Minimizing Weighted Penta-Objective Function using Local Search Methods and Branch and Bound Algorithm

Manal H. Ibrahim, Faez H. Ali, Hanan A. Chachan

Department of Mathematics, College of Science, Mustansiriyah University, Baghdad, Iraq2

***Abstract*— In this work, we will discuss a Multi (Penta) Objective Function (POF) for Single Machine Scheduling Problems (SMSPs) to minimize the total cost of sum weighted of the following objectives: earliness, tardiness, completion time, late jobs and late work ($\sum_{j=1}^{n}(\alpha_j E_j + \beta_j T_j + \theta_j C_j + \gamma_j U_j + \omega_j V_j)$), where $\alpha_j, \beta_j, \theta_j, \gamma_j$ and ω_j, are positive penalties of job j for each objective respectively. For this problem we will study the mathematical formulation of this problem, using Complete Enumeration and Branch and Bound (BAB) methods to find the optimal solution and Local Search Methods (LSM's) to find approximation solutions. Lastly, the practical results proved the efficiency of BAB and LSM's methods.**

***Keywords*—Machine Scheduling Problems, Multi Objective Function, Local Search Methods, Branch and Bound.**

INTRODUCTION

This This research focuses on the Machine Scheduling Problem (MSP), which is one of the most significant topics in Operations Research. Scheduling is the specifying of resources each time to find a set of tasks, which is a subset of the wider field of COP. The scheduling problem is critical in most industrial and production systems, as well as a variety of information processing contexts. It is also useful in transportation and distribution environments (Phrueksanant, 2013).

The single MSP (SMSP) has received a lot of attention over the last few decades. Johnson, (1954) contributed the first theoretical advancement in scheduling, which was quickly followed by the findings of both Jockson (1955). and Smith (1956).

I. LITERATURE REVIEW

For Although many real-life scheduling problems involves minimizing multiple objectives, little work has been done on these criteria and more less for the problems with set-up times. Emmons (1975) discussed the minimizing the number of tardy jobs and the sum of completion time without set-up time problem. Abdul Razaq and Zeghiar (1996) presented a BAB algorithm for minimizing total cost of completion time and the number of late jobs on a single machine without set-up time problem also.

In the last two decades LSM are often to tackle various kinds of problems. Thousands of articles on all kinds of variants of LSM and their applications were published. If LSMs are used correctly, they produce significantly better outcomes than classical heuristics (Arts, 1997).

M'Hallah and Alhajraf (2016) studied the $1//E_{max}^{W}$ problem, using approximated by employing an ant colony-based system with a decreased number of ants and colonies, as well as daemon operations that traverse the search space surrounding the ants using a variable neighborhood search. Harrath et al. (2019) studies and solve the $1/s_f/\sum(C_j + \omega_j T_j)$ by GA with new crossover operators.

The incorporation including both earliness and tardiness costs in the objective function is consistent with the just-in-time production philosophy, which focuses generating wealth only when they are necessary. As our knowledge, no published work on the subject exists of $1//\sum_{j=1}^{n}(\alpha_j E_j + \beta_j T_j + \theta_j C_j + \gamma_j U_j + \omega_j V_j)$ with unequal penalties.

This work has been arranged as follows: section 2 presents the mathematical model of the proposed problem, and section 3 decomposes the problem and provides some basic results. In section 4, it then introduced some heuristic methods for determining UB for the problem. In section 5, we use the exact methods to solve the proposed problem. The LSMs are used to solve the suggested problem in this paper in section 6. Section 7 contains the proposed mathematical calculations (models). Section 8 contained the concluding remarks and future extensions.

II. THE MATHEMATICAL FORMULATION OF THE DISCUSSED PROBLEM

The problem consists of n production (jobs) order with different processing times p_j, $j = 1, \dots, n$, on the single machine, and different due date d_j and positive penalties $\alpha_j, \beta_j, \theta_j, \gamma_j$ and ω_j for each unit of earliness time, tardiness time, completion time, number of tardy jobs and late work.

This problem, denoted by (P) can formally be stated as follows:

DOI: http://doi.org/10.24086/ICAFS2023/paper.906

$$V = \min_{\sigma \in S}\{Z(\sigma)\} = \min_{\sigma \in S} \sum_{J=1}^{n} \left(\alpha_{\sigma_j} E_j + \beta_{\sigma_j} T_j + \theta_{\sigma_j} C_j + \gamma_{\sigma_j} U_j + \omega_{\sigma_j} V_j\right)$$

Subject to:

$$\left.\begin{aligned} & C_1 = p_{\sigma_1} \\ & C_j = C_{j-1} + p_{\sigma_j}, \quad j = 2, \dots, n \\ & E_j \geq d_{\sigma_j} - C_j, \quad j = 1, \dots, n \\ & E_j \geq 0, \quad j = 1, \dots, n \\ & T_j \geq C_j - d_{\sigma_j}, \quad j = 1, \dots, n \\ & T_j \geq 0, \quad j = 1, \dots, n \\ & U_j = \begin{cases} 0, & if C_j \leq d_{\sigma_j} \\ 1, & if C_j > d_{\sigma_j} \end{cases}, j = 1, \dots, n \\ & V_j = \min\left\{T_j, p_{\sigma_j}\right\}, \quad j = 1, \dots, n \end{aligned}\right\} \quad \dots(\mathrm{P})$$

$\alpha_j, \beta_j, \theta_j, \gamma_j$ and ω_j are non − negative weights

Where σ_j denote the position of job j in the ordering σ. The set of all schedules is denoted by S. The P-problem is NP-hard.

III. DECOMPOSITION OF THE PROBLEM (P) AND SOME BASIC RESULTS

The problem (P) can be decomposed into three subproblem with a simple structure, and state some results which help us in solving the problem (P).

These sub problems are:

$$V_1 = \min_{\sigma \in S}\{Z_1(\sigma)\} = \min_{\sigma \in S} \sum_{J=1}^{n} \left(\alpha_{\sigma_j} E_j + \beta_{\sigma_j} T_j + \theta_{\sigma_j} C_j\right)$$

Subject to:

$$\left.\begin{aligned} & C_1 = p_{\sigma_1} \\ & C_j = C_{j-1} + p_{\sigma_j}, \quad j = 2, \dots, n \\ & E_j \geq d_{\sigma_j} - C_j, \quad j = 1, \dots, n \\ & E_j \geq 0, \quad j = 1, \dots, n \\ & T_j \geq C_j - d_{\sigma_j}, \quad j = 1, \dots, n \\ & T_j \geq 0, \quad j = 1, \dots, n \end{aligned}\right\} (\mathrm{SP}_1)$$

Also the sub problem

$$V_2 = \min_{\sigma \in S}\{Z_2(\sigma)\} = \min_{\sigma \in S} \sum_{J=1}^{n} \left(\gamma_{\sigma_j} U_j\right)$$

Subject to:

$$\left.\begin{aligned} & C_1 = p_{\sigma_1} \\ & C_j = C_{j-1} + p_{\sigma_j}, \quad j = 2, \dots, n \\ & U_j = \begin{cases} 0, & if C_j \leq d_{\sigma_j} \\ 1, & if C_j > d_{\sigma_j} \end{cases}, j = 1, \dots, n \end{aligned}\right\} \quad \dots(\mathrm{SP}_2)$$

And the subproblem

$$V_3 = \min_{\sigma \in S}\{Z_3(\sigma)\} = \min_{\sigma \in S} \sum_{J=1}^{n} \left(\omega_{\sigma_j} V_j\right)$$

Subject to:

$$\left.\begin{aligned} & C_1 = p_{\sigma_1} \\ & C_j = C_{j-1} + p_{\sigma_j}, \quad j = 2, \dots, n \\ & T_j \geq C_j - d_{\sigma_j}, \quad j = 1, \dots, n \\ & T_j \geq 0, \quad j = 1, \dots, n \\ & V_j = \min\left\{T_j, p_{\sigma_j}\right\}, \quad j = 1, \dots, n \end{aligned}\right\} \quad \dots(SP_3)$$

IV. OPTIMAL SOLUTION FOR P-PROBLEM

In order to solve the P-problem by using BAB algorithm, we need the UB and LB for the P-problem which is given in the following subsection.

A. Upper Bound (UB) for P-Problem

The most basic rules which are given an optimal solution for some problems, generate initial sequences for the upper bounding techniques for our problem. TABLE I summarizes these heuristics and their primary features.

TABLE I
Some Basic Rules For The P-Problem

Rule	Abbreviations	Detailed description
Shortest Weighted Processing Time	$SWPT$	This rule entails sequencing jobs the order of processing time to weight ratio (non-descending), This rule resolves $1//\sum W_j C_j$ problem (Graham et al. (1979)).
Earliest Due Date	EDD	This rule, entails sequencing jobs the order of due date (non-descending), This rule resolves $1//T_{max}$ problem (Jackson,1955).
Minimum Slack Time	MST	This rule sequencing the jobs according to their slack time $d_j - p_j$ (non-descending), This rule resolves $1//E_{max}$ problem (Hoogeveen, and van deVelde, 1990).
Modified Due Date	MDD	this rule includes modified the due date of the jobs to $max\ [t + p_j, d_j]$ (Hoogeveen, 2005).
Apparent Tardiness Cost	ATC	This rule is heuristic which choose the machine whenever it becomes available the jobs which are have the largest priority index $\frac{1}{p_j} \exp\{\frac{-\max(0, d_j - t - p_j)}{\varphi \bar{p}}\}$, where t is the current time, $\bar{p}$ is the average processing time, and φ is the look ahead empirical parameter (Lawler, 1973).
Lawler's Algorithm	LA	LA was created to tackle the $1//f_{max}$ issue, where $f_{max} \in [T_{max}, V_{max}, L_{max}]$ were employed to get the minimal f_{max} (Mahmood, 2001).
Moor's Algorithm	MA	MA solves the problem $1//\sum U_j$ to find minimum number of late jobs (Mahmood, 2001).

B. Lower Bound (LB) for P-Problem

To find the LB for our problem we have to decomposing it into three subproblems, which can be denoted by (SP_1), (SP_2) and (SP_3) respectively. Then the LB of the problem (P) is the

DOI: http://doi.org/10.24086/ICAFS2023/paper.906

sum of minimum values of the subproblems (SP_1), (SP_2) and (SP_3).

This decomposition subproblems have a better capability than (P). In order to calculate Z_1.

First we have to find the optimality for (SP_1) using the following procedure:

Step (1): Let σ be the SWPT.

Step (2): Calculate the value of cost function $Z_1(\sigma)$:

$$Z_1(\sigma) = LB_1 = \sum_{j=1}^{n} \left(\theta_{\sigma_j} C_{\sigma_j} + \beta_{\sigma_j} T_{\sigma_j} + \alpha_{\sigma_j} E_{\sigma_j}\right) = max(\sum_{j=1}^{n} d_j, \sum_{j=1}^{n} \min\theta_{\sigma_j}(\max(2C_{\sigma_j} - d_{\sigma_j}, C_{\sigma_j}))).$$

Second, while to minimum value (Z_2) for (SP_2) we need the following algorithm:

Step (1): Let π be the EDD rule.

Step (2): Using Moore algorithm (MA), find the minimum number of late jobs say k.

Step (3): Index the jobs in anon-decreasing order of their weights γ_j $(j = 1, \dots, n)$.

Step (4): $Z_2(\pi) = LB_2 = \sum_{j=1}^{k} \gamma_{\pi_j} U_{\pi_j}$.

Third, to solve optimality for (SP_3) to get $Z_3(\vartheta)$ by applying LA is described in section (4.1), let ϑ be the sequence obtained from LA, then:

$$Z_3(\vartheta) = LB_3 = \sum_{j=1}^{n} \omega_{\vartheta_j} V_{\vartheta_j}$$

Theorem (1) (Akturk, and Ozdemir 2001): Let P-problem consists of sum two objective functions. Let P1-subproblem depends on the 1st objective, and P2-subproblem depends on the 2nd objective then $Z_1 + Z_2 \leq Z$ where Z_1, Z_2 and Z are the minimum objective function values for the three problems respectively.

Lemma (1) (Akturk, and Ozdemir 2001): If L_i is a LB for (P_i) where $i = 1,2$, then $L_1 + L_2$ is a LB for P-problem.

Theorem (2): If L_i is a LB for (P_i) where $i = 1,2,3$, then $L_1 + L_2 + L_3$ is a LB for P-problem.

Proof: Since L_1 is a LB for subproblem-(SP1), L_2 is a LB for subproblem-(P2), and L_3 is a BL for subproblem-(P3), thus $L_1 + L_2 + +L_3 \leq Z_1 + Z_2 + Z_3$.

From Theorem (2) we have $Z_1 + Z_2 + Z_3 \leq Z$, a LB for P-problem.

Hence LB≥Z_1+Z_2+Z_3 is a LB for the P-problem since:

$$\text{LB} = \min_{\sigma \in S} \sum_{j=1}^{n} \left(\alpha_{\sigma_j} E_j + \beta_{\sigma_j} T_j + \theta_{\sigma_j} C_j + \gamma_{\sigma_j} U_j + \omega_{\sigma_j} V_{\sigma_j}\right) \geq Z_1 + Z_2 + Z_3$$

V. SOLVING FOR P-PROBLEM DEPENDING ON CEM AND BAB METHODS

The complete enumeration process yields all feasible solutions before selecting the best one. Its obvious, there are n! sequencing for SMSP with n jobs (Abdul-Razaq, 1987).

In this section, we present the BAB algorithm with new UB and LB to solve P-problem. The BAB algorithm steps are as follows:

BAB Method

Step (1): INPUT: n, p_j, d_j, and $w_j = \{\alpha_j, \beta_j, \theta_j, \gamma_j, \omega_j\}$, $1 \leq j \leq n$.

Step (2): Let $Z(\pi_i) = \sum_{j=1}^{n} (\alpha_{\pi_i j} E_{\pi_i j} + \beta_{\pi_i j} T_{\pi_i j} + \theta_{\pi_i j} C_{\pi_i j} + \gamma_{\pi_i j} U_{\pi_i j} + \omega_{\pi_i j} V_{\pi_i j})$ let π_i be the arrangement of jobs subject to the rules *R(i)*, where: *R=(SWPT, ATC, MDD, MST, EDD, LA, MA)* and $UB_i = Z(\pi_i), 1 \leq i \leq 7$.

Step (3): Set the $UB = min(UB_i)$, $1 \leq i \leq n$ at first level of BAB.

Step (4): calculate the LB(node)=cost of sequencing jobs + cost of unsequencing jobs, using the new LB.

Step (5): Branching from the node which has $LB(\text{node}) \leq UB$.

Step (6): Implementing Backtracking process to modify the UB At last level of BAB.

Step (7): If $LB \leq$ the last UB, then we decided that the last LB is an optimal solution.

Step (8): END.

A. VII. NEAR OPTIMAL SOLUTION BY USING LSM FOR P-PROBLEM

For In this section, we will discuss some LSMs (Simulated Annealing, Genetic Algorithm, Particle Swarm Optimization and BAT algorithm) for one of the types of SMSP, the POF under study and find near optimal solution for it and for large n ($n \leq 30000$) and acceptable computational time, as well as to prevent trying to solve problems that require lengthy calculations. As a result, the estimated approach can be described as a good term of capability that is used to find a near-optimal solution in a reasonable period of time (Reeves, 1997).

LSM is a family of methods that iteratively search through the set of solutions. Starting form an initial solution, a local search procedure moves from one feasible solution to a neighboring solution until some stopping criteria are met. If all of its neighbors have the same or a worse value of the objective function, a solution is said to be a local minimum with regard to a neighborhood function (Steinhöfel, et al. 2003). In the following, we will present solution approach for the problem under study.

B. Simulated Annealing Technique

This technique was created to prevent local minima. Because of two factors, it is a randomly selected LSM: First, a neighbor is chosen at random from a solution's neighborhood. Second, in addition to better-cost neighbors, which are always accepted if chosen, worse-cost neighbors are also accepted, but with decreasing probability over the course of the algorithm's execution. The randomly selected properties allow for iterative convergence to optimal solutions under some moderate conditions.

The following are the major components of simulated annealing (Gupta et al. 2002):

1. **Initialization:** The initial solution may be developed using heuristic techniques or randomly selected.

DOI: http://doi.org/10.24086/ICAFS2023/paper.906

2. **Neighborhood Generation:** There are additional methods for creating a neighborhood.
3. **Accepting Test:** The difference in value between the existing starting solution Z and the new value $\acute{Z}, \Delta = \grave{Z} - Z$, is computed and evaluated as follows:

a) If ($\Delta \leq 0$): Then, $\grave{Z}$ is accepted as the new current solution, and $Z = \grave{Z}$.
b) If ($\Delta > 0$): $\grave{Z}$ is accepted a move, when the temperature is known.

4. **Termination Test:** The algorithm is terminated after 50000 iteration at near optimal solution.

The starting solution for UB its can found by SWPT rule using its sequence $\sigma = (\sigma(1), \sigma(2), \dots, \sigma(n))$ and then calculating objective function value for the obtained sequence $f(\sigma)$, where $f(\sigma)$ is the value of objective function. The neighbor σ^* of the current solution σ is swap neighbor and compute its objective function value $f(\sigma^*)$ and m=50000 be the number of iteration, for each iteration j, $1 \leq j \leq m$.

C. Genetic Algorithms (Grefenstette, 1993)

Holland created Genetic Algorithms (GA) at 1975. The GA have been used to solve a wide range of issues, including gaming, cryptanalysis, machine learning, and COP. GAs execute a thorough search of search space using a population of candidate solutions. A collection of solutions is produced at random at first. Each of them is then assessed using a fitness function. After then, the algorithm enters a loop. A generation is any iteration in the loop that consists of two steps: selection and recombination.

The general GA is as follows

Step (1): Creating a population of m chromosomes.
Step (2): Evaluate the function $f(i)$ fitness value for each chromosome i.
Step (3): Calculate the selection probabilities $p(i)$ for each chromosome i where $p(i)$ is proportional to $f(i)$.
Step (4): Find new m children randomly from picking parents.
Step (5): GOTO Step (2) until a stopping condition is satisfied.

For the initial population (with *m*=2500) we using SWPT, EDD, MST, MA, LA, and the remaining solutions are generated randomly, and the same techniques used to find the new population, selection, crossover, mutation, and termination condition.

D. Particle Swarm Optimization (Konstantinos et al. 2010)

Based on how traveling birds find food, Eberhart and Kennedy (1995) created the particle swarm optimization (PSO) algorithm. A population member is represented in this algorithm by a particle with velocity (speed) and position. Every object in multi-faceted feature space moves by varying its position and speed in reply to its own and the population's expertise. As a result, velocity and position are updated as follows:

$$V_{ij}(t+1) = V_{ij}(t) + c_1 r_1 \left(P_{ij}(t) - X_{ij}(t)\right) + c_2 r_2 \left(P_{gj}(t) - X_{ij}(t)\right) \quad \dots(1)$$

$$X_{ij}(t+1) = X_{ij}(t) + V_{ij}(t+1) \quad \dots(2)$$

Where $i = 1,2,\dots,N, j = 1,2,\dots,n$, and n are the dimensions of the search space $A \subset R^n$; t is the iteration counter; $r_1, r_2 \in [0,1]$ are random variables. The weighting variables c_1 and c_2 are known as the social and cognitive parameters, respectively. A particle population is defined as a set $S = f[X_1, X_2, \dots, X_N]$, g of candidate solutions to an objective function to be optimized. The position of i^{th} particle be $X_i = (X_{i1}, X_{i2}, \dots, X_{in})^T \in A$ and each particle is assumed to move within the search space A iteratively. This is done by changing their position with a suitable position shift $V_i = (V_{i1}, V_{i2}, \dots, V_{in})^T$; called velocity, which is also constantly adjusted to enable particles to visit the certain location. The location in the dimension of search space with the best fitness for the particle is represented by $p_{ij}(t)$, while the best location found by the all particles is noted by $p_{gj}(t)$.

the starting population will be found as follows:

S_1: Swapping or Switching two jobs not on assignment or randomly in the sequence of s^*
S_2: Swapping or Switching two jobs (different than S_1) in the sequence of s^*.
S_3: Switch two adjacent jobs in the sequence of s^*
S_4: Arranging the first and second half of s^* in EDD rule.
S_5: Arranging the first and second half of s^* in SWPT rule.
S_6: Arranging the first and second half of s^* in MST rule.

E. Bat algorithm

Yang (2010) defined BAT as a research methodology that depends on knowledge and bat sound waves behavior. They can find their prey and many species of insects even when it is completely night. These are the only animals that have wings and extremely developed echolocation. Used the principles of pareto non-dominance and elitism to create a multi-objective extension of BAT. BATs hunt for prey by flying at random utilizing frequency, velocity, and location. The bat algorithm updates the frequency, velocity, and position of each bat in the population for future moves. Only if the new solution is approved is the pulse rate and loudness adjusted. The solutions' frequency, velocity, and location are computed. A local search strategy is performed to those solutions that fulfill a given requirement in the bat algorithm to increase the variety of viable solutions. The loudness A_i and the ulse rate r_i have to be updated in each iteration (Sheah and Abbas, 2021).

VI. COMPUTATIONAL EXPERIMENTS

A. The Problems Instances

For P-problem, we used the BAB algorithm described in section (5.2) to obtain the optimal solutions reasonable limits on computation time. The examples that are not solved optimality, we use the LSM's which mentioned in section 6. The results of the BAB method and LSMs are compared for ten (10) examples. The examples were generated randomly for $\forall j$, where $j \in \{1,2,\dots,n\}$, p_j was uniformly created in the interval [1,10] (Z field), while d_j was uniformly distributed from the interval $[(1 - TF - RDD/2)P, (1 - TF + RDD/2)P]$, where $P = \sum_{j=1}^{n} p_j$ and the parameters (*TF*) and (*RDD*)

DOI: http://doi.org/10.24086/ICAFS2023/paper.906

are said to be the (Average Tardiness Factor) in which it has the values (0.2, 0.4) and the (Related Range of Due Dates) with the values (0.2, 0.4, 0.6, 0.8, 1.0), and For each selected value of n (Monma and Potts, 1989).

The following symbols are used in the tables:
A(Z): The average optimal or near optimal values.
A(T): The average of the CPU-time (by seconds).
R: $0 \leq$ Real ≤ 1.

B. Computational Results

In this subsection, the results tables are represented, and for each table there is a brief comment for illustrating the contents.

TABLE II shows the results of the comparison between the BAB method and the CEM is reported. The problems instances are tested for n=6:10 jobs.

TABLE II
COMPARISON RESULTS OF CEM AND BAB METHODS

n	CEM		BAB	
	A(Z)	A(T)	Av(Z)	A(T)
6	349.4	R	349.4	R
7	601.9	R	601.9	R
8	699.6	R	699.6	R
9	795.5	5.0	795.5	4.1
10	1269.2	52.3	1269.2	55.3

TABLE III illustrates the results of the comparison between the BAB method and SA, Gam PSO, BAT. The problems instances are tested for n=6:10 jobs.

TABLE III
COMPARISON RESULTS OF BAB LSMS, n=6:10

n	BAB	SA	GA	PSO	BAT
6	349.4	349.4	349.4	349.4	349.4
7	601.9	601.9	601.9	601.9	601.9
8	699.6	699.6	699.6	699.6	699.6
9	795.5	797.2	795.5	795.8	796.1
10	1269.2	1273.1	1269.2	1271.5	1275.0

TABLE IV illustrates the comparative of LSM's (SA, GA, PSO, and BAT) for $n = 50,100,250,500,1000,2000,5000,10000,20000, and\ 30000$.

TABLE IV
COMPARISON RESULTS BETWEEN SA, GA, PSO, AND BAT.

n	SA		GA		PSO		BAT	
	A(Z)	A(T)	A(Z)	A(T)	Av(Z)	A(T)	A(Z)	A(T)
50	24575.9	2.2	21871.6	R	22408.6	R	24629.3	R
100	103737.3	3.3	93391.8	R	95914.9	R	103732.3	R
250	623673.2	6.4	582114.6	2.7	607694.2	R	624209.9	R
500	2425437.8	12.4	2326117.1	10.6	2421082.4	R	2425785.1	1.7
1000	9518916	23.9	9205823	38.9	9811368	R	9520030	75.3
2000	39384703.2	49.7	38696335	152.6	39705640.5	1.8	39385871.5	6.2
AV	8680173.9	16.3	8487608.9	51.2	8777351.4	1.8	8680709.9	27.7
5000	234531420	125.2	-	-	246378326.4	4.9	234532819	19.3
10000	938600102	282.6	-	-	980316892	9.9	938599675	57.1
20000	-	-	-	-	3942476042	19.9	3833442939	188.3
30000	-	-	-	-	8789498898	33.2	8337358970	363.7

Note: For unsolved examples we use the sign (-).

VII. Discussion and Conclusions

1. From table (2), we notice that the CEM and BAB methods gives an optimal solution for $n \leq 10$. Since this problem is NP-hard, we can't find an optimal solution for P-problem for $n > 10$ in reasonable time, therefore we applied LSMs to find approximate solutions.
2. The results in table (3), and from A(Z) column, we see that GA has the best performance results comparing the other three LSM's (PSO, SA, and BAT), while PSO is the best in A(T). All methods stopped for $n \leq 2000$, while PSO and BAT are stop for $n \leq 30000$.
3. As future work, we suggest to use more LSM's like Ant colony and Tabu Search to solve P-problem.

References

Aarts, E. H., & Lenstra, J. K. (2003). Local search in combinatorial optimization. Princeton University Press.

Abdul-Razaq, T. S. (1987). Machine scheduling problems: a branch and bound approach. London, Doctoral dissertation, UK: Keele University.

Abdul-Razaq, T. S., & Zeghiar, M. K. (1996). (1996). One machine scheduling to minimize total cost of completion time and number of tardy jobs. *Journal of Basrah Researchers*.

Akturk, M. S., & Ozdemir, D. (2001). A new dominance rule to minimize total weighted tardiness with unequal release dates. *European Journal of Operational Research*, 394-412.

Emmons, H. (1975). One machine sequencing to minimize mean flow time with minimum number tardy. *Naval Research Logistics Quarterly*, 585-592.

Graham, R. L., Lawler, E. L., & Lenstra, J. K. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. *In Annals of discrete mathematics, Elsevier.*, 287-326.

Grefenstette, J. J. (1993). Genetic algorithms and machine learning. *The sixth annual conference on Computational learning theory.* 3-4.

Gupta, J. N., Hennig, K., & Werner , F. (2002). Local search heuristics for two-stage flow shop problems with secondary criterion. *Computers & Operations Research*, 123-149.

Harrath, Y., Mahjoub, A., & Kaabi, J. (2019). A multi-objective genetic algorithm to solve a single machine scheduling problem with setup-times. *International Journal of Services and Operations Management*, 494-511.

Hoogeveen, H. (2005). Multicriteria scheduling. *European Journal of operational research*, 592-623.

Hoogeveen, J. A., & van deVelde, S. L. (1990). Polynomial-time algorithms for single-machine multicriteria scheduling. *Department of Operations Research, Statistics, and System Theory.*

Jackson, J. R. (1955). Scheduling a production line to minimize maximum tardiness. *Management Science Research Project.*.

Johnson, S. M. (1954). Optimal two-and three-stage production schedules with setup times included. *Naval research logistics quarterly*, 61-68.

Konstantinos, E., & Michael, N. V. (2010). Particle Swarm Optimization and Intelligence: Advances and

DOI: http://doi.org/10.24086/ICAFS2023/paper.906

Applications. *Information Science Reference, Hershey, New York*.

Lawler, E. L. (1973). Optimal sequencing of a single machine subject to precedence constraints. *Management science*, 544-546.

M'Hallah, R., & Alhajraf, A. (2016). Ant colony systems for the single-machine total weighted earliness tardiness scheduling problem. *Journal of Scheduling*, 191-205.

Mahmood, A. A. (2001). Solution procedures for scheduling job families with setups and due dates. Baghdad, M. Sc. Thesis, Iraq: University of AL-Mustansiriyah, College of Science, Dept. of Mathematics.

Monma, C. L., & Potts, C. N. (1989). On the complexity of scheduling with batch setup times. *Operations research*, 798-804.

Phrueksanant, J. (2013). Machine scheduling using the Bees Algorithm. *Doctoral dissertation*. United Kingdom: Cardiff University.

Reeves, C. (1997). Modern heuristics techniques for combinatorial problems. *Nikkan Kogyo Shimbun*.

Sheah, R. H., & Abbas, I. T. (2021). Using Multi-Objective Bat Algorithm for Solving Multi-Objective Non-linear Programming Problem. *Iraqi Journal of Science*, 997-1015.

Smith, W. E. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 59-66.

Steinhöfel, K., Albrech, A., & Wong, C. K. (2003). An experimental analysis of local minima to improve neighbourhood search. *Computers & Operations Research*, 2157-2173.

DOI: http://doi.org/10.24086/ICAFS2023/paper.906