ISSN 2347-1921
Volume 12 Number 3
Journalof Advances in Mathematics

Branch and Bound Method to Solve Multi Objectives Function

Mohammed KadhimAl- Zuwaini[1] , TahaniJabbarKahribt[2]
[1] Department of Mathematics, College of Computerfor Science andMathematics
Thi-Qar University, Thi-Qar, Iraq
Mkzz50@ yahoo.com
[2] Department of Mathematics,College of Education for Pure Science,
Thi-Qar University, Thi-Qar, Iraq
tahany@ yahoo.com

ABSTRACT

This paper presents a branch and bound algorithm for sequencing a set of n independent jobs on a single machine to minimize sum of the discounted total weighted completion time and maximum lateness, this problems is NP-hard. Two lower bounds were proposed and heuristic method to get an upper bound. Some special cases were proved and some dominance rules were suggested and proved, the problem solved with up to 50 jobs.

Keywords

Single machine; Scheduling; Discounted total weighted completion time; Maximum lateness.

1.INTRODUCTION

We consider the scheduling n jobs on a single machine to minimize the bi-criteria problems. Our objective is to find a schedule that minimize sum of discounted total weighted completion time and maximum lateness, with penalty of lateness equal to one the extension of the problem when the penalty of lateness equal or more than one was consider also. The all jobs are available at time zero. This problems is denoted by $1//\sum_{j=1}^{n} w_j(1-e^{-rc_j}) + L_{max}$ where $h=1$ and$1//\sum_{j=1}^{n} w_j(1-e^{-rc_j}) + L_{max}^{h}$ where $h \geq 1$ (h represent the penalty of lateness).It is minimizing of total discounted weighted completion time which it extends from the minimization of total weighted completion time, Rothkof (1966) [1]. Rothkopf and Smith (1984) [2] considered the total discounted weighted completion time as $1//\sum_{j=1}^{n} w_j(1-e^{-rC_j})$. Their study was more general cost function and discounted rate of r , $0<r<1$ of the problem can be solved as P-type optimally in polynomial time by the Weighted Discounted Shorter Processing Time (WDSPT) rule. The rule that schedules the jobs in non-decreasing order of ratio : $\frac{w_j e^{-rp_j}}{1-e^{-rp_j}}$[3]. Wang et.al(2006) consider the problem $F_m//\sum_{j=1}^{n} w_j(1-e^{-rC_j})$ is NP-hard [4]. Yunqiang Yin et al. (2012)[5] showed that the total weighted discounted completion time is polynomials solvable and optimal by considering the effects of position-dependent learning and time-dependent deterioration simultaneously. The problem total weighted discounted completion time studied by Lin Li et al. (2013) [6] showed the heuristics according to the corresponding problems without learning effect. Guochen Sun andYuewu Li (2013) [7] showed the problem $1/DT, D_{psd}/\sum_{j=1}^{n} w_j(1-e^{-rC_j})$ is past-sequence-dependent delivery time and deteriorating jobs. Hongjie Li et al.(2014) [8] studies the problem $1/p_j^X = a_j^X(1-kt)/w_j^A\left(1-e^{-rC_j^A}\right): f_{max}^B \leq U$ considers a scheduling environment in which there are two agents and a set of jobs, each of which belongs to one of the two agents and its actual processing time is defined as a decreasing linear function of its starting time. Each of the two agents competes to process its respective jobs on a single machine and has its own scheduling objective to optimize. The discounted cash flow (DCF) analysis is a method of evaluating a project, company, or assessing using the concepts of the time value of money. All future cash flows are estimated and discounted by using cost of capital to give their present values (PVs). The sum of all future cash flows both incoming and outgoing, is the net present value (NPV) which is taken as the value or price of the cash flows. Discounted cash flow calculations have been used in some form since money was first lent at interest in ancient times. Studies of ancient Egyptian and Babylonian mathematics suggest that they used techniques similar to discounting of the future cash flows. Following the stock market crash of 1929, discounted cash flow analysis gained popularity as an evaluation method for stocks. Irving Fisher in his 1930 book *The Theory of Interest* and John Burr Williams's 1938 text *The Theory of Investment Value* were the first who formally expressed the DCF method in modern economic terms. The DCF use has increased substantially in institutional ,investment property and business valuation sectors. It is frequently required by client, underwriters, financial advisers and administrators, and portfolio managers. It is used in investment finance, real estate development, corporate financial management and patent valuation.

The discounted cash flow formula is:

$$DCF = \frac{CF_1}{(1+r)^1} + \frac{CF_2}{(1+r)^2} + \cdots + \frac{CF_n}{(1+r)^n}$$

where CF: cash flow, r: discount rate, n: is the time in years before the future cash flow occurs.

Discounted cash flow analysis is an extension of simple cash flow analysis that takes into account the time value of money and the risks of investing in a project. DCF analysis can be divided into two main categories, the net present value method (NPV) and the internal rate of return method (IRR). Today the DCF model is the most commonly used tool among financial analysts when valuing a firm. It is documented that almost fifty percent of all financial analysts use a DCF method when valuing potential objects to acquire (Hult, 1998). The problem $1//L_{max}$ is solved in O(n log n) time by

Jackson's earliest due date (EDD) rule1 (i.e., by ordering the jobs according to non-decreasing due dates) Jackson (1955) [9]. Lawler and Moore (1969) [10] showed that the optimal schedule for $(1 / prec / L_{max})$ in $O(n^2)$ time. Horn (1974) [11] observed that $(1/r_j, p_j = 1 / L_{max})$ and ($1/pmtn, r_j / L_{max}$) were solved by the extended Jackson's rule. Lageweget al. (1976) [12] showed that the problem $1/prec, r_j, p_j = 1/L_{max}$ and $1/pmtn, prec, r_j / L_{max}$ can still be solved in polynomial time. Simons (1978) [13] presented a more sophisticated approach to solve the problem $(1/r_j, p_j = p / L_{max})$. Lenstra et al(1977) [14] showed the problem $(1 / r_j / L_{max})$ which is strongly NP-hard. Various branch and bound methods (BAB) exist for solving the problem $(1/prec, r_j / L_{max})$, Baker and Su (1974) (1982) [15][16]. Sen and Gupta(1983) [17] extended the method to the problem $1 // \sum_{j=1}^{n} C_j + L_{max}$ is NP-hard. The problem $1 // \sum_{j=1}^{n} C_j + L_{max} + E_{max}$ is NP-hard and it's studied by Sen et al. (1988) [18] that developed a branch-and-bound algorithm and derived lower bounds by means of the maximum potential improvement method. Hariri and Potts (1997) [19] proposed a branch and bound (BAB) to solve the $(1 / S_f / L_{max})$ problem.

Some approaches used for the problem of $1/S_f/L_{max}$ considerations are given by (Potts and Kovalyov, 2000) [20] and (Allahverdi et al., 2008) [21]. Christos and George (2010) [22] showed that the problem $1 / q_{psd} / L_{max}$can be solved by simple polynomial-time algorithms. Habibeh and Lai (2010) [23] used genetic algorithm for the problem $1 / S_f / L_{max}$. RadostawRudek (2011) [24] proved that the problem maximum lateness is NP-hard even if job processing time are described by linear function he proposed BAB algorithm and approximation to verify numerically their efficiency. Yunqiang Yin et al. (2012) [5] showed that the maximum lateness is polynomial solvable and optimal by considering the effects of position-dependent learning and time-dependent deterioration simultaneously. The problems maximum lateness studied by Lin Li et al. (2013)[6] showed the heuristics according to the corresponding problems without learning effect. Suh-Jenq Yang et al. (2013) [25] showed the problem of $(1/p_{jr}, q_{psd} / L_{max})$ is consider of past-sequence-dependent delivery times and the effects of deterioration and learning. Morteza and Mehde (2015) [26] proposed a branch and bound (BAB) to solve the problem $1/nr - a / L_{max}$. Spyros T. and Alekos T. (1993) showed that can be found optimal schedule under the condition of unit-length independent of no preemption jobs and identical or uniform machines with respect to the criterion L_{max}^h in $O(n^3)$ which is a significant extension of the well-known maximum lateness L_{max} [27].

2.Problem formulation

We take into consideration the problem of scheduling n jobs on a single machine to minimize the total cost that can be stated as follows: A set N={1,2,3,….,n} of n independent jobs has to be scheduled on a single machine in order to minimize a given criterion. This study applies the one machine scheduling problem with multiple objective function: The sum of discounted total weighted completion time and maximum lateness which is denoted by $1 // \sum_{j=1}^{n} w_j(1 - e^{-rC_j}) + L_{max}$. Under the conditions: Preemption is not allowed, no precedence relation among jobs is assumed and only one job i can be processed at a time. Given a schedule (1,……n), for each job j needs processing time p_j and a positive weighted w_j on the machine and ideally should be completed at its due date d_j. Discounted total weighted completion time $\sum_{j=1}^{n} w_j(1 - e^{-rC_j})$ and maximum lateness L_{max}, can be respectively defined as: $\sum_{j=1}^{n} w_j(1 - e^{-rc_j})$ $= \sum_{j=1}^{n} w_j(1 - e^{-r\sum_{i=1}^{j} p_j})$ and $L_{max} = \max\{L_j\} = \max\{C_j - d_j\}$, j= 1,…….n.

Our scheduling problem can be stated mathematically more precisely as follows:

Given a schedule δ = (1,2,3,….,n), then for each job $j \in \delta$ can calculate the discounted total weighted completion time $\sum_{j=1}^{n} w_j(1 - e^{-rC_j})$ and maximum lateness L_{max}. The objective is to find a schedule, $\sigma = (\sigma(1), \sigma(2), \ldots., \sigma(n))$ (belonging to a neighborhood ofδ).

That minimizes the total cost $\sum_{j=1}^{n} w_{\sigma(j)}(1 - e^{-rC_j}) + L_{max(\sigma)}$.

Let S be a set of all schedules,$|S|=n!$, then we can formulate our problem in mathematical form as:

$$H_1(\sigma) = min_{\sigma \in S}\left\{\sum_{j=1}^{n} w_{\sigma(j)}(1 - e^{-rC_j})\right\} + L_{\max(\sigma)}$$

$$\left.\begin{array}{l} Subject\ to: \\ \left.\begin{array}{ll} C_{\sigma(j)} \geq p_{\sigma(j)} & j = 1,2 \ldots\ldots\ldots, n \\ 0 < r < 1 & j = 1,2, \ldots\ldots., n \\ L_j = C_j - d_j & j = 1,2, \ldots\ldots\ldots, n \end{array}\right\} .. (E) \\ w_{\sigma(j)} \geq 1,\ d_{\sigma(j)} > 0, p_{\sigma(j)} > 0 \quad j = 1,2, \ldots\ldots\ldots, n \end{array}\right\} .. (A)$$

Also the multiple criteria: sum of discounted total weighted completion time and penalty maximum lateness was considered, which it signifies extension of the criteria $\sum_{j=1}^{n} w_{\sigma(j)}(1 - e^{-rC_j}) + L_{max(\sigma)}$, under the same condition (E), furthermore $h_j \geq 1$ is incurred as a penalty for late shipment. That's the problem that the cost of completing job j at a time C_j is denoted:

$$H_2(\sigma) = min_{\sigma \in \delta}\left\{\sum_{j=1}^{n} w_{\sigma(j)}(1 - e^{-rC_j}) + L_{max(\sigma)}^h\right\} \ldots (B)$$

Where $L_{max}^h = max_{1 \leq j \leq n}\{h_j L_j\}$ and $\sigma(j)$ denoted the position of job j in the ordering σ.

3. Decomposition of problem (A)

Because the complexity of problem (A) can be decomposed it into two subproblems which are a simple structure as the follow.

i. $1 // \sum_{j=1}^{n} w_j(1-e^{-rC_j})$ the discounted total weighted completion time which is solved by WDSPT rule [3] and the formulation mathematical form as:

$$\left.\begin{array}{ll} Z_1 = min_{\sigma(j)}\left\{\sum_{j=1}^{n} w_{\sigma(j)}(1-e^{-rc_{\sigma(j)}})\right\} & \\ Subject\ to & \\ C_{\sigma(j)} \geq p_{\sigma(j)} & j = 1,2,\dots\dots\dots,n \\ 0 < r < 1 & j = 1,2,\dots\dots\dots,n \\ w_{\sigma(j)} \geq 1,\ d_{\sigma(j)} > 0, p_{\sigma(j)} > 0 & j = 1,2,\dots\dots\dots,n \end{array}\right\} \dots (SA_1)$$

ii. $1 // L_{max}$ the maximum lateness which is solved by EDD rule [9] and the formulation mathematical form as:

$$\left.\begin{array}{ll} Z_2 = min_{\sigma(j)}\{L_{\max(\sigma)}\} & \\ Subject\ to & \\ C_{\sigma(j)} \geq p_{\sigma(j)} & j = 1,2,\dots\dots\dots,n \\ L_j = C_j - d_j & j = 1,2,\dots\dots\dots,n \\ d_{\sigma(j)} > 0, p_{\sigma(j)} > 0 & j = 1,2,\dots\dots\dots,n \end{array}\right\} \dots (SA_2)$$

Theorem (3.1)[28]: $Z_1 + Z_2 \leq H_1$ where Z_1, Z_2 and H_1 are the minimum objective function value of $(SA_1), (SA_2)$ and (A) respectively.

4. Special Cases

A machine scheduling problem of type NP-hard is not easily solved and it is more difficult when the objective function is multi objective. Using some Mathematical programming methods to find optimal solution for this kind of problem, such as dynamic programming and branch and bound method. Sometimes special cases for this problem can be solved. A special case for scheduling problem means finding an optimal schedule directly without using mathematical programming techniques. A special case, if it exists, depends on satisfying some conditions in order to make the problem easily solvable. These conditions depend on the objective function as well as the jobs [29]. In this section, some special cases for problem (A) and (B) are given.

Case (1):

If the jobs of a schedule π ordered according to $WDSPT$ and satisfy (JIT) $\forall$ job $j \in \pi$, then π gives optimal solution for the problems (A) and (B).

Proof: Since $\forall\, j \in \pi,\ C_j = d_j$, then $L_j = L_{max} = L^h_{max} = 0$. But $WDSPT$ gives an optimal solution for $\sum_{j=1}^{n} w_j(1-e^{-rC_j})$. So π is optimal solution for the problem (A) and (B). ■

Case (2):

The EDD schedule gives an optimal solution for the problems (A) and (B) if, $h_j = h, \forall j \in N$ and $(\sum_{j=1}^{n} w_j(1-e^{-rC_j})(EDD)) = \sum_{j=1}^{n} w_j(1-e^{-rC_j})(WDSPT)$.

Proof: Since EDD rule is optimal solution for $1 / h_j = h / L^h_{max}$. But $\sum_{j=1}^{n} w_j(1-e^{-rC_j})(EDD)) = \sum_{j=1}^{n} w_j(1-e^{-rC_j})(WDSPT)$ (given). So EDD gives an optimal to solution for problems (A) and (B). ■

Case (3):

If Lawler algorithm (LA) satisfy $L^h_{max}(LA) = L^h_{max}(WDSPT)$ then LA algorithm gives optimal value for the problem (B).

Proof: Since L^h_{max} is minimized byLawler's algorithm but $WDSPT$ rule gives minimize to $\sum_{j=1}^{n} w_j(1-e^{-rC_j})$ and $L^h_{max}(LA) = L^h_{max}(WDSPT)$. Hence LA algorithm gives an optimal solution for the problem (B). ■

Case (4) :

If $d_j = d$ and $h_j = h, \forall j \in N$. Then $WDSPT$ is optimal for the problem (B).

Proof: Since $d_j = d$ and $h_j = h$ then any sequence gives $minL_{max}$, so $WDSPT$ is optimal for the problem

$1/d_j = d, h_j = h \,/\, \sum_{j=1}^{n} w_j(1-e^{-rC_j}) + L^h_{max}$. ■

Case (5):

If $WDSPT$ rule satisfies$d_j = W_j + p_j, \forall j \in N$ then $WDSPT$ gives the optimal solution for the problem (A) and (B).

(W_j is waited time for job j).

Proof: Since $C_j = r_j + W_j + p_j$, and $d_j = W_j + p_j$. Then $C_j = d_j$ (because $r_j = 0 \forall j$) this means that for each job j, j is just in time (JIT) that is $L_{max} = L^h_{max} = 0$. Then $WDSPT$ rule gives the optimal solution for the problem (A) and (B). ■

Case (6):

If $p_j = p$ and $w_j = h_j = w, \forall j \in N$ then EDD rule gives an optimal solution for the problem (A) and (B).

Proof: Since $p_j = p$ and $w_j = h_j = w$ then any sequence gives optimal for $1/p_j = p, w_j = w / \sum_{j=1}^{n} w_j(1 - e^{-rC_j})$ problem. And EDD is optimal solution for$1/p_j = p, w_j = w/\sum_{j=1}^{n} w_j(1 - e^{-rC_j}) + L_{max}$ because EDD is optimal for $1 // L_{max}$. ■

Case (7):

If $WDSPT$ rule gives $minL^h_{max}$, then $WDSPT$ is optimal solution for the problem (B).

Proof: Since $WDSPT$ rule gives $\sum_{j=1}^{n} w_j(1 - e^{-rC_j})$ [3] and $WDSPT$ give $minL^h_{max}$ (given). So WDSPT is optimal solution for the problem (B). ■

Case(8):

If $L_{max}(WDSPT) = L_{max}(EDD)$ then $WDSPT$ is optimal for the problem (A).

Proof: Since L_{max} is minimized by EDD rule and $L_{max}(WDSPT) = L_{max}(EDD)$. Hence $WDSPT$ is optimal for the problem (A). ■

Case (9):

If $p_j = p$ and $WDSPT$ rule satisfies$d_j = jp, \forall\, j \in WDSPT$ then $WDSPT$ rule is optimal for the problem (A) and (B).

Proof: Since $C_j = \sum_{i=1}^{j} p_i$ be the completion time of the job j and $p_j = p \implies C_j = jp, \forall\, j \in WDSPT$ then $C_j = d_j$that is $L_j = 0, \forall\, j$. Hence $WDSPT$ is optimal for the problem (A) and (B). ■

Case (10):

If $WDSPT$ schedule gives $C_j = d_j$ for each $j = (1,2, \dots n)$ then $WDSPT$ s optimal for the problem (A) and (B).

Proof: See proof case (9). ■

Case (11):

If $d_j = d, \forall j\ (j = 1,2, \dots, n)$, then $WDSPT$ rule gives an optimal solution for the problem (A).

Proof: Since $d_j = d$, then any sequence gives $minL_{max}$, So $WDSPT$ is optimal for the problem $1 \,/\, d_j = d \,/ \sum_{j=1}^{n} w_j(1 - e^{-rC_j}) + L_{max}$.■

5. Dominance Rule

Because of branching scheme, the size of the search tree is directly linked to the length of the current sequence (which represents the number of nodes). Hence, a preprocessing step is performed in order to remove as many positions as possible. Reducing the current sequence is done by using several dominance rules. Dominance rules usually specify whether a node can be eliminated before its lower bound is calculated. Clearly, dominance rules are particularly useful when a node can be eliminated which has a lower bound that is less than the optimum solution [29]. Some of dominance rules are valid for minimization of the sum of discounted total weighted completion time and maximum lateness. As in the preprocessing step, similar dominance rules are also used within the branch and bound procedure to cut nodes that is dominated by others. These improvements lead to very large decrease in the number of nodes to obtain the optimal solution.

Below are the three dominance rules that are stated in order to decrease the number of nodes in search tree as well as decreasing the time.

Theorem (5.1): Let δ_k be a partial sequence which it's jobs are schedule $K \subset N$ for $i, j \in \overline{K} = N - K$ and T be completion time of the last job in k. If $p_i \leq p_j, w_i \geq w_j, d_i \leq d_j$ and $h_i \geq h_j$. Then $i \prec j$ in optimal schedule for the problem (B).

Proof:

Let $\pi = \delta_{kij}$ be a sequence where i and j are two jobs with $p_i \le p_j, w_i \ge w_j, d_i \le d_j$ and $h_i \ge h_j$. Let T be completion time of last job in δ_k.

Let $\acute{\pi} = \delta_{kji}$ a new sequence obtained (by interchange job i and j in original sequence).

π	δ_k	i	j
π'	δ_k	j	i

Fig. (1.1) The schedules π and π'

For π : $w_i(1-e^{-rC_i}) = w_i\left(1-e^{-r(T+p_i)}\right)$

$w_j(1-e^{-rC_j}) = w_j\left(1-e^{-r(T+p_{i+}p_j)}\right)$

$w_i\left(1-e^{-r(T+p_i)}\right) + w_j\left(1-e^{-r(T+p_{i+}p_j)}\right)$ .....(1)

For $\acute{\pi}$: $w_j(1-e^{-rC_j}) = w_j\left(1-e^{-r(T+p_j)}\right)$

$w_i(1-e^{-rc_i}) = w_i\left(1-e^{-r(T+p_j+i)}\right)$

$w_j\left(1-e^{-r(T+p_j)}\right) + w_i\left(1-e^{-r(T+p_j+p_i)}\right)$ .....(2)

From (1) – (2) we get

$w_i\left(1-e^{-r(T+p_i)}\right) + w_j\left(1-e^{-r(T+p_i+p_j)}\right) - w_j\left(1-e^{-r(T+p_j)}\right) - w_i\left(1-e^{-r(T+p_{j+}p_i)}\right)$

$= w_i - w_i e^{-r(T+p_i)} + w_j - w_j e^{-r(T+p_i+p_j)} - w_j + w_j e^{-r(T+p_j)} - w_i + w_i e^{-r(T+p_j+p_i)}$

$= -w_i e^{-r(T+p_i)} - w_j e^{-r(T+p_i+p_j)} + w_j e^{-r(T+p_j)} + w_i e^{-r(T+p_{j+}p_i)}$

$=e^{-rT}(-w_i e^{-rp_i} - w_j e^{-r(p_i+p_j)} + w_j e^{-rp_j} + w_i e^{-r(p_j+p_i)}$

$=e^{-rT}[(w_i - w_j)e^{-r(p_{i+}p_j)} + w_j e^{-rp_j} - w_i e^{-rp_i}]$

$=e^{-rT}\left[\frac{w_i-w_j}{e^{r(p_i+p_j}} + \frac{w_i e^{rp_i} - w_j e^{rp_j}}{e^{r(p_i+p_j)}}\right]$

$=e^{-rT}\frac{w_i-w_j+w_i e^{rp_i}-w_j e^{rp_j}}{e^{r(p_i+p_j)}}$

$=\frac{w_i(1-e^{rp_i})-w_j(1-e^{rp_j})}{e^{r(T+p_i+p_j)}} < 0$.....(3)

And for π :

$L^h_{max} = max\{L^h, L^h_j, L^h_i\}$

Where $L^h = \max_{i\in\delta_k}\{h_i L_i\}$

$L^h_i = \{h_i(C_i - d_i)\} = \{h_i(T + p_i - d_i)\}$

$L^h_j = \{h_j(C_j - d_j)\} = \{h_j(T + p_i + p_j - d_j)\}$

For schedule π'

$L'^h_{max} = max\{L'^h, L'^h_j, L'^h_i\}$

$L'^h_j = \{h_j(\acute{C}_j - d_j)\} = \{h_j(T + p_j - d_j)\}$

$L'^h_i = \{h_i(\acute{C}_i - d_i)\} = \{h_i(T + p_j + p_i - d_i)\}$

Since $\acute{C}_i \ge \acute{C}_j, d_i \le d_j$, $h_i \ge h_j$ and $L^h = L'^h$

Then $L'^h_i \ge L'^h_j, L'^h_i \ge L^h_i$ and $L'^h_i \ge L^h_j$

So $L'^{h}_{max} \geq L^{h}_{max}$ .....(4)

From (3) and (4) we get

$i \prec j$ in optimal solution for problem (B). ■

Lemma (5.1):

Let δ_k be a partial sequence which it's jobs are schedule $K \subset N$ for $i, j \in \overline{K} = N - K$ and T be completion time of the last job in k. If $p_i \leq p_j, w_i \geq w_j$ and $d_i \leq d_j$. Then $i \prec j$ in optimal schedule in the problem(A).

Proof:

By theorem(2.5.1) and EDD rule in optimal for L_{max} [9]. ■

Lemma (5.2):

Let δ_k be a partial sequence which it's jobs are schedule $K \subset N$ for $i, j \in \overline{K} = N - K$ and T be completion time of the last job in k. If $\frac{w_i e^{-rp_i}}{1-e^{-rp_i}} \leq \frac{w_j e^{-rp_j}}{1-e^{-rp_j}}$and $d_i \leq d_j$. Then i≺j in optimal schedule in the problem (A).

Proof:

Since WDSPT in optimal for the discounted total weighted completion time [3] and EDD in optimal for L_{max} [9]. ■

6. Branch and Bound (BAB) Method [30]

Our BAB method is based on forward sequencing branching rule for which nodes at level k of the search tree are corresponding to initial partial sequence in case if jobs are sequenced in first k positions. The LB at any node is the cost of scheduling jobs (this cost depends on the objective function) and the cost of un sequenced jobs (this cost depends on derived lower bound (LB)). At any level of the BAB method, if a node has LB ≥ UB, then this node is dominated. If the branching ends at a complete sequence of jobs then this sequence is evaluated, and if its value is less than the current (UB), this (UB) is reset to take that value. The procedure is then repeated until all nodes have been considered by using back tracking procedure. Backtracking procedure is the movement from the lowest level to the upper level in the BAB method.

6.1Upper bound (UB) Procedure

In this subsection, we propose a heuristic method which is applied once at the root node of search tree in (BAB) to find an upper bound (UB) on the minimization value of problem (A) and (B).

Heuristic (UB)

The Simulated Annealing is suggested to obtain a sequences to be an upper bound (UB) for the problem (A) and (B).

Algorithm UB [31]:

Let 's term

S_c : Candidate schedule

S_o : Best schedule found so far

S_k : schedule constructed at K iteration (K= iteration counter)

$G(S_o)$: Value of best schedule

$G(S_k)$: Value of schedule constructed at K iteration

$G(S_c)$: Value of candidate schedule

$$rand(x) < \exp(\frac{G(S_c) - G(S_k)}{\beta_k})$$

Where, x is a random variable having uniform distribution U[0; 1].

β_k is called cooling parameter in annealing terminology usually $\beta_k = a^k$, where $a \in [0,1]$.

Step (1): Initialize

Set K=1

Let S_k be a randomly create sequence

Let $S_k = S_o$

Then $G(S_k) = G(S_o)$

Step (2): Generate a perturbed sequence $S_c$with one of the neighborhoods operators and set $k = k + 1$.

Step (3): Evaluate the $G(S_c)$ values.

Step (4): If $G(S_c) < G(S_o)$, then let $S_k = S_c$, $G(S_o) = G(S_c)$. Set $k = k + 1$ and go to step **7**.

Step (5): Generate a random number x.

Step (6): If $rand(x) < \exp(\frac{G(S_c)-G(S_k)}{\beta_k})$, then let $S_k = S_c$, $K = N$.

Step (7): If $k < N$, then go to step **2**.

Else

Let $\beta_{k+1} = a\beta_k$ and $k = 0$.

Step (8): Stop best value of S_o stored in best.

7. The Lower Bound (LB)

In this subsection, two lower bound LB_1 and LB_2 are derived for the problems (A) and (B) respectively.

7.1The Lower Bound For Problem (A)

The lower bound for the problem (A) is based on decomposing (A) of two sub problems (SA_1) and (SA_2) as was shown in section (3). Then Z_1 was calculated to be the lower bound for(SA_1) by $(WDSPT)$ rule (sequencing the jobs in non-decreasing order of Weighted Discounted Shortest Processing Times) [3], and Z_2 was calculated to be the lower bound for(SA_2) by (EDD) rule (sequencing the jobs in non-decreasing order of due date) [9] and then applying Theorem (3.1) to get the first a lower bound LB_1 for problem (A).

Algorithm $\mathbf{LB_1}$

Step (1): Initialize order the un scheduling jobs by using WDSPT rule.

Step(2): Calculate the value of cost functionf_1 where

$f_1 = \sum_{j=1}^{n} w_j(1 - e^{-rC_j})$

Step (3): Re-order the jobs by using EDD rule.

Step(4): Calculate the value of cost functionf_2 where

$f_2 = L_{max}$.

Calculate $LB_1 = f_1 + f_2$

Algorithm $\mathbf{LB_2}$

Step (1) : Initialize order the un scheduling jobs by using WDSPT rule.

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

$Z_1 = \sum_{j=1}^{n} w_j(1 - e^{-rC_j})$.

Step (3) : Re-order jobs by using Lawler Algorithm (LA)[32].

Step (4) : Calculate the value of cost function$Z_2$where $Z_1 = L_{max}^h$

Step (5) :$\mathrm{LB}_2 = Z_1 + Z_2$.

Example (1): Data for the processing times, due dates, weighted times, denoted for lateness and discounted rate 0.1

Job	1	2	3	4	5
p_j	2	4	9	9	1
d_j	3	7	9	14	13
w_j	5	7	7	3	5
h_j	1	10	2	2	4

Solution:

Sequences (5,1,2,3,4)

UB= 35.6361

$f_1 = 13{,}6361$, $f_2 = 11$

$LB_1 = 24{,}6361.$

Z_1=13.6361 , Z_2=22

$LB_2 = 35{,}6361$

8. Computational Experience

An intensive work of numerical experimentations has been performed. Subsection (8.1) shows how instances (test problems) can be randomly generated.

8.1Test Problems

There exists in the literature a classical way to randomly generate test problems of scheduling problems[33].

- The processing time $p_j$is uniformly distributed in the interval [1,10].
- The due date d_j is uniformly distributed in the interval [P(1-TF-RDD/2),P(1-TF+RDD/2)]; where $P = \sum_{j=1}^{n} p_j$ depending on the relative range of due date (RDD) and on the average tardiness factor (TF).
- The an integer weights w_j were generated from uniform distribution [1,10].
- The an integer penalty h_j were generated from uniform distribution [1,10].

For both parameters, the values 0.2, 0.4, 0.6, 0.8 and 1.0 are considered. For each selected value of n (where n is the number of jobs), ten problems were generated.

8.2 Computational Experience with the Lower and Upper Bound of BAB Algorithm

The BAB algorithm was tested by coding it in MATLAB 7.10.0 (R2010a) and implemented on Intel (R) Core (TM) i7-4500UCPU @ 1.80 GHz,2.40 Hz with RAM 8.00 GB(2.45GB usable) personal computer.

Tables (1.1) and (1.2) show the results for problems (A) and Table (1.3) shows the results for problems (B) obtained by (BAB) algorithm. We list 10 problems for each value of n, where {5,10,15,20,25,30,35,40,45,50}, n∈{5,10,11,12} and the optimal value, upper bound (UB), initial lower bound (ILB), the number of generated nodes (Nodes), the computational time in second (Time), and the number of unsolved problems (Status).The stopping condition for the BAB algorithm was determined and we consider that the problem is unsolved (state is 1) that the BAB algorithm is stopped after a fixed period of time that is here after 1800 seconds (i.e. after 30 minutes). We observed from tables (1.2) and (1.3), the heuristics of upper bound is good algorithm. It gives the value for objective function equal to optimal or near optimal value for small value of n.

Table (1.1): The performance of initial lower bound, upper bound, number of nodes with ten jobs for each n

n	Av. of opt.	UB		LB		Av. of nodes	Av. of time	no. of un sol.
		Av. of UB	no. of opt.	Av. of LB	no. of opt.			
5	25.6578	25.6578	10	24.9073	3	10.9	0.00023	0
10	59.6866	60.0529	6	57.0832	0	192.5	0.01523	0
15	99.8799	101.052	7	97.4429	1	2941.4	0.354438	0

20	146.847	149.23	5	144.73	1	60826.2	3.26779	0
25	159.689	165.651	0	157.981	1	123916.5	7.57818	0
30	211.629	217.431	4	209.663	2	761409.8	53.5956	0
35	216.031	232.055	0	211.6	0	4139287.6	318.431	1
40	244.313	256.542	1	239.839	1	5253698.3	450.137	2
45	306.104	318.944	1	302.586	2	5423814.9	505.428	2
50	330.137	348.069	0	328.022	2	8449172	855.092	4

Table (1.2): The performance of initial lower bound, upper bound, number of nodes and computational time in second of BAB algorithm for $1//\sum_{j=1}^{n} w_j(1-e^{-rC_j})+L_{max}$ problem n=50

n	EX.	Optimal	UB	LB	Node	Time	Status
50	1	320.136	345.583	320.021	17719828	1800	1
	2	298.433	324.387	298.377	1525686	153.765	0
	3	285.128	321.918	278.514	17808736	1800.1	1
	4	420.383	424.383	420.383**	196705	19.6478	0
	5	332.393	357.206	322.388	4458796	470.553	0
	6	407.199	411.19	407.19	4750232	491.484	0
	7	359.522	367.898	357.898	18167237	1800	1
	8	294.33	327.253	294.257	17702994	1800	1
	9	222.313	231.336	209.657	993793	100.05	0
	10	361.537	369.537	361.537**	1167713	115.404	0

Table (1.3): The performance of initial lower bound, upper bound, number of nodes and computational time in second of BAB algorithm for $1//\sum_{j=1}^{n} w_j(1-e^{-rC_j})+L_{max}^{h}$ problem n=12.

n	EX.	Optimal	UB	LB	Node	Time	Status
12	1	185.808	191.109	182.653	20726565	1053.61	0
	2	80.9048	80.9048*	80.6157	13763	0.67007	0
	3	111.572	111.572*	108.088	2230161	109.249	0
	4	47.1936	49.471	45.3982	368	0.01382	0
	5	198.267	200.502	189.665	25751215	1265.49	0
	6	128.352	128.698*	122.414	10951668	585.324	0
	7	96.4971	96.4971*	93.8539	6136750	309.528	0
	8	97.2861	99.0753	89.5095	3846561	187.801	0
	9	101.935	102.901	94.2957	5466178	371.504	0
	10	190.476	198.13	184.873	30070298	1491.93	0

Optimal = the optimal value obtained by BAB method.

UB = upper bound.

ILB = initial lower bound.

Nodes = the number of generated nodes.

Time = Computational time in seconds.

* = The upper bound gives the optimal value.

** = The initial lower bound gives the optimal value.

$$\text{Status}=\begin{cases}0 & \text{if the problem is solve}\\ 1 & \text{if the problem isn't solved}\end{cases}$$

REFERENCES

[1] Rothkopf, M.H, "Scheduling Independent Tasks on Parallel Processors", Management Science, Vol. 12, pp. 437–447, (1966).

[2] Rothkopf, M.H and Smith S.A, "There are no Undiscovered Priority Index Sequencing Rules for Minimizing Total Delay Costs", Operations Research, Vol. 32, pp. 451–456, (1984).

[3] Pinedo, M.L., "Scheduling theory, algorithms, and systems", Springer Science +Business Media, LLC., New York (2012).

[4] Ji-Bo, W., Feng, S., Bo, J., and Li-Yan, W., "Permutation flow shop scheduling with dominant machines to minimize discounted total weighted completion time", Applied Mathematics and Computation 182 947–954 (2006).

[5] Yunqiang, Y., Min, L., Jinghua, H., and Mengchu, Z., "Single-Machine Scheduling With Job-Position-Dependent Learning and Time-Dependent Deterioration", Vol. 42, NO. 1, January (2012).

[6] Lin, L., Sheng-Wu Y,. Yu-Bin, W,. Yunzhang, H,. Ping, J., " Single machine scheduling jobs with a truncated sum-of-processing-times-based learning effect", Int J AdvManufTechnol 67:261–267 (2013).

[7] Guochen, S., and Yuewu, L., " Single-machine scheduling with past-sequence-dependent delivery times and deteriorating jobs" MathematicaAeterna, Vol. 3, No. 9, 799 – 806, (2013).

[8] Hongjie, L., Zeyuan, L., and Yunqiang, Y., "Some Single-Machine Scheduling Problems with Learning Effects and Two Competing Agents", The Scientific World Journal Volume Article ID 471016 (2014).

[9] Jakson, J.R., "Scheduling a Production Line to Minimize Maximum Tardiness " , Res. Report 43 Management science , Res. Project , University of California , Loss Angles, CA, (1955).

[10] Lawler, E.L., and Moore, J.M., "A functional equation and its application to resource allocation and sequencing problems", Management Sci. 1677-84 (1969).

[11] Horn, W. A., "Some simple scheduling algorithms", Naval Research Logistics Quarterly, 21(1):177-185 (1974).

[12] Lageweg, J.K., Lenstra and RinnooyKan, A.H.G., " Minimizing maximum lateness on one machine : computational experience and some applications", StatisticaNeerlandica 30, 25-41 (1976).

[13] Simons, B., "A fast algorithm for single processor scheduling", proc. 19th Ann. Symp. Foundations of computer Science, pp. 246-252 (1978).

[14] Lenstra, J.K., RinnooyKan A.H.G., and Brucker, P., " Complexity of machine scheduling problems", Ann. Discrete Math. 1, 343-362. (1977).

[15] Baker, R.K., and SU, Z., "Sequencing with due dates and early start time to minimize maximum tardiness", Naval Res. Logist. Quart. 21171-176 (1974).

[16] Carlier, J., "The one-machine sequencing problem" , European J. oper. Res. I I 42-47 (1982).

[17] Sen, T., and Gupta, S.K., "A branch-and-bound procedure to solve a bricriterion scheduling problem", llE Trans. 15, 84-88 (1983).

[18] Sen, T., Raiszadeh ,F.M.E., and Dileepan, P., "A branch and-bound approach to the bicriterion scheduling problem involving total flow time and range of lateness", Manage ment Sci. 34, 254-260 (1988).

[19] Hariri, A.M.A., and Potts, C.N., "Single machine scheduling with batch set-up times to minimize maximum lateness", Annals of Ops. Res. O, 75-92 (1997).

[20] Potts, C. N., Kovalyov, M.Y., "Scheduling with batching: a review" .European Journal of Operational Research 120, 228-249 (2000).

[21] Allahverdi, A., Ng, C. T., Cheng, T. C. E., and Kovalyov, M. Y., "A survey of scheduling problems with setup times or costs", European Journal of Operational Research, 187, 985-1032(2008).

[22] Christos, K., and George J.K., "Single-machine scheduling problems with past-sequence-dependent delivery times", Int. J. Production Economics 126. 264–266 (2010).

[23] Habibeh, N., and Lai, S.L., "Solving Single Machine Scheduling Problem with Maximum Lateness Using a Genetic Algorithm", Journal of Mathematics Research Vol. 2, No. 3; August (2010).

[24] Radostaw, R., " Minimizing maximum lateness in a single machine scheduling problem with processing time based aging effects", European J. Industrial Engineering Vol. x, No. x,xxxx Accepted 12 September (2011).

[25] Suh-Jenq, Y., Jia-Yuarn, G., Hsin-Tao, L., and Dar-Li, Y.," Single machine scheduling problem past sequence dependent delivery times and deterioration and learning effects simultaneously", ICIC International c ISSN 1349-4198 (2013).

[26] Morteza, S., and Mehde , S., " Minimizing Maximum Lateness in a Single Machine Scheduling Problem with a Fixed Availability Constraint", Scientific Research Volume: 4, Issue: 1, Pages: 155-165 (January 2015).

[27] Spyros ,T., and Alekos, T., " An optimal scheduling algorithmfor minimizing the maximum weightedlateness of unit-length independent tasks", Computers in Industry 22 283-289 (1993).

[28] Araibi, S.M., "Machine Scheduling Problem to Minimize Two and Three Objectives Function", M.Sc thesis, Dept. of mathematics, college of Education for PurSciences,Thi-QarUniversity (2012).

[29] Husein, N.A.," Machine Scheduling Problem to Minimize Multiple Objective Function", M.Sc. thesis, Dept. of mathematics, college of Education (Ibn AL-Haitham), Baghdad University (2012).

[30] Karrar, F., and Abdul-Razzaq, T.S., "Solving Multi-criteria Scheduling Problems", M.Sc. thesis University of Al-mustansiriyah, College of science, Dep. Of Mathematics (2014).

[31] Sergio, F. Fulvio, C. Antonio, C. Alberto, F. "A Simulated Annealing Algorithm for Single Machine Scheduling Problem with Release Dates, Learning and Deteriorating Effects", Proceedings of the World Congress on Engineering Vol I, WCE 2013, July 3 - 5, London, U.K.

[32] Lawler, E.L., "Optimal sequencing of a single machine subject to precedence constraints", Management Science 19/5, 544-546 (1973).

[33] Abdul-Razaq, T.S., Potts, C.N. and Van Wassenhove., "A survey of algorithms for the single machine total weighted tardiness scheduling problem", Discrete App. Math. 26(1990) 235-253.