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

(3s.) **v. 2026 (44) 10** : 1–11.
ISSN-0037-8712
doi:10.5269/bspm.80725

Solving Bi-Criteria of Total Earliness Times Jobs and Range of Lateness Machine Scheduling Problems by Using Local Search Methods

Wadhah Abdulelah Hussein*, Faez Hassan Ali, Zeyad M. Abdullah

ABSTRACT: In this paper, two different local search methods (LSMs) have been investigated in this work to solve the bi-criteria $1//(\sum E_j, R_{L)})$ and bi-objective $1//(\sum E_j + R_{L)})$ problems. The used LSMs are Bees Algorithm (BA) and Genetic Algorithm (GA), which are effective heuristic approaches. We are comparing the results of the two LSMs with the results of exact methods like Branch and Bound. In terms of optimal solutions and CPU time, this work demonstrates the strong performance of BA and GA when compared to exact and heuristic approaches

Keywords: Bi-Criteria Scheduling problems, local Search Methods, Bees Algorithm (BA) and Genetic Algorithm (GA).

Contents

1. Introduction

Many people agree that a major part of combinatorial optimization problems is machine scheduling problems (MSP). The main goal of this paper is to find the best solution to reduce a function [10]. When the MSP is an NP-hard problem, the computational time requirements are particularly high for large-sized problems; hence, heuristic strategies can be used to overcome these limitations. Recently, the term "local search methods" has been used to describe the development of heuristic techniques. The local search method provides high-quality solutions to NP-hard problems of realistic scale in a reasonable time. The Local scan Methods (LSMs), which start with a first solution, scan neighborhoods in order to find better ones [10]. In the last 70 years many researchers in introduced in general in COP, as a field of COP, some of these researchers are interest in MSP.

In 2014, Sajah [12], developed a nearly optimum solution to the $1//(T_{\max}+E_{\max})$ problem by employing LSMs (Simulated Annealing (SA) and Descent method (DM). The study also included the findings

* Corresponding author.
2020 *Mathematics Subject Classification:* 90B35, 90C27, 68W25.
Submitted December 10, 2025. Published March 12, 2026

of sophisticated computer assessments of SA and DM. Our testing findings indicate that the proposed algorithms have found exact and efficient solutions in most cases.

In 2016, Abdul-Razaq and Faez [3] determined the $1//\left(\sum C_j, R_{L)}\right)$ problem, they make use of BAB with using some LSM's (Local Insertion (LI) and Local Exchange (LE)) to find the solution problems with a large number of jobs. The purpose of LE and LI was to strengthen BAB rather than to replace it. As force multipliers, they made the BAB algorithm efficient enough to be useful for $1//\left(\sum C_j, R_{L)}\right)$ problems on a bigger scale.

In 2017, Shermeen [13] presented the problem $(T_{\max}, \mathrm{V}_{\max}, \sum V_j)$ and solved it by using some types of LSM's: The origin problem is resolved via neural networks (NN), particle swarm optimization (PSO), and the bee's algorithm (BA). For many nodes, the outcomes demonstrated the effectiveness of the PSO and BA approaches.

In 2019, Doha [1] examined a single multi-objective MSP. Minimizing four cost functions $(\sum C_j + \sum U_j + \sum T_j + T_{\max})$ is the goal by LSMs methods (DM and Genetic method(GA)) to obtain an optimal solutions for the considered problem, Also, we developed a simple algorithm (SPT-MA) to find a solution near the optimum solution. The (SPT-MA) algorithm proofs its good performance in solving the problem in a reasonable time. show that the SPT-MA method is the best compared with DM and GA in results and CPU-time, while DM is better than GA.

In 2019, Faez and Sajjad [6] One of the precise techniques is the Branch and Bound Technique (BABT); they suggested using Genetic Algorithm (GA) and Simulated Annealing (SA). To improve the outcomes of SA and GA, they suggested two more LSMs: Hybrid GA (HGA) and Improved GA (IGA). For n$\leq$ 2000, HGA demonstrated its effectiveness when compared to alternative techniques. Additionally, they employed the successive rule to minimize the problem's magnitude. Lastly, they implemented the recommended techniques in a real-world scenario they dubbed the Iraqi Cities Problem (ICP).

In 2022, Manal et al. [7] To find the most effective solutions for the Multi-Criteria Traveling Salesman Problem (MCTSP), two LSMs—the PSO and the BA—were explored. provided a few strategies for resolving the MCMSP. The outcomes demonstrated the effectiveness of the PSO and BA approaches for a sizable number of nodes (n).

In 2022, Zeyad et. al. [16] they conclude from this paper that using the new modified AZH2 algorithm to solve nonlinear fuzzy equations gives us high efficiency in solving with fewer iterations and with higher accuracy compared with other algorithms in the same field, where other algorithms can be developed in the future that may be more efficient in solving non-fuzzy problems linear. In 2021, Marwan and Zeyad et. al. [8] the method of conjugate gradient to be among the most useful methods to solve the problem of large-scale unconstrained optimization.

In 2022, Safanah and Faez [10] the LSMs, every recommended problem is resolved with a BA and a SA. Lastly, a fair amount of time is spent comparing the experimental outcomes of the LSMs with those of the BAB approach. These outcomes are guaranteeing LSMs effectiveness. They noticed a consistent pattern in all result tables, namely the significant impact of initial solutions on achieving the best results for SA and BA across different parameters indicated by 'n'. Notably, compared to BA and other approaches evaluated, SA produced findings with higher precision and used less CPU time.

The MSP idea is covered in section two. The BCMSP Problem's MSP mathematical formulation is presented in Section 3 and we introduce the mathematical formulation to drive new BOMSP subproblem From BCMSP for the two suggested problems, namely the sum of earliest landing time $\sum E_j$ and the Range of lateness R_L, this problem is NP-hard because of $\sum E_j$ and R_L are NP-hard problems. Also, in section 4, we have Local Search Algorithms (The Basic Notation) study the approximate method of the search tree. Clearly, if the algorithm selects always the best or at least a better-cost neighbor, the algorithm will end up in a local minimum, also, we have Bees Algorithm (BA) and we will prove some mathematical facts and look at result for problem bi-criteria. and we will Genetic Algorithm (GA). GA' s are optimization search algorithms which depend upon the idea of natural selection and natural. In Section 8, We recommend the use of LSMs. In order to determine the most effective solutions for BCMSP, these LSM are BA and GA. We show the outcomes of using the two proposed algorithms, BA and GA, on the two problems in Section 5. BOMSP and BCMSP. In Section 6, we have an analysis and discussion for Solving BCSER_L and NBOSER_LModels. In the end, we have section 7, Conclusions and Recommendations.

2. Basic: Concepts of Combinatorial Optimization Problems

One of the Combinatorial Optimization Problems (COP) is the MSP. In this section we will discuss some important basic concepts and notation of MSP. We state the notation that is used for the single machine, job j,(j=1,2,...,n). the job has:

$p_{(j)}$: $p_{(j)}$: processing time for a period of length $p_{(j)}$. $d_{(j)}$: A due date, the date on which the job j should be complete, the completion of the job j after its due date is allowed, but a penalty is incurred. s_j : A slack time of job j s.t. $s_j = d_j - p_j$. The completion time $C_{(j)}$: the time at which the processing of job j is completed such that $C_j = \sum_{k=1}^{1} p_k$. The lateness $L_j = C_j - d_j$. The earliness E_j=max{-L,0}. Range of lateness $R_L = L_{\max} - L_{\min} where L_{\max}(\sigma) = \max_{1\le j\le n}\{L_j\} and L_{\min}(\sigma) = \min_{1\le j\le n}\{L_j\}$.

Definition 2.1 *[3]: A schedule s is said to be an efficient schedule if we cannot found another schedule $s' satisfying f_j(s') \le f_j(s)$, j=1,2,...,k, with at least one of the above holding as a strict inequality, we call it an efficient schedule, another way says that s' dominates s.*

Definition 2.2 *[9]: Minimum Slack Time (MST) rule: The problem $1//E_{\max}$ is solved by sequencing all jobs in a non-decreasing order of slack time (s_j) i.e. ($s_1 \le s_2 \le \cdots \le s_n$).*

Definition 2.3 *[11]: Latest Processing Time (LPT) rule: Sequencing all jobs in non-increasing order of p_j i.e. ($p_1 \ge p_2 \ge \cdots \ge p_n$).*

Definition 2.4 *[4]:Let (f_o, g_o) be an efficient solution for Bi-criteria problem 1//(f,g), then the Euclidean distance (dist) for this solution is: dist=$\sqrt{f_o^2 + g_o^2}$.* ... (1)

Definition 2.5 *[2]:Shortest Processing Time (SPT) rule Jobs are sequenced in non-decreasing order of (p_j), this rule used to solve the problem 1// $\sum C_j$.*

Definition 2.6 *[5,2]:Earliest Due Date (EDD) rule Jobs are sequenced in non-decreasing order of (d_j), rule is used to minimize the problems $1//T_{\max}$ and $1//L_{\max}$.*

3. Mathematical Formulation for BCMSP and BOMSP Problem

3.1. Mathematical Formulation of BCMSP

By [14], BCMSP belongs to simultaneous optimization which depend on total earliest ($\sum E_j$) and Range of lateness (R_L) and it's denoted by ($BCSER_L$). We will try to find the efficient solutions, for the machine which can be written for a given schedule as: $\sigma = (1, 2, \cdots, n)$and can be written as [14]:

$$\left.\begin{aligned} & F = \min\left(\sum_{j=1}^{n} E_j,\ R_L\right) \\ \text{s.t.}\quad & C_1 = p_{\sigma(1)} \\ & C_j = C_{j-1} + p_{\sigma(j)}, \quad j = 2, 3, \ldots, n \\ & L_j = C_j - d_{\sigma(j)}, \quad j = 1, 2, \ldots, n \\ & E_j \ge d_{\sigma(j)} - C_j, \quad j = 1, 2, \ldots, n \\ & E_j \ge 0, \quad R_L(\sigma) \ge 0 \end{aligned}\right\} \quad \text{...(BCSER}_L\text{)}$$

This problem is NP-hard since the problems $1 \parallel \sum E_j$ and $1 \parallel R_L$ are NP-hard problems [14].

3.2. Mathematical Formulation of Subproblem (BOMSP)

In this subsection we discuss BOMSP as subproblem from the origin problem (BCSER$_L$). The BOMSP subproblem is recognized as a BOMSP with a non-linear objective function [14]. This variation, where the objective becomes non-linear, is referred to as Non-linear BOMSP (NBOMSP).which is depend on distance measures, so the NBOMSP which is derived from BCSER$_L$ [14] which comfortable as:

$$\left.\begin{aligned} \min f &= \sqrt{\left(\sum_{j=1}^{n} E_j\right)^2 + (R_L)^2} \\ \text{s.t.}\quad & C_1 = p_{\sigma(1)} \\ & C_j = C_{j-1} + p_{\sigma(j)}, \quad j = 2, 3, \ldots, n \\ & L_j = C_j - d_{\sigma(j)}, \quad j = 1, 2, \ldots, n \\ & E_j \geq d_{\sigma(j)} - C_j, \quad j = 1, 2, \ldots, n \\ & E_j \geq 0, \quad R_L(\sigma) \geq 0 \end{aligned}\right\} \quad \ldots(\text{NBOSER}_L)$$

4. Local Search Algorithms: The Basic Notation

LSMs are widely to obtain approximate solutions to compatibility problems Neighborhood research is a kind of LSMs from which one can study the method of changing adjacent pairs we will also study the approximate method of the search tree. Clearly, if the algorithm selects always the best or at least a better-cost neighbor, the algorithm will end up in a local minimum [8]. Local Search Methods (LSMs) share the following feature: Initialization [8]: The initial solution is the point from which the local search procedure is started, this could be a solution obtained from a heuristic or generated randomly, since a random solution may not satisfy the minimum of objective function. Neighborhood generation [8]: A "move" is made through the solution space s from one neighbor to another to select a neighbor s' of s. Acceptance test [8]: Each LSM has its own acceptance test to decide whether s' replaces as the current solution. Stopping criteria [8]: The stopping criterion is the method used to terminate the search process. In this paper, we propose to use two LSM's (BA and GA) to solve BCMSP Problem and new BOMSP Subproblem From BCMSP in order to obtain near optimal solutions for the large sized instances in a small consuming time [11].

4.1. Bees Algorithm (BA)

:

Ant colony optimization, particle swarm optimization, and marriage in honey bee optimization (MBO) are the three most well-known instances of swarm intelligence systems. A new method called MBO is based on the haploid-diploid genetic breeding of honey bees and is used to solve a specific class of propositional satisfiability problems. The flight of the queen bee to mate with drones, the creation of new broods, and the improvement of the broods' fitness are the three main MBO processes. Changing the colony's self-organization behavior to solve the problems is the challenging part. Honey bees' typical foraging behavior serves as the model for the Bees Algorithm (BA), a solution-finding algorithm. The BA's pseudo code in its most basic version is as follows [10]:

Several parameters must be set for the algorithm, specifically:

m: The quantity of scout bees.

ss: The number of sites chosen from a total of n sites viewed.

e: The number of top sites among the sites chosen by SS.

Nep: The quantity of bees attracted to the top o locations.

nsp: The quantity of bees enlisted for the remaining (ss-e) sites that were chosen.

ngh: The initial patch size, which takes into account the site, its surroundings, and the halting condition.

Algorithm (1): Bees Algorithm (BA)
Step 1: INPUT: m, ss, e, nep, nsp, The maximum number of iterations;
Step 2: Set up the population using random solutions;
Step 3: Start the population using random solutions.;
Step 4: REPEAT;
Step 5: UNTIL stopping criterion is met;
Step 6: OUTPUT: Near optimal or ideal results;
Step 7: END.

4.2. Genetic Algorithm (GA)

:
Using the concepts of natural selection and natural genetics, GA's are optimization search algorithms [13].
Algorithm (2): Genetic Algorithm (GA)
Step 1: Initialize Process Ge (P), i=0;
Step 3: Evolve Ge (P);
Step 4: WHILE (GA conditions have ceased) go to step 11;
Step 5: Generation i=i+1;
Step 6: Select Process Ge(P) from Ge(P-1);
Step 7: Crossover Process Ge (P);
Step 8: Mutate Process Ge (P);
Step 9: Evolve Ge (P);
Step 10: END (While);
Step 11: END.

5. Applying LSMs for Solving the Two BCSER$_L$ and NBOSER$_L$

This section recommends the use of LSMs. To solve BCSER_L, these LSMs use the Bees Algorithm (BA) and Al genetic (GA) to choose the most effective solutions. p_j and d_j overline values, or all samples, are produced at random such that We generate the values of p_j and d_j for all example randomly s.t. $p_j \in [1, 10]$ and

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

In a crucial situation that $d_j \geq p_j$, for $j = 1, 2, \ldots, n$.
The following significant abbreviations are now introduced:

Ex: An example number.
Av: Average.
NES: Number of efficient solutions.
Time: The average computation time required to solve an example (in seconds).
F: Objective function value of the BCSER$_L$ problem.
V: Objective function value of the NBOSER$_L$ problem.
R: A real number in the interval $(0, 1)$.
T/S: Average time per second.
EFF: Efficient solution.
OS: Optimal solution.

In this Section, we demonstrate the results of applying the two suggested algorithms BA and GA on the two problems BCSER$_L$and NBOSER$_L$

5.1. Applying LSM's for Solving Problem BCSER$_L$

:

In this paper we will compare the result of LSMs with BAB, Swap-Rule and Swap-DRds which obtain from [14]. displaying the result of using the two LSMs to solve the BCSER$_L$ problem. The comparative findings between CEM with BA and GA for n=3:2:11 are displayed in Table 1. Additionally, Table 2. displays the comparison findings for n=10:3:31 between BAB with BA and GA. The comparison findings of Swap-Rule and Swap-DRds with BA and GA for n = 50:50:450 are displayed in 3.4. shows the Comparison between Swap-Rule with BA and GA for results n= 500:500:4500. 5. shows the comparison results between BA with GA for n= 5000:1000:10000.

Table 1: Comparison results between CEM and BAB with BA and GA for $n = 3 : 2 : 11$.

n	CEM			BA			GA		
	EFF	**NES**	**T/S**	**EFF**	**NES**	**T/S**	**EFF**	**NES**	**T/S**
3	(7,9,5.3)	1.8	R	(7,9,5.3)	1.8	R	(7,9,5.3)	1.8	1.6
5	(2.2,17.0)	1	R	(2.6,19.6)	1.4	R	(2.2,17.0)	1	1.2
7	(7.7,21.2)	1.4	R	(14.3,30.6)	2.4	R	(7.7,21.2)	1.4	1.7
9	(2.0,26.8)	1	1.1	(4.2,35.3)	1.4	R	(2.0,27.2)	1	1.8
11	(5.0,25.0)	1	118.9	(11.4,37.0)	1.8	R	(5.7,26.1)	1.8	1.8
Av	**(5,19.1)**	**1.3**	**24**	**(8.1,25.6)**	**1.7**	**R**	**(5.1,19.4)**	**1.4**	**1.6**

Table 2: Comparison results between BAB with BA and GA for $n = 10 : 3 : 31$.

n	BAB			BA			GA		
	EFF	**NES**	**T/S**	**EFF**	**NES**	**T/S**	**EFF**	**NES**	**T/S**
10	(4.6,36.8)	1	1.5	(9.6,43.5)	1.2	R	(10.0,44.9)	1.6	R
13	(5.2,47.2)	1	1.2	(13.7,57.8)	1.8	R	(10.7,56.9)	1.6	R
16	(3.4,61.0)	1	4.6	(13.1,73.3)	1.4	R	(12.6,73.8)	1.6	R
19	(4.8,86.4)	1	20.6	(10.1,99.3)	2.0	R	(8.9,94.8)	1.4	R
22	(2.6,93.4)	1	42.2	(10.5,107.3)	1.8	R	(7.6,105.0)	1.6	R
25	(1.6,106.6)	1	107.8	(6.1,119.5)	1.6	R	(6.6,122.7)	1.4	R
28	(3.0,134.6)	1	250.6	(9.9,147.1)	1.8	R	(11.9,147.2)	1.8	R
31	(2.2,138.0)	1	516.5	(14.4,157.0)	1.4	R	(13.3,157.7)	1.6	R
Av	**(3.4,88)**	**1**	**118.1**	**(10.9,100.6)**	**1.6**	**R**	**(10.2,100.4)**	**1.6**	**R**

Table 3: Comparison results between Swap-Rule and Swap-DRDs with BA and GA for $n = 50 : 50 : 450$.

n	Swap-Rule		Swap-DRDs		BA		GA		T/S
	EFF	**NES**	**EFF**	**NES**	**EFF**	**NES**	**EFF**	**NES**	
50	(3.1,241.7)	1.2	(3.0,242.2)	1	(22.1,262.4)	1.6	(7.4,254.4)	2.4	2.7
100	(1.6,497.0)	1	(1.6,497.0)	1	(34.9,536.5)	1.4	(14.0,519.3)	2	5.5
150	(0.6,793.4)	1	(0.6,793.6)	1	(18.7,827.1)	1.2	(10.8,816.2)	2	8.3
200	(0.6,1082.0)	1	(0.6,1082.2)	1	(22.3,1116.2)	1.4	(18.1,1110.0)	2.6	11
250	(0.6,1308.0)	1	(0.6,1308.0)	1	(29.8,1350.6)	1.4	(14.1,1337.0)	2.2	12.5
300	(1.2,1600.2)	1	(1.2,1600.2)	1	(28.0,1636.5)	1.4	(18.5,1628.0)	2.6	17.8
350	(0.4,1867.0)	1	(0.4,1867.0)	1	(24.6,1900.0)	1.6	(16.4,1892.6)	3.4	16.9
400	(0.8,2174.6)	1.2	(0.6,2174.8)	1	(16.6,2202.3)	1.6	(15.2,2202.7)	2.6	17.9
450	(0.6,2415.8)	1	(0.6,2415.8)	1	(18.0,2446.6)	1.4	(15.2,2438.9)	2	19.6
Av	**(1.1,1331.1)**	**1**	**(1,1331.2)**	**1**	**(23.9,1364.2)**	**1.4**	**(14.4,1355.5)**	**2.4**	**12.5**

Note: The time (T/S) efficient solution in Table(3) for Swap-Rule, Swap-DRDs and BA is R.

Table 4: Comparison between Swap-Rule with BA and GA for results $n = 500 : 500 : 4500$.

n	Swap-Rule			BA			GA		
	EFF	**NES**	**T/S**	**EFF**	**NES**	**T/S**	**EFF**	**NES**	**T/S**
500	(0.4,2669.6)	1	R	(30.6,2713.9)	2.2	4.4	(9.1,2694.1)	2.4	23.9
1000	(0.0,5449.0)	2.1	1	(62.1,5513.1)	1.8	3.1	(48.0,5505.5)	3	45.5
1500	(0.0,8118.0)	1	6.2	(64.2,8185.7)	1.8	3	(49.5,8167.5)	3	49.4
2000	(0.0,10961.6)	1	13.5	(51.4,11021.9)	1.2	2.4	(28.6,11014.7)	2.2	66.1
2500	(0.0,13674.4)	1	25.3	(57.9,13729.4)	1.2	4.1	(28.6,13723.6)	1.8	85.1
3000	(0.0,16427.0)	1	42.5	(60.1,16494.8)	2.6	7.7	(26.4,16477.2)	2	91.9
3500	(0.0,19200.4)	1	66.1	(65.3,19267.3)	1.4	11	(21.6,19245.2)	2.2	117.8
4000	(0.0,22051.0)	1	96.1	(47.6,22113.1)	1.2	11.4	(38.8,22101.5)	2.8	170.8
4500	(0.0,24680.2)	1	134.4	(61.8,24744.1)	2.2	7.2	(32.2,24731.9)	3	157.1
Av	**(0.1,13692.3)**	**1.1**	**42.8**	**(55.7,13753.7)**	**1.7**	**6**	**(31.4,13740.1)**	**2.5**	**89.7**

Table 5: Comparison results between BA with GA for $n = 5000 : 1000 : 10000$.

n	BA			GA		
	EFF	**NES**	**T/S**	**EFF**	**NES**	**T/S**
5000	(70.6,27509.3)	2.2	33.1	(27.8,27498.8)	2.4	151.2
6000	(62.3,29426.6)	1.6	1.4	(5.0,32940.8)	1.6	220.4
7000	(9.2,38332.2)	1.6	30.2	(3.2,33824.1)	2.4	218
8000	(11.0,44092.2)	2.2	23.8	(2.6,44083.7)	1.4	263.4
9000	(6.0,49380.4)	1	30	(6.0,49379.0)	1.8	293.3
10000	(9.5,55082.4)	1.4	19.1	(7.8,55076.0)	3	303
Av	**(18.8,41223.2)**	**1.7**	**22.9**	**(8.7,41217.1)**	**2.1**	**241.6**

5.2. Applying LSM's for Solving Problem NBOSER$_L$

previous to displaying the outcomes of using the two LSMs for the NBOSER$_L$problem. The comparison findings of BAB with BA and GA for n=10:3:31 are displayed in Table 6., additionally, for n = 50:50:450, Table 7. displays the comparison results between Swap-Rule and Swap-DRds with BA and GA. The comparison of Swap-Rule with BA and GA for results n = 500,1000,1500,2000,2500,3000,3500,4000,4500 is displayed in Table 8.,Table 9. shows the comparison results between BA with GA for n= 5000:1000:10000.

Table 6: Comparative results between BAB with BA and GA for $n = 10 : 3 : 31$.

n	BAB		BA		GA	
	V	**T/S**	**V**	**T/S**	**V**	**T/S**
10	37.4	1.4	47.3	1.3	37.2	1.5
13	47.6	1.2	56.8	1.3	48.6	1.6
16	61.1	4.7	68.4	1.3	65.9	1.6
19	86.6	15.2	96.7	1.3	90.6	1.7
22	93.4	43.4	105.2	1.3	96.3	1.7
25	106.6	106	119.3	1.4	111.4	1.8
28	134.6	233.7	147.9	1.4	141.0	1.9
31	138.0	440.4	165.5	1.4	147.3	2.0
Av	**88.2**	**105.8**	**100.9**	**1.3**	**92.3**	**1.7**

Figure1 Shows the Comparative results between BAB with BA and GA in table 6. for n=10:3:31.

Figure 2 Shows the Comparative results between Swap-Rule, Swap-DRds, BA, and GA in table 7. for n =50,150,200,250,300,350,400,450.

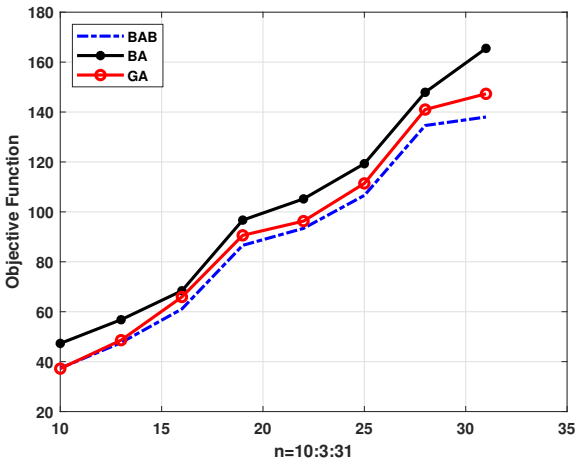


Figure 1: Comparative results between Swap-Rule and Swap-DRds with BA and GA for $n = 50 : 50 : 450$

Table 7: Comparative results between Swap-Rule and Swap-DRds with BA and GA for $n = 50 : 50 : 450$.

n	**Swap-Rule**		**Swap-DRds**		**BA**		**GA**	
	V	**T/S**	**V**	**T/S**	**V**	**T/S**	**V**	**T/S**
50	244.6	R	245.2	R	262.8	1.7	247.3	2.4
100	498.6	R	498.6	R	533.2	1.9	517	3.4
150	794	R	794.2	R	822.8	2.3	813	4.6
200	1082.6	R	1082.8	R	1113	2.6	1106.6	5.8
250	1308.6	R	1308.6	R	1349.8	3	1334.7	6.9
300	1601.4	R	1601.4	R	1638.2	3.4	1626.1	7.7
350	1867.4	R	1867.4	R	1905.6	3.7	1892.2	8.4
400	2175.2	R	2175.4	1.1	2211.7	4	2198.4	8.8
450	2416.4	R	2416.4	1.3	2448.7	4.4	2440.4	9.6
Av	**1332.1**	**R**	**1332.2**	**0.3**	**1365.1**	**3**	**1352.9**	**6.4**

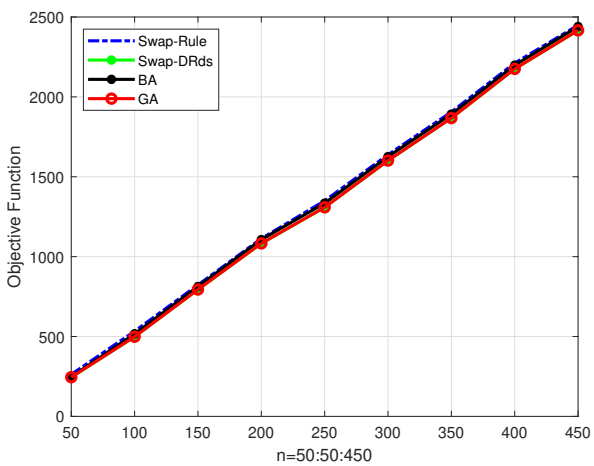


Figure 2: Comparative results between Swap-Rule, Swap-DRds, BA, and GA in table 7 for n =50:50:450.

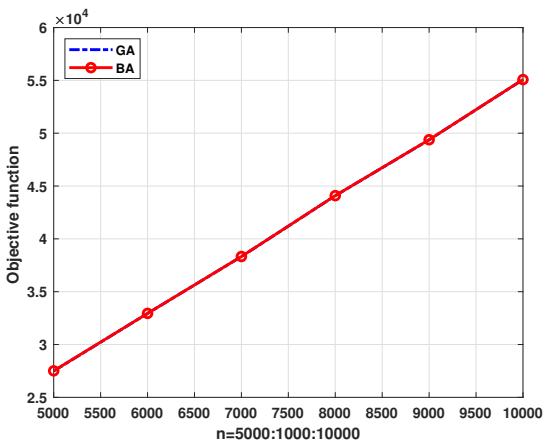
Table 8: Comparative between Swap-Rule with BA and GA for results $n = 500 : 500 : 4500$.

n	**Swap-Rule**		**BA**		**GA**	
	V	**T/S**	**V**	**T/S**	**V**	**T/S**
500	2670	R	2705.1	4.6	2692.7	10.3
1000	5449	2.1	5511	8.6	5495.4	16.7
1500	8118	5.8	8183.6	11.4	8170.1	24.1
2000	10961.6	12.8	11025.5	15.1	11014	29.6
2500	13674.4	24.2	13738.6	21.4	13732.1	37.3
3000	16427	40.8	16493.3	23.4	16478.1	44.8
3500	19200.4	63.7	19260.4	34.6	19250.9	54.7
4000	22051	94.1	22105.1	33.7	22098.9	60.9
4500	24680.2	131.9	24738.2	37.3	24728.3	69.3
Av	**13692.4**	**41.7**	**13751.2**	**21.1**	**13740.1**	**38.6**

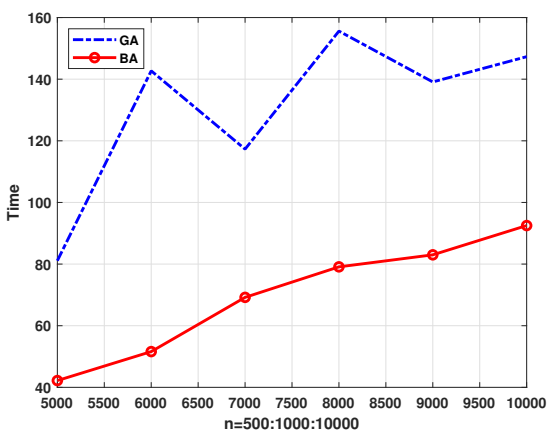
Table 9: Comparative results between Swap-Rule with BA and GA for $n = 5000 : 1000 : 8000$.

n	**BA**		**GA**	
	V	**T/S**	**V**	**T/S**
5000	27508.8	42.2	27492.5	81.1
6000	32945.2	51.6	32941.4	142.7
7000	38329.2	69.2	38323.8	117.3
8000	44090.8	79.1	44082.6	155.6
9000	49382.2	83	49375.8	139.1
10000	55077	92.5	55075.4	147.3
Av	**41222.2**	**69.6**	**41215.3**	**130.5**

Figure 3-a Shows the Comparative results objective function with n =5000,6000, 7000,8000,9000,10000 between BA, and GA in table 9., Figure 3-b. Shows the comparison results Time with n =5000,6000, 7000,8000,9000,10000 between BA, and GA in table 9.



(a) Comparative results Objective function between BA, and GA in table 9 for n =5000:1000:10000.



(b) Comparative results time between BA, and GA in table 9 for n =5000:1000:10000.

Figure 3: (3-a) and (3-b) Comparative results Objective function and time between BA, and GA in table 9 for n =5000:1000:10000.

6. Analysis and Discussion for Solving BCSER$_L$ and NBOSER$_L$ Models

1. The results presented in Table 1. indicate that the GA demonstrated the most best efficiency solutions comparing with results of exact CEM and its better from BA. These findings highlight GA's exceptional scalability and adaptability in effectively addressing bi-criteria scheduling challenges. It is useful for problem BCSER$_L$

2. A comparison of BA and GA reveals that GA consistently delivers slightly higher efficiency for solutions and a greater number of efficient solutions (NES) across different problem sizes, whereas BA exhibits greater variability. In general, GA proves to be more stable and dependable in its performance for the scheduling problems examined for problems BCSER$_L$ and NBOSER$_L$. (see Table 2, 6 and figure 1)

3. The comparison results of objective functions for GA and BA are closed to the results of Swap-Rule and Swap-DRds methods specially in 2nd criterion for problem BCSER$_L$ (see tables 3. and 4.) and for problem NBOSER$_L$ (see table 7. and figure 2)

4. In Table 8, The GA gives better solution than BA compared with Swap-Rule method, but BA gives best CPU-time compared with Swap-Rule and GA for problem NBOSER$_L$

5. Table 9 demonstrates that both BA and GA produce nearly identical solutions for objective function values (V) across all problem sizes, reflecting their comparable ability to deliver high-quality results. Nonetheless, GA consistently demands more computational time (T/S) compared to BA, so that while GA sustains solution quality, BA proves to be more efficient in terms of processing time, particularly for larger problem instances (see figure 3) for problem NBOSER$_L$

6. Figure 3.a. Shows the Comparative results objective function with n =5000,6000, 7000,8000,9000, 10000 between BA, and GA in table 9., Figure 3-b. Shows the comparison results Time with n =5000,6000, 7000,8000,9000,10000 between BA, and GA in table 9.

7. Conclusions and Recommendations

1. Our analysis demonstrated the excellent performance of two particular LSMs: BA and GA, in resolving both BCSER$_L$ and NBOSER$_L$. The higher performance of our suggested LSMs was demonstrated by comparative assessments against well-known exact methods like CEM and BAB, and different heuristic techniques. In general, the proposed LSM's are closed to each other, but GA is better.

2. In CPU-time the BA performed better than GA in all different examples for the two problems.

3. As future work, we suggest using BA and GA to solve tri-criteria problems in MSP such as $1\,||\,(\sum E_j,\ \sum T_j,\ R_L)$

4. To enhanced the performance of BA and GA in solving the two MSP's, we can start the initial population from good sequences by applying the heuristic methods: Swap-Rule and Swap-DRds, to improve the accuracy and CPU-time.

5. We suggesting a hybrid between BA and GA to design a new method can solve the two problems in good performance (accuracy and CPU-time).

References

1. Abbass, Doha A., *Using Branch and Bound and Local Search Methods to Solve Multi-Objective Machine Scheduling Problem*, in First International Conference of Computer and Applied Sciences (CAS), IEEE Xplore, (2019).

2. Faez, Hassan A.; Manal, Ghassan A., *Local Search Methods for Solving Total Completion Times, Range of Lateness and Maximum Tardiness Problem*, in 6th International Engineering Conference Sustainable Technology and Development (IEC-2020), Erbil, Iraq.

3. Faez, Hassan A.; Riyam, N. J.; Wadhah, A. H., *Solving Bi-Criteria and Bi-Objectives of Total Tardiness Jobs Times and Range of Lateness Problems Using New Techniques*, AI-Mustansiriyah Journal of Science 33(2), 27-35, (2022).

4. Faez, Hassan A.; Rasha J. M. and Wadhah A. H., *Exact and Near Pareto Optimal Solutions for Total Completion Time and Total Late Work Problem*, Iraqi Journal of Science 64(7), 4385-4398, (2023).

5. Faez, H. A.; Wadhah A. H., *The heuristic and metaheuristic methods to minimize the total completion and total tardiness jobs times problem*, International Journal of Nonlinear Analysis and Applications 13(2), 2025-2035, (2022).

6. Jasim, Sajjad M.; Faez, H. A., *Exact and Local Search Methods for Solving Travelling Salesman Problem with Practical Application*, Iraqi Journal of Science 60(5), 1138-1153, (2019).

7. Manal, G. A.; Faez, H. A., *The Best Efficient Solutions for Multi-Criteria Travelling Salesman Problem Using Local Search Methods*, Iraqi Journal of Science 63(10), 4352-4360, (2022).

8. Manal, G. A.; Faez, H. A.; Hanan, A. C., *Solving Multi Objectives function problem using branch and bound and local search methods*, International Journal of Nonlinear Analysis and Applications 13(1), 1619-1658, (2022).

9. Safanah, F. Y.; Faez, H. A., *Solving Maximum Early Jobs Time and Range of Lateness Jobs Times Problem Using Exact and Heuristic Methods*, Iraqi Journal of Science 65(2), 923-937, (2024).

10. Safanah, F. Y.; Faez, H. H.; Karrar, F. A., *Using local search methods for solving two multi-criteria machine scheduling problems*, Al-Mustansiriyah Journal of Science 34(4), (2023).

11. Safanah, F. Y., *New Methods for Solving Combinatorial Optimization Problems*, M.Sc. Thesis, Mustansiriyah University, College of Science, Department of Mathematics Science, (2023).

12. Sajah, J. I., *Single Machine Multicriteria Scheduling*, M.Sc. Thesis, Mathematics Department, College of Ibn AL-Haitham, University of Baghdad, (2014).

13. Shermeen, B. A., *Algorithms for Solving Multi-Criteria Scheduling Problems*, M.Sc. Thesis, Baghdad: Mathematics Department, College of Science, Mustansiriyah University, (2017).

14. Wadhah, A. H.; Zeyad, M. A.; Faez, H. A., *Solving Bi-Criteria of Total Earliness Jobs Times and Range of Lateness Problems by using Local Search Methods*, Al-Nahrain Journal of Science, (accepted 2025).

15. Zeyad, M. A. et al., *A New Shifted Conjugate Gradient Method Based on Shifted Quasi-Newton Condition*, Journal of Physics: Conference Series 1818, 012-105, (2021).

16. Zeyad, M. A. et al., *Modification of the new conjugate gradient algorithm to solve nonlinear fuzzy equations*, Indonesian Journal of Electrical Engineering and Computer Science 27(3), 1525-1532, (2022).

Wadhah Abdulelah Hussein,
Department of Mathematics,
College of Computer Science and Mathematics, Tikrit University,
Tikrit,Iraq.
E-mail address: wadhahabdulelah@st.tu.edu.iq

and

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

and

Zeyad M. Abdullah,
Department of Mathematics,
College of Computer Science and Mathematics, Tikrit University,
Tikrit,Iraq.
E-mail address: zeyaemoh1978@tu.edu.iq