Scheduling of Single Machine with Release Date to Minimize Multiple Objective Functions

Hanan A Cheachan¹, Manal H Ibrhim², Hussam A A Mohammed³

¹,² Department of Mathematics, College of Sciences, University of Mustansiriyah, Baghdad, Iraq.
³ Department of Mathematics, College of Education for Pure Sciences, University of Kerbala, Kerbala, Iraq.
¹ hanomh123@uomustanstansiriyah.edu.iq, ³ hussam.abidali@uokerbala.edu.iq.

Abstract. The aim of this paper is to derive several upper bounds and a lower bound, these bounds to be used in a branch and bound method. Because of the computational complexity of the our problem, optimal and near-optimal solutions are obtained by developed algorithms. All algorithms are tested and computational experiments is given in tables.

Keywords: Single Machine Scheduling, Branch and Bound Method, Descent Method, Simulation Acceptance Method.

1. Introduction

Scheduling theory comes into sight in 1950. These, problems be more combination be due to the many practical bands they want to take into account. Surprisingly the most of on scheduling problems, in pursuit, is the employ of double standard, which overwhelmingly let to numeration a widely factual resolution for the ruling creator. A research on multicriteria one-machine scheduling problems is established in [1]. They offer that three kinds of issues have been talked. The first one deals with issues in which a lexicographical request of the standard is decreased. The second category of issues looks a cambered collection of criteria. The third category of issues anxiety’s the fixing of whole firm Vilfredo Pareto optima [2].

Scheduling is a resolution-working practicability that plays an important part in most industrialization and duty méters. It is applied in insurance and output, in carriage and allocation, and in input processing and connection. The scheduling function in a corporation uses mathematical mechanism or heuristic style to assign restricted purses to the transformation of tasks. A suitable distribution of exchequer become the corporation to optimize its objectives and attain its target [3].

Discuss on double scheduling has been rare, mostly when contrasted to research in single criterion scheduling [4]. Lately, much research has been directed to scheduling problems with multiple criteria. Bülbül et al. [5], Li [6], Ronconi, and Kawamura [7], Wang et al. [8], Singh and Mahapatra. [9], Vakhania [10], Cheachan et al. [11] and Mohammed et al. [12].

In this paper, we survey bicriteria scheduling problem belong to the second class. In section (2) problem formulation and analysis are given. In section (3) Dominance Rule is given. In section (4), we propose a Branch and Bound (BAB) algorithm for the Optimal Solution (OS) for the problem. Special cases for the problem is given in section (5). The near-OS for the problem obtained by using some algorithms given in section (6). In section (7) computational experiments is given. The conclusion is given in section (8). Future work is given in section (9).

2. Figuration of the Issue and Analysis

The public issue of scheduling jobs on a single machine to reduce the gross set back can explain as pursue: A group of n independent jobs \( N = \{1,2,\ldots,n\} \) that has to add up to without preemption on a single machine that can lug at most one job at a time. The machine is supposed to be constantly ready from time zero onwards and no priority connection exists between jobs. Every job \( k, k \in N \) has a positive integer processing time \( p_k \), a release date \( r_k \) and ideally should be completed at its due date \( d_k \). For a given sequence \( \varphi \) of the jobs, completion time \( C_{\varphi(k)} \) of job \( k \), Tardiness \( T_{\varphi(k)} \) and the number of late jobs are given by:

\[ C_{\varphi(k)} = \sum_{i=1}^{k-1} p_i + r_k + T_{\varphi(k)} \]

\[ T_{\varphi(k)} = \max\{0, d_k - C_{\varphi(k)}\} \]

\[ L = \sum_{k=1}^{n} T_{\varphi(k)} \]

These terms are used to evaluate the performance of the scheduling algorithm.
\[ C_{\phi(1)} = R_{\phi(k)} + p_{\phi(k)} \quad k = 1, \]
\[ C_{\phi(k)} = \max\{C_{\phi(k-1)}, R_{\phi(k)}\} + p_{\phi(k)} \quad k = 2, \ldots, n, \]
\[ T_{\phi(k)} = \max\{C_{\phi(k)} - d_{\phi(k)}, 0\} \quad k = 1, \ldots, n, \]
\[ U_{\phi(k)} = \begin{cases} 0 & \text{if } C_{\phi(k)} \leq d_{\phi(k)} , \\ 1 & \text{otherwise} \end{cases} \quad k = 1, \ldots, n. \]

The objects are to discover a sequence \( \varphi \) that reduce the gross cost

\[ R = \sum_{k=1}^{n} C_{\phi(k)} + \sum_{k=1}^{n} U_{\phi(k)} + T_{\max}(\varphi). \]

The mathematical form of this issue, which indicates by (P) can be declared as follows:

\[
\begin{align*}
\text{min } R &= \min_{\varphi \in \delta} \left\{ \sum_{k=1}^{n} C_{\phi(k)} + \sum_{k=1}^{n} U_{\phi(k)} + T_{\max}(\varphi) \right\} \\
\text{s.t.} \\
C_{\phi(k)} &= R_{\phi(k)} + p_{\phi(k)} \quad k = 1 \\
C_{\phi(k)} &= \max\{C_{\phi(k-1)}, R_{\phi(k)}\} + p_{\phi(k)} \quad k = 2, 3, \ldots, n \\
T_{\phi(k)} &\geq C_{\phi(k)} - d_{\phi(k)} \quad k = 1, 2, \ldots, n \\
U_{\phi(k)} &= \begin{cases} 0 & \text{if } C_{\phi(k)} \leq d_{\phi(k)} , \\ 1 & \text{otherwise} \end{cases} \quad k = 1, 2, \ldots, n \\
C_{\phi(k)} &> 0, T_{\phi(k)} \geq 0, R_{\phi(k)} \geq 0, p_{\phi(k)} > 0 \quad k = 1, 2, \ldots, n \\
\end{align*}
\]

where \( \varphi(k) \) denotes the position of job \( k \) in the ordering \( \varphi \) and \( \delta \) denotes the set of all enumerated schedules.

The problem (P) is decomposed into three subproblems (SP1), (SP2) and (SP3) with a simple structure as follows:

\[
\begin{align*}
N_1 &= \min_{\varphi \in \delta} \left\{ \sum_{k=1}^{n} C_{\phi(k)} \right\} \\
\text{s.t.} \\
C_{\phi(k)} &= R_{\phi(k)} + p_{\phi(k)} \quad k = 1 \\
C_{\phi(k)} &= \max\{C_{\phi(k-1)}, R_{\phi(k)}\} + p_{\phi(k)} \quad k = 2, 3, \ldots, n \\
C_{\phi(k)} &> 0, R_{\phi(k)} \geq 0, p_{\phi(k)} > 0 \quad k = 1, 2, \ldots, n \\
\end{align*}
\]

\[
\begin{align*}
N_2 &= \min_{\varphi \in \delta} \{T_{\max}(\varphi)\} \\
\text{s.t.} \\
C_{\phi(k)} &= R_{\phi(k)} + p_{\phi(k)} \quad k = 1 \\
C_{\phi(k)} &= \max\{C_{\phi(k-1)}, R_{\phi(k)}\} + p_{\phi(k)} \quad k = 2, 3, \ldots, n \\
T_{\phi(k)} &\geq C_{\phi(k)} - d_{\phi(k)} \quad k = 1, 2, \ldots, n \\
T_{\phi(k)} &\geq 0, R_{\phi(k)} \geq 0, p_{\phi(k)} > 0 \quad k = 1, 2, \ldots, n \\
\end{align*}
\]

\[
\begin{align*}
N_3 &= \min_{\varphi \in \delta} \left\{ \sum_{k=1}^{n} U_{\phi(k)} \right\} \\
\text{s.t.} \\
U_{\phi(k)} &= \begin{cases} 0 & \text{if } C_{\phi(k)} \leq d_{\phi(k)} , \\ 1 & \text{otherwise} \end{cases} \quad k = 1, 2, \ldots, n \\
\end{align*}
\]

3. Precedence Rules

We will introduce a set of rules of dominance that use the method of BAB branching and adherence.
Theorem 1. If $\varphi_K$ be a subsequence which it’s jobs are scheduled $K \subset N$, for $a,b \in \overline{K} = N - K$, and let $\tau$ be the completion of the last job in $\varphi_K$. If $p_a \leq p_b$, $\mathcal{R}_a \leq \mathcal{R}_b \leq \tau$ and $d_a \leq d_b \leq \tau$. Then job $a$ proceed job $b$ in the optimal solution for the problem (P).

Proof. Let $\varphi_2 = (\varphi_K, b, a)$ be the schedule which is acquired by interchanging jobs $a$ and $b$ in $\varphi_1 = (\varphi_K, a, b)$.

The total completion time of jobs $a$ and $b$ in $\varphi_1$ is:
$$C_a + C_b = 2\tau + 2p_a + p_b$$
(1)

The total completion time of jobs $a$ and $b$ in $\varphi_2$ is:
$$C_a' + C_b' = 2\tau + 2p_b + p_a$$
(2)

From (1) and (2) we get
$$\sum_{a \in \varphi_1} C_a \leq \sum_{b \in \varphi_2} C_b$$
(3)

For schedule $\varphi_1$, the maximum tardiness is:
$$T_{max} = \max\{T_m, T_b, T_a\}$$
where $T_m = \max_{k \in \varphi_K} \{T_k\}$, $T_b = \max\{C_b - d_b, 0\}$ and $T_a = \max\{C_a - d_a, 0\}$.

For schedule $\varphi_2$, the maximum Tardiness is:
$$T'_{max} = \max\{T_m, T_b', T_a'\}$$
where $T_b' = \max\{C_b - d_b, 0\}$ and $T_a' = \max\{C_a' - d_a, 0\}$.

Since $C_b < C_a$ and $d_a \leq d_b$, then $T_a' \geq T_b'$, so $T'_{max} = \max\{T_m, T_a\}$.

Then
$$T'_{max} \geq T_{max}$$
(4)

From (3) and (4) we get:
$$\sum_{a \in \varphi_1} C_a + \sum_{b \in \varphi_2} U_d + T_{max} \leq \sum_{a \in \varphi_1} C_a' + \sum_{b \in \varphi_2} U_d + T_{max}.$$

Theorem 2. If $\varphi_K$ be a subsequence which its jobs are schedule $K \subset N$, for $a,b \in \overline{K} = N - K$, if $p_a \leq p_b$, $d_a \leq d_b$, $b, a$ are early in $\varphi_1$ and $\varphi_2$. Respectively and $\tau > \max(\mathcal{R}_a, \mathcal{R}_b)$, then job $a$ proceed job $b$ in the optimal solution for the problem (P).

Proof. The tardiness late jobs of $a$ and $b$ in $\varphi_1$ and $\varphi_2$ are equal to zero and are have:
$$\sum_{a \in \varphi_1} C_d \leq \sum_{a \in \varphi_2} C_d'$$
since $p_a \leq p_b$, we get:
$$\sum_{a \in \varphi_1} C_d + \sum_{a \in \varphi_2} U_d + T_{max} \leq \sum_{a \in \varphi_1} C_d' + \sum_{a \in \varphi_2} U_d + T_{max}.$$

Theorem 3 [6]. $N_1 + N_2 + N_3 \leq \mathcal{R}$ where $N_1, N_2, N_3$ and $\mathcal{R}$ are the minimum thematic task amount of (SP1), (SP2), (SP3) and (P) respectively.

4. BAB Method
In the following two subsections, the Upper Bound (UB) and Lower Bound (LB) are be derived.

4.1 Derivation of UB
We can find UB for our problem (P) by using three heuristic methods, these heuristics are as follows:

Heuristic 1. Sorting the jobs $(1, 2, \ldots, n)$ by non-decreasing order of $\mathcal{R}_i$ (i.e. $\mathcal{R}_1 \leq \mathcal{R}_2 \leq \cdots \leq \mathcal{R}_n$).
If $\varphi = (\varphi(1), \varphi(2), \ldots, \varphi(n))$ is obtained by Heuristic 1, then $UB_1 = \sum_{k=1}^{n} C_{\varphi(k)} + \sum_{k=1}^{n} U_{\varphi(k)} + T_{max}(\varphi)$. 
Heuristic 2. Sorting the jobs by non-decreasing order of $R_j + p_j$ (i.e. $R_1 + p_1 \leq R_2 + p_2 \leq \cdots \leq R_n + p_n$).
If $\varphi = (\varphi(1), \varphi(2), \ldots, \varphi(n))$ is obtained by Heuristic 2, then $UB2 = \sum_{k=1}^{n} C_{\varphi(k)} + \sum_{k=1}^{n} U_{\varphi(k)} + T_{max}(\varphi)$.

Heuristic 3. Choose minimum $R_k + p_k$ and scheduling first and the remaining jobs $(n - 1)$ sorting by SPT rule (i.e. $p_1 \leq p_2 \leq \cdots \leq p_{n-1}$).
If $\varphi = (\varphi(1), \varphi(2), \ldots, \varphi(n))$ is obtained by Heuristic 3, then $UB3 = \sum_{k=1}^{n} C_{\varphi(k)} + \sum_{k=1}^{n} U_{\varphi(k)} + T_{max}(\varphi)$.

4.2. Derivation of $LB$
A $LB$ for problem (P) is founded on decomposition (P) into three subproblems (SP1), (SP2) and (SP3) as shown in section (2), then calculate $N1$ to find the $LB$ for (SP1), $N2$ to exist the $LB$ for (SP2), and $N3$ to exist the $LB$ for (SP3) then calculate $N1, N2$ and $N3$ to be $LB$s for (SP1), (SP2) and (SP3), respectively, then applying Theorem 3 to get $LB$ for our problem (P).
For subproblem (SP1) we using the lower, which suggested by Al-Zuwaini [13], then:
$LB(SP1) = nR* + \alpha - R,$
where $R* = \min\{R_k\}, R = \sum_{k=1}^{n} R_k$ and $\alpha$ be an OS for $1|R_k = 0| \sum_{k=1}^{n} C_{k}$ (i.e. $\alpha$ is obtained by SPT-rule).
For the subproblems (SP2) and (SP3), we relaxed the release date and the problem become $1|R_k = 0|T_{max}$ and $1|R_k = 0| \sum_{k=1}^{n} U_k$, then sorting the jobs by $EDD$ rule (i.e. $d_1 \leq d_2 \leq \cdots \leq d_n$) and calculate $T_{max} = \max(C_k - d_k, 0), LB(SP2) = T_{max}$ while using Moor's algorithm for subproblem (SP3) then:
$LB(SP3) = \sum_{k=1}^{n} U_k$.
Then $LB = LB(SP1) + LB(SP2) + LB(SP3)$.

5. Special Cases of the Problem (P)
Case 1. If $p_1 \leq p_2 \leq \cdots \leq p_n, R_1 \leq R_2 \leq \cdots \leq R_n$ and $C_k = d_k \forall k \in \varphi$, in a schedule $SPT$-rule, then $SPT$ is an OS for problem (P).
Proof. Since $C_k = d_k \forall k \in SPT$-rule, then $T_{max} = \sum_{k=1}^{n} U_k = 0$. The problem (P) reduced to $1|R_k| \sum_{k=1}^{n} C_{k}$ but the problem solved in $SPT$-rule [2].

Case 2. If $C_k = d_k \forall k \in$ a schedule $\varphi$ and the preemptive is allowed then $\varphi$ given an OS for the problem $1|R_k| pmtn| \sum_{k=1}^{n} C_{\varphi(k)} + \sum_{k=1}^{n} U_{\varphi(k)} + T_{max}(\varphi)$.
Proof. Since $T_{max} = \sum_{k=1}^{n} U_k = 0$ in $\varphi$, then the problem (P) reduced $1|R_k| pmtn| \sum_{k=1}^{n} C_{\varphi(k)}$, but this problem was solved by $SRPT$ - rule [9]. Then $\varphi$ given an OS for the problem $1|R_k| pmtn| \sum_{k=1}^{n} C_{\varphi(k)} + \sum_{k=1}^{n} U_{\varphi(k)} + T_{max}(\varphi)$ provided that $C_k = d_k \forall k \in \varphi$.

Case 3. Any schedule grand’s an OS for the problem (P) if $R_k = R, p_k = p$, and $d_k = d \forall k \in \varphi$.
Proof. Since $\sum_{k=1}^{n} C_{\varphi(k)} = nR + \left(\frac{n^2 + n}{2}\right)p, T_{max} = \max\{(R + np) - d, 0\}$ and
$U_{\varphi(k)} = \begin{cases} 0 & \text{if } C_{\varphi(k)} \leq d_{\varphi(k)} \text{ for } k = 1, 2, \ldots, n \\ \text{o.w.} & \end{cases}$
in any schedule.
Then any schedule is optimal for the problem $1|R_k = R, p_k = p, d_k = d| \sum_{k=1}^{n} C_{k} + \sum_{k=1}^{n} U_k + T_{max}$ (because the three quantities are constant).

Case 4. For the problem (P), if $R_j = R, d_j = d \forall j = 1, 2, \ldots, n$ and
(i) If $\sum_{j=1}^{n} C_j(EDD) = \sum_{j=1}^{n} C_j(SPT)$, then $EDD$ schedule is the OS.
(ii) If $T_{max}(SPT) = T_{max}(EDD)$, then $SPT$ schedule is the OS.
Proof (i). From condition \( \mathcal{R}_j = \mathcal{R}, d_j = d \ \forall j = 1,2,...,n \) any order given an OS for the problem \( 1|\mathcal{R}_k = \mathcal{R}, d_k = d|T_{\text{max}} \).
Now, since \( \sum_{k=1}^{n} c_k(\text{EDD}) = \sum_{k=1}^{n} c_k(\text{SPT}) \), then EDD minimize of the problem \( 1|\mathcal{R}_k = \mathcal{R}, p_k = p, d_k = d|\sum_{k=1}^{n} c_k + \sum_{k=1}^{n} T_k + T_{\text{max}} \).

Proof (ii). From condition \( \mathcal{R}_j = \mathcal{R}, d_j = d \ \forall j = 1,2,...,n \) any order given OS for the problem \( 1|\mathcal{R}_k = \mathcal{R}, p_k = p, d_k = d|\sum_{k=1}^{n} c_k + \sum_{k=1}^{n} U_k + T_{\text{max}} \).

Case 5. If \( \mathcal{R}_k = \mathcal{R} \) and SPT schedule gives \( d_i + p_j \leq d_j \ \forall i < j \) and \( T_{\text{max}}(\text{SPT}) = T_{\text{max}}(\text{EDD}) \), then SPT is the OS for the problem (P).

Proof. Since \( d_i + p_j \leq d_j \ \forall i < j \) in SPT schedule then \( d_i - p_i \leq d_j - p_j \ \forall i < j (P_i > 0) \). Thus SPT given OS for both criteria \( \sum_{k=1}^{n} U_k \) and \( \sum_{k=1}^{n} C_k \) and from condition \( T_{\text{max}}(\text{SPT}) = T_{\text{max}}(\text{EDD}) \). Then SPT OS for the problem (P).

6. Local Search Method (LSM) [14]

LSM consists of the next steps.
- Select a first schedule \( S \) to be the initial solution and calculate the amount of the objective task \( \vartheta(S) \).
- Select a neighbour \( S' \) of the actual solution \( S \) and calculate \( \vartheta(S') \).
- Experiment whether to take the get about from \( S \) to \( S' \). If the get about is taken, then \( S \) exchange \( S \) as the actual solution; do it another way \( S \) is keep as the actual solution.
- Experiment whether the algorithm should cutoff. If it cutoff, output the preferable solution created; otherwise, recurrence to the neighbour descent step in the following subsections we will discuss some LSMs.

6.1. Simulated Annealing (SA)

Simulated annealing is an algorithmic method that is fit to flight from local minima. It is a randomized LSM for two criteria: First, from the neighbourhood of a solution a neighbour, which are always agree abled if they are selected, worse-cost neighbours are also accepted, although with a likelihood that is by degrees decreased in the course of the algorithm's fulfilment. The randomized kind enables asymptotic assemblage to optimum solutions under sure n mild conditions. None the less, the power landscape, which is specified by the objective function and the neighbourhood framework, may confess many and/or “deep” local minima. Therefore, avert local minima is a critical part of the performance of the algorithm [9].

SA Algorithm
Step (1). initialization: determining the temperature parameter \( T_0 \) and the cooling schedule: \( r (0 < r < 1) \) and also the termination criterion. Generate and evaluate an initial candidate solution (perhaps at random); call this current solution \( c \).
Step (2). Generate a new neighboring solution \( m \), by making a small change in the current permutation of jobs and evaluate this new solution.
Step (3). Accept this new solution as the current solution if:
The objective value of new solution, \( f(m) \) is better than of the current solution \( f(c) \).
If \( f(m) \) is worse than \( f(c) \), the value of acceptance is determined by probability function given by \( \exp(-(f(m) - f(c))/T_k) \) which must be greater than a uniformly generated random number "rand" where \( 0 < \text{rand} < 1 \).
Step (4). Check the stop criteria and update temperature parameter; (i.e. \( T_k = r^{	ext{th}} T_{k-1} \) and return to step (2) until some stopping criteria are met.
6.2. Improvement Descent Algorithm (DM)[15]

**Step (1).** Choose initial solution \( s \in S \).

**Step (2).** Generate \( s' \) neighbourhood for \( s \) (by insert or swap) and set \( \Delta = \vartheta(s') - \vartheta(s) \), if \( \Delta < 0 \) then set \( s = s' \).

**Step (3).** If \( \vartheta(s') \geq \vartheta(s) \), \( \forall s' \in N(S) \), then stop; else return to step (2).

6.3. Threshold Acceptance Method (TA)[16]

The threshold acceptance procedure has the advantage that they can quit a domestic minimum. They have the abuse that it is potential to get back to solutions formerly show round. Thus vibration concerning domestic minima is potential and this may do command to a situation where frequently computational time is a dust on a little portion of the resolution collection. For specifics of threshold acceptance framework see [10].

The LSMs (DM, SA and TA) stopped when iteration = 1000 iterations.

7. Computational Test

We headmost current how models can be randomly created. The processing time \( p_k \) is uniformly deal with the period [1,10]. The release date \( \varrho_k \) is uniformly deal in the period [1,10]. The period \( [\tau(1 - \tau F - RDD/2), \tau (1 + \tau F + RDD/2)] \) is uniformly deal gives due dates \( d_k \); where \( \tau = \sum_{k=1}^{n} p_k \), relying on the proportional Range of Due Date (RDD) and on the average Tardiness Factor (\( \tau F \)). For both parameters, the amounts 0.2, 0.4, 0.6, 0.8, 1 are looked. For each elected amount of \( n \) two problems were created for each of a number of parameters producing 10 problems for each amount of \( n \).

The BAB algorithm was tested in Fortran Power Station and LSMs (DM, SA and TA) were tested by coding them in Matlab R2009b and running on Pentium (R) at 2.20 GHz with Ram 2 GB computer processor-type PDCT 4400. In the Table 1 \( n = 9 \) jobs and 12 jobs we list 10 problems for each value of \( n \). Test problems are tested to show the efficiency of our LB used in a BAB algorithm to obtain the OS. Results of comparing the LB, UBs and the OSs are given in Table 1. The first column is the number of jobs of problems. The second column gives the value of an OS found by using BAB algorithm. The third column gives the value of the Initial LB (ILB). The fourth, fifth, six columns give the value of our UBs (UB1, UB2, UB3). The seven column gives the number of nodes (Nodes).
Table 1. The execution of $I L B$ and $U B$s of $B A B$ algorithm.

| $n$ | $N$ | $Opt.$ | $I L B$ | $U B_1$ | $U B_2$ | $U B_3$ | $Nodes$ |
|-----|-----|-------|--------|---------|---------|---------|---------|
| 9   | 1   | 288   | 228    | 360     | 342     | 367     | 19588   |
|     | 2   | 284   | 259    | 293     | 291     | 294     | 5613    |
|     | 3   | 252   | 218    | 376     | 305     | 287     | 8775    |
|     | 4   | 298   | 259    | 345     | 321     | 346     | 15739   |
|     | 5   | 223   | 186    | 225     | 267     | 240     | 7375    |
|     | 6   | 202   | 174    | 324     | 289     | 292     | 10731   |
|     | 7   | 215   | 173    | 327     | 282     | 282     | 11258   |
|     | 8   | 319   | 300    | 333     | 494     | 325     | 38807   |
|     | 9   | 178   | 137    | 281     | 254     | 259     | 6500    |
|     | 10  | 189   | 171    | 295     | 282     | 309     | 7516    |

$Opt.$ = The optimal value obtained by $B A B$ algorithm.

Table 1 shows that the $L B$ and $U B$s, the number of nodes and computational time for the 10 problems of $n = 9$ and $n = 11$ jobs, we watch that whenever $n$ growing the number of nodes and computational time growing.

Table 2. Comparison $O S$s in $B A B$ with $D M, S A, T A$

| $n$ | $N$ | $Opt.$ | $D M$ | $S A$ | $T A$ |
|-----|-----|-------|-------|-------|-------|
| 10  | 1   | 289   | 298   | 291   | 339   |
|     | 2   | 265   | 281   | 271   | 401   |
|     | 3   | 262   | 294   | 276   | 356   |
|     | 4   | 340   | 373   | 472   | 343   |
|     | 5   | 289   | 296   | 473   | 417   |
|     | 6   | 277   | 286   | 279   | 379   |
|     | 7   | 190   | 289   | 302   | 279   |
|     | 8   | 284   | 295   | 301   | 398   |
|     | 9   | 259   | 294   | 287   | 297   |
|     | 10  | 330   | 393   | 395   | 360   |

No. of optimal = 0

7.1. Comparative Results of $L S M$s

Table 3 shows the comparison of $L S M$s which are $D S, S A$ and $T A$. The results in a Table 3 show that $T A$ has good performance results, followed by $D S$ and $S A$. 

7
Table 3. Comparison of *LSMs*

| n  | N   | DS   | SA   | TA   | Best  |
|----|-----|------|------|------|-------|
| 80 | 1   | 17265| 17490| 16898| 16898 |
|    | 2   | 17350| 17437| 19295| 17350 |
|    | 3   | 17429| 17380| 18308| 17380 |
|    | 4   | 17390| 17467| 17365| 17365 |
|    | 5   | 17368| 17377| 17385| 17368 |
|    | 6   | 17185| 17418| 17223| 17185 |
|    | 7   | 17498| 17513| 16085| 16085 |
|    | 8   | 17494| 17446| 16242| 16242 |
|    | 9   | 17427| 17416| 18524| 17416 |
|    | 10  | 17316| 17475| 17400| 17316 |

| No. of best | 4 | 2 | 4 |
|-------------|---|---|---|
| Av. Time    | 10.12071 | 10.15951 | 10.02871 |

| 200 |
|-----|
| 1   | 113143 | 113067 | 112392 | 112392 |
| 2   | 113062 | 113156 | 100918 | 100918 |
| 3   | 113195 | 113105 | 111850 | 111850 |
| 4   | 113056 | 113186 | 112149 | 112149 |
| 5   | 113096 | 113082 | 120129 | 120129 |
| 6   | 113102 | 112880 | 112880 | 112880 |
| 7   | 113190 | 113150 | 102502 | 102502 |
| 8   | 113161 | 113142 | 115675 | 115675 |
| 9   | 113099 | 113043 | 113043 | 113043 |
| 10  | 113118 | 112935 | 109802 | 109802 |

| No. of best | 0 | 3 | 7 |
|-------------|---|---|---|
| Av. Time    | 10.27401 | 10.24431 | 10.04991 |

| 600 |
|-----|
| 1   | 952297 | 952425 | 952425 | 952297 |
| 2   | 954850 | 954197 | 954197 | 954197 |
| 3   | 954103 | 954210 | 954210 | 954103 |
| 4   | 955354 | 954130 | 954130 | 954130 |
| 5   | 954028 | 954251 | 954251 | 954028 |
| 6   | 955429 | 955332 | 955332 | 955332 |
| 7   | 954951 | 954202 | 954202 | 954202 |
| 8   | 952919 | 954152 | 954152 | 952414 |
| 9   | 954733 | 953168 | 953168 | 953168 |
| 10  | 955425 | 955431 | 955431 | 955425 |

| No. of best | 5 | 5 | 5 |
|-------------|---|---|---|
| Av. Time    | 10.52421 | 10.41611 | 10.13641 |

| 1000 |
|------|
| 1   | 2805627 | 2805615 | 2804765 | 2804765 |
| 2   | 2805389 | 2805584 | 2682848 | 2682848 |
| 3   | 2805666 | 2805536 | 2729962 | 2729962 |
| 4   | 2805605 | 2805455 | 2791608 | 2791608 |
| 5   | 2805602 | 2805621 | 2748204 | 2748204 |
| 6   | 2805560 | 2805606 | 2663106 | 2663106 |
| 7   | 2805536 | 2805536 | 2757554 | 2757554 |
| 8   | 2805443 | 2805643 | 2772540 | 2772540 |
| 9   | 2805623 | 2805501 | 2787764 | 2787764 |
| 10  | 2805505 | 2805616 | 2742417 | 2742417 |

| No. of best | 0 | 0 | 10 |
9. Outcomes

In this study, the problem of scheduling jobs on one machine to minimize tri-criteria with a release date is considered. The three criteria to be minimized are \( \sum_{k=1}^{n} C_k \), \( \sum_{k=1}^{n} U_k \) and \( T_{\text{max}} \). We present a BAB algorithm to find OS for the problem of minimizing a linear function (i.e. \( \sum_{k=1}^{n} C_k \), \( \sum_{k=1}^{n} U_k \) and \( T_{\text{max}} \)). A computational test for BAB algorithm on a large set of test problems are given and BAB algorithm solve our test problem up to (11) jobs. The NP-hardness of this problem and the OSs of the BAB algorithm are not always quickly. Hence, this problem is solved by using LSMs as DS, SA and TA algorithms. Also, we determination on the results of overall computations test of LSMs.

The major conclusion to be pulled from our account results is that TA is a more effective method for our problem followed by DS and SA. Whereas the computational time of SA is very small followed by the computational times of DS and SA.

9. Outlook Work

A delightful future discuss subject would involve experimentation with the following problems:

\[
F2|\text{RI}_k| \sum_{k=1}^{n} C_k + L_{\text{max}}
\]

\[
1|\text{RI}_k| \sum_{k=1}^{n} C_k + L_{\text{max}} + E_{\text{max}}
\]

References

[1] Hoogeveen J A 1992 Single machine bi-criteria scheduling PhD Dissertation Center for Mathematics and Computer science Amsterdam (The Netherlands).
[2] T’kindt V and Billaut J C 1999 Some guidelines to solve multicriteria scheduling problems IEEE pp 463-468.
[3] Lenstra J K, Rinnooy Kan A H G and Brucker P 1977 Complexity of machine scheduling problems Annals of Discrete Mathematics vol 1 pp 343-362.
[4] Wu C C, Hsu P H, Chen J C and Wang N S 2011 Genetic algorithm for minimizing the total weighted completion time scheduling problem with learning and release times Computers & Operations Research vol 38 pp 1025-1034.
[5] Bülbül K, Kaminsky P and Yano C 2007 Preemption in single machine earliness/tardiness scheduling Journal of Scheduling vol 10 pp 271-292.
[6] Li G. 1997 Single machine earliness and tardiness scheduling European Journal of Operational Research vol 96 pp 546-558.
[7] Ronconi D P and Kawamura M S 2010 The single machine earliness and tardiness scheduling problem: lower bounds and a branch-and-bound algorithm Computational & Applied Mathematics vol 29(2) pp 107-124.
[8] Wang X J, Zhang C Y, Gao L and Li P G 2008 A survey and future trend of study on multi-objective scheduling International Conference on Natural Computation vol 6 pp 382-391.
[9] Singh M R and Mahapatra S S 2016 A quantum behaved particle swarm optimization for flexible job shop scheduling Computers & Industrial Engineering vol 93 pp 36-44.
[10] Vakhania N 2018 Scheduling a Single Machine with Primary and Secondary Objectives Algorithms vol 11(6) p 80.
[11] Cheachan H A, Mohammed H A A and Cheachan F A, 2013 Fuzzy distances and their applications on fuzzy scheduling Eastern-European Journal of Enterprise Technologies vol 3/4 (63) pp 23-30 ISSN 1729-3774.
[12] Mohammed H A, Cheachan F A, Cheachan H A and Cheachan R A 2019 A New Perspective to a Fuzzy Distance Function and Its Applications on Fuzzy Due Dates and Cost Function Journal of Engineering and Applied Sciences vol 14 (No.07-08) to get release in April 2019 ISSN 1816-949X.
[13] Al-Zuwaini M K 2000 Single machine scheduling to minimize total cost of flow time with unequal ready times Journal of Basrah Research vol 25(3) 9 pp 4-108.
[14] Moscato P 1989 On evolution, search, optimization, genetic algorithm and martial arts: toward memetic algorithms Caltech concurrent computation program c3p Technical Report California 826.
[15] Glover F 1994 A user's guide to tabu search Annals of Operations Research vol 41 pp 3-28.
[16] Brucker P 2007 Scheduling Algorithms Springer Berlin Heidelberg New York Fifth Edition.