity with due-dates and is used here to minimize due-date related costs. In order to get the problem closer to real world situation, two different due-dates are considered for each job. A linear programming formulation is proposed for solving the problem.

Since the single machine total weighted tardiness is known to be strongly NP-hard, three different algorithms based on Tabu search mechanism are proposed. These algorithms differ in their neighborhood generation methodology. Extensive numerical experiments were conducted in order to observe and compare the algorithms performance.

Future research works can focus on the following:

Considering new conditions such as deterioration and setup time in the problem to make it close to real-world situation.

Using other meta-heuristics such as genetic algorithm to solve the problem and comparing their performances.

Considering other types of Tabu search in order to solve the problem.

References

[1] Lenstra, J.K., Rinnooy, A.H.G., Kan, Brucker, P., *Complexity of Machine Scheduling Problems*, Ann. Discret. Math. 1, 1977, pp. 343–362.

[2] Emmons, H., *One Machine Sequencing to Minimize Certain Functions of Job Tardiness*, Operations Research 17, 1969, pp. 701–715.

[3] Lawler, E.L., *A Pseudo Polynomial Algorithm for Sequencing Jobs to Minimize Total Tardiness*, Annals of Discrete Mathematics 1, 1977, pp. 331–342.

[4] Potts, C.N., Van Wassenhove, L.N., *A Decomposition Algorithm for the Single Machine Total Tardiness Problem*, Operations Research letters 1, 1982, pp. 177–181.

[5] Koulamas, C., *The Single-Machine Total Tardiness Scheduling Problem: Review and Extensions*, European Journal of Operational Research. 1998.

[6] Zhou, H., Cheung, W., Leung, L.C., *Minimizing Weighted Tardiness of Job-Shop Scheduling Using a Hybrid Genetic Algorithm,* European Journal of Operational Research 194, 2009, pp. 637–649.

[7] Grossoa, A., Della Croceb; F., Tadeib, R., *An Enhanced Dynasearch Neighborhood for the Single-Machine Total Weighted Tardiness Scheduling Problem*, Operations Research Letters 32, 2004, pp. 68 – 72.

[8] Fuh-Der Chou, *A Jigsaw Intelligent Genetic Algorithm for Single-Machine*, total weighted tardiness scheduling problems with sequence dependent setups, 2001.

[9] Glover, F., Tabu search—Part 1. ORSA Journal of Computing 1, 1989, pp. 190–206.

[10] Glover, F., Tabu search—Part 2. ORSA Journal of Computing 2, 1990, pp. 4–32.

[11] Glover, F., Kelly, J.P., Laguna, M., *Genetic Algorithms and Tabu Search: Hybrids for Optimization*. Computers and Operations Research 22, 1995, pp. 111–134.

[12] Angel, E., Bampis, E., *A Multi-Start Dynasearch Algorithm for the Time Dependent Single-Machine Total Weighted Tardiness Scheduling Problem*, European Journal of Operational Research 162, 2005, pp. 281–289

[13] Anghinolfi, D., Paolucci , M., *A New Discrete Particle Swarm Optimization Approach for the Single-Machine Total Weighted Tardiness scheduling Problem with Sequence-Dependent Setup Times*, European Journal of Operational Research 193, 2009, pp.73–85.

[14] Selim Akturk, M., Deniz Ozdemir, *A New Dominance Rule to Minimize Total Weighted Tardiness with unequal Release Dates*, European Journal of Operational Research 135, 2001, pp. 394-412.

[15] FatihTasgetirena, M., Quan-KePan, Yun-ChiaLiang, *A Discrete Differential Evolution Algorithm for the Single Machine Total Weighted Tardiness Problem with Sequence Dependent Setup Times,* Computers & Operations Research 36, 2009, pp.1900 – 1915.

[16] Stavros, G., Kolliopoulosa, George Steiner, *Approximation Algorithms for Minimizing the Total Weighted Tardiness on a Single Machine,* Theoretical Computer Science 355, 2006, pp. 261 – 273.

[17] Wojciech Bozejko a, Jo´zef Grabowski , Mieczysław Wodeck, Block approach—tabu search algorithm for single machine total weighted tardiness problem, Computers & Industrial Engineering 50, 2006, pp. 1–14.

[18] Umit Bilge, Mujde Kurtulan, Furkan Kirac, A tabu search algorithm for the single machine total weighted tardiness problem, European Journal of Operational Research 176, 2007, pp.1423–1435.

[19] Hans Kellerer, Vitaly A. Strusevich, A fully polynomial approximation scheme for the single machine weighted total tardiness problem with a common due date, Theoretical Computer Science 369, 2006, pp.230 – 238.

[20] Wang, X., Tang, L., *A Population-Based Variable Neighborhood Search for the Single Machine Total Weighted Tardiness Problem*, Computers & Operations Research 36, 2009, pp. 2105 – 2110.

[21] Chou, F., *An Experienced Learning Genetic Algorithm to Solve the Single Machine Total Weighted Tardiness Scheduling Problem,* Exp. Sys. with Appl., 36, 2009, pp. 3857-3865.