



Science Forum (Journal of Pure and Applied Sciences)

Journal Homepage: <https://atbuscienceforum.com.ng/index.php/jpas>



Solving Single Machine Bi-Criteria and Bi-Objectives Scheduling Problems of Total Tardiness and Range of Lateness Using Exact and Local Search Methods

Sajjad Majeed Jasim

Department of Mathematics, College of Science, Mustansiriyah University, Baghdad, Iraq

Abstract

In manufacturing and production planning environments, where efficiency depends on optimizing time-based performance measures, scheduling operations on a single machine is a foundational challenge. This study examines a single-machine scheduling problem (MSP) focusing on minimizing the total tardiness of jobs and the range of lateness as evaluation criteria. The problem is formulated in two ways: a bi-criteria model $1/(\sum T_j, R_L)$ and a bi-objective model $1/(\sum T_j + R_L)$ each of them is NP-hard problem. In theoretical part: Several special cases of these formulations are analyzed to better understand their characteristics. Some cases were provided that lead to the optimal solution, and others led to efficient solutions. While in practical part: To obtain effective solutions, local search methods such as the Artificial Bee Colony algorithm and Tabu Search are proposed and applied. In addition, some exact and modified solution techniques are used to solve the problems and verify the obtained results. Computational experiments demonstrate the efficiency and accuracy of the proposed approaches in solving both scheduling models.

Keywords

Machine scheduling problem (MSP),
Total tardiness,
Range of lateness,
Branch and Bound Method,
Complete Enumeration Method.



1. Introduction

Scheduling is one of the most important decision-making procedures in production systems. The goal of Machine Scheduling Problems (MSP) is to optimize one or more performance metrics by allocating resources to jobs throughout predefined time periods. Since the mid-1950s, an extensive amount of research has been done on these types of problems (Allahverdi et al. 2008).

Scheduling is one of the most investigated topics at the moment since it is so significant both practically and conceptually. Textile, metallurgical, and chemical industries are just a few of the many industries that employ these problems, most of which are classified as NP-Hard (Jacob and Arroyo 2016). In general, scheduling is the process of assigning machines to jobs such that every task may be completed within the allotted time. Finding the best order to complete these tasks order to minimize a given objective function is the difficult part (Mitić et al. 2023). Multi-objective MSP's (MOMSP's) and multi-criteria MSP's (MCMSP's) are essential to operations research because they improve scheduling efficacy, reduce operating costs, and optimize resource use (Costa et al. 2025).

The most important literature survey for the last years is that of (Ali 2016) in her M. Sc. suggested minimize the Bi-Criteria MSP (BCMSP) $1/(\sum C_j, R_L)$ problem, she proposed and applied several (exact and approximate) algorithms, which give approximate set of efficient solutions. (Abdul Razaq and Motair 2017), solves MOMSP $(1/F(\sum C_j + \sum T_j + T_{\max} + E_{\max}))$, they suggested using the Branch and Bound (BAB) technique with dominance rule (DR) and local search methods (LSM's) to solve the MOMSP. (Chachan and Jaafar 2020) study the MOMSP $(1/r_j/\sum(C_j + T_j + E_j + U_j + V_j))$ and solved examples up to 16 jobs using a BAB technique with enhanced constraints for single-MSP (SMSP). (Abbas 2022) studied the MCMSP $(1/F(\sum U_j, \sum T_j, T_{\max}))$, decomposed it into subproblems, proposed simple solution algorithms, and compared

their results with optimal solutions. (Hameed et. al 2023) study used heuristics, Genetic Algorithm (GA), and Particle Swarm Optimization (PSO) to solve a MOMSP (the sum of total completion time, total tardiness, total earliness, and total late work with setup times) that included setup times, obtaining results that were nearly complete enumeration. (Neamah and Kalaf 2024) in they study solved the multi-objective SMSP (MOSMSP) $F = \sum C_j + \sum V_j + E_{\max}$ using CEM, BAB, and heuristics up to $n = 4000$ jobs. (Jabbar et. al 2025) study proposes a MOSMSP model and applies the Meerkat Clan Algorithm (MCA) to obtain efficient solutions. The results show that MCA provides good performance compared with the BAB method, especially for larger problem sizes.

The main outlines of this paper are as follows: Section 2 discusses the basic concepts of MSP. The mathematical models for the problem are formulated in sections 3, which include both the BCMSP $(1/(\sum T_j, R_L))$ (which is the main MSP) and Bi-Objective MSP (BOMSP) $(1/(\sum T_j + R_L))$. Sections 4 and 5 introduce special cases for main problem and subproblem. Exact and LMS's are used to solve the two problems in section 6, while in section 7 the experimental results of applying the solving methods are introduced. The final sections 8 and 9 contain the analysis and discussion of results and conclusions respectively.

2. Basic Concepts of MSP

We begin by defining a few basic concepts, focusing primarily on the performance standards without delving into the specifics of the machine environment. We use $(1, \dots, n)$ to indicate the estimated number of jobs, n . These jobs are to be scheduled on a set of machines that can only handle one job at a time and are always available starting at time zero. Here, we only specify that the notations apply to a single machine, jobs j , where $(j=1, \dots, n)$ (Ali et al. 2022):

p_j : processing time of job j .

d_j : due date, which is the date for finishing job j .

s_j : Slack time of job j s.t. $s_j = d_j - p_j$.

We can now calculate the following the objectives for job $j, j=1,2,\dots,n$, given a sequence of jobs.

- The completion time C_j , where $C_j = \sum_{k=1}^j p_k$.
- The lateness $L_j = C_j - d_j$.
- The tardiness $T_j = \max\{C_j - d_j, 0\}$, where T_j denotes the individual tardiness of job j , while $\sum T_j$ denotes the total tardiness over all jobs.
- The earliness $E_j = \max\{d_j - C_j, 0\}$.

Definition (1) The Shortest Process Time (SPT) rule (Yang et al. 2025): minimizes a single objective function $\sum C_j$ for

$1/\sum C_j$ problem by sequencing the jobs in a non-decreasing order of p_j , (s.t. $p_1 \leq p_2 \leq \dots \leq p_n$).

Definition (2) Earliest due dates (EDD) rule (Jaafar and Chachan 2024): This rule minimizes a single objective function a L_{max} for $1/L_{max}$ problems and T_{max} for $1/T_{max}$ problems. It is based on the sequencing of the jobs in a non-decreasing order of d_j , (s.t. $d_1 \leq d_2 \leq \dots \leq d_n$).

Definition (3) Minimum Slacks Time (MST) rule (Ahmed 2022): This rule minimizes the single objective function E_{max} for $1/E_{max}$ problem by sequencing the jobs in a non-decreasing arrangement of their slack time $s_j = d_j - p_j$, (s.t. $s_1 \leq s_2 \leq \dots \leq s_n$).

Definition (4) The Latest Processing Times (LPT) rule (Yousif 2023): states that all jobs should be sequenced in a non-increasing order of processing times ($p_1 \geq p_2 \geq \dots \geq p_n$).

Definition (5) (Li et al. 2025): "Optimize" means to a solution in a multi-criteria decision problem where it is impossible to improve one objective without making the other worse.

Definition (6) (Hoogeveen 2005): In an MSP, we called the feasible schedule **Pareto optimal**, or we can say its non-dominated (or efficient) w. r. t. the performance of two criteria f and g if there is no feasible schedule π such that both $f(\pi) \leq f(\sigma)$ and $g(\pi) \leq g(\sigma)$, where at least one of the inequalities is strict.

Definition (7) (Huang et al. 2025): 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:

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

Remark (1) (Ali et al. 2022):

(a). $\sum T_j$ and R_L are NP-hard problems.

(b). $E_{max} = -L_{min}$ if $L_{min} \leq 0$.

3. The Mathematical Formulation of BCMSP and BOMSP

3.1. Mathematical Formulation of BCMSP

In this paper we suggest studying the BCMSP $1/(\sum T_j, R_L)$ (denoted by **BC – STR_L**), which can be formulated for a given schedule $\sigma = (1, 2, \dots, n)$ as follows:

$$F = \min(\sum T_j, R_L)$$

s.t.

$$C_1 = p_{\sigma(1)}$$

$$C_j \geq p_{\sigma(j)}, \quad j = 1, 2, \dots, n$$

$$d_j \geq p_j, \quad j = 1, 2, \dots, n$$

$$C_j = \sum_{k=1}^j p_k, \quad j = 1, 2, \dots, n$$

$$L_j = C_j - d_{\sigma(j)}, \quad j = 1, 2, \dots, n$$

$$L_{\max} = \max_{1 \leq j \leq n} \{L_j\}, \quad L_{\min} = \min_{1 \leq j \leq n} \{L_j\}$$

$$R_{L(\sigma)} = L_{\max(\sigma)} - L_{\min(\sigma)}$$

$$T_j = \max\{L_j, 0\}, \quad j = 1, 2, \dots, n$$

$$T_j, R_{L(\sigma)} \geq 0$$

... (BC - STR_L)

Notice that BCMSP-STR is NP-hard problem (see Remark (1-a)).

3.2. Mathematical Formulation of Subproblems BOMSP

While schedule $\sigma = (1, 2, \dots, n)$ is used to represent the BOMSP $1/(\sum T_j + R_L)$ of the BCMSP, (BO - STR_L), and can be formulated as follows:

$$F = \min(\sum T_j + R_L)$$

s.t.

$$C_1 = p_{\sigma(1)}$$

$$C_j \geq p_{\sigma(j)}, \quad j = 1, 2, \dots, n$$

$$d_j \geq p_j, \quad j = 1, 2, \dots, n$$

$$C_j = \sum_{k=1}^j p_k, \quad j = 1, 2, \dots, n$$

$$L_j = C_j - d_{\sigma(j)}, \quad j = 1, 2, \dots, n$$

$$L_{\max} = \max_{1 \leq j \leq n} \{L_j\} \text{ and } L_{\min} = \min_{1 \leq j \leq n} \{L_j\}$$

$$R_{L(\sigma)} = L_{\max(\sigma)} - L_{\min(\sigma)}$$

$$T_j = \max\{L_j, 0\} \quad j = 1, 2, \dots, n$$

$$T_j(\sigma), R_L(\sigma) \geq 0$$

... (BO - STR_L)

Notice that BO - STR_L is NP-hard problem (see Remark (1-a)).

Note: The bi-objective model minimizes the equally weighted scalar sum $(\sum T_j + R_L)$, whereas the bi-criteria model minimizes a Pareto criterion vector $(\sum T_j, R_L)$. These are distinct formulations.

Proposition (1): For a sequence σ , if $p_j = p, \forall j$, then $\sigma \equiv MST \equiv EDD$.

Proof: Let job i proceeds job j if $s_i \leq s_j$, subjects to $\sigma \equiv MST$ rule. By definition (3) $s_j = d_j - p_j$ and $p_j = p, \forall j$, then:

$$s_i \leq s_j \Leftrightarrow d_i - p \leq d_j - p \text{ then } d_i \leq d_j, \text{ that means the jobs are subject to } \sigma \equiv EDD, \text{ then } \sigma \equiv MST \equiv EDD.$$

Proposition (2): For a sequence σ , if $d_j = d, \forall j$, then $\sigma \equiv MST \equiv LPT$.

Proof: Using the same method as proof of proposition (1) and since $d_i = d_j = d, \forall i, j$, then

$$s_i \leq s_j \Leftrightarrow d - p_i \leq d - p_j \text{ then } \Leftrightarrow -p_i \leq -p_j \Leftrightarrow p_i \geq p_j. \text{ that means the jobs are subject to } \sigma \equiv LPT, \text{ then } \sigma \equiv MST \equiv LPT.$$

4. Special Cases for Problem BC – STR_L

Case (4.1): For problem BC – STR_L, if $\sigma \equiv EDD \equiv MST$, then:

1. If $L_{max}(\sigma) = 0$, then we obtain an optimal solution.
2. If $L_{max}(\sigma) < 0$, then the problem converts to R_L and obtaining the optimal solution.

Proof:

1. If $L_{max}(\sigma) = 0$, then $T_{\sigma(j)} = 0, \forall j$ then $\sum T_{\sigma(j)} = 0$ then:

(a). If $L_{min}(\sigma) = 0$, then $1/(\sum T_j, R_L(\sigma)) \equiv 1/(0,0)$.

(b). If $L_{min}(\sigma) < 0$, then from remark (1-b) where: $E_{max}(\sigma) = -L_{min}(\sigma)$, then:

$1/(\sum T_j, R_L(\sigma)) \equiv 1/(0, 0 - L_{min}(\sigma)) \equiv 1/(0, E_{max}(\sigma)) \equiv 1/E_{max}(\sigma)$, this problem has optimal solution for problem BC – STR_L since $\sigma = MST$.

2. If $L_{max}(\sigma) < 0$, then $T_{\sigma(j)} = 0, \forall j$ then $\sum T_{\sigma(j)} = 0$ and by remark (1-b), then:

$$1/(\sum T_{\sigma(j)}, R_L(\sigma)) \equiv 1/(0, R_L(\sigma)) \equiv 1/R_L(\sigma) \equiv 1/(0, L_{max}(\sigma) - L_{min}(\sigma)) \equiv 1/(L_{max}(\sigma) + E_{max}(\sigma))$$

to prove that σ is an optimal solution for $1/(L_{max}(\sigma) + E_{max}(\sigma))$:

since $\sigma = EDD$, then by definition (2), $L_{max}(\sigma = EDD) \leq L_{max}(\sigma')$ for every feasible sequence σ'

and since $\sigma = MST$, then by definition (3), $E_{max}(\sigma = MST) \leq E_{max}(\sigma')$ for every feasible sequence σ'

this leads to $L_{max}(\sigma) + E_{max}(\sigma) \leq L_{max}(\sigma') + E_{max}(\sigma')$

then σ is optimal solution for $1/R_L(\sigma) \equiv 1/(L_{max}(\sigma) + E_{max}(\sigma))$.

Case (4.2): For problem BC – STR_L, if $C_j = d_{\sigma(j)}, \forall j$, then the problem converted to $1/(0,0)$.

Proof: If $C_j = d_{\sigma(j)}$ then $L_j = L_{max}(\sigma) = L_{min}(\sigma) = 0$ then $\sum T_{\sigma(j)} = R_L(\sigma) = 0$, then $1/(\sum T_{\sigma(j)}, R_L(\sigma)) \equiv 1/(0,0)$.

Case (4.3): For problem BC – STR_L, if $p_j = p$ and $C_j \leq d_j, \forall j$, then:

1. If $L_{max} = 0$ the problem converted to E_{max} solved by $\sigma = MST$.
2. If $L_j < 0, \forall j$, then the problem converted to $1/R_L$ solved by $\sigma = EDD$.

Proof: If $C_j = jp \leq d_j, \forall j$, that mean all jobs are early $\sum T_j = 0$ then:

1. If $L_{max} = 0$, then $R_L = -L_{min} = E_{max}$, since $p_j = p$, then from proposition (1), $\sigma \equiv EDD \equiv MST$ and from case (4.1-1) then we obtain an optimal solution for $1/E_{max}(\sigma)$.
2. If $L_j < 0, \forall j$, $1/(\sum T_j, R_L(\sigma)) \equiv 1/(0, R_L(\sigma)) \equiv 1/R_L(\sigma)$

since $p_j = p$, then from proposition (1), $\sigma \equiv EDD \equiv MST$ and from case (4.1-2): σ is optimal solution for $1/R_L(\sigma)$.

Case (4.4): For problem BC – STR_L, if $d_j = d, \forall j$, and $d \geq C_{max}$ then the sequence σ efficient solution is obtained by $(\sum T_j, R_L(\sigma)) = (0, C_{max} - p_{\sigma(1)})$ where $p_{\sigma(1)}$ is the largest processing time.

Proof: since $d \geq C_{max}$, then $\sum T_j = 0$, then $1/(\sum T_j, R_L) \equiv 1/R_L$

Let's have any sequence σ

since $d_j = d, \forall j$ then the lateness $L_j = C_j - d$ and since $C_1 < C_2 < \dots < C_{max}$

then $L_{\max(\sigma)} = C_{\max} - d$ and $L_{\min(\sigma)} = p_{\sigma(1)} - d$.

$$R_L(\sigma) = L_{\max(\sigma)} - L_{\min(\sigma)} = C_{\max} - d - p_{\sigma(1)} + d = C_{\max} - p_{\sigma(1)}.$$

To obtain minimum R_L we must take ($C_1 = p_{\sigma(1)}$) as high as possible, so we need the $\sigma = LPT$ rule (or putting the largest processing time first by default) to obtain optimal solution for $1/R_L = 1/(C_{\max} - p_{\sigma(1)})$.

Case (4.5): For problem BC – STR_L, If $p_j = p$ and $d_j = d, \forall_j$, then $\exists$ unique efficient solution with $n!$ sequences with constant objective function, then this problem will be:

1. $1/(\sum T_j, R_L) = 1/(0, p(n-1))$ if all jobs are early.
2. $1/(\sum T_j, R_L) = 1/(\frac{np(n-1)}{2}, p(n-1))$ if $d = p$.
3. $1/(\frac{(n-k)}{2}(p(n+k+1) - 2d), p(n-1))$ if $np > d > p$.

Proof: As known:

1. If all jobs are early, this means $d \geq C_{\max} = np$ and $\sum T_j = 0, C_j = jp$, then:

$$L_j = jp - d \text{ and } L_{\max} = \max\{L_j\} = np - d \text{ and } L_{\min} = \min\{L_j\} = p - d.$$

$$\therefore R_L = L_{\max} - L_{\min} = np - d - (p - d) = p(n-1),$$

$$\therefore 1/(\sum T_j, R_L) \equiv 1/(0, p(n-1))$$

2. If $d = p$, this mean all jobs are late (except for the first job, which is on time), then

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

$$L_{\min} = 0 \text{ and } L_{\max} = p(n-1) \text{ then } R_L = p(n-1)$$

$$\text{Since } T_j = L_j = p(j-1) \text{ then } \sum_{j=1}^n T_j = p \sum_{j=1}^n (j-1) = p \frac{n(n-1)}{2}$$

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

3. If $np > d > p$, in this case, there are a number of jobs, the number of which is $k = \lfloor \frac{d}{p} \rfloor$ represents the greatest number of jobs that may be finished early or on scheduled time without tardiness. It acts as a line in the schedule between jobs that are on time and those that are not.

$$L_{\max} = C_{\max} - d = np - d$$

$$L_{\min(\sigma)} = p_{\sigma(1)} - d = p - d$$

$$R_L = p(n-1)$$

$$\sum T_j = \sum \max\{0, jp - d\} = \frac{(n-k)}{2}((k+1)p - d + np - d) = \frac{(n-k)}{2}(p(n+k+1) - 2d)$$

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

5. Special Cases for Problem BO – STR_L

Case (5.1): For problem BO – STR_L, if $\sigma \equiv EDD \equiv MST$, then:

1. If $L_{\max}(\sigma) = 0$, then we obtain an optimal solution.
2. If $L_{\max}(\sigma) < 0$, then the problem converts to R_L and obtaining the optimal solution.

Proof:

1. From case (4.1-1), then $1/(\sum T_j + R_L(\sigma)) \equiv 1/(0 + 0 - L_{\min}(\sigma)) \equiv 1/E_{\max}(\sigma)$, since $\sigma = MST$, then we obtain an optimal solution.
2. From case (4.1-2), then $1/(\sum T_{\sigma(j)} + R_L(\sigma)) \equiv 1/(L_{\max}(\sigma) + E_{\max}(\sigma))$, since $\sigma \equiv EDD \equiv MST$ then σ is optimal solution.

Case (5.2): For problem BO – STR_L, if $C_j = d_{\sigma(j)}, \forall j$, then the problem converted to $1/(0)$.

Proof: From case (4.2), $1/(\sum T_j + R_L(\sigma)) \equiv 1/(0 + 0) \equiv 1/(0)$.

Case (5.3): For problem BO – STR_L, if $p_j = p$ then:

1. If $C_j \leq d_{\sigma(j)}, \forall j$, then
 - (a). If $L_{\max} = 0$ the problem converted to $E_{\max}$ solved by $\sigma = MST$.
 - (b). If $L_j < 0, \forall j$, then the problem converted to $1/R_L$ solved by $\sigma \equiv EDD \equiv MST$.
2. If $C_j > d_{\sigma(j)}$ except $(C_1 = p = d_{\sigma(1)})$, for $j = 2, \dots, n$, then can be solved by $\sigma = EDD$.

Proof: Since $L_j = jp - d_{\sigma(j)}$ then:

1. (a). If $L_{\max} = 0$, $1/(\sum T_j + R_L(\sigma)) \equiv 1/E_{\max}(\sigma)$, then from case (4.3-1) we obtain an optimal solution for $1/E_{\max}(\sigma)$.
- (b). If $L_j < 0$, $1/(\sum T_j + R_L(\sigma)) \equiv 1/(0, R_L(\sigma)) \equiv 1/R_L(\sigma)$, then from case (4.3-2) we obtain an optimal solution for $1/R_L(\sigma)$.
2. If $C_j > d_{\sigma(j)}$ except $(C_1 = p = d_{\sigma(1)})$, for $j = 2, \dots, n$, then $L_{\min} = 0$

$$\sum T_j = \sum jp - \sum d_{\sigma(j)} = \frac{np(n+1)}{2} - \sum d_{\sigma(j)} \text{ and } R_L = L_{\max}(\sigma)$$

$$\sum T_j + R_L = \frac{np(n+1)}{2} - \sum d_{\sigma(j)} + L_{\max}(\sigma) \text{ solved by } \sigma = EDD \text{ rule.}$$

Case (5.4): For problem BO – STR_L, if $d_j = d, \forall j$, and $d \geq C_{\max}$ then the sequence σ optimal solution is obtained by $(\sum T_j + R_L(\sigma)) = C_{\max}(\sigma) - p_{\sigma(1)}$, where $p_{\sigma(1)}$ is the largest processing time.

Proof: by case (4.4) then $(\sum T_j + R_L(\sigma)) = C_{\max} - p_{\sigma(1)}$, then $\sigma = LPT$ rule to obtain optimal solution.

Case (5.5): For problem BO – STR_L, If $p_j = p$ and $d_j = d, \forall j$, then $\exists$ unique efficient solution with $n!$ sequences with constant objective function, then this problem will be:

1. $1/(\sum T_j + R_L) = 1/p(n-1)$ if all jobs are early.
2. $1/(\sum T_j + R_L) = 1/p(n-1)(\frac{n}{2} + 1)$ if $d = p$.
3. $1/\left(\frac{(n-k)}{2}(p(n+k+1) - 2d) + p(n-1)\right)$ if $np > d > p$.

Proof: As known:

1. If all jobs are early, by case (4.5-1): $1/(\sum T_j + R_L) \equiv 1/p(n-1)$
2. If $d = p$, by case (4.5-2): $1/(\sum T_j + R_L) = 1/(p(n-1)(\frac{n}{2} + 1))$.
3. If $np > d > p$, by case (4.5-3): $1/(\sum T_j + R_L) = 1/\left(\frac{(n-k)}{2}(p(n+k+1) - 2d) + p(n-1)\right)$.

6. Exact, Modified and LSMs for Solving BC – STR_L and BO – STR_L

6.1. Complete Enumeration Method (CEM)

The standard technique known as "complete enumeration" creates every possible timetable before selecting the optimal one. A MOMSP with n tasks has $n!$ distinct sequences for SMSP. This approach might be time-consuming because, even for large n (Neamah and Kalaf 2024).

6.2. Modified Branch and Bound (MBAB) Method

By implicitly enumerating every solution in the solution set—that is, evaluating progressively smaller subgroups of the solutions set the BAB method finds the best answers. These subsets may be thought of as sets of solutions to corresponding subproblems of the main problem. In order to precisely identify a solution that maximizes (minimizes) the MSP, the BAB approach is employed (Ahmed 2022). The MBAB used here is therefore classified as an incomplete method: it explores a pruned subset of the search tree and does not guarantee the globally optimal solution (Ali et al. 2022).

6.3. Local Search Methods for SMSP

6.3.1. Bee Colony Method

A metaheuristic optimization method called the Bee Colony (BA) algorithm draws inspiration from honey bee foraging. This method divides the bee colony into three groups: scout bees, hired bees, and observer bees, each of which represents a different stage of the hunt. Finding food sources and informing other bees about their quality are the responsibilities of employed bees. In light of this shared knowledge, bystander bees strengthen their search inside their communities and probabilistically choose prospective food sources.

A food supply is considered depleted or ineffective if it does not improve after a certain number of trial attempts. As a result, the matching bee stops using

that source and turns into a scout bee, which hunts for other food sources by haphazardly exploring different areas of the search space. The algorithm can successfully strike a balance between exploration and exploitation because to this cooperative method (Huang et al. 2025).

In this paper we are suggesting using BA for the two BC(O)MSP problems. The suggested BA steps are as follows:

Bees Algorithm (BA)

Step 1: INPUT: Food Number, MaxIter, limit, LS_prob, n , p_j and d_j .

Step 2: Food sources (population): Generate three solutions using EDD, SPT and MST rule and generate remaining solutions randomly.

Step 3: Estimate the population's fitness.

Step 4: REPEAT

Step 5: (Employed Bee Phase): Select sites using smart swap (choosing tardy jobs) and apply local search with probability LS_prob and estimate fitness's.

Step 6: (Onlooker Bee Phase): Select best solutions from the **step 4** and generate a neighboring solution and apply local search with probability LS_prob and estimate fitness's.

Step 7: (Scout Bee Phase): If trial counter exceeds limit, replace the solution with a random one.

Step 8: UNTIL MaxIter is reached.

Step 9: OUTPUT: Final efficient (optimal) or near efficient (optimal) solution.

END.

The BA algorithm primary control parameters are LS_prob (probability of applying local search) = 0.25, FoodNumber (number of food sources) = 20, MaxIter (maximum number of iterations) = 2000, and limit (abandonment threshold for scout activation) = 40.

6.3.2. Tabu Search Method

Since its introduction by Glover et al. (1997), Tabu Search (TS) has been used extensively to solve COP, including several scheduling applications. In contrast

to traditional LSM's, TS enhances the capacity to escape local optima by allowing the study of non-improving solutions, as long as they are not categorized as tabu. In order to stop the search from returning to previously visited solutions, the approach records the characteristics of recent moves that are utilized to change one solution into another. The following are essential elements of a standard Tabu Search framework: a specified neighbourhood structure, move characteristics, a tabu list to limit certain transitions, aspiration criteria to override tabu status under some circumstances, and termination criteria to regulate the halting process (Li and Gao 2016). In this paper we suggesting using TS for the two BC(O)MSP problems. The suggested TSA steps are as follows:

Tabu Search Algorithm (TSA)

Step 1: INPUT: MaxIter, TabuTenure, Neighborhood size, RestartIter, n , p_j and d_j .

Step 2: Generate initial solution (EDD, SPT and MST rule) and estimate fitness and choose minimum.

Step 3: REPEAT

Step 4: Use insert (or swap) moves to create neighborhood solutions.

Step 5: Choose the most acceptable alternative that meets the desire condition or is not tabu.

Step 6: Update current solution and Tabu list. If improvement occurs, update best solution.

Step 7: Use diversity (random restart) if **RestartIter** is not improving.

Step 8: Until the maximum number of iterations is achieved.

Step 9: OUTPUT: Best solution.

END.

MaxIter = 2000, TabuTenure (tabu duration) = round $(0.05 \times n) + 5$, Neighborhood = 40 (number of potential motions each iteration), and RestartIter (threshold for diversification) = 600 for diversification control

are the parameter values used by the TSA. In order to avoid cycling, a Tabu list ($n \times n$ matrix) is also kept up to date. When stagnation happens, a no-improvement counter is utilized to initiate a random restart.

7. Applying BA and TSA for Solving BC – STR_L and BO – STR_L

For each n , the values used were all randomly generated for five examples, with $p_j \in [1,10]$ and

$$d_j \in \begin{cases} [1,30], & 1 \leq n \leq 29. \\ [1,40], & 30 \leq n \leq 99. \\ [1,50], & 100 \leq n \leq 999. \\ [1,70], & \text{otherwise.} \end{cases}$$

There are five different examples examined for each problem size n . All instance data and the CEM and MBAB findings were obtained direct from the referenced published article (Ali et al. 2022). To ensure a fair and consistent comparison, the same instances were compared to the suggested BA and TSA algorithms. Since all instance data is fixed and obtained from the reference source, each algorithm was run as a single deterministic run per instance; no random seeds were needed. The objective function value is used to directly evaluate each solution's fitness; lower values for **BO – STR_L** suggest better solutions, and Pareto dominance is used to compare **BC – STR_L** solutions. Move selection in BA uses smart swap to target tardy jobs, whereas insert and swap moves in TSA are evaluated across a neighborhood of size 40.

To prevent distortion in average computations, the value was recorded as $R = 1$ second in cases when CPU time was less than one second. The MATLAB source code for both BA and TSA is available from the author upon request. MATLAB R2024b was used to program the exact and local search algorithms, which were then evaluated on an Intel(R) Core (TM) i5 (13420H) CPU operating at 2.1 GHz with 16 GB of RAM and running Windows 11 Home 64-bit.

Now we introduce the following important abbreviations:

F : Objective function of BCMSP

- G** : Objective function of BOMSP
- NEF** : Represents the average number of efficient (non-dominated) solutions across the 5 tested instances for each problem size n . For each instance, the algorithm generates a set of non-dominated solutions, and NEF is the arithmetic mean of these counts over the 5 instances.
- Av** : Represents the arithmetic mean of each objective function value independently, computed across all tested values of n presented in the table.
- R** : $0 < \text{Real} \leq 1$, When calculating the average, the value is set $R=1$.
- T/S** : CPU-Time per second.
- E** : Error, $E = \text{Proposed} - \text{Reference}$. $E < 0$: proposed method found a better (lower) solution. $E > 0$: reference found a better solution. In Tables 1 and 4 the reference is CEM; in Tables 2 it is MBAB; in Table 3 and 5 the reference is BA.

The Pareto archive consists of all non-dominated solutions found during the search. A solution σ is included if no other feasible solution simultaneously achieves a lower or equal value on both $\sum T_j$ and R_L with at least one strict improvement.

Table 1. A comparison results between BA and TSA with CEM and the BC – STR_L problem.

N	CEM			BA				TSA			
	F	T/S	NEF	F	T/S	NEF	E	F	T/S	NEF	E
5	(10.7,12.3)	R	1.2	(10.7,12.3)	R	1.2	(0,0)	(10.7,12.3)	R	1.2	(0,0)
7	(34.1,18.9)	R	2.2	(34.1,18.9)	R	2.2	(0,0)	(34.1,18.9)	R	2.2	(0,0)
9	(108.7,37.7)	3.7	4.4	(108.7,37.7)	R	4.4	(0,0)	(108.7,37.7)	R	4.4	(0,0)
11	(143,44.3)	150.3	4	(143,44.3)	R	4	(0,0)	(143,44.3)	R	4	(0,0)
Av	(74.1,28.3)	39	3	(74.1,28.3)	R	3	(0,0)	(74.1,28.3)	R	3	(0,0)

Table 2. A comparison results between MBAB with BA and TSA for the BC – STR_L problem.

n	MBAB			BA				TSA			
	F	NEF	T/S	F	NEF	T/S	E	F	NEF	T/S	E
5	(10.7,12.3)	1.2	R	(10.7,12.3)	1.2	R	(0,0)	(10.7,12.3)	1.2	R	(0,0)
10	(79.8,30.3)	3.2	R	(73,31.8)	4.4	R	(-6.8,1.5)	(74.0,31.6)	4.6	R	(-5.8,1.3)
15	(252.1,55.8)	4.2	R	(243.4,58.2)	5.4	R	(-8.7,2.4)	(248.0,58.0)	5.8	R	(-4.1,2.2)
20	(540.1,82.7)	5	R	(497.7,81.9)	6.4	R	(-42.4,-0.8)	(501.3,81.9)	4.6	R	(-38.8,-0.8)
25	(1036.5,120.5)	4.6	R	(987.7,121.5)	6.4	R	(-48.8,1)	(1007.1,121.5)	5.4	R	(-29.4,1)
30	(1442.1,134.7)	4.4	R	(1345.9,136.4)	7	R	(-96.2,1.7)	(1360.4,138)	5.2	R	(-81.7,3.3)
35	(2111.5,174)	5.8	55.4	(2042.4,172.1)	6	R	(-69.1,-1.9)	(2077.2,173)	6	R	(-34.3,-1)
40	(2516.4,190.2)	3.8	58.2	(2422.8,187.7)	5.4	R	(-93.6,-2.5)	(2455.9,187.5)	5	R	(-60.5,-2.7)
Av	(998.7,100.1)	4.0	14.9	(952.9,100.2)	5.3	R	(-45.7,0.2)	(966.8,100.5)	4.7	R	(-31.8,0.4)

Table 3. A comparison results between BA and TSA for the BC – STR_L problem.

n	BA			TSA			
	F	NEF	T/S	F	NEF	T/S	E
50	(3922.2,234.9)	7.2	R	(3936.9,235.8)	7	R	(14.7,0.9)
80	(11403.4,415.8)	8.2	1.9	(11453.0,418.0)	5.8	R	(49.6,2.2)
100	(17473.1,518.4)	6.4	1.2	(17559.7,519.7)	6.8	R	(86.6,1.3)
300	(168213.3,1610.6)	10.2	1.3	(168419.7,1614.9)	6.6	R	(206.4,4.3)
700	(930873.8,3797.7)	6.2	2.1	(933006.5,3800.4)	3.6	R	(2132.7,2.7)
1000	(1936053.7,5440.1)	9.6	2.7	(1995624.1,5444.4)	4.2	R	(59570.4,4.3)
3000	(17791961.6,16422.2)	5.2	6.9	(18794039.7,16418.8)	2.2	1.3	(1002078.1,-3.4)
Av	(2979985.9,4062.8)	7.6	2.4	(3132005.7,4064.6)	5.2	1.1	(152019.79,1.8)

Table 4. A comparison results between CEM and BA and TSA for the BO – STR_L problem.

n	CEM		BA			TSA		
	G	T/S	G	T/S	E	G	T/S	E
5	22.8	R	22.8	R	0	22.8	R	0
7	51	R	51	R	0	51	R	0
9	141.6	3.8	141.6	R	0	141.6	R	0
11	184.2	44.6	184.2	R	0	184.2	R	0
Av	99.9	12.6	99.9	R	0	99.9	R	0

Table 5. A comparison results between BA and TSA for the BO – STR_L problem.

n	BA		TSA		
	G	T/S	G	T/S	E
30	1460.2	R	1463.2	R	3
70	8495	R	8480.4	R	-14.6
100	17881.6	R	17835.6	R	-46
300	168790.8	R	168590.2	R	-200.6
700	925490	R	925128	R	-362
1000	1899487	R	1898773.4	R	-713.6
3000	17201661.4	1.4	17201098.6	2.1	-562.8
Av	2889038	1.1	2888767.1	1.2	-270.9

8. Analysis and Discussion of Results

1. The accuracy of the suggested methods is verified using small-scale examples, for the BC – STR_L and the BO – STR_L, the findings presented in Tables 1 and 4, for example, sizes (n = 5-11), demonstrate a perfect match between the objective function values derived by CEM, BA, and

TSA (E = 0 in all cases). Note that equal average objective values are consistent with, but do not by themselves establish, equality of the Pareto fronts; run-by-run comparison of the non-dominated sets would be required for that stronger conclusion. This precise agreement attests to the ability of both suggested

algorithms to get at the best result in small scale instances.

2. When ($n = 35$ and 40), Table 2 shows a sharp increase in the CPU time needed by the MBAB, going from insignificant values to (55.4 and 58.2) seconds at ($n = 35$) and ($n = 40$), respectively. In contrast, the running times of BA and TSA are almost constant at the same sizes. For the **BO – STR_L** problem, the pattern is even more pronounced in Table 4, as the CEM running time increases to (44.6) seconds at just an ($n = 11$). These results confirm that CEM and MBAB techniques are computationally unfeasible beyond very small instance sizes ($n = 35$ and 40) due to combinatorial explosion, while increased up to instances hundreds of times larger (up to $n = 3000$ in Tables 3 and 5), the suggested algorithms retain great computing efficiency with running times of no more than a few seconds.
3. Instances in which the local search algorithms perform better than the MBAB, note that the error values (E) for both BA and TSA are mostly negative ($-$) for ($n \geq 10$) is a noteworthy finding in Table 2. This suggests that both algorithms obtained better (lower) objective values than those produced by MBAB, rather than just approximating them. For example, in the first objective, BA outperformed MBAB by (96.2) units at ($n = 30$). Despite being classified as a near-exact method, this pattern indicates that the MBAB does not always ensure a globally optimal solution for this composite objective formulation. As a result, its results should be interpreted as a strong reference benchmark rather than an absolute optimality guarantee at medium-to-large instance sizes.
4. Depending on how the aim is formulated, BA and TSA function differently, an interesting comparison can be seen when comparing Tables 3 and 5. For the **BC – STR_L** problem (Table 3), BA consistently outperforms TSA in solution quality across most of tested sizes ($50 \leq n \leq 1000$), with

the difference reaching (59,570) units at ($n = 1000$). Conversely, the **BO – STR_L** problem results (Table 5) show that TSA outperforms BA for most sizes ($70 \leq n \leq 3000$), with a difference of (713.6) units at ($n = 1000$). This divergence implies that the two algorithms' relative superiority is dependent on the objective function's structure rather than being absolute (disaggregated bi-criteria versus aggregated bi-objective formulation).

5. Table 3, which represents a trade-off between the two objectives at this specific scale, showed a substantial widening of the gap between BA and TSA in the first objective (exceeding one million units) at the largest tested size ($n = 3000$), along with a marginal advantage for TSA in the second objective ($E_2 = -3.4$). This suggests that when the instance size gets closer to the upper end of the tested range, the relative performance of the two methods could not be consistently consistent across all objective components. This is a problem that needs research in future work with larger sample sizes.

9. Conclusions

1. This study presents a unified set of closed-form optimality results for a structured specialized cases of the **BC – STR_L** and **BO – STR_L** scheduling problems. We show when $\sigma \equiv \text{EDD} \equiv \text{MST}$, that minimizes the R_L and, in the case of early jobs, simultaneously achieves the optimum of both $\sum T_j$ and R_L . It is demonstrated that two structural conditions automatically induce such ordering rule coincidences: equal processing times, where $\text{EDD} \equiv \text{MST}$ and the problem is reduced to $1/E_{\max}$ or $1/R_L$; and equal due dates, where $\text{MST} \equiv \text{LPT}$, R_L admits the closed form $C_{\max} - p_{\sigma(1)}$ and, in the absence of a large-due-date constraint, an explicit finite Pareto front. It is also shown that $\sigma \equiv \text{EDD} \equiv \text{MST}$ minimizes the **BO – STR_L**

problem $G = \sum T_j + R_L$ unconditionally under equal processing times.

2. The performance of two suggested algorithms, BA and TSA, for solving bi-criteria and bi-objective single-machine scheduling problems for instance sizes ranging from ($n = 5$ to 3000) was evaluated in this study compared exact methods (CEM) and Modified (MBAB). The results show that both algorithms give highly accurate solutions on small-sized instances ($n \leq 11$), confirming their reliability and accuracy. Even with the fact that the exact methods exhibit an exponential growth in computational time as the size of the problem increases, they become impractical beyond ($n = 35$ and 40). Conversely, BA and TSA keep near-negligible running times even at ($n = 3000$), demonstrating their strong scalability for large-scale applications.
3. Notably, both BA and TSA sometimes performed better in terms of solution quality for medium-sized instances ($10 \leq n \leq 40$) than the MBAB (They usually perform better in objective $F_2 = \sum T_j$). This implies that MBAB does not guarantee the global optimum results for this composite objective and should be considered a strong benchmark and not an absolute reference. Furthermore, it was found that the relative performance of BA and TSA depended on the formulation of the objective: TSA performed better on the bi-objective problem ($BO - STR_L$), whereas BA generally outperformed TSA on the bi-criteria problem ($BC - STR_L$). This indicates to the specific structure of the objective function should be taken into consideration when choosing an algorithm.
4. To further develop this method to larger and more varied problem instances, future research may investigate hybrid methods which combine the complementary strengths of TSA and BA.
5. As future work, we suggesting to use LSM's (like particle swarm optimization, simulated annealing, genetic algorithm, ..., etc) to find efficient and

approximation solutions for ($BC - STR_L$ and $BO - STR_L$) problems.

Conflicts of interest

The authors declare no conflict of interest.

Acknowledgments

We are especially grateful to Mustansiriyah University in Baghdad, Iraq (www.uomustansiriyah.edu.iq) for their support.

References

- Abbas, D. A. (2022). Search method for solving multicriteria scheduling problem. *International Journal of Nonlinear Analysis and Applications*, 13(1), 1709-1720. <https://doi.org/10.22075/ijnaa.2022.5786>
- Abdul Razaq, T. S., & Motair, H. M. (2017). Solving composite multi objective single machine scheduling problem using branch and bound and local search algorithms. *Al-Mustansiriyah Journal of Science*, 28(3), 118-125. <https://doi.org/10.23851/mjs.v28i3.122>
- Ahmed, M. G. (2022). *Exact and approximation methods for solving combinatorial optimization problems* (Doctoral dissertation). Department of Mathematics, College of Science, Mustansiriyah University, Baghdad, Iraq.
- Ali, F. H., Jawad, R. N., & Hussein, W. A. (2022). 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), 27-35. <https://doi.org/10.23851/mjs.v33i3.1135>

Ali, S. A. (2016). *A comparison of local search algorithms for multicriteria scheduling problems* (Master's thesis). Department of Mathematics, College of Science, Mustansiriyah University, Baghdad, Iraq.

Allahverdi, A., Ng, C. T., Cheng, T. C. E., & Kovalyov, M. Y. (2008). A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, 187(3), 985-1032. <https://doi.org/10.1016/j.ejor.2006.06.060>

Chachan, H. A., & Jaafar, H. A. (2020). Exact solutions for minimizing cost function with five criteria and release dates on single machine. *Ibn AL-Haitham Journal for Pure and Applied Sciences*, 33(3), 140-157. <https://doi.org/10.30526/33.3.2479>

Costa, A., Corsini, R. R., Pagano, D., & Fernandez-Viagas, V. (2025). A new composite heuristic to minimize the total tardiness for the single machine scheduling problem with variable and flexible maintenance. *Computers & Operations Research*, 173, 106849. <https://doi.org/10.1016/j.cor.2024.106849>

Hameed, A. S., Chachan, H. A., & Ali, H. J. (2023). Optimizing multi-objective problem with setup times. *AIP Conference Proceedings*, 2414(1), 040003. <https://doi.org/10.1063/5.0117217>

Hoogeveen, H. (2005). Multicriteria scheduling: Information processing and decision making. *European Journal of Operational Research*, 167(3), 592-623. <https://doi.org/10.1016/j.ejor.2004.07.011>

Huang, X., Yin, J., & Deng, G. (2025). A hybrid discrete artificial bee colony optimization algorithm for the no-wait job shop problem with tardiness criterion. *Journal of Algorithms & Computational Technology*, 19, 17483026251322104. <https://doi.org/10.1177/17483026251322104>

Jaafar, H. A., & Chachan, H. A. (2024). Local search algorithms for solving a function with five-objectives and release dates on one-machine. *Galoitica: Journal of Mathematical Structures and Applications*, 10(2), 44-51. <https://doi.org/10.54216/GJMSA.0100204>

Jabbar, T., Kalaf, B. A., & Rehman, E. (2025). Solving multi-objective machine scheduling problem using the Meerkat Clan algorithm. *Journal of Economics and Administrative Sciences*, 31(147), 158-169. <https://doi.org/10.33095/Oss7v861>

Jacob, V. V., & Arroyo, J. E. C. (2016). ILS heuristics for the single-machine scheduling problem with sequence-dependent family setup times to minimize total tardiness. *Journal of Applied Mathematics*, 2016, 9598041. <https://doi.org/10.1155/2016/9598041>

Li, S., Wei, J., Sun, Y., & Irangabiye, A. J. N. (2025). Optimizing two-agent single-machine parallel batch scheduling: Achieving Pareto efficiency in makespan and maximum cost objectives. *RAIRO - Operations Research*, 59(4), 2219-2240. <https://doi.org/10.1051/ro/2025092>

Li, X., & Gao, L. (2016). An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem. *International Journal of Production Economics*, 174, 93-110. <https://doi.org/10.1016/j.ijpe.2016.01.016>

Mitić, P., Savić, S. P., Djordjevic, A., Erić, M., Sukić, E., Vidojević, D., & Stefanovic, M. (2023). The problem of machine part operations optimal scheduling in the production industry based on a customer's order. *Applied Sciences*, 13(19), 11049. <https://doi.org/10.3390/app131911049>

Neamah, N. M., & Kalaf, B. A. (2024). Solving tri-criteria: Total completion time, total late work, and maximum earliness by using exact, and heuristic methods on single machine scheduling problem. *Iraqi Journal for Computer Science and Mathematics*, 5(3), 14-25. <https://doi.org/10.52866/ijcsm.2024.05.03.002>

Yang, S., Meng, L., Ullah, S., Zhang, C., Sang, H., & Zhang, B. (2025). MILP modeling and optimization of three-stage flexible job shop scheduling problem with assembly and AGV transportation. *Chinese Journal of Mechanical Engineering*, 38(1), 1-15. <https://doi.org/10.1186/s10033-025-01281-z>

Yousif, S. F. (2023). *New methods for solving combinatorial optimization problems* (Master's thesis). Department of Mathematics, College of Science, Mustansiriyah University, Baghdad, Iraq.