Minimizing the Total Completion Times, the Total Tardiness and the Maximum Tardiness

Tariq S. Abdul-Razaq
Zainab M. Ali
Dept. of Mathematics / College of Science / University of Al-Mustansiriyah

Received in: 25 February 2015, Accepted in: 26 April 2015

Abstract

In this paper, the main work is to minimize a function of three cost criteria for scheduling n jobs on a single machine. We proposed algorithms to solve the single machine scheduling multiobjective problem. In this problem, we consider minimizing the total completion times, total tardiness and maximum tardiness criteria.

First a branch and bound (BAB) algorithm is applied for the $1//\sum C_i+\sum T_i+T_{max}$ problem. Second we compare two multiobjective algorithms one of them based on (BAB) algorithm to find the set of efficient (non dominated) solutions for the $1//(\sum C_i, \sum T_i, T_{max})$ problem.

The computational results show that the algorithm based on (BAB) algorithm is better than the other one for generated the total number of non dominated solutions.

Keywords: Multiobjective optimization, total Completion times, total Tardiness, maximum Tardiness, non dominated solutions.

1. Introduction

For many years, scheduling researchers focused on single regular performance measures that are non-decreasing in job completion time. The single machine scheduling problem has been extensively investigated during the last decades [1],[2],[3],[4],[5],[6],[7]. Until the late 1980s, it was common practice that in the objective function only one performance criterion was taken into account.

In practice, however, quality is a multidimensional notion. A firm, for instance, judges a production scheme on the basis of a number of criteria, for example, work in-process inventories and observance of due dates. If only one criterion is taken into account, then the outcome is likely to be unbalanced, no matter what criterion is considered. If everything is set on keeping work-in-process inventories low, then some products are likely to be completed far beyond their due dates, whereas, if the main goal is to keep the customers satisfied by observing due dates, then the work-in-process inventories are likely to be large. In order to reach an acceptable compromise, one has to measure the quality of a solution on all important criteria. This notice has led to the development of the area of multicriteria scheduling.

However, decision makers evaluate schedules according to more than one measure. Since using multiple criteria is more realistic, several multicriteria scheduling papers have appeared in the scheduling literature. Most of these papers are on single machine bicriteria scheduling problems and the minimization of couples of criteria.

Smith (1956) [18] studied a particular case of $1/\bar{d}$ /F ($\sum C_i, T_{max}$) problem where a condition of $T_{max}=0$ is imposed. Vanwassenhove & Gelders (1980) [21] extended this problem to the $1//F(\sum C_i, T_{max})$ with $T_{max}\neq 0$. The set of efficient points is characterized and a pseudo-polynomial algorithm to enumerate all these point is given. Nelson et al.(1986) [16] developed a BAB algorithm to solve the problem for mean flow time ($\sum C_i$), number of tardy jobs ($\sum U_i$) and maximum tardiness (T_{max}) simultaneously. Hoogveen and Van de Velde (1995) [9] showed that exactly the same approach can be used to solve the $1//F(\sum C_i, f_{max})$problem. Esswein and T'kindt (2002) [8] proposed a branch and bound algorithm which efficiently enumerates the set of non-dominated solutions for the $1//F(\sum W_i C_i, f_{max})$ problem. Tadie et al. (2002) [20] proposed a procedure that takes advantage of an algorithm for finding the Pareto-optima set by applying specially developed constraints to a branch and bound algorithm for the $1//F(\sum T_i, T_{max})$ problem. To find the set of efficient point for $1//F(\sum C_i, E_{max})$ problem, Kurz and Canterbury (2005) [11] used genetic algorithm and AL-Assaf (2007) [5] proposed a polynomial algorithm within special range. Recently, Oyetunji and Oluleye (2008) [17] have used a heuristic approach for minimizing total completion time and number of tardy jobs simultaneously on single machine with release date (i.e., for the $1/r_i/F(\sum C_i, U_i)$ problem. Abbas (2009) [1] presented algorithms for many bicriteria scheduling problems on a single machine with release dates. The two criteria to be minimized are C_{max} and $\sum C_i$. He presented optimal solution for the two hierarchical problems of the $1/r_i/F(\sum C_i, C_{max})$ problem. Recently multicriteria scheduling problems has been studied by several researchers in different directions ([14],[13],[3],[4])

In this paper, we consider the problem of scheduling a set $N=\{1,2,\ldots,n\}$ of n jobs on a single machine to minimize a variety of multicriteria may be stated as follows. Each of n jobs (numbered 1, . .. , n) is to be processed on a single machine which can handle only one job at a time. Associated with job i its processing time p_i and its due date d_i . All jobs are available for processing at time zero. The main object is to find a set of Pareto optimal solutions for the $1//F(\sum C_i, \sum T_i, T_{max})$ problem.

This paper begins with some notation and basic concepts of multicriteria scheduling problems are given in section 2. Formulations and some algorithms are given in section 3. A branch and bound algorithm for the $1//\sum C_i+\sum T_i+T_{max}$ is given in section 4. Characterizing

efficient solutions for the 1//F($\sum C_i$, $\sum T_i$,T_{max}) and algorithms are given in sections 5. In section 6 computational experiments is given.

2. Notation, basic concepts and analysis

The following notation will be used in this paper

n : number of jobs
p_i : processing time of job i.
d_i : due date of job i.
C_i : completion time of job i.
T_i : the tardiness of job i.
T_{max} : Max{T_i}, the maximum tardiness.
LB : lower bound.
UB : upper bound.
BAB : branch and bound.
CE : Complete Enumeration method.

Usually in multicriteria decision making each solution is represented as a point in the criteria space. The dimensionality of this space is equal to the number of criteria. Different states of the art of multicriteria scheduling can be found in the literature Nagar et al. 1995[15], Hoogeveen 2005[10]. Analysis of these works is under-lines [19].

- The necessity of knowing the results of the domain of multicriteria optimization to understand well the difficulties related to taking into account conflicting criteria.
- The need for a typology enables us to formalize the different types of problems and to unify the notations of these problems.
- The need for a knowledge of the results on single criteria scheduling problems.

An example, many scheduling problems in the production domain involve several criteria. It is clear the need to produce “just-in-time”. This need translates into two wishes, one is not to deliver to the client late, the other is not to store the finished products. To produce “just in time” therefore a trade-off production slightly late and not too early.

The total completion time ($\sum C_i$) and the maximum tardiness (T_{max}) are the most famous measures among the scheduling objectives in industrial applications. Minimizing ($\sum C_i$) involves maintaining the work in process inventory at a low level. Minimizing (T_{max}) involves reducing the penalties incurred for late jobs. Also minimizing total tardiness ($\sum T_i$) involves reducing the total penalties incurred for late jobs.

This paper considers the scheduling problem which involves three criteria on a single machine. The multicriteria scheduling problems are generally divided into three classes. In the first class, the problem involves minimization one of the criteria ($\sum C_i$,T_{max},$\sum T_i$,), which is more important than the others, subject to the constraints that the other criteria have to be optimized. In the second class, all the criteria are considered equally important and the problem involves finding efficient set of solutions. In the third class, all criteria are weighted differently and an objective function as the sum of weighted functions is defined. The problems considered in this paper belong to the three classes above.

In order to describe the multi-objective optimization in general, consider a problem with k ($k \geq 2$) conflicting objective functions ($f_i: R^n \rightarrow R$) that are to be minimized simultaneously. That is, we wish to find a solution, $x=(x_1,x_2\ldots,x_k)$, from the set of feasible solutions X, that solves the problem $\min_{x \epsilon X} f(x) = \{f_1(x), f_2(x),\ldots,f_k(x)\}$. Since it is assumed that the objectives conflict, there is no single value of x that minimizes all objectives simultaneously as in the single objective sense, so ‘‘optimal’’ must be defined differently.

3. Minimizing total flow time, total tardiness and maximum tardiness

Let N={1,2,…,n} be the set of jobs which are available at time zero and require processing on a single machine . Each job jϵN has processing time p_j and a due date d_j , given a sequence σ=(σ(1),…,σ(n)) of jobs, produce the earliest completion time $C_{\sigma(j)} = \sum_{i=1}^{j} P_{\sigma(i)}$, the tardiness of job j, $T_{\sigma(j)} = Max\{C_{\sigma(j)} - d_{\sigma(j)}, 0\}$, consequently we have total flow times $\sum_{i\epsilon N} C_{\sigma(i)}$, total tardiness $\sum_{i\epsilon N} T_{\sigma(i)}$ and maximum tardiness $T_{max}(\sigma) = \max_{j\epsilon N}\{T_{\sigma(j)}\}$.

The optimality criteria are based on the completion times $C_{\sigma(i)}$ of the jobs j in the schedule σ :

- The minimization of the sum of flow times $C_{sum} = \sum_{j\epsilon N} C_{\sigma(j)}$
- The minimization of the maximum tardiness $T_{max}(\sigma) = Max\{T_{\sigma(1)}, \dots, T_{\sigma(n)}\}$.
- The minimization of the total tardiness $T_{sum} = \sum_{j\epsilon N} T_{\sigma(j)}$

Consider the problem denoted by the 1//F($\sum C_i$,$\sum T_i$,T_{max}) problem (TP).

We will try to find efficient (Pareto optimal) solutions for (TP) which can be written as:

$$\left.\begin{array}{ll} Min\begin{Bmatrix}\sum C_i\\ \sum T_i\\ T_{max}\end{Bmatrix} & \\ s.t. & \\ C_i \geq P_i & i = 1, \dots, n \\ C_i = C_{(i-1)} + P_i & i = 2, \dots, n \\ T_i \geq C_i - d_i & i = 1, \dots, n \\ T_i \geq 0 & i = 1, \dots, n \end{array}\right\} \dots (TP)$$

This problem (TP) is difficult to solve and find the set of all efficient (Pareto optimal) solutions, we will propose efficient algorithms to find approximate set of efficient solutions for the problem (TP).

3.1 Special cases for the problem (TP)

Consider the following problems, which are special cases of the problem (TP), the object is to find a schedule that minimize the multicriteria for the following problems:

1- 1//Lex($\sum C_i$,$\sum T_i$,T_{max}) problem (TP1)
2- 1//Lex($\sum C_i$,T_{max} ,$\sum T_i$) probem (TP2)
3- 1//Lex(T_{max} ,$\sum C_i$,$\sum T_i$) problem (TP3)
4- 1//Lex(T_{max} ,$\sum T_i$,$\sum C_i$) problem (TP4)
5- 1//Lex($\sum T_i$,$\sum C_i$,T_{max}) problem (TP5)
6- 1//Lex($\sum T_i$,T_{max} ,$\sum C_i$) problem (TP6)
7- 1//$\sum C_i$+$\sum T_i$+T_{max} problem (TP7)

3.1.1 The 1//Lex($\sum C_i$,$\sum T_i$,T_{max}) problem (TP1)

This problem can be defined as:

$$\left.\begin{array}{l} Min\{T_{max}\} \\ s.t. \\ \sum_{i=1}^{n} C_i = C^*, C^* = \sum_{i=1}^{n} C_i(SPT) \\ \sum T_i = T, T\epsilon[\sum T_i(EDD), \sum T_i(SPT)] \end{array}\right\} \dots (TP_1)$$

Since in this problem (TP_1), the $\sum_{1=1}^{n} C_i$ is the more important function and should be optimal, then the following simple algorithm (ATP_1) gives the best required result.

Algorithm (ATP_1) for 1//Lex($\sum C_i$,$\sum T_i$,T_{max}) problem (TP_1):

Step (0): Order the jobs by SPT rule and calculate ($\sum C_i$,$\sum T_i$,T_{max}) point .

Step (1): If there exist a tie (jobs with equal processing times), order these jobs by EDD rule.

Example (1): Consider the problem (TP_1) with the following data:

P_i=(2,4,4,9) , d_i=(5,10,6,12)

The SPT rule gives the feasible schedules (1,2,3,4) and (1,3,2,4). The SPT* sequence (1,3,2,4) with ($\sum C_i$,$\sum T_i$,T_{max})=(37,7,7). According to the algorithm (ATP_1) above for the 1//Lex($\sum C_i$,$\sum T_i$,T_{max}) problem, the SPT*(1,3,2,4) gives minimum ($\sum C_i$,$\sum T_i$,T_{max}).

Note that the sequence (1,3,2,4) is SPT and EDD. It is well known that SPT rule minimizes $\sum C_i$, EDD rule minimizes T_{max} and if SPT and EDD are identical then $\sum T_i$ is minimized.

3.1.2 The 1//Lex($\sum C_i$,T_{max} , $\sum T_i$) problem (TP_2)

This problem can be defined as:

$$\left.\begin{array}{l} Min \sum T_i \\ s.t. \\ \sum_{i=1}^{n} C_i = C^*, C^* = \sum_{i=1}^{n} C_i(SPT) \\ T_{max} = T, T \in [T_{max}(EDD), T_{max}(SPT)] \end{array}\right\} \ldots (TP_2)$$

Since in this problem (TP_2) the $\sum_{i=1}^{n} C_i$ is the more important function and should be optimal, then the above algorithm (ATP_1) also gives the required result for this problem (TP_2).

3.1.3 The 1//Lex(T_{max} ,$\sum C_i$,$\sum T_i$) problem (TP_3)

This problem can be defined as:

$$\left.\begin{array}{l} Min \sum_{i=1}^{n} T_i \\ s.t. \\ T_{max} = T^*, T^* = T_{max}(EDD) \\ \sum_{i=1}^{n} C_i = C, C \in [\sum_{i=1}^{n} C_i\,(SPT), \sum_{i=1}^{n} C_i(EDD)] \end{array}\right\} \ldots (TP_3)$$

Since in this problem (TP_3) the T_{max} is the more important function and should be optimal, then the following algorithm (ATP_3) gives the best possible solution.

Algorithm (ATP_3) for 1//Lex(T_{max} ,$\sum C_i$,$\sum T_i$) problem (TP_3):

Step (0): Solve 1//T_{max} problem to find $T^*=T_{max}(EDD)$.

Step (1): Determined $D_i=d_i+T^*$, $\forall i \in N$, N={1,…,n}

Step (2): Let $t=\sum_{i \in N} p_i$, k=n.

Step (3): Using Smith backward algorithm to find a job $j \in N$ satisfy $D_j \geq t$, $p_j \geq p_i$ (if there exists a tie choose the job j with largest due date). Assign job j in position k.

Step (4): Set $t=t-p_j$, $N=N-\{j\}$, k=k-1, if k=1 go to step (5), otherwise go to step (3) .

Step (5): For the resulting sequence find ($\sum C_i$,$\sum T_i$,T_{max}).

Example (٢): Consider the problem (TP_3) with the following data:
P_i=(1,5,4,3) , d_i=(10,6,6,5), $T_{max}(EDD)=T^*$=6, t=13, $D_i=d_i+T^*$=(16,12,12,11)
Hence Smith backward algorithm gives the schedule (4,3,2,1) with ($\sum C_i$,$\sum T_i$,T_{max})=(35,10,6).

3.1.4 The 1//Lex(T_{max} ,$\sum T_i$,$\sum C_i$) problem (TP_4):

This problem can be defined as:

$$\left.\begin{array}{l} Min\sum C_i \\ \\ s.t. \\ \\ T_{max}=T^*, T^*=T_{max}(EDD) \\ \sum T_i=T, T\in[\sum T_i(EDD), \sum T_i(SPT)] \end{array}\right\}\ldots(TP_4)$$

Since in this problem (TP_4) the T_{max} is the more important function and should be optimal, then the above algorithm (ATP_3) also gives the required result for this problem (TP_4).

3.1.5 The 1//Lex($\sum T_i$,$\sum C_i$,T_{max}) problem (TP_5) and also the 1//Lex($\sum T_i$,T_{max} ,$\sum C_i$) problem (TP_6):

The two problems (TP_5) and (TP_6) are NP_hard problems, since the 1//($\sum T_i$) problem is NP-hard [6].

4. The 1//($\sum C_i$ +$\sum T_i$ +T_{max}) problem (TP_7):

This problem can be defined as:

$$\left.\begin{array}{ll} Min\left\{\sum_{i=1}^{n} C_i+\sum_{i=1}^{n} T_i+T_{max}\right\} & \\ s.t. & \\ C_i\geq P_i & i=1,\ldots,n \\ C_i=C_{(i-1)}+P_i & i=2,\ldots,n \\ T_i\geq C_i-d_i & i=1,\ldots,n \\ T_i\geq 0 & i=1,\ldots,n \end{array}\right\}\ldots(TP_7)$$

The aim of problem (TP_7) is to find a processing order σ=(σ(1),…,(σ(n)) of the jobs on a single machine to minimize the sum of the total completion times, the total tardiness and the maximum tardiness
($\sum C_{\sigma(i)}+\sum T_{\sigma(i)}+T_{max}(\sigma)$) σ∈S (where S is the set of all feasible solutions), which is a single object and can be minimized by BAB method.

4.1 Special cases for the problem (TP_7):

Case 1: If the SPT schedule gives $C_i\leq d_i\ \forall i\in N$ then this SPT schedule gives an optimal value for (TP_7).
Proof: It is clear. ■
Case 2: If SPT rule and EDD rule are identical, then there exists an optimal solution for the problem (TP_7).
Proof: It is clear. ■

4.2 Decomposition of problem (TP_7):

Let M= $Min\{\sum_{i=1}^{n} C_i+\sum_{i=1}^{n} T_i+T_{max}\}$
This problem can be decomposed into three subproblems (SP1), (SP2) and (SP3)

$$\left.\begin{aligned} V_1 &= min \sum_{i=1}^{n} C_i \\ s.t & \\ C_i &\geq P_i \qquad i = 1,\dots,n \\ C_i &= C_{(i-1)} + P_i \qquad i = 2,\dots,n \end{aligned}\right\} \dots (SP1)$$

$$\left.\begin{aligned} V_2 &= min \sum_{i=1}^{n} T_i \\ s.t & \\ T_i &\geq C_i - d_i \qquad i = 1,\dots,n \\ T_i &\geq 0 \qquad i = 1,\dots,n \end{aligned}\right\} \dots (SP2)$$

$$\left.\begin{aligned} V_3 &= \min\{T_{max}\} \\ s.t & \\ T_i &\geq C_i - d_i \qquad i = 1,\dots,n \\ T_i &\geq 0 \qquad i = 1,\dots,n \end{aligned}\right\} \dots (SP3)$$

This decomposition has the following properties:
First (SP1),(SP2) and (SP3) have simpler structure than the multicriteria problem (TP_7). Second it is easy to solve optimality for SP_1 and SP_3 by applying SPT rule and EDD rule respectively and also to get a lower bound for (SP_2).

4.3 Derivation of lower bound (LB) and upper bound for problem (TP_7):

The lower bound (LB) is based on decomposing problem (TP_7) into three subproblems (SP_1), (SP_2) and (SP3). Then calculate V_1 to be the minimum value for (SP_1), V_2 to be lower bound for (SP_2) and V_3 to be the minimum value for (SP_3). Then applying the following theorem:

Theorem (1)[12]: $V1+V2+V3 \leq M$ where V1,V2,V3 and M are the minimum objective function values of (SP1),(SP2),(SP3) and (TP_7) respectively. ■

To get a lower bound LB for the problem (TP_7):
For (SP_1) we compute V_1 by sequencing the jobs in SPT order to find the minimum total completion times ΣC_i. For (SP_2) we compute a lower bound for V_2 by sequencing the jobs in EDD order to find the minimum maximum tardiness (T_{max}(EDD)).
Since $T_{max}(EDD) \leq \sum T_i(opt)$. Hence T_{max}(EDD) is a lower bound for V_2 (i.e $T_{max}(EDD) \leq V_2$).
For (SP_3) we compute V_3 by sequencing the jobs in EDD order to find minimum maximum tardiness.
Then applying theorem (1) to obtain $LB=V_1+T_{max}(EDD)+V_3$.

To calculate a simple upper bound (UB) order the jobs by SPT rule and then set $UB=\Sigma C_i+\Sigma T_i+T_{max}$ for this SPT order.

4.4 Dominance rules (DR)

For dominance rules, if it can be shown that an optimal solution can always be generated without branching from a particular node of the search tree, then that node is dominated and can be eliminated. It is clear that (DR) are particularly useful when a node can be eliminated before its LB is calculated, and this (LB) is less than the optimal solution. The following result for (DR) for problem (TP_7) is given next.

Lemma (1)[7]: If $p_i \leq p_j$ and $d_i \leq d_j$, then there exists an optimal sequence in which job i is sequence before job j.

Hence the (BAB) method now can be used to find the optimal solution for the problem (TP_7), with the help of dominance rule for the problem (TP).

5. Characterizing efficient solutions for the problem (TP):

We first characterize the set of efficient points and then propose an efficient algorithms to find most of the efficient points.

5.1 Some results for the $1//(\sum C_i, \sum T_i, T_{max})$ problem (TP):

Proposition (1): If the SPT schedule gives $C_i \leq d_i \ \forall i \in N$, then this SPT schedule is the only efficient solution for the problem $1//(\sum C_i, \sum T_i, T_{max})$.
Proof: It is clear. ■

Proposition (2): There exists an efficient solution for problem (TP) $1//(\sum C_i, \sum T_i, T_{max})$ that satisfies the SPT rule.
Proof: Suppose first, that all processing times are different. The unique SPT sequence (SPT^*) gives the absolute minimum of $\sum C_i$. Hence there is no sequence $\sigma \neq SPT^*$ such that:

$$\sum_{i=1}^{n} C_i(\sigma) \leq \sum_{i=1}^{n} C_i\,(SPT^*)\,, \sum_{i=1}^{n} T_i(\sigma) \leq \sum_{i=1}^{n} T_i(SPT^*)\ and\ T_{max}(\sigma) \leq T_{max}(SPT^*) \ldots (1)$$

with at least one strict inequality.
If more than one SPT sequence exists (jobs with equal processing times), let SPT^* be a sequence satisfying the SPT rule and the jobs with equal processing times are ordered in EDD rule to minimize $\sum T_i(SPT^*)$ and $T_{max}(SPT^*)$. Note that if we have SPT^* is not unique we can prove that every SPT^*sequence is an efficient, it is clear that sequence that does not satisfy the SPT rule cannot dominate an SPT^* sequence (1). Note if σ is an SPT but not SPT^* sequence it cannot dominate SPT^* since:

$$\sum_{i=1}^{n} C_i(\sigma) = \sum_{i=1}^{n} C_i(SPT^*)\,, \sum_{i=1}^{n} T_i\,(SPT^*) \leq \sum_{i=1}^{n} T_i(\sigma)\ and\ T_{max}(SPT^*) \leq T_{max}(\sigma) \quad \ldots (2)$$

Hence, all SPT^* sequences are efficient. ■

Example (3): Consider the problem (TP) with the following data:
$P_i=(2,2,4,7)$, $d_i=(4,6,5,9)$. The SPT sequence (1,2,3,4) gives $(\sum C_i, \sum T_i, T_{max})=(29,9,6)$ is one of the efficient solutions for the problem (TP).

Proposition (3): If SPT rule and EDD rule are identical, then there exists only one efficient solution for (TP) $1//(\sum C_i, \sum T_i, T_{max})$.
Proof: It is clear. ■

Example (4): Consider the problem (TP) with the following data:
$P_i=(1,3,3,7)$, $d_i=(4,6,8,9)$. The sequence (1,2,3,4) (which is SPT and EDD) gives the only one efficient point $(\sum C_i, \sum T_i, T_{max})=(26,5,5)$.

Proposition (4): If $T_{max}(EDD)=0$, then there exists an efficient sequence for $1//(\sum C_i, \sum T_i, T_{max})$ problem obtained by Smith backward algorithm (SBA).
Proof: If $T_{max}(EDD)=0$, then it's clear that SBA gives a schedule with $C_i \leq d_i$ for each $i \in N$ and this schedule also gives minimum $\sum C_i$ with $\sum T_i=0$ and $T_{max}=0$. This schedule cannot be dominated by any other schedule since $\sum C_i$ is the minimum for all schedules with $\sum T_i=0$ and $T_{max}=0$. Hence this schedule obtained by SBA is efficient for $1//(\sum C_i, \sum T_i, T_{max})$ problem.■

Example (5): consider the problem (TP) with the following data:
$P_i=(5,3,4,6)$, $d_i=(10,12,13,18)$. The schedule (2,1,3,4) obtained by SBA gives the efficient point $(\sum C_i, \sum T_i, T_{max})=(41,0,0)$.

5.2 Algorithm (ATP) for determination of Pareto points:

From the above results, We now propose algorithm (ATP) to determine the set of efficient solutions for the $1//(\sum C_i,\sum T_i,T_{max})$ problem (TP).

Algorithm (ATP) for finding efficient solutions for the problem $1//(\sum C_i,\sum T_i,T_{max})$ (TP) by (modifying Lawler algorithm):

Step (0): Put $\Delta=\sum p_i$, $N=\{1,2,\ldots,n\}$, $K=n$, $t=C_{max}=\sum p_i$ and $\sigma=(\varphi)$.
Step (1): Calculate T_i, $\forall i\in N$ by using Lawler algorithm, where $T_i=Max\{C_i-d_i,0\}$.
Step (2): Find a jop $j^*\in N$, such that $T_{j^*}\leq\Delta$ and $P_{j^*}\geq P_i$, $\forall j^*$, $i\in N$ and $T_i\leq\Delta$, if $P_{j^*}=P_i$ choose the job with largest d_{j^*} then assign job j^* in position K of $\sigma=(\sigma(K),\sigma)$.
If no job j^* with $T_{j^*}\leq\Delta$ go to step (6).
Step (3): Set $t=t-P_{j^*}$, $N=N-\{j^*\}$, $K=K-1$, if $K>1$ go to step (1), otherwise go to step (4).
Step (4): Compute $\sum C_{\sigma(i)}$, $\sum T_{\sigma(i)}$, $T_{max}(\sigma)$ for the resulting sequence jobs $\sigma=(\sigma(1),\ldots,\sigma(n))$.
Step (5): Put $\Delta=T_{max}(\sigma)-1$, $N=\{1,\ldots,n\}$, $K=n$, $t=C_{max}=\sum P_i$ and $\sigma=(\varphi)$, go to step (1).
Step (6): Stop.

Example (6): To illustrate algorithm (ATP), we consider the problem (TP) with the following data: $p_i=(4,6,2,5,5)$, $d_i=(20,9,4,7,10)$.
Table (1) show the results of efficient solutions obtained by algorithm (ATP)

5.3 Branch and Bound algorithm for determination of Pareto points:

We now propose another algorithm to determine the set of efficient solutions for the $1//(\sum C_i,\sum T_i,T_{max})$ problem (TP).
Algorithm (ZTP) for finding most efficient solutions for this problem, this algorithm depends on the techniques of branch and bound (BAB) method and the definition of efficient solutions as follows:

Algorithm (ZTP) for finding efficient solutions for the problem $1//(\sum C_i,\sum T_i,T_{max})$ (TP)

Step (0): Find the upper bound (UB) by SPT rule, that is sequencing the jobs in non-decreasing order of their processing time P_i $(i=1,2,\ldots,n)$ for this order σ compute $\sum C_{\sigma(i)}$, $\sum T_{\sigma(i)}$ and $T_{max}(\sigma)$ and set UB=$(\sum C_i,\sum T_i,T_{max})$ at the parent node of the search tree. UB is efficient by proposition (1) and add this efficient solution to the set of solutions SE. Also if $T_{max}(EDD)=0$, then there exists an efficient sequence obtained by proposition (3), and add this efficient solution to the set SE.
Step (1): For each node in the search tree of BAB method, i.e, for each partial sequence of jobs σ compute a lower bound $LB(\sigma)$ as follows:
$LB(\sigma)$=cost of sequence jobs (σ) for the objective functions + cost of unsequence jobs obtained by sequence the jobs in SPT rule.
Step (2): Branch from each node with $LB\leq UB$.
Step (3): At the last level of the BAB method, we get a set of solutions, if $(\sum C_i,\sum T_i,T_{max})$ denote the outcome is added to the set SE unless it is dominated by the previously obtained efficient solutions in SE.
Step (4): Stop.

Note that the set of efficient solutions (SE) is update with the solutions $\sigma\epsilon S$ (set of all feasible solutions). A solution σ is added to the set SE if $\sigma\notin SE$ and it is not dominated by any solution of SE. The solutions of SE dominated by σ are removed from SE. In this study, the algorithm (ZTP) stops when maximum CPU time is reached (1800 sec.). The set SE of non-dominated solutions is initialized with a solution σ_1 (SPT rule) in which the jobs are arranged in non-decreasing order of the processing time.

Example (7): To illustrate algorithm (ZTP), we consider again example (6) for the problem (TP) with the following data:

P_i=(4,6,2,5,5) , d_i=(20,9,4,7,10). Using algorithm (ZTP) to find the set of efficient solutions.

It is clear from table (2) that algorithm (ZTP) generating all efficient solutions (see table (1)) for problem (TP).

6. Computational experiments:

6.1 Problem instances

The performance of the BAB algorithm for the problem (TP_7) is compared on 5 problem instances for each n with the complete enumeration method (CEM). The sizes of these instances are n=4,5,6,7,8,9,10 and the large sizes are n=11,12,13,14,15,16,17,18. The problems were generated randomly, similar to that of [2]. For each job j, the processing time p_j was uniformly generated from the uniform distribution [1,10]. Also, for each job j, an integer due date d_j is generated from the uniform distribution [(1-TF-RDD/2)TP , (1-TF+RDD/2)TP], where TP is the total processing time of all the jobs, TF is the tardiness factor, and RDD is the relative range of the due dates. For the two parameters TF and RDD, the values 0.2, 0.4, 0.6, 0.8, 1.0 are considered. For each selected value of n, one problem was generated for each of the five values of parameters producing 5 problems. Whenever a problem remained unsolved within the time limit of 1800 seconds, computation was abandoned for that problem.

Also the algorithms (ATP) and (ZTP) were tested on the problem (TP). The cardinal measure is used for each algorithm. We compare the number of efficient (non dominated) solutions that obtained by the two algorithms (ATP) and (ZTP). We denoted by SE_1 and SE_2 the set of efficient solutions (approximated Pareto fronts) obtained by the algorithms (ATP) and (ZTP) respectively. It is clear that the optimal Pareto front for our problem (TP) is not known, it is only known for small $n\leq10$, which is given by complete enumeration method (CEM). The performance of an algorithm is then measured in term of the quality of the solutions obtained by this algorithm with respect to the solutions in (CEM).

All the algorithms were tested by coding them in Matlab 8.3.0 (R2014a) and implemented on Intel (R) core (TM) i5 CPU (3210M) @ 2.50 GHZ, with RAM 4.00 GB personal computer.

6.2 Comparison of results

The results of the BAB algorithm (ATP and ZTP) for the problem (TP_7) for the test problems are compared with the results of complete enumeration method (CEM). Comparison of these results is given in table (3) for $n\leq10$. The results for $n>10$ for the BAB is given in table (4).

Also the two algorithms were run for all the instances of the test problems given in subsection (6.1) for the problem (TP). The sets ES_1 and ES_2 contain the efficient solutions found by the algorithm Alg(ATP) and Alg(ZTP) respectively. The performance of these algorithms is compared on 5 problem instances for each n. The cardinal measure is calculated for the two sets ES_1 and ES_2 and given in tables (5) and (6).

Conclusion

In this paper, a single machine scheduling problem, with total completion times, total tardiness and maximum tardiness has been considered. The objective is to find a schedule that minimizes the sum of completion times, sum of tardiness and maximum tardiness. We have given simple approximation algorithms, by which we can efficiently generate Pareto-optimal solutions. The computational results show that with our BAB algorithm (ZTP) we can find Pareto-optimal solutions for some number of jobs n, a comparison is mode with the results of CEM for $n\leq10$, and for $n\geq10$ a comparison is mode with the results of algorithm (ATP). Also a branch and bound algorithm is given to find the optimal solution for the sum of the three objectives.

The research may be extended to a more general case in which we have more than three objectives.

References

1. Abbas, I.T., (2009), "The performance of multicriteria scheduling in one machine", M.Sc. thesis, Univ. of Al-Mustansiriyah, College of Science, Dept. of Mathematics.
2. Abdul-Razzaq, K. F. , (2014), "Solving Multicriteria Scheduling Problems", M. Sc. Thesis, University of AL-Mustansiriyah, College of Science, Dept. of Mathematics.
3. Ahmed, M. G. , (2012), "A single machine scheduling problem to minimize the sum of total Completion times and total Late works", Res. Al-Mustansiriyah Journal of Science 23, 7, 117-130.
4. Al Zuwaini, M. K. ; Al Saidy, S. K. and Abdul-Razaq, T. S. , (2011), "Comparison study for some local search methods for multiple objective function in a single machine scheduling problem", Journal of Basrah Research (science) 37, 4, 103-113.
5. Al-Assaf, S.S., (2007), "Solving multiple objectives scheduling problems", M.Sc. thesis, Univ. of Al-Mustansiriyah, College of Science, Dept. of Mathematics.
6. Du, J. and Leung, (1990), "Minimizing total tardiness on one machine is NP- Hard", Mathematics of operation Res. journal, 15, 3, 483-495.
7. Emmons, H. , (1969), "One machine sequencing to minimize certain functions of job tardiness", Operation Research 17, 701-715.
8. Esswein,C., and T'kindt, V., (2002), "Resolution of a single machine scheduling problem with a maximum cost criterion and the weighted sum of completion times", available on line http://www.adeit.uv.es/pms2002/papers/N57.pdf.
9. Hoogeveen, J.A., (2005), "Invited review Multicriteria Scheduling" , European Journal of operational Research 167,592-623.
10. Hoogeveen, J.A., and Van de Velde, S.L., (1995), "Minimizing total completion time and maximum cost simultaneously is solvable in polynomial time", Operations Res. Letters 17, 205-208.
11. Kurz, M.E., and Canterbury, S., (2005), "Minimizing total flowtime and maximum earliness on a single machine using multiple measures of fitness", Genetic and Evolutionary Computation Conference, 803-809.
12. Mahmood, A. A. , (2001), "Solution procedures for scheduling job families with setups and due dates" , M. Sc. Thesis, University of AL-Mustansiriyah , College of Science, Dept. of Mathematics.
13. Mahmood, A. A., (2014), "Approximation solution for multicriteria scheduling problems", Res. Al-Rafidain University College for Sciences 1, 34, 161-179.
14. Mahmood, A. A. and Abdul-Razaq, T. S., (2014), "Exact algorithm for minimizing the sum of total late work and maximum late work problem", Res. Diyala Journal for Pure Science 10, 1, 39-50.
15. Nagar, A., Jorge, H., and Sunderesh H., (1995), "Multiple and bi-criteria Scheduling: A literature survey", European Journal of Operational Research North-Holland, 81, 88-104.
16. Nelson, R.T.; Sarin, R.K., and Daniels, R.L., (1986), "Scheduling with multiple performance measures: The one machine case", Management Science, 32, 464-479.
17. Oyetunji , E.O., and Oluleye, A.E., (2008), "Heuristics for minimizing total completion time and number of tardy jobs simultaneously on single machine with release time", Res. Journal of Applied Sciences, 3,147-152.
18. Smith, W.E., (1956), " Version optimization for single stage production", Naval Res. Logistics Quarter, 3, 59-66.

19. T'Kindt, V. and Billaut, J.C. , (2006), "Multicriteria Scheduling : Theory, Models and Algorithms" Springer-verlag Berlin Heidelberg ,2nd ed.
20.Tadei, R.; Grosso, A., and Della Croce, F., (2002), "Finding the pareto-optima for the total and maximum tardiness single machine problem", Discrete Applied Mathematics 124, 117-126.
21.Vanwassenhove, L.N., and Gelders, L.F., (1980), "Solving a bicriterion scheduling problem", European Journal of Operational Res. 4, 42-48.

Table No.(1) :The results of efficient solutions for the example (6) obtained by CEM and algorithm (ATP).

Sequence	$\sum C_i$	$\sum T_i$	T_{max}
(3,1,4,5,2)*	57	23	13
(3,4,1,5,2)	58	19	13
(3,1,4,2,5)*	58	24	12
(3,4,5,1,2)	59	15	13
(3,4,1,2,5)	59	20	12
(3,4,5,2,1)*	61	13	9
(3,4,2,5,1)*	62	14	8

Note that (*) indicates that the efficient schedule obtained by algorithm (ATP).

Table No.(2) :The results of efficient solutions for example (7) by BAB method Algorithm (ZTP).

Sequence	$\sum C_i$	$\sum T_i$	T_{max}
(3,1,4,5,2)	57	23	13
(3,4,1,5,2)	58	19	13
(3,1,4,2,5)	58	24	12
(3,4,5,1,2)	59	15	13
(3,4,1,2,5)	59	20	12
(3,4,5,2,1)	61	13	9
(3,4,2,5,1)	62	14	8

Table No.(3): Comparison of results of minimum sum of CE and BAB methods for the problem (TP_7).

$1//\sum C_i+\sum T_i+T_{max}$											
n	EX	Sum.CE	Sum.BAB	n	EX	Sum.CE	Sum.BAB	n	EX	Sum.CE	Sum.BAB
4	1	44	44	5	1	99	99	6	1	174	174
	2	31	31		2	82	82		2	148	148
	3	108	108		3	66	66		3	105	105
	4	130	130		4	152	152		4	99	99
	5	66	66		5	146	146		5	199	199
7	1	227	227	8	1	316	316	9	1	257	257
	2	188	188		2	276	276		2	510	510
	3	284	284		3	418	418		3	243	243
	4	232	232		4	202	202		4	258	258
	5	103	103		5	282	282		5	210	210
10	1	397	397								
	2	490	490								
	3	340	340								
	4	523	523								
	5	350	350								

Table No.(4): The results of minimum sum of BAB method for the problem (TP_7).

$1//\sum C_i+\sum T_i+T_{max}$										
n	EX	Sum.BAB	n	EX	Sum.BAB	n	EX	Sum.BAB	n	EX	Sum.BAB
11	1	727	12	1	630	13	1	650	14	1	978
	2	586		2	598		2	575		2	887
	3	671		3	441		3	652		3	623
	4	460		4	689		4	573		4	383
	5	430		5	630		5	619		5	962
15	1	895	16	1	1046	17	1	*	18	1	1287
	2	966		2	791		2	758		2	*
	3	779		3	898		3	919		3	*
	4	909		4	925		4	*		4	1532
	5	916		5	643		5	*		5	1515

In tables (3) and (4) we have:
n: number of jobs
EX: Example number
Sum.CE: The optimal value obtained by Complete Enumeration method (CEM).
Sum.BAB: The optimal value obtained by BAB method with dominance rule.
(*): Indicates that the problem unsolved within the limit of 1800 seconds.

It is clear from table (4) that the BAB method is satisfactory for solving small sized problems. However, a sharper lower bound is needed to cut down the size of the search tree when the number of jobs exceeds 10.

Table No.(5): Shows the performance of the two proposed algorithms with (CEM) for n≤10 for the problem (TP).

Efficient solutions ($\sum C_i, \sum T_i, T_{max}$)				
n	EX	|CEM|	Alg(ATP) $\|ES_1\|$	Alg(ZTP) $\|ES_2\|$
4	1	1	1	1
	2	1	1	1
	3	1	1	1
	4	1	1	1
	5	2	1	2
5	1	1	1	1
	2	1	1	1
	3	1	1	1
	4	2	1	2
	5	3	1	3
6	1	2	1	2
	2	1	1	1
	3	1	1	1
	4	6	2	6
	5	5	2	5
7	1	1	1	1
	2	2	1	2
	3	6	3	6
	4	3	2	3
	5	9	2	9
8	1	1	1	1
	2	1	1	1
	3	6	2	6
	4	3	1	3
	5	5	1	5
9	1	1	1	1
	2	3	1	3
	3	3	1	3
	4	19	2	19
	5	8	3	8
10	1	2	1	2
	2	1	1	1
	3	7	2	7
	4	7	2	7
	5	10	3	10
total		127	49	127

Table No.(6): Shows a comparison of these two algorithms (ATP and ZTP) for the problem (TP).

Efficient solutions ($\sum C_i, \sum T_i, T_{max}$)			
n	EX	Alg(ATP)	Alg(ZTP)
		$\|ES_1\|$	$\|ES_2\|$
11	1	1	1
	2	2	8
	3	2	8
	4	3	6
	5	6	39
12	1	2	8
	2	2	14
	3	2	10
	4	2	12
	5	9	118
13	1	1	3
	2	3	26
	3	6	54
	4	6	46
	5	4	36
14	1	2	7
	2	2	8
	3	2	8
	4	4	21
	5	1	1
15	1	2	2
	2	5	5
	3	4	15
	4	1	3
	5	3	6
16	1	2	8
	2	1	9
	3	1	5
	4	4	4
	5	2	22
17	1	2	*
	2	2	12
	3	1	3
	4	4	*
	5	4	*
18	1	1	5
	2	2	*
	3	7	*
	4	1	2
	5	2	2

In tables (5) and (6) we have:
n: number of jobs
EX: Example number
$|ES_1|$: The cardinal number of efficient solutions for the set ES_1.
$|ES_2|$: The cardinal number of efficient solutions for the set ES_2.
(*): Indicates that the problem unsolved within the limit of 1800 seconds.

It is clear from the results of tables (5) and (6) that Alg.(ZTP) is better than the Alg.(ATP) for generating the total number of non dominated solutions for the NP-hard multicriteria problem (TP).

تصغير مجموع أوقات الأتمام ، مجموع التأخير اللاسالب وأكبر تأخير لاسالب

طارق صالح عبد الرزاق
زينب محروز علي

الجامعة المستنصرية \ كلية العلوم \ قسم الرياضيات

استلم البحث في : 25شباط ٢٠١٥، قبل البحث في: 26 نيسان ٢٠١٥

الخلاصة

في هذا البحث، العمل الرئيسي هو تصغير دالة لثلاثة معايير والحاصلة من جدولة n من الاعمال على ماكنة واحدة. اقترحنا خوارزميات لحل مسألة جدولة الماكنة متعددة الأهداف. وفي هذه المسألة أخذنا بنظر الاعتبار تصغير الأهداف مجموع أوقات الاتمام ، مجموع التأخير اللاسالب وأكبر تأخير لاسالب.

أولاً خوارزمية التفرع والتقيد استخدمت للمسألة $(1//\sum C_i+\sum T_i+T_{max})$. ثانياً تم مقارنة خوارزميتان للدوال متعددة الأهداف واحداهما تعتمد على طريقة التفرع والتقيد في أيجاد مجموعة الحلول الكفؤة (غير المهيمن عليها) للمسألة $1//(\sum C_i,\sum T_i,T_{max})$. ومن النتائج الحسابية تبين ان الخوارزمية التي تعتمد على خوارزمية التفرع والتقيد هي الأفضل من الأخرى في ايجاد العدد الكلي للحلول غير المهيمن عليها.

الكلمات المفتاحية: متعددة الحلول المثلى، مجموع أوقات الاتمام ، مجموع التأخير اللاسالب ،أكبر تأخير لاسالب، الحلول غير المهيمن عليها.