Bol. Soc. Paran. Mat.
©SPM – E-ISSN-2175-1188
SPM: www.spm.uem.br/bspm

(3s.) **v. 2026 (44) 12** : 1–17.
ISSN-0037-8712
doi:10.5269/bspm.83885

Solving Bicriteria Total Tardiness Times and Maximum Tardiness Machine Scheduling Problems

Tamara Sabah Fadhel* and Faez Hassan Ali

ABSTRACT: This research addresses the Single Machine Scheduling Problem (MSP) under two performance criteria: maximum tardiness $T_{\max}$ and total tardiness $\sum T_j$. Given that this problem is classified as NP-hard, the study focuses on developing efficient solution methodologies that balance solution quality with computational efficiency. The MSP is mathematically formulated through two models: a bi-criteria MSP (BCMSP) and a bi-objective MSP (BOMSP) based on minimizing the Euclidean distance. The theoretical framework involves deriving dominance rules (DR) and analytical results for specific cases, enabling efficient (optimal) solutions to be identified directly under certain conditions. On the practical side, exact solution techniques were developed using the Branch and Bound (BAB) method. Furthermore, two heuristic algorithms (HA's) were proposed to handle large-scale instances. Experimental results, involving instances with up to $n \leq 4000$ jobs, demonstrate that the proposed algorithms can find efficient solutions within reasonable computational times.

Keywords: Single machine scheduling, bi-criteria, maximum tardiness, total tardiness, heuristic algorithms, Branch and Bound Method.

Contents

* Corresponding author.
2020 *Mathematics Subject Classification:* 90B35.
Submitted May 01, 2026. Published June 09, 2026.

1. Introduction

Combinatorial Optimization (OP) is one of the most important areas of Operations Research. It is concerned with finding the optimal solution among a finite but very large number of feasible solutions defined on a discrete solution space. Many practical problems arising in industrial, manufacturing, and service systems can be formulated as combinatorial optimization problems (COP's), which explains the increasing interest in this field.

Machine Scheduling Problems (MSP) constitute a fundamental class of COP's. The main objective of MSP is to determine an optimal schedule for processing a set of jobs on one or more machines while satisfying technological constraints and optimizing one or more performance criteria. Single MSP's (SMSP) have been extensively studied in the literature since they represent the core structure upon which more complex scheduling models are built [1].

Koulamas (2009) [2] presented a comprehensive review of the single machine total tardiness problem with objective function ΣT_j. The study analyzed exact algorithms, dynamic programming approaches, and heuristic methods. It was emphasized that exact solution approaches are computationally feasible only for small problem sizes, typically with $n \leq 25$, whereas heuristic and approximation methods are necessary for larger instances. In [3] Toksarı (2016) studied the single MSP with the objective of minimizing the maximum tardiness ($T_{\max}$) under the effects of learning, deterioration, and setup times. An exact BAB algorithm was proposed and tested on problem instances with up to $n = 30$ jobs. The computational results showed that the proposed method is efficient for small and medium-sized instances, while the computational time increases significantly as the number of jobs increases, reflecting the NP-hard nature of the problem. In (2020) Assel et al. [4] they investigate some methods to solve the BCMSP $(1//(\sum C_j, \sum E_j))$, two heuristics are proposed which introduced good results, and BAB method produced exact results for $n \leq 20$ in a reasonable time. Faez et al. (2022) [5], they proposed new techniques to solve BCMSP $(1//(\sum T_j, R_L))$ BOMSP $(1//(\sum T_j + R_L))$, by using new some new exact and approximation methods which are produced good results. The results which are obtained from applying the newly proposed methods are compared with the exact method (CEM), then compared results of the heuristics with each other's to obtain the most efficient method. Hameed et al. (2023) [6] introduce a multi-objective scheduling problem is considered which is the problem $1//(\sum C_j + \sum T_j + \sum E_j + \sum V_j)$. The jobs are divided into families. Two local search methods (genetic algorithm (GA) and particle swarm optimization (PSO)) are used to solve this study's problem. Within each family. The results in which given in tables have showed that the GA and PSO gave values and times near or equal to the values of the known CEM. Five heuristics are compared with the GA and PSO methods where the results are acceptable. In [7] Bouška et al. (2024) proposed a deep learning-driven heuristic algorithm for MSP with the objective of minimizing the total tardiness $\sum T_j$. The proposed approach was evaluated on large-scale instances with up to $n = 800$ jobs. Numerical experiments demonstrated that the algorithm achieved high-quality solutions with very small optimality gaps compared to classical heuristics, while maintaining reasonable computational times. In [8] Heeger et al. (2024) investigated the computational complexity of single MSP involving maximal tardiness and related tardiness measures. The authors proved that several variants combining $T_{\max}$ with tardiness-based objectives are strongly NP-hard. This result further confirms that exact methods are unsuitable for large-scale instances and motivates the development of efficient heuristic and hybrid approaches. Safanha et al. (2025) [9] solving BCMSP and BOMSP by using Swarm Intelligence (AI) represented by PSO. The discussed BCMSP $1//(E_{\max}, R_L)$ and the BOMSP is $1//(E_{\max} + R_L)$. Comparison results of a simulation for exact (CEM and BAB), heuristic method, and simulated annealing with proposed PSO has been made. The results prove the good efficiency of PSO in solving the two problems.

From the reviewed studies, it is clear that minimizing $T_{\max}$ focuses on controlling the worst-case delay, whereas minimizing ΣT_j provides a global performance measure of the scheduling system. Exact methods such as BAB algorithm are effective only for small values of n, while heuristic and learning-based approaches are more appropriate for large-scale problems (see [?]). These observations motivate the present research, which considers the bi-criteria objective function $(T_{\max}, \Sigma T_j)$. Then we derive non-linear multi_objective MSP (NBOMSP) depends on Euclidean distance.

In Section 2, we discuss the most important basic concepts of MSP and some definitions of rules and Pareto optimal with Euclidean distance. While in Section 3, we discuss the dominance rules for MSP.

The mathematical formulation of the proposed MCNSP and MUNSP are presented in Section 4. The special cases for the problems are discussed in Section 5. In Section 6, we suggest exact and heuristic methods for solving the two proposed problems. While in Section 7, we apply the proposed methods to the BC model POMSP. In Section 8, we introduce the analysis and discussion of the results which are obtained in Section 7. Lastly, the conclusion and future works are introduced in Section 9.

2. Basic Concepts of Machine Scheduling Problems

In this section, we will discuss the most important concepts of MSP with $N = \{1, 2, \ldots, n\}$ set of jobs, which represents the most important basic concepts of MSP. Consider a set of n jobs to be processed on MSP [13]:
p_j: which means that job j has to be processed for a period of time.
d_j: due date, the date when the job should be completed. The completion of job after its due date is allowed, but a penalty is incurred, and when d is constant for all jobs, then it is called a common due date (d).
s_j: slack time s.t. $s_j = d_j - p_j$.
C_j: Completion time of job j s.t. $C_j = \sum_{k=1}^{j} p_k$.
L_j: The lateness $L_j = C_j - d_j$.
The tardiness of job j is defined as: $T_j = \max\{C_j - d_j, 0\}$, with maximum tardiness $T_{\max} = \max\{T_j\}$.

Definition 2.1 ((Shortest Processing Time (SPT) rule) [14]) *Jobs are sequencing in non-decreasing order of* p_j *this rule is used to minimize the problem* $1//\sum C_j$.

Definition 2.2 ((Minimum Slack Time (MST) rule) [15]) *In the rule sequencing all jobs in non-decreasing order of slack time* (s)*, this rule solved the problem* $1//E_{\max}$.

Definition 2.3 ((Earliest Dut Date (EDD) rule) [14]) *Jobs are sequencing in non-decreasing order of* d_i*, this rule is used to minimize* $T_{\max}$ *for* $1//T_{\max}$.

Definition 2.4 (Efficient Solution (EF) [16]) *An efficient solution is an ideal solution that cannot be made better. This means there is no other solution that surpasses it in all respects simultaneously. If you try to improve one aspect, you will inevitably notice a deterioration in at least one other aspect.*

Definition (4) (Pareto Optimal (PO)): In MSP, a feasible efficient solution (solution is an optimal solution that cannot be made better [16]) for schedule α is considered PO when we compare its performance according to two criteria, f and g. This means that there is no other feasible schedule π that can achieve performance that is better than or equal to both criteria ($f(\pi) \leq f(\alpha)$ and $g(\pi) \leq g(\alpha)$) without being worse in at least one of them [17,19].

Definition (5) [18]: In m-Euclidean space, the Euclidean distance defined as the length of the straight-line segment connecting any two points. Lets have two points $P_i = (x_{i1}, x_{i2}, \ldots, x_{im}), i = 0, 1$, the distance $D(P_0, P_1)$ between the two points can be calculated as follows:

$$D(P_0, P_1) = \sqrt{\sum_{j=1}^{m} (x_{1j} - x_{0j})^2} \tag{1}$$

If P_0 is the origin point and $P = (x_1, x_2, \ldots, x_m)$ is any point in the Euclidean space, then:

$$D(P) = \sqrt{\sum_{j=1}^{m} x_j^2} \tag{2}$$

3. Dominance Rules for MSP

Several Dominance Rules (DRs) can be used to reduce the current sequence. DRs can be helpful in determining whether or not the obtained schedule is good because they typically specify some (all) of the path to obtain a good value for the objective function. In order to find a more effective schedule to solve the MSP, the DRs are also helpful in reducing computation time. Finally, in order to get a good solution in a reasonable amount of time, the DRs are helpful in canceling numerous nodes in the BAB method [4].

Definition 3.1 ([15]) *: If G is a graph with n vertices, then the adjacency matrix of G is the matrix $A(G) = [a_{ij}]$, whose i^{th} and j^{th} component is 1 if there exists at least one edge between V_i and V_j and zero otherwise, where:*

$$a_{ij} = \begin{cases} 0, & \text{if } i = j \text{ or } i \nrightarrow j, \\ 1, & \text{if } i \rightarrow j, \\ a_{ij} \text{ and } \overline{a}_{ij}, & i \leftrightarrow j. \end{cases}$$

Remark (1): For MSP, we have:
1. If $p_i \leq p_j$ and $s_i \leq s_j$ we obtain $A_1(G)$.
2. If $p_i \leq p_j$ and $d_i \leq d_j$ are obtain $A_2(G)$.

4. Construction the Mathematical Formulation for the New BCMSP and BOMSP

4.1. Construction the Formulation of the New BCMSP

As a part from the theoretical part of this paper, in this section, the construction of mathematical formulation of the suggested BCMSP and the derived BOMSP sub-problem is introduced. The BCMSP considers two tardiness-based objective functions, namely $T_{\max}$ and $\sum T_j$.

The proposed BCMSP is NP-hard, since the single machine problem $1//\sum T_j$ is known to be NP-hard. Consequently, any bi-criteria MSP that includes $\sum T_j$ as one of its objective functions, such as the BCMSP with objective vector $(T_{\max}, \sum T_j)$, is also NP-hard.

Let $\sigma = (1, 2, \ldots, n)$ denote any feasible sequence of n jobs processed on a single machine. Then:

$$\left.\begin{aligned} & F = \min\left(T_{\max}, \sum_{j=1}^{n} T_j\right) \\ \text{s.t.}\quad & C_1 = p_{\sigma(1)} \\ & C_j = C_{j-1} + p_{\sigma(j)}, \quad j = 2, 3, \ldots, n \\ & T_j = \max\left\{C_j - d_{\sigma(j)},\ 0\right\}, \quad j = 1, 2, \ldots, n \\ & T_{\max} = \max\left\{T_j\right\}, \quad j = 1, 2, \ldots, n \\ & T_{\max} \geq 0 \end{aligned}\right\} \quad \text{(BC-TmST)}$$

Proposition 4.1 *The problem* BC-TmST *has an efficient solution for* $\sigma = \text{EDD}$.

Proof: Let σ^* be the EDD rule, and let π be any sequence. Then σ^* is an efficient sequence which gives an efficient solution for the problem BC-TmST, since

$$T_{\max}(\sigma^*) \leq T_{\max}(\pi) \quad \text{and} \quad \sum T_{\sigma^*(j)} \leq \sum T_{\pi(j)}$$

with at least one strict inequality.

That is true because $\sigma^* = \text{EDD}$ gives an optimal solution for $T_{\max}$ and good solution $\sum T_j$; therefore, $\sigma^* = \text{EDD}$ gives an efficient solution for BC-TmST. □

4.2. Construction the Formulation of the New BOMSP

Now we will discuss the mathematical formulation of the proposed BOMSP, which is derived from the main BCMSP, is presented. The BOMSP represents the single-objective version obtained by optimizing the first objective function of the bi-criteria problem.

Accordingly, the objective of the BOMSP is to minimize the maximum tardiness time. The formulation of the BOMSP is defined based on the same constraints used in the BCMSP. This MSP dented by BO-TmST.

This MSP dented by BO-TmST.

Let $\sigma = (1, 2, \ldots, n)$ be any sequence. The objective function of the BOMSP is given as:

$$\left.\begin{aligned} V &= \min \sqrt{(T_{\max})^2 + \left(\sum_{j=1}^{n} T_j\right)^2} \\ \text{s.t.}\quad C_1 &= p_{\sigma(1)} \\ C_j &= C_{j-1} + p_{\sigma(j)}, \quad j = 2, 3, \ldots, n \\ T_j &= \max\left\{C_j - d_{\sigma(j)}, 0\right\}, \quad j = 1, 2, \ldots, n \\ T_{\max} &= \max\left\{T_j\right\}, \quad j = 1, 2, \ldots, n \\ T_j, C_j &\geq 0 \end{aligned}\right\} \quad \text{(BO-TmST)}$$

Proposition 4.2 *Any optimal solution of a* BO-TmST *is an ES of a* BC-TmST *yields an efficient schedule for the original bi-criteria problem.*

Proof: Let σ^* be an optimal solution of a BO-TmST derived from the BC-TmST. Assume, for the sake of contradiction, that σ^* is not an ES for the BC-TmST. Then, $\exists$ a feasible schedule σ s.t.

$$T_{\max}(\sigma) \leq T_{\max}(\sigma^*) \quad \text{and} \quad \sum T_{\sigma^*(j)} \leq \sum T_{\pi(j)},$$

with at least one strict inequality.

This implies that σ dominates σ^*, which contradicts the optimality of σ^*. Therefore, σ^* must be an ES for the BC-TmST. □

5. Special Cases for BC-TmST and BO-TmST Problems

5.1. Special Case For BC-TmST problem

Case (5.1.1): For problem BC-TmST, let σ be the sequence such that

$$C_{\sigma(j)} \leq d_{\sigma(j)}, \quad \forall j.$$

Then

$$\left(T_{\max(\sigma)}, \sum T_{\sigma(j)}\right) = (0, 0).$$

Proof: Since

$$C_{\sigma(j)} \leq d_{\sigma(j)}, \quad \forall j,$$

this means all jobs are early. Then

$$L_j = C_j - d_{\sigma(j)} \leq 0.$$

So,

$$T_{\max}(\sigma) = \max\{T_j\} = 0. \tag{3}$$

And

$$T_{\sigma(j)} = \max\{L_j, 0\} = 0.$$

Then:

$$\sum T_{\sigma(j)} = 0. \tag{4}$$

From relations (1) and (2), we obtain:

$$F_{(\sigma)} = \left(T_{\max(\sigma)}, \sum T_{j(\sigma)}\right) = (0, 0).$$

□

Case (5.1.2): For BC-TmST, if $p_j = p$, then:

1. If $C_j \leq d_j$, then
$$\left(T_{\max}, \sum T_j\right) = (0, 0).$$
2. If $C_j > d_j$, then
$$\left(T_{\max}, \sum T_j\right) = \left(T_{\max}, \frac{pn(n+1)}{2} - \sum d_{\sigma(j)}\right),$$
when $\sigma = $ EDD.

Proof:

1. If
$$C_j = jp \leq d_j, \quad \forall j,$$
then by Case (5.1.1),
$$\left(T_{\max}, \sum T_j\right) = (0, 0).$$
2. If
$$C_j = jp > d_j$$
except
$$p \leq d_{\sigma(1)},$$
this means all jobs are late:
$$T_j = jp - d_j.$$
Hence,
$$\sum T_j = \sum jp - \sum d_{\sigma(j)} = \frac{pn(n+1)}{2} - \sum d_{\sigma(j)}. \tag{5}$$

From relation (3):

$$\left(T_{\max}, \sum T_j\right) = \left(T_{\max}, \frac{pn(n+1)}{2} - \sum d_j\right).$$

Where $\sigma = $ EDD useful for $T_{\max}$ and

$$\frac{p(n+1)}{2} - \sum d_j$$

is fixed.

□

Case (5.1.3): For BC-TmST problem, if $d_j = d, \ \forall j$, then:

1. If $C_j \leq d, \ \forall j$, then
$$\left(T_{\max}, \sum T_j\right) = (0, 0).$$

2. If $C_j > d$, except $p_{\sigma(1)} \leq d$, then

$$\left(T_{\max}, \sum T_j\right) = \left(n - d, \sum_{j=2}^{n} C_j - (n-1)d\right),$$

when $\sigma =$ EDD by using SPT rule.

Proof: If $d_j = d$, then by relations (1) and (2):

1. If
$$C_j \leq d, \quad \forall j,$$
then by Case (5.1.1),
$$\left(T_{\max}, \sum T_j\right) = (0, 0).$$

2. If $d < C_j$, then
$$L_j = C_j - d \geq 0,$$
and
$$T_j = \max\{L_j - d, 0\} = C_j - d.$$
Also,
$$L_j = C_j - d. \tag{6}$$
Hence,
$$T_{\max} = \max\{C_1 - d, C_2 - d, \ldots, C_n - d\} = C_n - d. \tag{7}$$
From relation (4), and taking the sum from $j = 2, \ldots, n$, since $T_1 = 0$:
$$\sum_{j=2}^{n} T_j = \sum_{j=2}^{n} C_j - \sum_{j=2}^{n} d.$$
Therefore,
$$\sum_{j=2}^{n} T_j = \sum_{j=2}^{n} C_j - (n-1)d. \tag{8}$$

From (4) and (5):

$$\left(T_{\max}, \sum T_j\right) = \left(n - d, \sum_{j=2}^{n} C_j - (n-1)d\right).$$

□

Case (5.1.4): For BC-TmST problem, $d_j = d$ and $p_j = p$, then:

1. If $C_j \leq d, \ \forall j$, then
$$\left(T_{\max}, \sum T_j\right) = (0, 0).$$

2. If $C_j > d$, except $C_1 = p \leq d$, then:

   (a) if $p = d$, then
$$\left(T_{\max}, \sum T_j\right) = \left(p(n-1), \frac{pn(n-1)}{2}\right).$$

(b) if $p < d$, then

$$\left(T_{\max}, \sum T_j\right) = \left(np - d, \frac{p(n(n+1)-2)}{2} - (n-1)d\right).$$

Proof:

1. If

$$C_j \leq d, \quad \forall j,$$

then by Case (5.1.1),

$$\left(T_{\max}, \sum T_j\right) = (0,0).$$

2. If $C_j > d_j$, except $C_1 = p \leq d$:

(a) If $p = d$, then

$$L_j = C_j - p = jp - p = p(j-1).$$

Therefore,

$$T_j = \max\{p(j-1), 0\} = p(j-1). \tag{9}$$

Hence,

$$T_{\max} = \max\{0, p, 2p, \ldots, (n-1)p\} = (n-1)p. \tag{10}$$

From relation (7):

$$\sum_{j=2}^{n} T_j = \sum p(j-1) = p\sum (j-1) = \frac{pn(n-1)}{2}. \tag{11}$$

From relations (8) and (9), we obtain:

$$\left(T_{\max}, \sum T_j\right) = \left(p(n-1), \frac{pn(n-1)}{2}\right). \tag{12}$$

(b) If $p < d$, and $jp > d$, $j = 2, 3, \ldots, n$, then

$$L_j = C_j - d = jp - d.$$

Therefore,

$$T_j = \max\{jp - d, 0\} = jp - d. \tag{13}$$

Hence,

$$T_{\max} = \max\{T_j\} = \{p - d, 2p - d, \ldots, np - d\} = np - d. \tag{14}$$

From relation (11), starting from $j = 2$:

$$\sum_{j=2}^{n} T_j = \sum jp - \sum d = \frac{p(n(n+1)-2)}{2} - (n-1)d. \tag{15}$$

By relations (12) and (13):

$$\left(T_{\max}, \sum T_j\right) = \left(np - d, \frac{p(n(n+1)-2)}{2} - (n-1)d\right).$$

Any sequence gives an efficient solution. □

5.2. Special Case For BO-TmST problem

Case (5.2.1): For the BO-TmST problem, let σ be a sequence. If

$$C_{\sigma(j)} \leq d_{\sigma(j)}, \quad \forall j,$$

then all jobs are early.

Proof: Since

$$C_{\sigma(j)} \leq d_{\sigma(j)}, \quad \forall j,$$

it for the sequence σ, then by Case (5.1.1):

$$F = \sqrt{(T_{\max})^2 + \left(\sum T_j\right)^2} = \sqrt{0+0} = 0.$$

□

Case (5.2.2): For problem BO-TmST, if $p_j = p$, $\forall j$:

1. If $C_j \leq d_j$, then
$$F = \sqrt{(T_{\max})^2 + \left(\sum T_j\right)^2}.$$
2. If $C_j > d_j$, then
$$F = \sqrt{(T_{\max})^2 + \left(\frac{pn(n+1)}{2} - \sum d_{\sigma(j)}\right)^2}, \quad \sigma = \text{EDD}.$$

Proof:

1. If $C_j \leq d_j$, by Case (5.2.1), then
$$F = 0.$$
2. If $C_j > d_j$, except $p \leq d_{\sigma(1)}$, then by Case (5.1.2):
$$F = \sqrt{(T_{\max})^2 + \left(\sum T_j\right)^2} = \sqrt{(T_{\max})^2 + \left(\frac{pn(n+1)}{2} - \sum d_{\sigma(j)}\right)^2}.$$

□

Case (5.2.3): For BO-TmST problem, if $d_j = d$, $\forall j$, then:

1. If $C_j \leq d_j$, $\forall j$, then by Case (5.2.1),
$$F = 0.$$
2. If $C_j > d_j$, by using SPT rule, except $p \leq d_{\sigma(1)}$, then
$$F = \sqrt{(n-d)^2 + \left(\sum_{j=2}^{n} C_j - (n-1)d\right)^2}.$$

Proof:

1. If $C_j \leq d_j$, $\forall j$, then by Case (5.2.1),
$$F = 0.$$

2. If $C_j > d_j$, by using SPT rule, except $p \leq d_{\sigma(1)}$, by using Case (5.1.3):

$$F = \sqrt{(T_{\max})^2 + \left(\sum T_j\right)^2} = \sqrt{(n-d)^2 + \left(\sum_{j=2}^{n} C_j - (n-1)d\right)^2}.$$

□

Case (5.2.4): For BO-TmST problem, $d_j = d,\ p_j = p$, then:

1. If $C_j \leq d_j$, then by Case (5.2.1),
$$F = 0.$$

2. If $C_j > d$, then:

   (a) if $p = d$, then
$$\sqrt{(T_{\max})^2 + \left(\sum T_j\right)^2} = \sqrt{(p(n-1))^2 + \left(\frac{pn(n-1)}{2}\right)^2}.$$

   (b) if $p < d$, then
$$\sqrt{(T_{\max})^2 + \left(\sum T_j\right)^2} = \sqrt{(np-d)^2 + \left(\frac{p(n(n+1)-2)}{2} - (n-1)d\right)^2}.$$

Proof:

1. If $C_j \leq d_j$, then by Case (5.2.1),
$$F = 0.$$

2. If $C_j > d$, then:

   (a) if $p = d$, then
$$\sqrt{(T_{\max})^2 + \left(\sum T_j\right)^2} = \sqrt{(p(n-1))^2 + \left(\frac{pn(n-1)}{2}\right)^2}.$$

   (b) if $p < d$, then
$$\sqrt{(T_{\max})^2 + \left(\sum T_j\right)^2} = \sqrt{(np-d)^2 + \left(\frac{p(n(n+1)-2)}{2} - (n-1)d\right)^2}.$$

□

Example (1): For the special cases which are mentioned in Subsections (5.1.1) and (5.2.1), we introduce an example for $n = 4$, as seen in Table 1.
Examples for special cases of BC(BO)-TmST problem.
Remark (2): From Table 1, for cases 4.1.1, 4.2.1, 4.3.1, and 4.1, where the conditions $C_j \leq d_j$ or $C_n \leq d$ are satisfied, the values of both objective functions are equal to zero, i.e.,

$$\left(T_{\max}, \sum T_j\right) = (0, 0).$$

Therefore, the corresponding schedules are both optimal and efficient for the bi-criteria problem. For cases 4.2.2, 4.3.2, 4.2.a, and 4.2.b, although an optimal solution exists for the bounded-objective problem BO-TmST, the obtained schedule is not efficient for the bi-criteria problem $(T_{\max}, \sum T_j)$, since it is dominated by at least one feasible schedule with respect to the two objective functions.

Table 1: Examples of special cases for n=4.

Case	p_i **and** d_j	**Condition**	BC-TmST **Efficient solution**	BO-TmST **Optimal solution**	**Values**
5.1.1 5.2.1	$p_i = 3,\ 5,\ 4,\ 2$ $d_j = 10,\ 12,\ 15,\ 18$	$C_j \leq d_j$	$(0,0)$	0	$(0,0) = 0$
5.1.2(1) 5.2.2	$p_i = 5,\ 5,\ 5,\ 5$ $d_j = 5,\ 10,\ 15,\ 20$	$p_j = p$ $jp \leq d_j$	$(0,0)$	0	$(0,0) = 0$
5.1.2(2) 5.2.2	$p_i = 5,\ 5,\ 5,\ 5$ $d_j = 5,\ 9,\ 14,\ 19$	$p_j = p$ $jp > d_j$	$\left(T_{\max}, \frac{pn(n+1)}{2} - \sum d_j\right)$	$\sqrt{(T_{\max})^2 + (\sum T_j)^2}$	$(1,3) = 3.2$
5.1.3(1) 5.2.3	$p_i = 4,\ 6,\ 3,\ 5$ $d_j = 20,\ 20,\ 20,\ 20$	$d_j = d$ $C_n \leq d$	$(0,0)$	0	$(0,0) = 0$
5.1.3(2) 5.2.3	$p_i = 3,\ 4,\ 5,\ 6$ $d_j = 6,\ 6,\ 6,\ 6$	$d_j = d$ $d < C_j$	$\left(n-d, \sum_{j=2}^{n} C_j - (n-1)d\right)$	$\sqrt{(T_{\max})^2 + (\sum T_j)^2}$	$(12,19) = 22.5$
5.1.4(1) 5.2.4	$p_i = 4,\ 4,\ 4,\ 4$ $d_j = 16,\ 16,\ 16,\ 16$	$p = d$ $C_j \leq d_j$	$(0,0)$	0	$(0,0) = 0$
5.1.4(2)a 5.2.4	$p_i = 4,\ 4,\ 4,\ 4$ $d_j = 4,\ 4,\ 4,\ 4$	$p = d$ $C_j \geq d_j$ Except $C_1 = p \leq d$	$\left(p(n-1), \frac{pn(n-1)}{2}\right)$	$\sqrt{(p(n-1))^2 + \left(\frac{pn(n-1)}{2}\right)^2}$	$(12,24) = 26.8$
5.1.4(2)b 5.2.4	$p_j = 4,\ 4,\ 4,\ 4$ $d_j = 5,\ 5,\ 5,\ 5$	$p < d$ $jp > d,\ j = 2,3,4$	$\left(np-d, \frac{p(n(n+1)-2)}{2} - (n-1)d\right)$	$\sqrt{(T_{\max})^2 + (\sum T_j)^2}$	$(11,19) = 21.9$

6. New Solving Techniques (NST's) for BC(O) – TmST Problems

6.1. New Heuristic Techniques (NHT's) for BC(O) – TmST Problems

In this sub-section, two heuristics for solving BC(O) – TmST problems are proposed.

The first proposal heuristic algorithm is act as follows: for the sequence $\sigma = (1, 2, \ldots, n)$, the first i jobs arranged as EDD rule, the rest jobs arranged in MST rule, where $i = 1, 2, \ldots, n-1$, so, we obtain $\sigma_1, \sigma_2, \ldots, \sigma_n$ sequences. then calculated $F(\sigma_i)$, so $S = \{F(\sigma_1), F(\sigma_2), \ldots, F(\sigma_{n-1})\}$ the set S will be filtered to obtain efficient (optimal) solution(s). This algorithm is called EDD to MST heuristic method for the two problems which is denoted by BC(O)E2MM, with following steps:

Algorithm (1): BC(O)E2MM Algorithm

Step 1: Input: $n - p_j$ and d_j.

Step 2: Set $\sigma = R(1,2) = (EDD, MST)$, $S = \emptyset$.

Step 3: For $k = 1:2$

Step 4: For $i = 2 : n-1$

Step 5: If $k = 1$, then $j = 2$, Else $j = 1$;

Step 6: divided σ into two parts, the first part $[1 : i]$ arranged by $R(k)$ while the second part contains the $[i+1 : n]$ arranged by $R(j)$ rule to obtain $\sigma'(i)$ where $\sigma'(i) =$ first part+second part. Calculates $F(\sigma'(i))$ and update $S = S \cup F(\sigma'(i))$.

Endfor $\{i\}$.

Endfor $\{k\}$.

Step 8: Filter for set S to obtain an efficient (optimal) solution.

Step 9: End

The second method firstly depends on finding DR. Then applying MST and EDD rules, then swap job $j \in [1, n]$ to the position $k = 1, \ldots, n$ to obtain new σ_k. If it contradicts with DR, it will be ignored; otherwise, we will compute the objective function and add it to the set of efficient solutions. Lastly, we filter this set to obtain efficient (optimal) sequences. This method is denoted by BC(O)MEDRM. The steps of BC(O)MEDRM are as follows:

Algorithm (2): BC(O)MEDRM Algorithm.

Step 1: Input: $n - p_j$ and d_j,

Step 2: Construct $X = [s^*, d^*]$ for MST and EDD respectively, and $S = \emptyset$.

Step 3: Find $A_1(G)$ and $A_2(G)$ matrices of DR, depending on X vector.

Step 4: For $j = 1 : n - 1$

Swap job j to the last position to obtain σ_j. If σ_j does not contradict with $A_1(G)$ and $A_2(G)$, calculate $F(\sigma_j)$ and set $S = S \cup F(\sigma_j)$. Otherwise, ignore σ_j.

Endfor$\{j\}$

Step 5: Filter the set S to obtain efficient (optimal) solutions for problem.
Step 6: End;

6.2. Exact Solving Techniques (EST's) for BC(O)_TmST Problems

The CEM will be discussed, as EST, in this section and propose new BAB technique with new LB and UB in order to solve the BC(O)-TmST problems. The propose UB depends on MST rule while the unsequence part of the obtained sequence depends on the LB with EDD rule.In algorithm (3) we describe the BC(O)BAB algorithm steps.

Algorithm (3): Branch and Bound (BC(O)BAB) Algorithm
Step 1: Input: $n - p_j$ and d_j.
Step 2: Compute $UB = F(MST)$, and $S = \emptyset$.
Step 3: Branch from the node (n_j) with $LB(n_j) \leq UB$, where $LB = F(sequence + unsequence(EDD))$.
Step 4: The set S is updated in each level of the Branching process.
Step 5: Continue until all nodes are considered and we get the efficient (optimal) solution(s) from $F(S)$.
Step 6: stop.

7. Applying NST's for BC(O) – TmST Problems

The NST's will be applied to two problems, BC(O) – TmST, in this section. First, we applied the NST to five randomly selected examples for each n. Determine the average of the solutions that were found for each example and all of the examples that were taken. Among the generated examples was the generation of random $p_j \leq d_j$, where $p_j \in [1, 10]$ will $d_j \in [1, 30]$, and this range grows as n increases.
Some important symbols used in the tables:
F: mean of objective function of problem BC – TmST.
V: mean of objective function of problem BO – TmST.
ME: mean of number of efficient solutions.
MT: mean of time.
MD: mean of distance.
ATS: Average of total sum.
R: Real number s.t. $R \in (0, 1)$.

7.1. Applying NST for BC – TmST Problem

In each NST, there are a number of efficient solutions for problem BC – TmST, so we have to check the ME, and then calculate the best solution from these solutions, we applied the distance law (see relation (2)) and also took the MD. Also we calculate MT, lastly, we heck the ATS. In table 2 we compared the results of exact methods (CEM and BCBAB) for $n = 3 : 11$. In table 3 we introduce the results of the two heuristics BCE2MHM and MaEDRHM compared with BCBAB for $n = 4 : 3 : 19$, while in table 4 we compare results of the BCE2MHM and MaEDRHM for $n = 500 : 500 : 4000$.

Note: In table 3, the number of efficiency solutions (AT) for BCE2MHM, MaEDRHM is R.

7.2. Applying NST for BO – TmST Problem

In table 5 the results of the two heuristics BOEMM and BOMEDRM compared with BOBAB for $n = 5 : 4 : 21$ while in table 6 we compare the results of BOEMM and BOMEDRM for $n = 500 : 500 : 4000$.

Table 2: Comparison results of the CEM and BCBAB for BC–TmST problem for $n = 3 : 11$.

n	CEM					BCBAB				
	ME	**F**	**MT**	**MD**	**F/ME**	**ME**	**F**	**MT**	**MD**	**F/ME**
3	1.6	(8.2, 9.8)	R	12.8	(5.1, 6.1)	1.6	(8.2, 9.8)	R	12.8	(5.1, 6.1)
4	1.6	(12.2, 19.6)	R	23.1	(7.6, 12.3)	1.0	(12.2, 19.8)	R	23.3	(12.2, 19.8)
5	1.6	(15.2, 27.6)	R	31.5	(9.5, 17.3)	1.4	(14.6, 27.9)	R	31.5	(10.5, 19.9)
6	1.8	(21.2, 45.9)	R	50.5	(11.8, 25.5)	1.8	(21.2, 46.7)	R	51.3	(11.8, 25.9)
7	2.6	(23.3, 53.1)	R	58.0	(9.0, 20.4)	2.8	(23.5, 53.8)	R	58.7	(8.4, 19.0)
8	2.0	(26.1, 67.0)	R	71.9	(13.1, 33.5)	2.0	(26.1, 67.6)	R	72.4	(13.1, 33.8)
9	3.0	(30.6, 83.7)	3.7	89.1	(10.2, 27.9)	3.0	(30.6, 85.7)	R	91.0	(10.2, 27.9)
10	2.4	(39.3, 132.1)	39.0	137.8	(16.4, 55.0)	2.4	(39.3, 135.9)	R	141.4	(16.4, 56.6)
ATS	**2.1**	(22.0, 54.9)	**5.3**	**59.3**	(10.5, 26.1)	**2.0**	(21.9, 55.9)	**R**	**60.3**	(11.0, 28.0)

Table 3: The results of the two heuristics BCEMM and BCMEDRM compared with BCBAB for BC–TmST for $n = 4 : 3 : 19$.

n	BCBAB				BCEMM			BCMEDRM		
	ME	**F**	**MT**	**MD**	**ME**	**F**	**MD**	**ME**	**F**	**MD**
4	1.0	(12.2, 19.8)	R	23.3	1.0	(12.2, 20.8)	24.1	1.2	(13.1, 20.7)	24.5
7	2.8	(23.5, 53.8)	R	58.7	1.8	(25.9, 82.1)	86.1	2.6	(26.7, 73.1)	77.8
13	2.1	(55.4, 229.5)	5.0	236.1	1.6	(54.6, 323.2)	327.8	2.2	(55.7, 304.0)	309.1
16	2.2	(69.8, 367.1)	1.7	373.7	1.6	(68.6, 466.4)	471.4	2.8	(69.8, 481.5)	486.5
19	2.4	(92.9, 592.9)	182.1	600.1	1.8	(92.7, 790.4)	795.8	2.6	(93.9, 734.5)	740.5
ATS	**2.2**	(50.4, 252.6)	**37.8**	**258.4**	**1.6**	(50.8, 336.6)	**340.4**	**2.2**	(51.8, 322.8)	**326.9**
	F/ME=(22.9, 120.3)				**F/ME**=(31.8, 210.4)			**F/ME**=(23.5, 146.7)		

Table 4: Comparison results of the BCEMM and BCMEDRM for the problem BC–TmST for $n = 500 : 500 : 4000$.

n	BCEMM				BCMEDRM			
	ME	**F**	**MT**	**MD**	**ME**	**F**	**MT**	**MD**
500	1.2	(2671.9, 662890.9)	1.7	662896.3	6.8	(2676.8, 653228.9)	1.8	653234.4
1000	1.8	(5356.6, 2664869.3)	5.7	2664874.7	8.8	(5365.2, 2626600.4)	7.4	2626605.9
1500	1.4	(8283.1, 6162053.3)	13.9	6162058.8	8.8	(8291.7, 6086752.3)	22.5	6086757.9
2000	1.2	(10864.1, 10818180.1)	28.6	10818185.6	8.4	(10875.6, 10708910.7)	53.3	10708916.2
2500	1.2	(13652.6, 17000681.2)	47.7	17000686.7	6.6	(13661.2, 16824254.5)	103.4	16824260.0
3000	1.8	(16550.2, 24778173.5)	74.9	24778179.0	11.2	(16561.7, 24520547.9)	179.0	24520553.5
3500	1.0	(19220.8, 33480857.4)	110.0	33480862.9	8.0	(19234.2, 233184587.8)	288.7	33184588.6
4000	1.2	(21978.1, 43833391.1)	156.2	4333396.6	9.4	(29193.3, 43399437.2)	447.3	43399447.0
ATS	**1.4**	(12333.2, 17425137.1)	**54.9**	**17425141.5**	**8.5**	(13132.5, 39118499.6)	**137.9**	**17125545.4**
	F/ME=(8809.4, 12446526.5)				**F/ME**=(1545.0, 4602176.4)			

Table 5: The results of the two heuristics BOEMM and BOMEDRM compared with BOBAB for BO–TmST for $n = 5 : 4 : 21$.

n	**BOBAB**		**BOEMM**		**BOMEDRM**	
	V	**MT**	**V**	**MT**	**V**	**MT**
5	30.1	R	33.6	R	32.3	R
9	84.4	R	112.8	R	102.2	R
13	233.3	R	326.8	R	316.5	R
17	419.2	93.2	517.9	R	509.1	R
21	652.0	161.2	883.8	R	867.1	R
ATS	**283.8**	**127.2**	**374.9**	**R**	**365.4**	**R**

Figure1 shows the results of the two heuristics BOEMM and BOMEDRM compared with BOBAB for $n = 5 : 4 : 21$ which are mentioned in Table5 for BO–TmST problem.

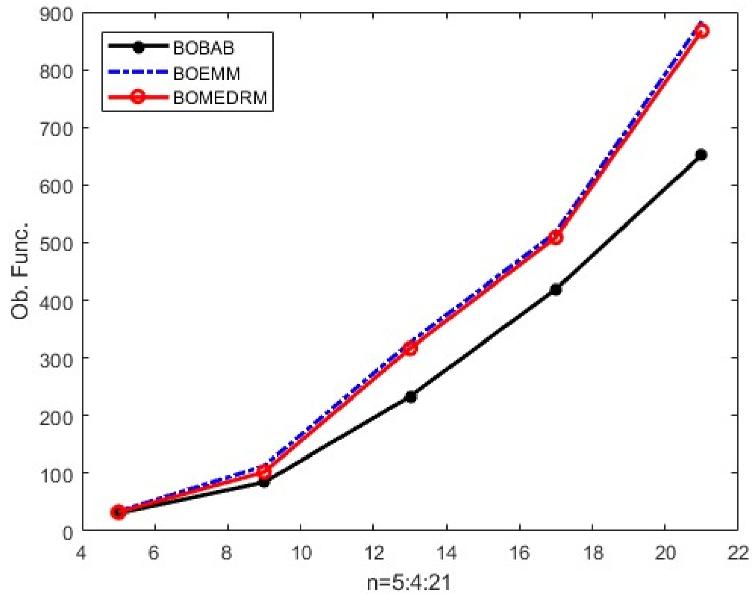

Figure 1: the results of the two heuristics BOEMM and BOMEDRM compared with BOBAB for $n = 5 : 4 : 21$ which are mentioned in Table (5) for BO–TmST problem

8. Analysis and Discussion of the Results for BC(O) – TmST Problems

For problem BC–TmST:

1. From Table2, we notice the accuracy of BCBAB compared with CEM, for $n \leq 11$ (see F, MD and F/ME criteria).
2. The results of the two heuristics, in Table3, the BCEMM and BCMEDRM have good accuracy compared with results of BCBAB, for $n = 4 : 3 : 19$, and the BCMEDRM is better than BCEMM in accuracy (see the MD and F/ME criteria).
3. In Table4, we notice that the accuracy results of BCMEDRM is better than BCEMM for $n = 500 : 500 : 4000$.
4. The CPU-time of the heuristic method BCEMM is better than BCMEDRM (see Table4).

Table 6: Comparison results of BOEMM and BOMEDRM for the problem BO–TmST for $n = 500 : 500 : 4000$.

n	**BOEMM**		**BOMEDRM**	
	V	**MT**	**V**	**MT**
500	662468.0	R	651294.5	1.2
1000	2663316.0	3.6	2624859.9	5.2
1500	6131511.4	9.7	6112760.4	17.8
2000	10817394.3	20.4	10694280.1	43.8
2500	17000168.9	35.7	16816782.3	88.1
3000	24762298.9	58.8	24634444.8	157.2
3500	33480862.9	86.4	33256284.2	267.3
4000	43822023.7	127.0	43529951.9	447.2
ATS	**17334759.51**	**42.7**	**17290082.26**	**128.5**

For problem BO–TmST:

5. In Table5, we see that the accuracy results of BOEMM and BOMEDRM are close to the results of BAB for $n = 5 : 4 : 21$ (see Figure1).

6. In Table6, for $n = 500 : 500 : 4000$, we notice that the accuracy results of BOMEDRM is better than BOEMM.

7. Results indicate that BOEMM achieves superior efficiency in terms of computational speed (CPU-Time (MT)), as shown in Table6.

9. Conclusions and Future Work

1. New different methods were proposed to solve the BCMSP (BC–TmST) and the subproblem BOMSP (BO–TmST).

2. The BC(O)BAB method proved its efficiency as an EST for small and medium-sized instances, providing optimal results for $n \leq 21$.

3. The proposed heuristic methods, BC(O)EMM and BC(O)MEDRM, demonstrated high effectiveness in generating efficient solutions for large-scale instances $n \leq 4000$.

4. Experimental results showed that the BC(O)EMM algorithm is significantly faster in terms of MT compared to other methods.

5. The BC(O)MEDRM heuristic provided high-quality results and, in several cases, achieved better objective values (V) than the BOEMM algorithm.

6. The integration of DRs within the algorithms led to a substantial reduction in the search space and improved the overall computational performance.

7. As future work, extending the model to a tri-criteria objective by adding $(T_{\max}, \sum T_j, \sum C_j)$.

8. discuss NP-hard problem with release dates, like $(1/r_j/(T_{\max}, \sum T_j))$.

9. Generalizing the proposed algorithms to the flow shop environment $(f_2//T_{\max}, \sum T_j)$.

10. Applying local search metaheuristics methods (like Genetic Algorithm, Bee Algorithm, ...) for both BCMSP and BOMSP.

References

1. Pinedo, M. L., *Scheduling: Theory, Algorithms, and Systems*, 5th Edition, Springer Published, (2016), https://doi.org/10.1007/978-3-319-26580-3.
2. Koulamas, C., *The single-machine total tardiness scheduling problem: Review and extensions*, European Journal of Operational Research, (2009), https://doi.org/10.1016/j.ejor.2008.04.012.
3. Toksarı, M. D., *A branch and bound algorithm for the single-machine total weighted tardiness problem*, Computers & Operations Research, (2016), https://doi.org/10.1016/j.cor.2015.11.008.
4. Jawad, Aseel Aboud; Ali, Faez Hassan; Hasanain, Wafaa Sayyid, *Using Heuristic and Branch and Bound Methods to Solve Multi-Criteria Machine Scheduling Problem*, Iraqi Journal of Science 61(8), 2055–2069, (2020), https://doi.org/10.24996/ijs.2020.61.8.21.
5. Ali, Faez Hassan; Jawad, Riyam Noori; Hussein, Wadhah Abdulleh, *Solving Bi-Criteria and Bi-Objectives of Total Tardiness Jobs Times and Range of Lateness Problems Using New Techniques*, Al-Mustansiriyah Journal of Science 33(3), (2022), https://doi.org/10.23851/mjs.v33i3.1135.
6. Hameed, A. S.; Chachan, H. A.; Ali, H. J., *Optimizing multi-objective problem with setup times*, in AIP Conference Proceedings 2414(1), 040046, AIP Publishing, (2023), https://doi.org/10.1063/5.0117217.
7. Bouška, Š.; Šůcha, P.; Novák, M.; Hanzálek, Z., *Deep learning-driven scheduling algorithm for a single machine problem minimizing total tardiness*, Computers & Operations Research 158, 106849, (2024), https://doi.org/10.1016/j.ejor.2023.02.
8. Heeger, K.; Hermelin, D.; Pinedo, M. L.; Shabtay, D., *Minimizing the number of tardy jobs and maximal tardiness on a single machine is NP-hard*, Operations Research Letters 52(2), 123–129, (2024), https://doi.org/10.48550/arXiv.2404.02784.
9. Faisal, Safanah Y.; Ahmed, Manal G.; Ibrahim, Manal H.; Ali, Faez H., *Using Swarm Intelligence to Solve Bicriteria and Biobjective Machine Scheduling Problems*, 10th International Science Conference of Mathematics and its Applications (ISCMA), Iraqi Journal of Science 66(11), 5015–5024, (2025), DOI: 10.24996/ijs.2025.66.11.26.
10. Gao, L.; Pan, Q.; Wang, H., *An improved iterated greedy algorithm for the single machine total tardiness problem*, Journal of Scheduling 24(4), 451–468, (2021), https://doi.org/10.1007/s10951-021-00678-7.
11. Zhang, R.; Li, X.; Wang, Y., *A hybrid heuristic algorithm for the single machine total tardiness scheduling problem*, Computers & Operations Research 139, 105630, (2022), https://doi.org/10.1016/j.cor.2021.105630.
12. Huang, K. W. W.; Lin, B. M. T., *Deep Q-networks for minimizing total tardiness on a single machine*, Mathematics 13(1), 62, (2025), https://doi.org/10.3390/math1313062.
13. Ali, F. H.; Ahmed, M. Gh., *Optimal and Near Optimal Solutions for Multi Objective Function on a Single Machine*, in 1st International Conference on Computer Science and Software Engineering (CSASE2020), Duhok, Kurdistan Region–Iraq, IEEE Iraq Section, (2020), DOI: 10.1109/CSASE48920.2020.9142053.
14. Ali, F. H.; Ahmed, M. Gh., *Local Search Methods for Solving Total Completion Times, Range of Lateness and Maximum Tardiness Problem*, in 6th International Engineering Conference, Sustainable Technology and Development, IEEE (IEC-2020), Erbil, Iraq, (2020).
15. Ahmed, M. Gh., *Exact and Approximation Methods for Solving Combinatorial Optimization Problems*, Ph.D. Thesis, Mathematics Department, College of Science, Mustansiriyah University, (2022).
16. Hassan, D. A.; Mehdavi-Amiri, N.; Ramadan, A. M., *A heuristic approach to minimize three criteria using efficient solutions*, Indonesian Journal of Electrical Engineering and Computer Science 6(1), 334–341, (2022).
17. Hoogeveen, J. A., *Invited Review Multicriteria Scheduling*, European Journal of Operational Research 167, 592–623, (1976).
18. West, D. B., *Introduction to Graph Theory*, Prentice Hall, Upper Saddle River, (2001).
19. Hussein, Wadhah Abdulelah; Ali, Faez Hassan; Abdullah, Zeyad M., *Solving Bi-Criteria of Total Earliness Times Jobs and Range of Lateness Machine Scheduling Problems by Using Local Search Methods*, Boletim da Sociedade Paranaense de Matemática, Brazil, (2026), doi:10.5269/bspm.80725.

Tamara Sabah Fadhel,
Department of Mathematics,
College of Science, Mustansiryah University,
Baghdad, Iraq.
E-mail address: tamarasf2024@uomustansiriyah.edu.iq

and

Faez Hassan Ali,

Department of Mathematics,
College of Science, Mustansiryah University,
Baghdad, Iraq.
E-mail address: faezhassan@uomustansiriyah.edu.iq