Available online at www.qu.edu.iq/journalcm
JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS
ISSN:2521-3504(online) ISSN:2074-0204(print)

# Solving Multi-Objectives Function with Release Dates Problem using Branch and Bound Methods

***Omar osama daowd [a], Professor Dr.Hanan Ali chachan [b] ,Professor Dr. Hazim G. Daway [c]***

[a]*Department of Mathematics,College of sciences,Mustansireah University,Baghdad Iraq, omarmaysan@yahoo.com*

[b]*Department of Mathematics,College of sciences,Mustansireah University,Baghdad Iraq, hanomh@uomustansiriyah.edu.iq*

[c]*Department of Physics,College of sciences,Mustansireah University,Baghdad Iraq, hazimdo@uomustansiriyah.edu.iq*

ARTICLE INFO

*Article history:*
Received: 10 /04/2022
Rrevised form: 01 /06/2022
Accepted : 12 /06/2022
Available online: 14 /06/2022

*Keywords:*

Multi-Objective Problem (MOP), Branch and Bound (BAB) method, unequal release times, Gray Wolfe algorithm.

ABSTRACT

**This research study focuses on solving a multi-objective job scheduling problem on a single machine. This problem involves the summation of four objective functions namely the total flow time ($F_J$), tardiness ($T_J$), earliness ($E_J$), and late work ($V_J$) of each of the n jobs with unequal release dates $r_J$, $j = 1, \cdots, n$. This is formulated as $1/r_j/\sum_{j=1}^{n}(F_J + T_J + E_J + V_J)$.**

**We present some special cases that yield optimal solutions. also, we propose a branch and bound algorithm in order to find the exact (optimal) solution for it, by derive and use a good lower, and using some heuristics methods to find the upper bound of seven including Gray Wolfe algorithm (GW), and Bat algorithm (BAT), this algorithm was given to find the optimal solution for problems of size up to 17 functions.**

MSC..

https://doi.org/ 10.29304/jqcm.2022.14.2.948

## 1. Introduction

### 1.1. Scheduling Importance

In this subsection, the problem of scheduling multiple jobs on a single machine using several performance metrics and varying or unequal release dates $r_J$, $j = 1, \cdots, n$ is described. The goal is to minimize the total flow time ($F_J$), tardiness ($T_J$), earliness ($E_J$), and late work ($V_J$), and optimize the overall performance of the machine. The problem is denoted by $1/r_j/\sum_{j=1}^{n}(F_J + T_J + E_J + V_J)$, and

∗Corresponding author : omar osama dawod

Email addresses: omarmaysan@yahoo.com

Communicated by 'sub etitor' : Hanan Ali Ghachan , Hazim G. Daway

it is drawn from the 3-filed $\alpha /\beta/ \gamma$ approach by Graham et al. (1979). The thrust of early research on the theory of job scheduling was based mainly on a single performance criterion, which may not effectively capture the reality in applications both in industrial production and service delivery. Due to the growing importance of scheduling in decision-making in real-world applications, there is a need for a multi-criteria assessment of performance and job quality (Chen and Sheen, 2007). Furthermore, the production philosophy referred to as just-in-time (JIT) is based on the premise that a sense of urgency and tardiness may be damaging, for example, tardiness can result in customer loss or payment delays. Earlyness, on the other hand, results in inventory carrying costs, insurance, and other expenses (Ali and Mehdi, 2012). These and several other factors have drawn the attention of researchers to the investigation of the scheduling problem involving several criteria (two or more), and/ or multi-processing time on more than one machine. In Belbachir et al. (2021), the flow shop scheduling problem was considered, while in this research, we look at the scheduling problem using four criteria (mentioned above) for evaluating performance with unequal release times, and a to the best of our knowledge, research is still lacking in this area.

### 1.2. Scheduling History:

The earliest mention of the scheduling problem in the literature dates back to the mid-1950's (1954-1956) (Johnson, 1954; Smith, 1956). The intervening years between 1956 and 1973 did not indicate significant activity until 1973 when Lawler (1973) suggested a method for reducing the maximum cost $(f_{max})$. After that, scheduling difficulties sparked a lot of attention, resulting in a significant number of studies introducing excellent approaches for determining optimality. Many researchers focused on scheduling issues as an objective criterion in subsequent years. However, the expansion in manufacturing businesses, services such as airlines, health care, and even military operations, provide real-life ramifications to study the multi-criteria scheduling problem. In addition, the complexity and diversity inherent in resource allocation in the real-world has stimulated growth in

the new field of multi-criteria scheduling. The minimization of a maximum function (Minimax) and minimization of a sum function (Minisum) are the two types of objective criteria that can be applied in this regard (T'kindt and Billaut, 2002). As a result, research in the field of multi-criteria scheduling has been expanding since the 1980s till now.

### 1.3 Problem Representation:

The problem addressed in this work is that of scheduling the set $N$, of n jobs, $N = \{1, \dots, n\}$ on a single machine. Each job $j$, $j \in N$ has a processed time that is an integer $p_j$, a release date $r_j$, and due date $d_j$. Given a schedule $\sigma = (1, \dots, n)$, the flow time of the job $j$ , $F_j$ can be defined as $F_j = C_j - r_j$ where $C_j$ is the completion time for job $j$, and is given as

$$C_1 = r_1 + p_1 \, , C_j = \max\{r_j, C_j\} + p_j, \qquad \text{for } j = 2 \, , \dots , n.$$

The tardiness of the $j^{th}$ job, is defined by $T_j = \max\{C_j - d_j, 0\}$ , and earliness by $E_j = \max\{d_j - C_j, 0\}$. The corresponding late work of job j given by $V_j = min\{T_j, P_j\}$. Let is a schedule in and be a set of all viable solutions. Our issue (P) has the following mathematical form:

$$M = \text{Min } F(\sigma) = \text{Min}_{\sigma \epsilon \delta} \{ \sum_{j=1}^{n} ( F_{\sigma(j)} + T_{\sigma(j)} + E_{\sigma(j)} + V_{\sigma(j)} ) \}$$

Subject to:

$$\left.\begin{aligned}
& C_{\sigma(1)} = r_{\sigma(1)} + \rho_{\sigma(1)} \\
& C_{\sigma(j)} = Max\{r_{\sigma(j)} \, , C_{\sigma(j-1)}\} + \rho_{\sigma(j)} \qquad j = 2, \dots, n \\
& \mathcal{T}_{\sigma(j)} = Max\{C_{\sigma(j)} - d_{\sigma(j)}, 0\} \qquad j = 1, \dots, n \\
& E_{\sigma(j)} = Max\{d_{\sigma(j)} - C_{\sigma(j)}, 0\} \qquad j = 1, \dots, n \\
& \mathcal{V}_{\sigma(j)} = Min\{\mathcal{T}_{\sigma(j)} \, , \rho_{\sigma(j)}\}
\end{aligned}\right\} \quad \dots(P)$$

The purpose is to find a processing sequence $\sigma = (\sigma(1), \ldots, \sigma(n))$ that will minimize the sum of the overall flow times, total tardiness, total earliness, and total late work for the problem (P).

## 2. Materials and Methods

In this section, an overview of the methods employed to solve the challenge (P) is presented. The branch and bound (BAB) method is the major approach for solving the problem, whereas the bat algorithm (BA) and the Gray Wolf algorithm (GW) are used to acquire upper bounds from the exact methods.

### 2.1. BAB Method

By undertaking an implicit enumeration of all possi8ble solutions in the solution set (i.e. evaluating subsets of the solution set sequentially), the branch and bound (BAB) algorithm discovers the optimal subsets. These subsets can be viewed as sets of solutions to corresponding sub-problems of the main problem. As a result, the (BAB) method is employed to determine an exact solution that optimizes (minimizes) the problem (Blazewicz et al., 2007).

A procedure for the (BAB) is introduced in this study by suggesting severl upper bounds and a lower bound.

#### 2.1.1. Upper bounds

Seven upper bounds shall be presented in this subsection, namely

1- $\boldsymbol{UB_1}$: SPT., sort the jobs in a non-decreasing order of processing time ($P_j$).

2- $\boldsymbol{UB_2}$: LRT., sort the jobs by large release time ($R_j$).

3- $\boldsymbol{UB_3}$: MRT., sort the jobs by minimum release time ($R_j$).

4- $\boldsymbol{UB_4}$: LW., sort the jobs using the Lawler algorithm.

5- $\boldsymbol{UB_5}$: MPRT., sort the jobs by the minimum of the sum of processing time and release time $(P_j + R_j)$.

6- $\boldsymbol{UB_6}$: determine the upper bound by applying the bat algorithm (BA).

**Bat Algorithm (BA)**

Due to the basic rules of Bat algorithm (BA) in chapter two, the parameter x and v refers to position and velocity of solution respectively, in our work we will regard these parameters as a control parameters to choose method of transfer to next neighborhood, and the loudness as a parameter in accepting the movement, we start by initialize the value of $x$ , $v$, and the loudness $A$ , by a random values in [0,1], also start by population of bats (schedules) of size $CN$, some of these schedules are known (or/and) several are random, the procedure start by compute the value of cost function for all schedules in this population and chose the schedule with minimum value to be the Global bat algorithm solution ($S^*$), and then for each solution ($s$) in the population, we use its cost value, with the parameter $x\ and\ v$ to find, its neighborhood (new solution). the condition of accepting this new solution ($s'$) to be the current solution ($s$) is depending on the cost value of ($s'$) and the parameter of the loudness ($A$), this procedure will be repeated iteratively for $N - times$ ($N$ the number of iterations), at each iteration the value of $x\ and\ v$ are updated, and the value of $A$ will be decreasing, and save the best solution found yet as local Bat solution, and check if it is best than the global solution then update $S^*$. the following equation describe the updating the value of both $x\ and\ v$,

$$v(t) = v(t-1) + \left(x^*(t) - Bval * x(t-1)\right) * f(t).$$

$$x(t) = x^*(t-1) + v(t).$$

Where: $Bval$ is the current global solution. and

$$x^*(t) = x(t-1) * val.$$

$$f(t) = fmin + (fmax - fmin) * B(t).$$

$$R(t) = R(t-1)(1 - e^{-\gamma t}).$$

$$A(t) = A(t-1) * \alpha \frac{Bval}{val}.$$

Where: $val$ is the current local solution, $B(t)$ is a random vector, $\alpha, \epsilon \ and \ \gamma \in (0,1)$.

The method of generate the new solution depend on the value of loudness $A$ and position $x$, that is: $if\ A(t) < \alpha\ *\ A(t-1)\ or\ \frac{x(t)}{x(t-1)} > 1 + \epsilon$ , then, generate the new solution by swap any two jobs in old sequence, otherwise, generate the new solution by swap any pair of two jobs by other pair in old schedule, i.e., exchange the position of four jobs.

Finally, we accept the new sequence $s'$ as a current solution if the cost of new sequence less than the cost current solution $s$, or $if\ \frac{Bval}{val} > R(t)$.

The algorithm terminates after finish all iterations $CN\ and\ N$, or if the procedure exceeds fixed period of time.

In our work we initialize the value of parameters as follows:

For the parameters , $\epsilon, \gamma \in (0,1)$, $\alpha = 0.9$ , $\epsilon = 0.125$ , $and\ \ \gamma = 0.9$ , and for the number of initial solutions (population size $CN$ ) $CN = 50$, that iterative by the index $j = 1{:}\,CN$.

and for the number of iterations $(N)$ , $N = 200$, that iterative by the index $i = 1{:}\,N$.

The algorithm terminates after finish all iterations $CN = 50\ and\ N = 200$, or if the procedure exceeds 10 minutes (600 seconds).

In next figure we introduce the Flow chart of the Bat algorithms:

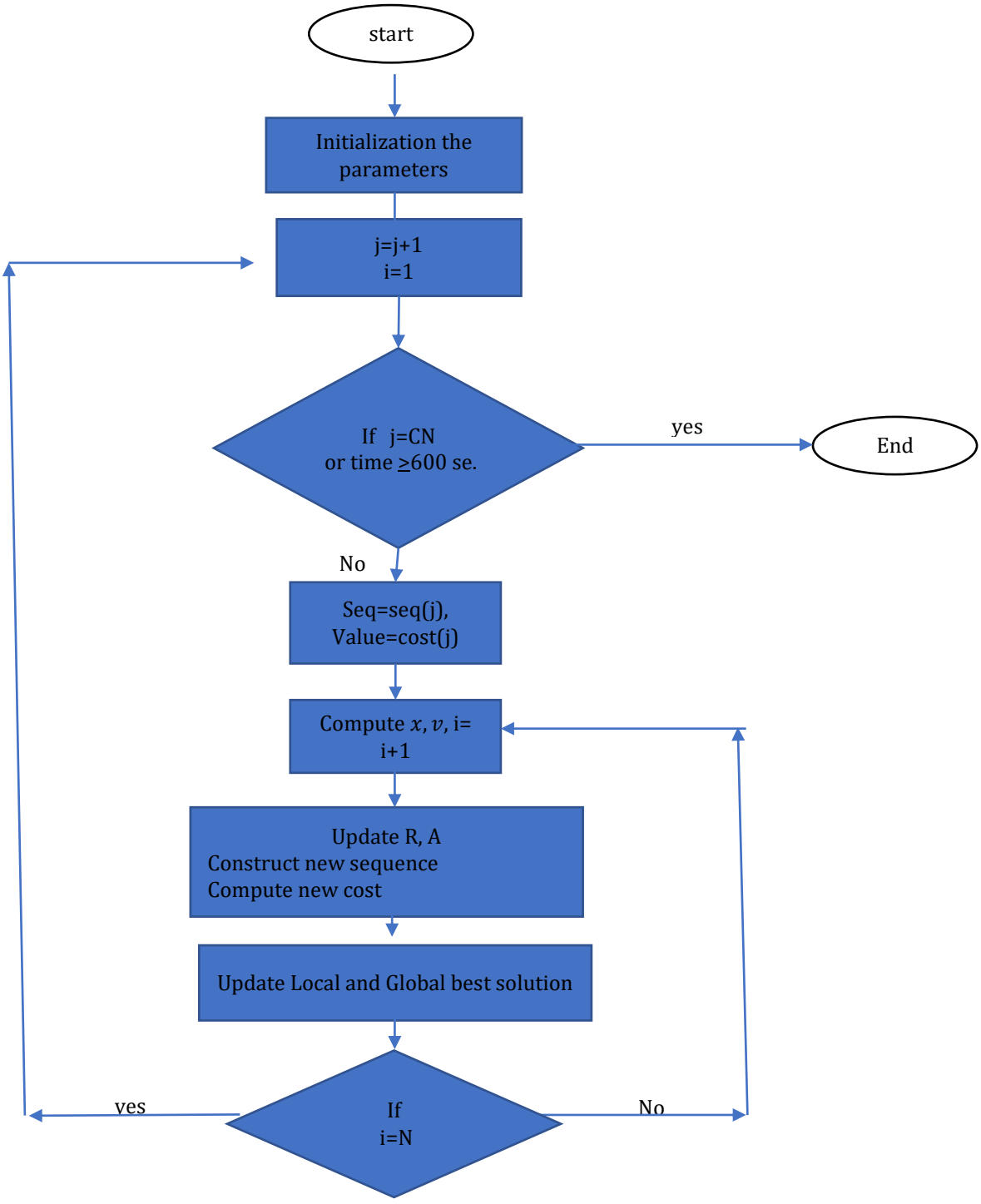


Figure 1. Flowchart of Bat algorithm

7- $\boldsymbol{UB_7}$**:** "Finally, this upper bound is found by applying" (GW) The Gray Wolf Algorithm

**Gray Wolf Algorithm**

To adapting the GW algorithm in order to appley it in machine scheduling theory, for any group of wolf with size $N$, we have to consider each wolf in this group represent a one solution, and then find its cost of objective function value, by reordering the solutions in group by nondecreasing order of cost function value, we get the classification of the group, into three subgroup $A, B, and\ D$, and save the best solution found yet, then by special considered swarm technics we construct a new group of solutions (wolfs) by using the subgroups $A, B, and\ D$, and then evaluate the new group by compute the values of cost function for this new population (group) and rearrangement it according to these values to find the subgroup $A, B, and\ D$, again, and update the best solution found yet if any. repeat this process until fixed number of iterations ($iter.$) is reached, or after a fixed period of time, the last best solution is the GW solution. The suggested the parameters and steps of above procedure as follows:

Step 1:

initialize the iterations $iter = 50$, the size of group $N = 50$, and the first random group 50 solutions, (the five special solutions - SPT, MRT, MPRT, LW, and LRT - are insert in this group).

Step 2:

Evaluation the solutions in group, and classifies into subgroups $A, B, and\ D$, which is the first, second and third solution.

Step 3:

Enhance the solution $A, B, and\ D$, until 50-times, and save the best solution founded yet and update the best GW solution if the new solution is dominated, and then construct new group (population of 50 solution), by made crossover between $A, B\ and\ D$. And add some random solutions.

Step 4: if any of termination criteria (complete all iterations or exceed 10 minute), not met go to step 2; else stop and consider the last best solution as the GW solution.

In next figure we introduce the Flow chart of the Gray Wolfs algorithms:

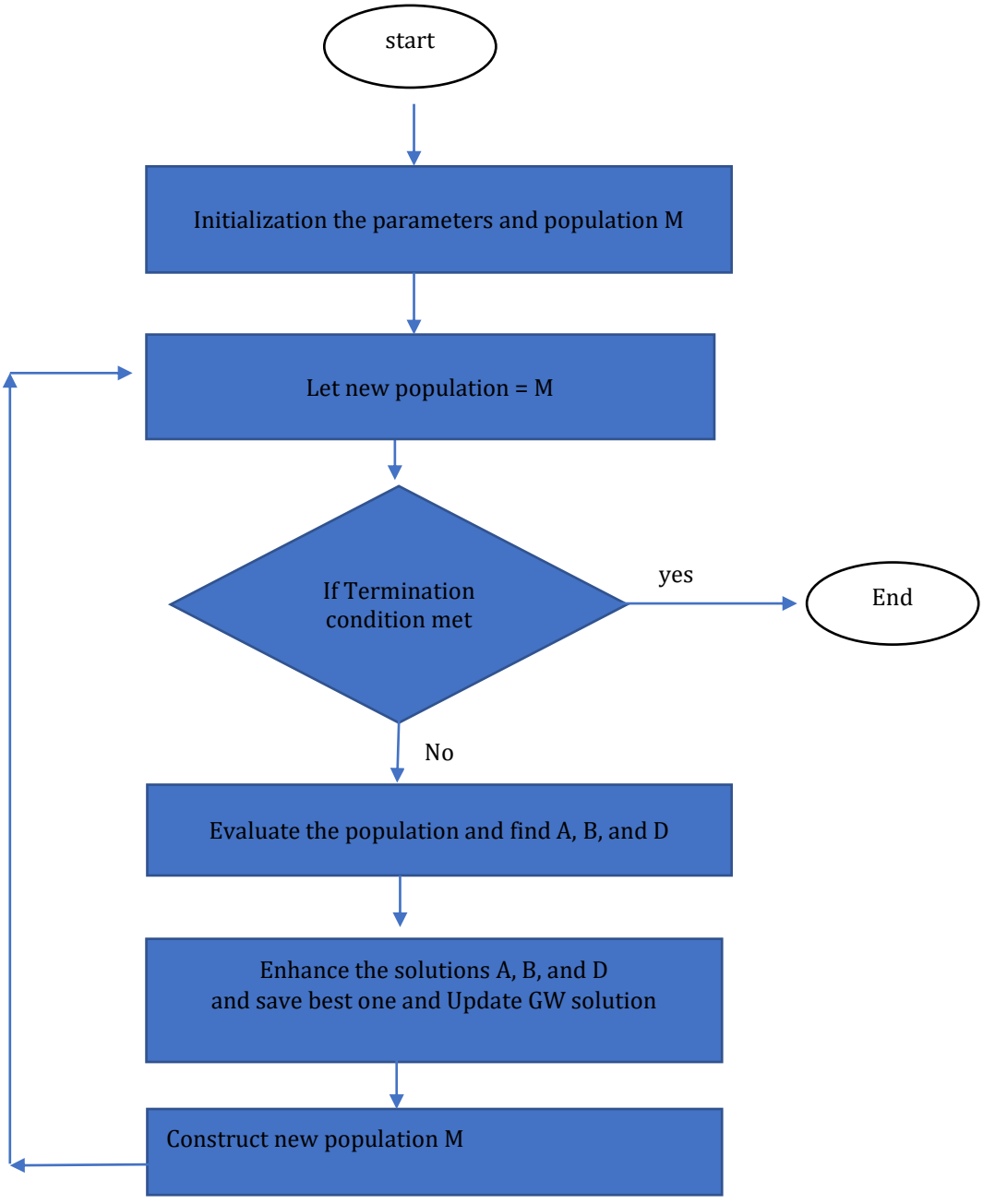


Figure 2. Flowchart for the Gray Wolfe algorithm.

### 2.1.2 Lower bound

The optimization problem given by (P) can be decomposed into two subproblems $P_1$ and $P_2$ so as to have a less complicated structure. The two subproblems are given as

$$m1 = Min_{\sigma\epsilon\delta}\{\sum_{j=1}^{n}(F_{\sigma(j)} + T_{\sigma(j)} + E_{\sigma(j)})\}$$

Subject to:

$$C_{\sigma(1)} = r_{\sigma(1)} + \rho_{\sigma(1)} \qquad i=1$$

$$C_{\sigma(j)} = Max\{r_{\sigma(j)}, C_{\sigma(j-1)}\} + \rho_{\sigma(j)} \quad , \quad j = 2, \dots, n$$

$$\mathcal{T}_{\sigma(j)} = Max\{C_{\sigma(j)} - d_{\sigma(j)}, 0\} \quad , \quad j = 1, \dots, n$$

$$E_{\sigma(j)} = Max\{d_{\sigma(j)} - C_{\sigma(j)}, 0\} \qquad j = 1, \dots, n$$

...($P_1$)

and

$$m_2 = Min_{\sigma\epsilon\delta}\ (\sum_{j=1}^{n} \mathcal{V}_{\sigma(j)})$$

Subject to:

$$C_{\sigma(1)} = r_{\sigma(1)} + \rho_{\sigma(1)} \qquad i = 1$$

$$C_{\sigma(j)} = Max\{r_{\sigma(j)}, C_{\sigma(j-1)}\} + \rho_{\sigma(j)} \qquad j = 2, \dots, n$$

$$\mathcal{T}_{\sigma(j)} \geq 0 \qquad j = 1, \dots, n$$

$$\rho_{\sigma(j)} > 0 \qquad j = 1, \dots, n$$

$$\mathcal{V}_{\sigma(j)} = Min\{\mathcal{T}_{\sigma(j)}, \rho_{\sigma(j)}\} \qquad j = 1, \dots, n$$

.... ($P_2$)

Denote the lower bounds of the subproblems P1 and P1 as LB1 and LB2 respectively. The two lower bounds LB1 and LB2 are obtained such that the aggregate of LB1 and LB2 becomes a lower bound LB for the whole problem P, LB=LB1+LB2. When S is the set of all feasible solutions, and $\delta$ is a subset of S.

1- Lower bound for subproblem P1 (LB1)

The subproblem P1 can be given as

$$Min\ Z(\delta) = \underset{\delta \in S}{Min}\left\{\sum_{j=1}^{n}\left(E_{\delta_j} + T_{\delta_j} + F_{\delta_j}\right)\right\}$$

$$= \underset{\delta \in S}{Min}\sum_{j=1}^{n}\left\{Max\left\{d_{\delta_j} - C_{\delta_j}, 0\right\} + Max\left\{C_{\delta_j} - d_{\delta_j}, 0\right\} + \left\{C_{\delta_j} - r_{\delta_j}\right\}\right\}$$

$$= \underset{\delta \in S}{Min}\sum_{j=1}^{n}\left\{Max\left\{d_{\delta_j} - C_{\delta_j}, C_{\delta_j} - d_{\delta_j}, 0\right\} + \left\{C_{\delta_j} - r_{\delta_j}\right\}\right\}$$

$$= \underset{\delta \in S}{Min}\sum_{j=1}^{n}\left\{Max\left\{d_{\delta_j} - C_{\delta_j} + C_{\delta_j} - r_{\delta_j}, C_{\delta_j} - d_{\delta_j} + C_{\delta_j} - r_{\delta_j}, C_{\delta_j} - r_{\delta_j}\right\}\right\}$$

$$= \underset{\delta \in S}{Min}\sum_{j=1}^{n}\left\{Max\left\{d_{\delta_j} - r_{\delta_j}, 2C_{\delta_j} - d_{\delta_j} - r_{\delta_j}, C_{\delta_j} - r_{\delta_j}\right\}\right\} \dots\dots\dots\dots (1)$$

Since the third term $C_{\delta_j} - r_{\delta_j}$ is always between $d_{\delta_j} - r_{\delta_j}$ and $2C_{\delta_j} - d_{\delta_j} - r_{\delta_j}$ , then we can write the objective function $Z(\delta)$ as:

$$= \underset{\delta \in S}{Min}\sum_{j=1}^{n}\left\{Max\left\{d_{\delta_j} - r_{\delta_j}, 2C_{\delta_j} - d_{\delta_j} - r_{\delta_j}\right\}\right\} \dots\dots\dots\dots (2)$$

This means that the cost of scheduling job $\delta_j$ is $(\delta_j)$ , given by:

$$Z(\delta_j) = \begin{cases} d_{\delta_j} - r_{\delta_j} & if \quad C_{\delta_j} - r_{\delta_j} \le d_{\delta_j} - r_{\delta_j} \\ 2C_{\delta_j} - d_{\delta_j} - r_{\delta_j} & otherwise \end{cases}$$

*i.e.*, $Z(\delta_j)$ is equal to $d_{\delta_j}$ if job j is early and $Z(\delta_j)$ is equal to $2C_{\delta_j} - d_{\delta_j} - r_{\delta_j}$ if job j is tardy.

Also, we can write the objective of the P1 in other from as the following:

Let

$$ER = \{j: j \in \delta\ , C_j - r_j \leq d_j - r_j\}\ and$$

$LT = \{j: j \in \delta\ , C_j - r_j > d_j - r_j\}$ so that:

$$\sum_{j\in\delta}(E_j + T_j + F_j) = \sum_{j\in\delta}(E_j + T_j + C_j - r_j)$$

$$= \sum_{j\in ER}(E_j + C_j - r_j) + \sum_{j\in LT}(T_j + C_j - r_j)$$

$$= \sum_{j\in ER}(d_j - C_j + C_j - r_j) + \sum_{j\in LT}(T_j + C_j - r_j - d_j + d_j)$$

$$= \sum_{j\in ER} d_j - r_j + 2\sum_{j\in LT} T_j + \sum_{j\in LT} d_j - r_j \quad \ldots\ldots\ldots\ldots\ldots (3)$$

It is clear from equation $(1)$ that a lower bound $(LB1)$ is obtained by sequencing the jobs by SPT rule.

Hence, we can prove that:

$$Min\ Z(\delta) \geq \underset{\delta\in S}{Min}\left\{Max\left\{\sum_{j=1}^{n} d_{\delta_j} - r_{\delta_j}\ , \sum_{j=1}^{n} Max\left\{2C_{\delta_j} - d_{\delta_j} - r_{\delta_j}\ , C_{\delta_j} - r_{\delta_j}\right\}\right\}\right\}$$

It is an LB1 for problem P1 since:

$$\underset{\delta\in S}{Min}\sum_{j=1}^{n} Max\left\{d_{\delta_j}-r_{\delta_j}\,,2C_{\delta_j}-d_{\delta_j}-r_{\delta_j}\,,C_{\delta_j}-r_{\delta_j}\right\}$$

$$\geq \underset{\delta\in S}{Min}\left\{Max\left\{\sum_{j=1}^{n} d_{\delta_j}-r_{\delta_j}\,,\sum_{j=1}^{n} Max\left\{2C_{\delta_j}-d_{\delta_j}-r_{\delta_j}\,,C_{\delta_j}-r_{\delta_j}\right\}\right\}\right\}$$

Put $y_{\delta_j}=Max\left\{2C_{\delta_j}-d_{\delta_j}-r_{\delta_j}\,,C_{\delta_j}-r_{\delta_j}\right\}$

To show that

$$\underset{\delta\in S}{Min}\sum_{j=1}^{n} Max\left\{d_{\delta_j}-r_{\delta_j}\,,y_{\delta_j}\right\}\geq \underset{\delta\in S}{Min}\left\{Max\left\{\sum_{j=1}^{n} d_{\delta_j}-r_{\delta_j}\,,\sum_{j=1}^{n} y_{\delta_j}\right\}\right\}$$

Since $d_{\delta_j}-r_{\delta_j}$ and $y_{\delta_j}$ are positive integers, hence it is clear that

$$\underset{\delta\in S}{Min}\sum_{j=1}^{n} Max\left\{d_{\delta_j}-r_{\delta_j}\,,2C_{\delta_j}-d_{\delta_j}-r_{\delta_j}\,,C_{\delta_j}-r_{\delta_j}\right\}$$

$$\geq \underset{\delta\in S}{Min}\left\{Max\left\{\sum_{j=1}^{n} d_{\delta_j}-r_{\delta_j}\,,\sum_{j=1}^{n} Max\left\{2C_{\delta_j}-d_{\delta_j}-r_{\delta_j}\,,C_{\delta_j}-r_{\delta_j}\right\}\right\}\right\}$$

$$\therefore\ LB=\underset{\delta\in S}{Min}\left\{Max\left\{\sum_{j=1}^{n} d_{\delta_j}-r_{\delta_j}\,,\sum_{j=1}^{n} Max\left\{2C_{\delta_j}-d_{\delta_j}-r_{\delta_j}\,,C_{\delta_j}-r_{\delta_j}\right\}\right\}\right\}\ldots(4)$$

Hence a lower bound (LB1) for our (P1).

2- Lower bound for subproblem P2 (LB2)

The subproblem P2 is given as

$$Min\ w(\delta) = Min_{\delta \epsilon S}\ (\sum_{j=1}^{n} \mathcal{V}_{\delta_{(j)}})$$

Clear that:

$$Vmax(\delta) \leq \sum_{j=1}^{n} \mathcal{V}_{\delta_{(j)}} \quad \forall\ \delta \epsilon S$$

$\because Vmax(lw) \leq Vmax(\delta),\ \ \forall\ \delta \epsilon S.$ when $lw$ is the Lowler sequence for Vmax.

$$\therefore\ Vmax(lw) \leq \sum_{j=1}^{n} \mathcal{V}_{\delta_{(j)}} \quad \forall\ \delta \epsilon S.$$

So that LB2= $Vmax(lw)$ is a lower bound for problem P2. As we mention above LB=LB1 + LB2 is a Lower bound for problem P.

LB is used in BAB to find the exact solution for our (P).

## 3. Results

### 3.1 Experimental Results:

In this section, the optimal value, upper bound (UB), lower bound (LB), number of created nodes (nodes), processing time (time), and number of unsolved problems are computed. These were the factors considered in this study. To determine the number of unsolved problems, there are two criteria for stopping the BAB algorithm and concluding that the problem is unsolved. These are based on stopping the BAB algorithm either after a fixed number of nodes or a fixed time period. The second criteria was used in this study and the procedure was terminated after (30) minutes. We list (10) examples for each value of (n), where n ∈ {3,... ,19}. Also in these tables: LB=the lower bound. ILB = initial lower bound. UB = upper bound. Optimal = the optimal value, NOON = number of open nodes, time =computational time in seconds, Status 1 if example solved 0 if not. As we will explain later our algorithm, can solves

the problem P up to size of 18 jobs, the next tables 1 to 4 contained 10 examples for n= {5, 9, 13, 17} and the table 5 involve the summary of results for each integer n in the interval [3,19].

### 3.2 Analysis and Problems Instances

The BAB algorithm was applied to the optimization problem (P) with four criteria to find optimal solution. The algorithm coding was achieved using MATLAB, and tested on the optimization problem (P) with (4, 5, ..., 30) jobs. The test problems were generated as follows: For each job j: an integer processing time $P_j$ is generated from the uniform distribution [1,10]; an integer release time $r_j$, is generated from the uniform distribution [1,5]; an integer due date $d_i$ are uniformly distributed in the interval [A, B], where

$$A= P\left(1-TF-\frac{RDD}{2}\right),\ B=P\left(1+ TF+\frac{RDD}{2}\right),\ \text{where } P=\sum_{j=1}^{n} P_j \ .$$

depends on the average tardiness and the relative range of due dates (RDD) (TF). The values 0.2, 0.4, 0.6, 0.8, and 1 were taken into account for both parameters. Two problems were generated for each of the five parameter values for each value of n, resulting in ten problems for each value of n.

### 3.3. Computational results

The computational results from the study are presented in Tables 1-5 for different configurations of n, the number of jobs. Each table contains the results (i.e., optimal values by (BAB), the problem's upper bound, the problem's starting lower bound, the number of nodes, and the execution time). Ten tasks are checked for each n, with a maximum execution time of 1800 seconds as the halting condition. The symbols used in the tables are as follows:

**n**: number of jobs

**Ex**: number of examples.

**Opt(BAB) : t**he optimal value of the function using BAB algorithm.

**UB**: the upper bound of the problem.

**ILB**: the initial lower bound of the problem.

**NOON** : number of open nodes.

**Time:** The problem's total execution time (by seconds).

**NOSP:** number of solved problems.

Table 1. Result of BAB when n=5

| Table (1) | | | | | | |
|---|---|---|---|---|---|---|
| EX | Opt(BAB) | UB | ILB | NOON | Time | Status |
| 1 | 152 | 152 | 132 | 83 | 0.00363 | 1 |
| 2 | 132 | 162 | 121 | 17 | 0.001456 | 1 |
| 3 | 108 | 109 | 95 | 48 | 0.002207 | 1 |
| 4 | 160 | 160 | 142 | 37 | 0.001804 | 1 |
| 5 | 139 | 139 | 110 | 63 | 0.002125 | 1 |
| 6 | 163 | 163 | 135 | 64 | 0.002124 | 1 |
| 7 | 101 | 101 | 81 | 36 | 0.001682 | 1 |
| 8 | 104 | 113 | 83 | 45 | 0.001692 | 1 |
| 9 | 140 | 140 | 111 | 72 | 0.002471 | 1 |
| 10 | 116 | 119 | 98 | 73 | 0.002519 | 1 |

Table 2. Result of BAB when n=9

| Table (2) | | | | | | |
|---|---|---|---|---|---|---|
| EX | Opt(BAB) | UB | ILB | NOON | Time | Status |
| 1 | 353 | 410 | 310 | 1019 | 0.029144 | 1 |
| 2 | 349 | 372 | 309 | 1684 | 0.044841 | 1 |
| 3 | 396 | 411 | 367 | 1057 | 0.028392 | 1 |
| 4 | 423 | 434 | 379 | 5736 | 0.15634 | 1 |
| 5 | 123 | 139 | 104 | 2190 | 0.061093 | 1 |
| 6 | 340 | 370 | 296 | 3852 | 0.11829 | 1 |
| 7 | 366 | 394 | 345 | 1177 | 0.051156 | 1 |
| 8 | 359 | 373 | 307 | 3826 | 0.134008 | 1 |
| 9 | 312 | 320 | 282 | 3558 | 0.130299 | 1 |
| 10 | 270 | 296 | 243 | 1701 | 0.051617 | 1 |

Table 3. Result of BAB when n=13

| Table (3) | | | | | | |
|---|---|---|---|---|---|---|
| EX | Opt(BAB) | UB | ILB | NOON | Time | Status |
| 1 | 604 | 616 | 548 | 148079 | 3.7590673 | 1 |
| 2 | 765 | 842 | 645 | 1008184 | 26.59887 | 1 |
| 3 | 651 | 671 | 588 | 413609 | 11.295591 | 1 |
| 4 | 752 | 809 | 712 | 400864 | 10.244974 | 1 |
| 5 | 703 | 796 | 647 | 150205 | 4.0155927 | 1 |
| 6 | 782 | 796 | 720 | 371568 | 9.8067975 | 1 |
| 7 | 590 | 605 | 544 | 175043 | 5.0673961 | 1 |
| 8 | 486 | 541 | 425 | 710207 | 17.875304 | 1 |
| 9 | 559 | 616 | 484 | 241975 | 6.0155772 | 1 |
| 10 | 789 | 819 | 692 | 783452 | 19.866373 | 1 |

Table 4. Result of BAB when n=17

| Table (4) N=17 | | | | | | |
|---|---|---|---|---|---|---|
| EX | Opt(BAB) | UB | ILB | NOON | Time | Status |
| 1 | 1347 | 1388 | 1262 | 71980503 | 1800.0002 | 0 |
| 2 | 1259 | 1340 | 1164 | 40525868 | 1108.72235 | 1 |
| 3 | 1152 | 1187 | 1081 | 43561393 | 1478.1002 | 1 |
| 4 | 999 | 1067 | 895 | 30835918 | 915.339141 | 1 |
| 5 | 1356 | 1394 | 1266 | 60554855 | 1532.7624 | 1 |
| 6 | 886 | 960 | 815 | 15644304 | 397.748429 | 1 |
| 7 | 1144 | 1216 | 1062 | 14546635 | 371.420793 | 1 |
| 8 | 1012 | 1073 | 924 | 19894958 | 496.366831 | 1 |
| 9 | 879 | 920 | 801 | 62964586 | 1574.74239 | 1 |
| 10 | 1043 | 1249 | 961 | 2862186 | 72.0961537 | 1 |

Table 5. Result of BAB when n=3:19

| Table (5) | | | | | | |
|---|---|---|---|---|---|---|
| N | Opt(BAB) | UB | ILB | NOON | Time | NOSP |
| 3 | 56.5 | 58.4 | 46.5 | 8.5 | 0.00070198 | 10 |
| 4 | 74.3 | 76.5 | 61.3 | 23.4 | 0.00204054 | 10 |
| 5 | 131.5 | 135.8 | 110.8 | 53.8 | 0.0021709 | 10 |
| 6 | 212.7 | 213.8 | 179.1 | 224 | 0.00860399 | 10 |
| 7 | 173.7 | 181.3 | 147.6 | 382.3 | 0.01326462 | 10 |
| 8 | 266.6 | 281.5 | 231.2 | 1157.7 | 0.03388743 | 10 |
| 9 | 329.1 | 351.9 | 294.2 | 2580 | 0.08051798 | 10 |
| 10 | 439.4 | 459.6 | 393.2 | 16763.9 | 0.43654682 | 10 |
| 11 | 515.7 | 548.4 | 468.2 | 26311.8 | 0.71140979 | 10 |
| 12 | 614.8 | 642.6 | 553.5 | 267439 | 7.3783194 | 10 |
| 13 | 668.1 | 711.1 | 600.5 | 440318.6 | 11.4545542 | 10 |
| 14 | 854.3 | 900 | 779.6 | 1304810.5 | 34.23946997 | 10 |
| 15 | 921.6 | 1006.1 | 844.7 | 6555583.8 | 174.2030229 | 10 |
| 16 | 946.3 | 980.2 | 867.5 | 28775867.5 | 716.5200295 | 10 |
| 17 | 1107.7 | 1179.4 | 1023.1 | 36337120.6 | 974.7298887 | 9 |
| 18 | 1150.2 | 1231.2 | 1054 | 53992822.5 | 1371.172138 | 4 |
| 19 | 1388.5 | 1484.7 | 1278.4 | 65163747 | 1720.651798 | 1 |

## 4. Discussion

Due to official standard, that consider the BAB algorithm get the optimal solution if it completes all procedures by time less than 0.5-hour i.e. (1800 second), for each example, and we say that the BAB algorithm can solve the problem of size (n), if it can solve %50 from the considered examples from the size (n). So, for our problem we get the BAB active until n=17, from above tables 4.1-4.5, we see that the algorithm can solve 10/10 examples from n=3 to 16 while it solves 9/10 for n=17 due to time condition that related to number of open nodes, that increased in problem of size n=18, so that BAB solves just 4/10 problems therefore it is cannot solve P for n=18, same as for n=19 it is solve only one example from the considered 10 examples, consequence from this two cases the algorithm will

consume more time, (more than 1800 second) to finished its procedure for any $n \geq 18$, as a result the BAB algorithm with the lower bound that introduced in the previous section guaranteed to solve problem P of size $n \leq 17$.

## References


1- Ali, Simin Poursheikh, and Mehdi Bijari (2012). Minimizing maximum earliness and tardiness on a single machine using a novel heuristic approach. Proceedings of the 2012 International Conference on Industrial Engineering and Operations Management 2012.
2- Blazewicz, J., Ecker, K.H., Pesch, E., Schmidt, G. and Weglarz, J. 2007. Handbook on Scheduling: From Theory to Applications. Springer. Berlin. Heidelberg. New York.
3- Chen, Wei-Yang, and Gwo-Ji Sheen (2007). Single-machine scheduling with multiple performance measures: minimizing job-dependent earliness and tardiness subject to the number of tardy jobs. International Journal of Production Economics. Vol. 109. No. 1-2, p.p. 214-229.
4- Driss Belbachir, Fatima Boumediene, Ahmed Hassam, Latéfa Ghomri, "Adaptation and parameters studies of CS algorithm for flow shop scheduling problem", international Journal of Electrical and Computer Engineering (IJECE) ,Vol. 11, No. 3, June 2021, pp. 2266~2274.
5- Graham, R. L., Lawler, E. L., Lenstra, J. K., & Kan, A. R (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. In Annals of discrete mathematics, Vol. 5.Elsevier, p.p. 287-326.
6- Johnson, S.M. 1954. Optimal two and three stage production schedules with set-up times included. Naval Res. Logist Quart. 1: 61-68.
7- Lawler E.L. 1973. Optimal sequencing of a single machine subject to precedence constraints. Management Science 19: 544-546.
8- Smith W.E. 1956 Various Optimizers for single-stage production. Naval. Res. Logist Quart. 3: 59- 66.
9- T'kindt, V. and Billaut, J. 2002. Multi-criteria Scheduling: Theory, Models and Algorithms. Second edition. Springer Berlin Heidelberg New York.