Int. J. Nonlinear Anal. Appl. 13 (2022) 2, 2025–2035
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2022.6548

The heuristic and metahuristic methods to minimize the total completion and total tardiness jobs times problem

Faez Hassan Ali[a,*], Wadhah Abdulleh Hussein[b]

[a]Mathematics Department, College of Science, Mustansiryah University, Baghdad, Iraq

[b]Mathematics Department, College of Science, University of Diyala, Iraq

(Communicated by Javad Vahidi)

Abstract

In this paper, we found some methods for solving one of the bicriteria machine scheduling problems (BMSP). Our discussed problem is represented by the total completion and the total tardiness jobs times $(1//(\sum C_j, \sum T_j))$ simultaneously. In order to solve the suggested BMSP, some new heuristic and metaheuristic methods are proposed which are produced good results. The results of the new suggested methods are compared with the exact method; like the complete enumeration method (CEM) and Branch and Bound (BAB) method, then compared results of the heuristics with each other's to obtain to the most efficient method.

Keywords: Branch and Bound method, complete enumeration method, heuristic, metahurisitc
2020 MSC: 11Y16, 33F10

1 Introduction

There are many known exact and approximation solution methods to solve the Machine Scheduling Problem (MSP): The exact solutions are obtained from the Complete Enumeration Method (CEM), Branch and Bound (BAB) method and Dynamic Programming (DP) method [8].

Multicriteria optimization depends on conflicting objective functions which establish a set called Pareto optimal solutions set (or Efficient solution), which is considered as vicarious of one optimal solution. This set includes one (or many) solution(s) that no other solution(s) is better than this (these) solution(s) according to the objective functions. In the literature, there are two approaches for multicriteria scheduling problems [1]; the hierarchical approach and the simultaneous approach.

The most important last five years' literature surveys are: some efficient algorithms are proposed for solving these problems. Ali [3], in part of his thesis, solves the $1//(\sum C_j, \sum T_j)$ problem, he suggested a new BAB and Neural Network (NN) with two local search methods (LSMs) to solve the problem. Abdul-Razaq and Ali [4] solve the $1//(\sum C_j, \sum R_L)$ problem, they use BAB with new upper and lower bounds and then use some LSM's to solve problems with a high number of jobs. Ali and Abdul-Kareem (2017) [5], solve a multicriteria MSP by minimize three objective functions $(T_{\max}, V_{\max}, \sum V_j)$ simultaneously, by using complete enumeration (CEM) method and BAB which are considered exact methods. And then they propose some new heuristic methods, for a large number of jobs

*Corresponding author
Email address: faezhassan@uomustansiriyah.edu.iq (Faez Hassan Ali)

Received: January 2022 *Accepted:* April 2022

and use the best heuristic method to obtain an upper bound for BAB. Chachan and Hameed [10], try to solve a new multiobjective problem $1//(\sum(C_j+T_j+E_j+V_j))$, they suggest using BAB for solving this problem, where a proposed number of upper and lower bounds, also they found a number of dominance rules to minimize the number of modes in the search tree.

Gallo and Capozzi [11], using Simulated Annealing (SA) to solve a problem of scheduling n jobs on m machines that minimizes the total completion time ($\sum\sum C_j^k$). SA is examined together with its key parameters (freezing, tempering, cooling, and a number of contours to be explored), and the choices made in identifying these parameters are illustrated to generate a good algorithm that efficiently solves the scheduling problem.

Chachan et al. [9] discussed the $1//(\sum(C_j+T_j+E_j+V_j+U_j))$ problem. Their paper investigates the theoretical analysis, discussion and proofs for various special cases for this problem. In addition, the effect of dominance rules in decreasing the CPU-time in solving the problem is discussed. Therefore, they used the BAB to obtain the optimal solution for this problem.

Ali and Ahmed [2] introduced a multicriteria objectives function $1//(\sum C_j, R_L, T_{\max})$ P-problem in single MSP which is solved by BAB and some heuristic methods. Some special cases are introduced and proved to find efficient solutions for problems. Then in they solved $1//(\sum C_j + R_L + T_{\max})$ P_1-problem to find good or optimal solutions by using exact and heuristic methods [7]. Lastly, the Bees Algorithm and Particle Swarm Optimization are used for solving the two suggested problems [6].

In section two we will discuss the mathematical formulation of $1//(\sum C_j, \sum T_j)$ problem (P-problem) and its special case (P_1-problem). In section three we propose two heuristic methods for $1//\sum T_j$ and the two suggested problems. The simulated annealing is introduced as a metaheuristic method to solve the two problems is introduced in section four. The practical and comparative results are introduced in section five. While in section six we will show some analysis and discussion of the results which are introduced in section five. Lastly, in section seven the most important conclusions and some recommendations are introduced.

2 Some Important Notations

First we have to introduce the following notations:

- n: The number of jobs.
- p_j: The processing time of jobs j.
- d_j: The Due date of jobs j.
- C_j: The Completion time of job j, where $C_j = \sum_{k=1}^{j} p_k$.
- T_j: The Tardiness of job j, $T_j = \max\{C_j - d_j, 0\}$.
- $\sum C_j$: Total completion time.
- $\sum T_j$: Total Tardiness.
- OP: Optimal Value of P_1-problem.

3 Machine Scheduling Problem

In this paper we need some basic definitions.

Definition (1): Shortest Processing Time (SPT) rule [13]: Jobs are sequenced in non-decreasing order of processing times (p_j), this rule used to solve the problem $1//\sum C_j$.

Definition (2): Earliest Due Date (EDD) rule [12]: Jobs are sequenced in non-decreasing order of due date (d_j), rule is used to minimize the problem $1//T_{\max}$.

Definition (3) [12]: The term "optimize" in a multicriteria resolution-making problem indicates a solution about which there is no way of developing or improving any objective without worsening the other objective.

Definition (4) [13]: A schedule S is said to be an efficient schedule if we cannot found another schedule S' satisfying $f_j(S') \leq f_j(S')$, $j = 1, 2, ..., k$, with at least one of the above holding as a strict inequality. Another way we say that S' dominates S.

4 Dominance Rules

Dominance Rules (DR's) can usually determine some (all) parts of the permutation or the sequence to obtain an optimized value for the objective function of the problem. The DR's can be very useful to determine whether a node in the BAB method can be ignored or eliminated without calculating its LB. The DR's are also useful within BAB to cancel all the nodes which are dominated by others. These improvements imply a very large decrease in the number of nodes and then in CPU-time to obtain the optimal solution for the discussed problem.

Definition (5) [15]: Let G be a graph with n vertices, then the matrix $A(G) = [a_{ij}],\ i, j = 1, 2, ..., n$, whose i^{th} and j^{th} element is 1 if there is at least one edge or path between vertex V_i and vertex V_j and zero otherwise, this matrix is called the adjacency matrix of graph G, where:

$$a_{ij} = \begin{cases} 0, & if\ i = j\ or\ i \neq j \\ 1, & if\ i \to j \\ a_{ij}\ and\ \bar{a}_{ij}, & if\ i \leftrightarrow j \end{cases}$$

5 Mathematical Formulation for $1//(\sum C_j, \sum T_j)$ Problem

The object can describe as a set of n jobs $N = 1, 2, ..., n$ on a single machine to find $\sigma \in S$ (where S is the set of all feasible schedules) so they can be us full to specify whether that minimize the multi-criteria $(\sum C_j, \sum T_j)$. The $1//(\sum C_j, \sum T_j)$ problem can be written as:

$$(P) \begin{cases} \min\{\sum C_j, \sum T_j\} & \\ \text{subject to:} & \\ C_j \geq p_{\sigma(j)}, & j = 1, 2, ..., n \\ C_j = C_{(j-1)} + p_{\sigma(j)}, & j = 2, 3, ..., n \\ T_j \geq C_j + d_{\sigma(j)}, & j = 1, 2, ..., n \\ T_j \geq 0, & j = 1, 2, ..., n \end{cases}$$

For P-problem, we can deduce subproblem: The $1//\sum C_j + \sum T_j$ Problem:

$$(P_1) \begin{cases} \min\{\sum C_j, \sum T_j\} & \\ \text{subject to:} & \\ C_j \geq p_{\sigma(j)}, & j = 1, 2, ..., n \\ C_j = C_{(j-1)} + p_{\sigma(j)}, & j = 2, 3, ..., n \\ T_j \geq C_j + d_{\sigma(j)}, & j = 1, 2, ..., n \\ T_j \geq 0, & j = 1, 2, ..., n \end{cases}$$

The goal for the P_1-problem is to find suitable sequence of the jobs on a single machine to reduce the total of completion time and total of tardiness jobs, which is considered just a single object.

6 Heuristic Methods for $//\sum T_j$, P and P_1-Problems

As most scheduling problems are NP-hard, and the computational requirement to solve such problems using BAB or DP methods might require much time, many researchers have developed heuristic algorithms to solve them in an efficient and effective way.

The heuristic (or approximate) method can be defined as follows (Reeves [16]): A heuristic is a technique or an algorithm that seeks well for optimal or near-optimal solutions at a reasonable computational time without no guarantee of optimality, or even in many cases to check how close this solution is to the optimal solution?

In the next section we will discuss some heuristic methods for $(1//\sum T_j)$, P and P_1-problems.

7 Heuristic Methods for $1//\sum T_j$

The problem $1//\sum T_j$ considered is NP-hard, so two heuristic or approximate methods are proposed for solving this problem to obtain good solutions. The first suggested heuristic method is using the EDD rule for the Sum

of Tardiness (which we called EDD-ST). The idea of this method we start to arrange the jobs by EDD rules and calculate the objective function, and then put the 2nd job in 1st position and then arrange the other jobs by EDD rules and calculate the objective function, we continue this process until obtaining n feasible sequences are obtained. The algorithm of EDD-ST is as follows:

EDD_ST Heuristic Algorithm

Step(1): **INPUT** $n,\ p_j$ and $d_j,\ j = 1, 2, 3, ..., n$.

Step(2): Arrange jobs in EDD rule (γ_1), and calculate $F_1(\gamma_1) = \sum T_j(\gamma_1)$, let $\gamma = \gamma_i$

Step(3): **FOR** $i = 2, ..., n$, job i in the first position of γ_{i-1} to obtain γ_i, then calculate $F_i(\gamma_i)$.

Step(4): **IF** $F_i(\gamma_i) \leq F(\gamma)$ **THEN** $\gamma = \gamma_i$.

ELSE GOTO Step(3).

ENDIF.

Step(5): **OUTPUT:** The optioned of sequence γ with $F(\gamma)$ value.

Step(6): **END.**

The second method depends using DR of Sum Tardiness (DR-ST).

Remark 7.1. [1] For $1//T_{\max}$ problem if $p_i \leq p_j$ and $d_i \leq d_j$, then there exists an efficient solution in which job i is sequenced before job j.

Remark 7.1 is can be useful for $1//\sum T_i$ problem to obtain optimal solution.

Example 7.2. Suppose we have the following MSP for $n = 4$:

n	**1**	**2**	**3**	**4**
p_j	6	1	3	4
d_j	25	7	8	6

By using remark 7.1 we obtain the following $DR : 2 \to 1,\ 3 \to 1,\ 4 \to 1,\ 2 \to 3$. Then the adjacency matrix:

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & a_{24} \\ 1 & 0 & 0 & a_{34} \\ 1 & \bar{a}_{24} & \bar{a}_{34} & 0 \end{bmatrix}$$

Here we obtain three sequences $\pi_1 = (2, 3, 4, 1),\ \pi_2 = (2, 4, 3, 1)$ and $\pi_3 = (4, 2, 3, 1)$, we notice that π_3 gives an optimal solution $\sum T_j = 0$.

The DR-ST (DR of $(1//\sum T_j)$) method is summarized by finding a sequence arranged with minimum p_j and d_j and which it's not contradicted with DR of the problem and then calculate the objective function. The algorithm of DR-ST is stated as follows:

DR_ST Heuristic Algorithm

Step(1): **INPUT:** n, p_j and $d_j,\ j = 1, 2, ..., n$.

Step(2): Apply remark 7.1 to find adjacency matrix A of DR, $N = \{1, 2, ..., n\}$.

Step(3): **FOR** $i = 1, 2, ..., n$, find a sequence σ with minimum p_j and d_j which is not contradiction with matrix A, if $\exists$ more than one job with same p_j or d_j (or both) then break tie arbitrary.

Step(4): **IF** $F_i(\sigma_i) \leq F(\sigma)$, **THEN** $\sigma = \sigma_i$

ELSE GOTO Step(3).

ENDIF.

Step(5): **OUTPUT:** The optioned sequence σ with $F(\sigma)$ value.

Step(6): **END.**

8 Heuristic Method for P and P_1-problems

The heuristic methods for P and P_1-problems are considered as a development of the heuristic methods for $1//\sum T_j$ problem.

8.1 Heuristic Methods for P-problem

The first heuristic method depends on SPT and EDD. Since the SPT rule solving the $1//\sum C_j$ problem in polynomial time, so we suggest to improve the heuristic method EDD-ST by order the jobs by SPT rule firstly, and then calculate the objective function, and then put the second job in first place and the other jobs still arranged by SPT rule and calculate the objective function, and so on until obtain n sequences. The process repeated for EDD rule for P-problem (so we called it SPT-EDD-SCST(F)). The algorithm of SPT-EDD-SCST(F) is as follows:

SPT_EDD_SCST(F) Heuristic Algorithm

Step(1): INPUT $n,\ p_j$ and $d_j,\ j = 1, 2, 3, ..., n,\ \delta = \phi$.

Step(2): Arrange jobs in SPT rule (σ_1), and calculate $F_1 1(\sigma_1) = (\sum C_j(\sigma_1), \sum T_j(\sigma_1))$;
$\delta = \delta \cup \{F_{11}(\sigma_1)\}$.

Step(3): FOR $i = 2, ..., n$, put job i in the first position of σ_{i-1} to obtain σ_i and calculate $F_{1i}(\sigma_i) = (\sum C_j(\sigma_i), \sum T_j(\sigma_i))$;
$\delta = \delta \cup \{F_{1i}(\sigma_i)\}$.
ENDFOR;

Step(4): Arrange jobs in EDD rule (π_1), calculate $F_{21}(\pi_1) = (\sum C_j(\pi_1), \sum T_j(\pi_1))$;
$\delta = \delta \cup \{F_{21}(\pi_1)\}$.

Step (5): FOR $i = 2, ..., n$, put job i in the first position of π_{i-1} to obtain π_i and calculate $F_{2i}(\pi_i) = (\sum C_j(\pi_i), \sum T_j(\pi_i))$;
$\delta = \delta \cup \{F_{2i}(\pi_i)\}$.
ENDFOR;

Step(6): Filter set δ to obtain as a set of efficient solution of P-problem

Step(7): OUTPUT The set of efficient solutions δ.

Step(8): END.

The idea of the second heuristic method is dependent on the heuristic DR-ST which is described in subsection Heuristic Methods for $1//\sum T_j$. The suggested method will be improved to applied on P-problem (so we called it DR-SCST(F)). The algorithm of DR-SCST(F) is as follows:

DR_SCST(F) Heuristic Algorithm

Step(1): INPUT: n, p_j and $d_j,\ j = 1, 2, ..., n$.

Step(2): Apply remark 7.1 to find DR and adjacency matrix A;
$\sigma = \emptyset, N = 1, 2, ..., n.\ \delta = \emptyset$.

Step(3): Sort the jobs with minimum p_j which is not contradiction with matrix A say σ_1, if $\exists$ more than one job break tie arbitrary, $\delta = \delta \cup \{\sigma_1\}$.

Step(4): Sort the jobs with minimum d_j which is not contradiction with matrix A say σ_2, if $\exists$ more than one job break tie arbitrary, $\delta = \delta \cup \{\sigma_2\}$.

Step(5): Find the dominated sequence set δ' from δ.

Step(6): Calculate $F(\delta')$.

Step(7): OUTPUT The set of efficient solutions δ'.

Step(8): END.

8.2 Heuristic Methods for P_1-problem

For P_1-problem, we can use the same two heuristic methods which are discussed in subsection heuristic methods for P-problem.

9 Simulated Annealing

Simulated annealing (SA) [11] is a scheme represented by the physical annealing process. This name refers to the simulation of the annealing process which is related to an annealing schedule of decreasing in the temperature. SA can be considered as a local optimization technique, where the initial solution is always enhanced by small local effects until none of these effects can improve the solution. The initial state or solution of a thermodynamic system, which is considered the original Metropolis acceptance criterion, was chosen at energy ($Cost$) and temperature ($Temp$ *or* t). Holding constant t, the initial setting of the system is perturbed to produce a new setting and calculate the energy ΔC. The new setting is accepted without conditions if ΔC is negative whereas it is accepted if ΔC is positive with a probability given by the Boltzmann factor $e^{-\Delta C/Temp}$ to stay away from trapping in the local optima. A simple scheme of SA [14] is as follows:

Simulated Annealing Algorithm

[ch']=SA(ch)

$ch' = ch$;

$Cost$ = Evaluate(ch');

$Temp = Initial\ Temperature$;

WHILE ($Temp > Final\ Temperature$) **DO**

ch_1 = Mutate (ch');

$NewCost$ = Evaluate(ch_1);

$\Delta C = NewCost - Cost$;

IF ($\Delta C \leq 0$) OR ($e^{-\Delta C/Temp} > Rand$) THEN

$Cost = NewCost$;

$ch' = ch_1$;

ENDIF

$Temp = cooling\ rate \times Temp$;

ENDWHILE

Return the best solution;

END

It's important to mention that *cooling rate* is 0.95, $Temperature$ is 10000, and final $Temperature$ is 0, $Rand$ as a uniform real random.

10 Practical Results Of P And P_1-Problems

The values of p_j and d_j for all example are generated randomly s.t. $p_j \in [1, 10]$ and

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

with condition $d_j \geq p_j$, for $j = 1, 2, ..., n$.

The BAB method is obtained from [1].

Now we introduce the following important abbreviations:

- Av: Average.
- $ANES$: Average number of efficient solutions.
- AAE: Average of Absolute Error $AE = |a_1 - a_2|$.
- ARE: Average of Relative Error $RE = |a_1 - a_2|/a_1$.

- AT/S: Average of Time per second.
- $ASOF$: Average of Single Objective Function.
- $AMOF$: Average of Multi-Criteria Objective Function.
- R: $0 <$ Real < 1.
- F: Objective Function value for P-problem.
- F_1: Objective Function value for P_1-problem.

It's important to mention that the results of applying all solving methods are revered for 5 experiments.

11 Comparison Results of $1//\sum T_j$

Table 1 shows a comparison between heuristics EDD_ST and DR_ST compared with the CEM results method for solving $1//\sum T_j$ problem, $n = 4 : 10$.

Table 1: Comparison of EDD_ST and DR_ST with CEM results for $1//\sum T_j$ problem, $n = 4 : 10$.

n	**CEM**		**EDD_ST**			**DR_ST**		
	OP / **AAST**	**AT/S**	**ASOF**	**AT/S**	**ARE**	**ASOF** / **AAST**	**AT/S**	**ARE**
4	6.4	R	6.6	R	0.03	6.8	R	0.06
5	10.2	R	12.4	R	0.22	12.0	R	0.18
6	18.0	R	20.4	R	0.13	21.6	R	0.20
7	30.6	R	38.8	R	0.27	41.2	R	0.35
8	80.6	R	99.6	R	0.24	104.2	R	0.29
9	102.4	7.7	121.6	R	0.19	129.8	R	0.27
10	65.8	86.1	99.0	R	0.50	104.8	R	0.59
Av	**44.9**	**13.4**	**56.9**	**R**	**0.23**	**60.1**	**R**	**0.28**

Where the AAST is the average of the averages of ST.

Table 2 shows a comparison results between heuristics EDD_ST and DR_ST methods to solve $1//\sum T_j$ problem for $n = 30, 70, 100, 300, 700, 1000, 3000$ and 7000.

Table 2: comparison between EDD_ST and DR_ST results for $1//\sum T_j$ problem for $n = 30, 70, 100, 300, 700, 1000, 3000$ and 7000.

n	**EDD_ST**		**DR_ST**	
	ASOF / **AAST**	**AT/S**	**ASOF** / **AAST**	**AT/S**
30	1848.0	R	1909.2	R
70	11778.0	R	11876.6	R
100	25205.6	R	25328.6	R
300	243685.2	R	243047.8	R
700	1317493.0	2.1	1309202.4	R
1000	2695619.6	2.5	2685643.0	2.3
3000	24619935.0	2.7	24499290.0	65.7
7000	134737606.8	12.3	132618239.2	838.2
Av	**20456646.4**	**2.5**	**20174317.1**	**113.3**

12 Comparison Results of P-problem

Comparison results between SPT-EDD-SCST(F), DR-SCST(F) and SA(F) with efficient results of CEM(F) for P-problem are shown in Table 3, $n = 4 : 10$.

Table 3: Comparison between SPT-EDD-SCST(F), DR-SCST(F), SA(F) with CEM(F) for P-problem, $n = 4 : 2 : 10$.

n	CEM(F)			SPT-EDD-SCST(F)			DR-SCST(F)			SA(F)		
	AMOF	AT/S	ANES	AT/S	ANES	AAE	AT/S	ANES	AAE	AT/S	ANES	AAE
4	(53.9,8.9)	R	2.4	R	2.2	(0.7,0.4)	R	2	(3.9,0.5)	R	2.2	(0.1,0.1)
6	(99.9,22.9)	R	6.4	R	4.0	(1.1,2.0)	R	1.8	(1.8,2.5)	R	5.2	(0.2,2.0)
8	(208.9,84.6)	R	4.6	R	2.0	(1.1,3.6)	R	1.0	(6.9,5.0)	R	2.4	(2.9,5.4)
10	(203.9,70.8)	31.2	7.6	R	2.4	(1.1,5.9)	R	1.0	(8.3,7.4)	R	2.0	(4.4,6.2)
Av	(141.7,46.8)	7.8	5.3	R	2.7	(1.0,3.0)	R	1.45	(5.2,3.9)	R	3.0	(1.9,3.4)

Notice that the Heuristics SPT-EDD-SCST(F), DR-SCST(F) and SA(F) give good results compared with CEM(F) for P-problem.

In Table 4 we compare the results obtained from heuristic SPT-EDD-SCST(F), DR-SCST(F) and SA(F) with BAB(F) for P-problem, $n = 12 : 2 : 20$.

Table 4: Results of comparison of SPT-EDD-SCST(F), DR-SCST(F) and SA(F) with BAB(F) for P-problem, $n = 12 : 2 : 20$.

n	BAB(F)			SPT-EDD-SCST(F)			DR-SCST(F)			SA(F)		
	AMOF	AT/S	ANES	AT/S	ANES	AAE	AT/S	ANES	AAE	AT/S	ANES	AAE
12	(298.8,138.4)	R	3.6	R	2.0	(4.4,4.4)	R	1.0	(1.8,3.4)	R	1.0	(0.8,6.4)
14	(426.4,233.9)	R	3.2	R	1.6	(0.5,1.6)	R	1.0	(2.8,2.5)	R	1.6	(1.6,4.1)
16	(577.7,333.6)	R	8.0	R	2.4	(2.8,6.2)	R	1.0	(11.7,9.0)	R	1.2	(8.9,7.9)
18	(732.3,459.7)	5.9	8.6	R	2.4	(0.3,5.0)	R	1.0	(9.7,6.9)	R	1.2	(9.1,7.2)
20	(789.3,476.9)	822.3	12.2	R	2.8	(3.3,13.1)	R	1.0	(11.7,10.3)	R	2.6	(7.0,13.1)
Av	(564.9,328.5)	169.4	7.1	R	2.24	(2.3, 6.1)	R	1.0	(7.5,6.4)	R	1.52	(5.5,7.7)

Table 5 shows a comparison results between heuristics methods SPT-EDD-SCST(F), DR-SCST(F) and SA(F) to solve P-problem for $n = 30, 70, 100, 300, 700, 1000$ and 3000.

Table 5: A comparison results between SPT-EDD-SCST(F), DR-SCST(F) and SA(F) for P-problem for different n.

n	SPT-EDD-SCST(F)		DR-SCST(F)		SA(F)	
	AMOF	AT/S	AMOF	AT/S	AMOF	AT/S
30	(1885.6,1346.7)	R	(1878.2,1344.8)	R	(1884.5,1347.5)	R
70	(9589.5,8182.6)	R	(9582.0,8162.4)	R	(9582.9,8184.7)	R
100	(19882.6,17410.3)	R	(19872.8,17381.6)	R	(19873.3,17416.5)	R
300	(174982.0,167200.7)	R	(174971.0,167127.8)	R	(174971.5,167201.3)	R
700	(940239.0,921650.8)	2.7	(940239.0,921320.0)	R	(940239.0,921650.4)	R
1000	(1930322.8,1893990.0)	9.7	(1930322.8,1893291.0)	R	(1930322.8,1893989.4)	3.2
3000	(17294695.8,17185179.8)	372	(17294695.8,17184480.4)	25.3	(17294695.8,17185179.8)	121.9

13 Comparison Results of P_1-problem

In Table 6 we show a comparison between the optimal results of CEM (F_1) and the results of the heuristics SPT-EDD-SCST (F_1), DR-SCST (F_1) and SA (F_1), $n = 4 : 10$, for P_1-problem.

Notice that the heuristics SA (F_1), SPT-EDD-SCST (F_1) and DR-SCST (F_1) give good objective values compared with CEM (F_1), and that cam be noticed from AAE, for P_1-problem.

Figure 1 shows the comparison results of CEM (F_1), SPT-EDD-SCST (F_1), DR-SCST (F_1) and SA (F_1) which are obtained from Table 6, for P_1-problem, for $n = 4 : 10$.

The results of applying BAB (F_1) and the three heuristics for P_1-problem, $n = 11 : 18$, which give results in reasonable CPU-time (Time ≤ 600 seconds), all are described in Table 7.

Table 8 describes the average of efficient solution for P_1-problem for $n = 30, 70, 100, 300, 700, 1000$ and 3000 using SA (F_1) compared with heuristic methods SPT-EDD-SCST (F_1) and DR-SCST (F_1).

Table 6: Comparison between CEM (F_1) and SPT-EDD-SCST (F_1), DR-SCST (F_1) and SA (F_1), $n = 4 : 10$, for P_1-problem.

n	CEM (F_1)		SPT-EDD-SCST (F_1)			DR-SCST (F_1)			SA (F_1)		
	OP	AT/S	ASOF	AT/S	AAE	ASOF	AT/S	AAE	ASOF	AT/S	AAE
4	61.2	R	62.2	R	1.0	63.6	R	2.4	61.2	R	0.0
5	75.8	R	76.2	R	0.4	75.8	R	0.0	75.8	R	0.0
6	118.2	R	122.0	R	3.8	122.2	R	4.0	118.2	R	0.0
7	163.0	R	167.2	R	4.2	166.4	R	3.4	163.8	R	0.8
8	291.2	R	293.0	R	1.8	291.6	R	0.4	291.2	R	0.0
9	332.4	2.7	336.8	R	4.4	334.8	R	2.4	332.8	R	0.4
10	269.2	29.6	275.4	R	6.2	273.8	R	4.6	269.42	R	0.2
Av	187.3	4.6	190.4	R	3.1	189.7	R	2.5	187.5	R	0.2

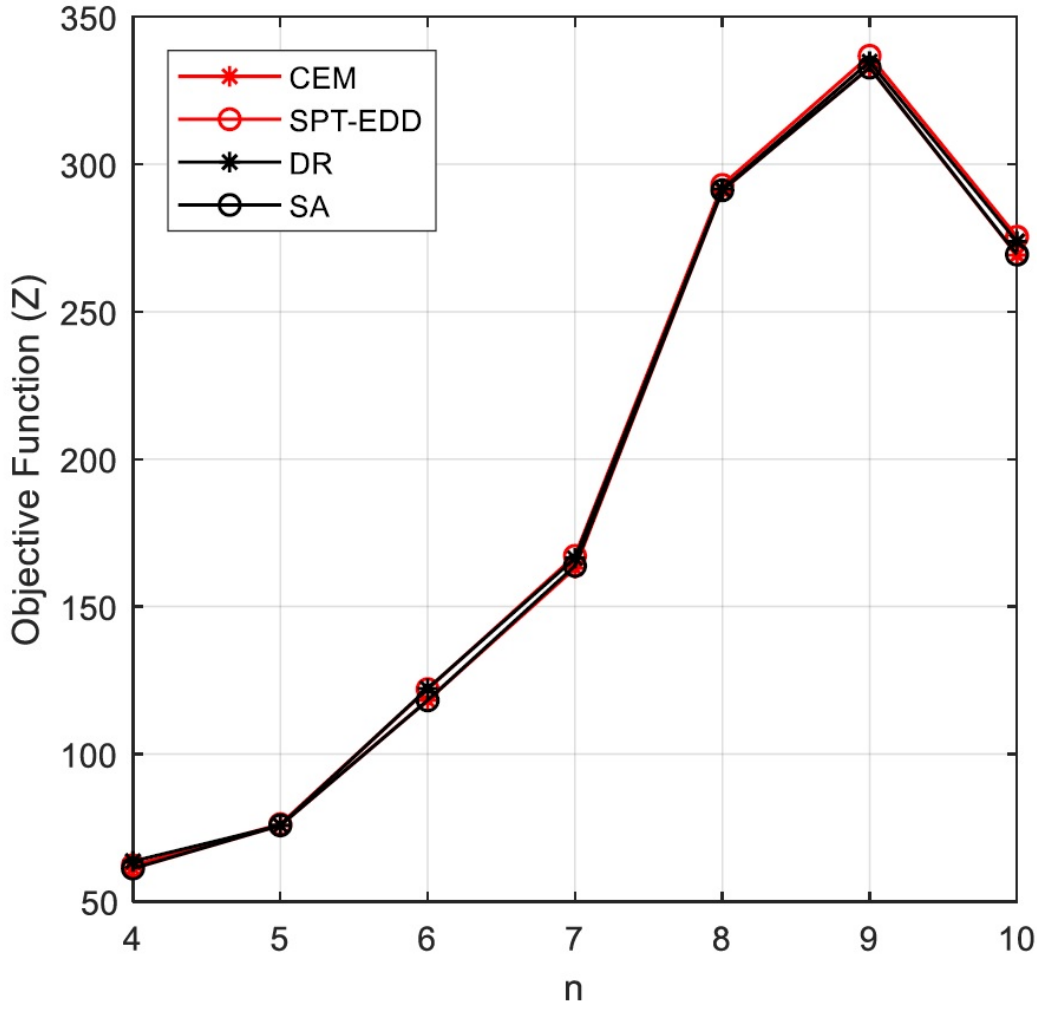

Figure 1: Comparison results of CEM (F_1), SPT-EDD-SCST (F_1), DR-SCST (F_1) and SA (F_1) for $n = 4 : 10$.

14 Analysis And Discussion Of Results

- For $1//\sum T_j$ problem, we noticed that EDD-ST better than DR-ST for $n \leq 10$ (see Table 1) while DR-ST better than EDD-ST for $n \leq 7000$ in accuracy but in CPU-time EDD-ST is better (see Table 2).
- For P-problem: the SPT-EDD-SCST (F) is the best among DR-SCST (F) and SA (F) in accuracy and all heuristics are closed to each other in CPU-time for $n \leq 20$, (see Tables 3 and 4) while DR-SCST (F) is best in accuracy and CPU-time for $30 \leq n \leq 3000$ (see Table 5).
- For P_1-problem: in accuracy and CPU-time, we ensure that SA (F_1) is the best among SPT-EDD-SCST (F_1)

Table 7: The comparison of BAB (F_1) results, with three heuristics for P_1-problem, for $n = 11 : 18$.

n	BAB (F_1)		SPT-EDD-SCST (F_1)		DR-SCST (F_1)		SA (F_1)	
	OP	AT/S	ASOF	AT/S	ASOF	AT/S	ASOF	AT/S
11	440.4	R	443.2	R	442.6	R	441.2	R
12	435.4	R	442.6	R	438.8	R	435.4	R
13	478.4	25.5	483.4	R	481.8	R	480.2	R
14	659.2	R	661.0	R	660.0	R	659.4	R
15	683.4	R	686.0	R	685.4	R	683.6	R
16	904.0	3.3	908.8	R	908.6	R	904.6	R
17	839.6	38.3	844.0	R	842.4	R	840.4	R
18	1186.4	4.4	1189.2	R	1189.2	R	1186.6	R
Av	703.4	8.9	707.3/3.9	R	706.1/2.7	R	703.9/0.5	R

Table 8: Results of comparison of SA (F_1) with SPT-EDD-SCST (F_1) and DR-SCST (F_1) for P_1-problem, for $n = 30, 70, 100, 300, 700, 1000$ and 3000.

n	SA (F_1)		SPT-EDD-SCST (F_1)		DR-SCST (F_1)	
	ASOF	AT/S	ASOF	AT/S	ASOF	AT/S
30	3214.6	R	3228.8	R	3223.0	R
70	17737.0	R	17768.4	R	17744.4	R
100	37311.0	R	37289.0	R	37254.4	R
300	458095.8	1.2	342173.4	R	342098.8	R
700	2631810.8	2.3	1861889.80	1.2	1861559.0	R
1000	5432999.6	5.9	3824312.8	4.2	3823613.8	R
3000	49373285.0	128.3	34479875.6	12.9	34479176.2	25.7
Av	8279207.7	19.7	5795219.7	2.6	5794952.8	3.7

and DR-SCST (F_1) for $n \leq 20$ (see Tables 6 and 7) while SPT-EDD-SCST (F_1) and DR-SCST (F_1) are closed and better than SA (F_1) for $30 \leq n \leq 3000$, (see Table 8).

15 Conclusion

From this paper, we obtained the following conclusions:

- New different heuristic are suggested to solve $1//\sum T_j$ problem for one machine, the experimental results proved that the MST-ST and DR-ST methods give good results.
- We develop the heuristic methods of $1//\sum T_j$ for convenient $1//(\sum C_j, \sum T_j)$ and we got two new heuristics SPT-EDD-SCST and DR-SCST with good performance.
- For P and P_1-problem, we used SPT-EDD-SCST and DR-SCST and suggest 3rd metaheuristic method (SA) which gives good results for $n \leq 3000$.
- We suggest using new LSM for solving P and P_1-problems, like Genetic Algorithm (GA) and Ant Colony Optimization (ACO).

Acknowledgments

The authors would like to thanks Mustansiriyah university (www.uomustansiriyah.edu.iq) Baghdad – Iraq for its support in the present work.

References

[1] T.S. Abdul-Razaq and F.H. Ali, *Algorithm for scheduling a single machine to minimize total completion time and total tardiness*, Basrah J. Sci. **34** (2016), no. 2, 113–132.

[2] M.G. Ahmed and F.H. Ali, *Efficient algorithms to solve tricriteria machine scheduling problem*, Al-Rafidain Univ. College Sci. **46** (2020).

[3] F.H. Ali, *Improving exact and local search algorithms for solving some combinatorial optimization problems*, Ph. D. Thesis, Mustansiriyah University, College of Science, Dept. of Mathematics, 2015.

[4] S.A. Ali, *A comparison of local search algorithm for multicriteria scheduling problems*, M.Sc. Thesis, University of Al-Mustansiriyah, College of Science, Dept. of Mathematics, 2016.

[5] F.H. Ali and S.B. Abdul-Kareem, *Scheduling a single machine to minimize max tardiness, max late work and total late work*, Mathematics and Statistics Journal, 3 (2017), no. 1, 1–17.

[6] F.H. Ali and M.G. Ahmed, *Local search methods for solving total completion times, range of lateness and maximum tardiness problem*, 6th Int. Engin. Conf. "Sustainable Technology and Development" (IEC), IEEE, 2020, pp. 103–108.

[7] F.H. Ali and M.G. Ahmed, *Optimal and near optimal solutions for multi objective function on a single machine*, Int. Conf. Comput. Sci. Software Engin. (CSASE), IEEE, 2020, pp. 152–156.

[8] J. Blazewics, K.H. Ecker, E. Pesch, G. Schmidt and J. Weglarz, *Scheduling computer and manufacture processes*, Springer Science and Business Media, 2001.

[9] H.A. Chachan, F.H. Ali and M.H. Ibrahim, *Branch and bound and heuristic methods for solving multi-objective function for machine scheduling problems*, 6th Int. Engin. Conf. "Sustainable Technology and Development"(IEC), IEEE, 202), pp. 109–114.

[10] H.A. Chachan and A.S. Hameed, *Exact methods for solving multi-objective problem on single machine scheduling*, Iraqi J. Sci. **60** (2019), no. 8, 1802–1813.

[11] C. Gallo and V. Capozzi, *A simulated annealing algorithm for scheduling problems*, J. Appl. Math. Phys. **7** (2019), no. 11, 2579–2594.

[12] J.A. Hoogeveen, *Minimizing maximum earliness and maximum lateness on a single machine*, Proc. the 1st Integer Program. Combin.Optim. Conf. 1991, pp. 283–295.

[13] J.A. Hoogeveen, *Single machine scheduling to minimize a function of two or three maximum cost criteria*, J. Algorithms **21** (1996), 415–433.

[14] S.M. Jasim and F.H. Ali, *Exact and local search methods for solving travelling salesman problem with practical application*, Iraqi J. Sci. **60** (2019), no. 5, 1138–1153.

[15] B. Kolman, *Introductory linear algebra with applications*, Macmillan Publishing Company, 1988.

[16] C.R. Reeves, *Modern heuristic techniques for combinatorial problems*, John Wiley and Sons, New York, 1993.