BICRITERIA SCHEDULING OF EQUAL LENGTH JOBS WITH RELEASE DATES ON IDENTICAL PARALLEL MACHINES

Hari Balasubramanian, John Fowler, Ahmet Keha
Industrial Engineering, Arizona State University

Abstract We consider bicriteria scheduling problems on parallel, identical machines with job release dates. The jobs are assumed to have equal processing times. Our main goal in this paper is to report complexity results for bicriteria problems in this environment when the number of machines is assumed to be a fixed or constant value. The results are based on a straightforward use of the dynamic program of Baptiste (2000). When the the number of machines is given, we show that it is possible to minimize, in polynomial time, a composite linear objective function involving 1) sum of completion times and total tardiness, and 2) under a given condition the sum of weighted completion times and total tardiness. We then use a technique proposed by Aneja and Nair (1979) to generate extreme schedules on the efficient frontier for the first problem and a subset of them for the second one. We also show that it is possible to generate in polynomial time the set of Pareto Optimal points for bicriteria problems with C_{max} as one of the criteria and either of $\sum w_jC_j$ and $\sum T_j$ as the other.

Keywords: bicriteria scheduling, parallel machines, equal processing time jobs.

1. Introduction

We consider the problem of scheduling n jobs $J_1, J_2, ..., J_n$ on m parallel, identical machines.Each job has a release date r_j, a due-date d_j and weight w_j. All jobs have the same processing time p. The goal is to find a completion time C_j for each job j to solve some bicriteria optimization problems. The completion times have to be feasible, that is 1) jobs start after their release date i.e. $\forall j, C_j - p \geq r_j$ and 2) no more than m machines are used at any time t,i.e $\forall t, | \ \{J_j | C_j - p \leq t < C_j\} \ | \leq m$. Additionally, all schedules are non-preemptive.

Since criteria are often conflicting, it is rare that a single solution is best for all of them. In any multicriteria problem it is important to point out the nature of optimization being performed given that such trade-offs exist.

Let S be the set of feasible solutions for a bicriteria optimization problem and $z_1(X)$ and $z_2(X)$ be the objective values for criteria z_1 and z_2 (both of which need to be minimized) for a feasible solution $X \in S$. We follow, for the most part, the notation and terminology of T'Kindt and Billaut (2002) and Hoogeoveen (2004).

It may be of interest to minimize a secondary criterion z_2 given the optimal value of the primary criterion z_1. This is called lexicographic or hierarchial optimization represented as Lex (z_1, z_2). Sometimes the decision-maker may have a linear function in mind that he intends to minimize: a composite function of the form $F_l(z_1, z_2) = \alpha * z_1 + (1 - \alpha) * z_2$, where $0 \leq \alpha \leq 1$. But perhaps of most interest - and in many cases the most difficult to generate - is the set of Pareto optimal or non-dominated solutions.

DEFINITION 1 *A solution X^* is Pareto optimal or non-dominated if there exists no other solution $X \in S$ for which $z_1(X) \leq z_1(X^*)$ and $z_2(X) \leq z_2(X^*)$ where at least one of the inequalities is strict.*

DEFINITION 2 *A solution X^* is said to be weakly Pareto optimal if there exists no other solution $X \in S$ for which $z_1(X) < z_1(X^*)$ and $z_2(X) < z_2(X^*)$*

Let $co(ND)$ represent the convex hull of all the non-dominated solutions when each solution in plotted on in criteria space (with each axis representing a criterion) based on the values the solution has for the two criteria. Hooegeveen (2004) defines the *efficient frontier* as the lower envelope of $co(ND)$ and an *extreme point* as a Pareto optimal solution that is also a vertex of the efficient frontier. The set of extreme points is of interest as sometimes it is not possible to generate all the Pareto optimal solutions. It is worthwhile to note that there exists an extreme point that is an optimal solution to the linear combination $F_l(z_1, z_2) = \alpha * z_1 + (1 - \alpha) * z_2$, for any α such that $0 \leq \alpha \leq 1$ The extreme points are also a subset of the *supported* solutions, defined by T'Kindt and Billaut (2002) as set of Pareto optimal points located on the efficient frontier. The set of *non-supported* solutions are set of set of Pareto optimal points that do not lie on the efficient frontier.

Insert figure 1 here.

In this paper, we consider the minimization of a linear combination of criteria $F_l(z_1, z_2) = (\alpha * z_1 + (1 - \alpha) * z_2$, where $0 \leq \alpha \leq 1$, and z_1 and

z_2 are the two objectives) for the following problems. In the notation of T'Kindt and Billaut (2002), we have 1) $P_m|r_j, p_j = p|F_l(\sum C_j, \sum T_j)$ and 2) $P_m|r_j, p_j = p|F_l(\sum w_j C_j, \sum T_j)$. In the next step, we generate the extreme schedules on the efficient frontier for the first problem and a subset of the extreme schedules on the efficient frontier for the second one.

We also present algorithms to generate the set of Pareto optimal points for some bicriteria problems. The technique used is the $\epsilon - constraint$ approach and is written in the γ field of the $\alpha|\beta|\gamma$ scheduling notation as $\epsilon(z_1|z_2)$ (T'Kindt and Billaut (2002)). It implies the minimization of z_1 given a fixed value of z_2. Our bicriteria problems are represented as:
1) $P_m|r_j, p_j = p|\epsilon(\sum w_j C_j|C_{max})$ and 2) $P_m|r_j, p_j = p|\epsilon(\sum T_j|C_{max})$.

2. Related Work

The recent survey Hoogeoven (2004) is a comprehensive study of multicriteria scheduling research to date. The survey reveals the dearth of theoretical results in bicriteria problems involving parallel machines, mostly because very few single criteria problems are have polynomial time complexities. Recently, however, some results have emerged in this area. Baptiste and Brucker (2004) show that $P_m||Lex(\sum C_j, \sum U_j)$ (the minimization of the total number of tardy jobs given the optimal value of the total completion time) is solvable in pseudopolynomial time and that $P_2||Lex(\sum C_j, \sum U_j)$ is NP-hard. Gupta *et al.* (2003) prove further complexity results: they show that$P||Lex(\sum C_j, C_{max})$ is strongly NP-hard and thus $P||Lex(\sum C_j, \sum U_j)$ is strongly NP-hard as well; $P||Lex(\sum C_j, T_{max})$ can be solved in pseudopolynomial time; and $P_m||Lex(\sum C_j, \sum U_j)$ can be solved in polynomial provided the processing times are all different. In a later paper, Gupta and Ruiz-Torrez (2004)propose heuristic algorithms for generating a set of efficient (non-dominated) schedules on identical parallel machines involving total flow-time and total number of tardy jobs. Sarin and Prakash (2004) propose polynomial algorithms for bicriteria scheduling of various pairs of traditional scheduling objectives of in a parallel machine environment with equal release dates assuming that all jobs have equal processing times. Their approach is lexicographic optimization i.e. they assume a primary measure and a secondary measure. Angel *et al*(2003) identify a class of multiobjective optimization problems possessing a fully polynomial time approximation scheme (FPTAS) for computing an ε-approximate Pareto curve. T'kindt et al. (2001) consider a bicriteria scheduling problem on unrelated parallel machines applicable to a in the production of glass bottles.

In the last few years, a slew of parallel machine scheduling problems with release dates, whose complexities had previously been unknown, were shown to be polynomial (Baptiste (2000), Baptiste *et al.* (2002)). A key characteristic which makes these problems tractable is the polynomial number of time points at which jobs can start or finish in a schedule. We return to this property later in section 5.

For our bicriteria optimization requirements, we use the the dynamic program of Baptiste (2000) that solves $\sum w_j C_j$ and $\sum T_j$ optimally (each criteria individually) and works for sum functions ($\sum f_j$) that are non-decreasing for all f_i and monotonous for all $f_i - f_j$. For notational convenience we call this dynamic program DPB (Dynamic Program by Baptiste).

3. Minimizing a composite linear function

As stated earlier, DPB works for functions of the type $\sum f_j$ that are non-decreasing and monotonous, formally defined below, (Baptiste, 2000):

DEFINITION 3.1 *The functions f_i are non-decreasing, i.e., $\forall t_1, \forall t_2 > t_1, f_i(t_1) \leq f_i(t_2)$ and the functions $f_i - f_j$ are monotonous, i.e., either $\forall t_1, \forall t_2 > t_1, (f_i - f_j)(t_1) \leq (f_i - f_j)(t_2)$ or $\forall t_1, \forall t_2 > t_1, (f_i - f_j)(t_1) \geq (f_i - f_j)(t_2)$.*

We now consider $P_m|r_j, p_j = p|F_l(\sum C_j, \sum T_j)$ scheduling problem.

PROPOSITION 3.1 *$f_i - f_j$ is monotonous for any composite linear function of the form $\alpha * \sum C_j + (1 - \alpha) * \sum T_j$, where $0 \leq \alpha \leq 1$.*

Proof: In $f_i - f_j$, for any pair of jobs i and j at any time point, the $\alpha * \sum C_j$ part of the composite linear function is always 0. Therefore when the function is compared at any two time points t_1 and $t_2, t_1 \leq t_2$, only the $(1 - \alpha) * \sum T_j$ can be non-zero. Since $f_i - f_j$ is monotonous for $(1 - \alpha) * \sum T_j$, it is monotonous for $\alpha * \sum C_j + (1 - \alpha) * \sum T_j$ as well.

We continue with the $P_m|r_j, p_j = p|F_l(\sum w_j C_j, \sum T_j)$ scheduling problem. Here, the linear function has the form: $\alpha * \sum w_j C_j + (1 - \alpha) * \sum T_j$. In the analysis that follows, we attempt to characterize the values of α for which $f_i - f_j$ is monotonous.

Let $w_{min} = min\{w_i - w_j | J_i, J_j, w_i > w_j, d_i > d_j\}$. Then we have the following proposition.

PROPOSITION 3.2 *For any values of α that satisfy $\alpha \geq 1/(1 + w_{min})$, the composite linear functions $f_i - f_j$ involving weighted completion time and tardiness are monotonous.*

Proof: Note that for those pairs of jobs for which, $w_i \geq w_j$ and $d_i \leq d_j$, $f_i - f_j$ is monotonous. So we focus our attention on pairs of jobs for which $w_i > w_j$ and $d_i > d_j$. We consider six exhaustive cases based on the locations of the time points t_1 and t_2.

1. $t_1, t_2 \leq d_i$: $f_i - f_j$ is monotonous for any pair of jobs as tardiness of the jobs is zero and only the $\alpha * \sum w_j C_j$ part can be non-zero.

2. $t_1, t_2 > d_j$ and $t_1, t_2 \leq d_i$: Here for $f_i - f_j$ to be monotonous,

$$\alpha(w_i - w_j)t_1 - (1-\alpha)(t_1 - d_j) \leq \alpha(w_i - w_j)t_2 - (1-\alpha)(t_2 - d_j)$$

$$\alpha * (w_i - w_j)(t_2 - t_1) \geq (1-\alpha)(t_2 - t_1)$$

$$(1-\alpha)/\alpha \leq (w_i - w_j)$$

3. $t_1, t_2 > d_i$: Here we need,

$$\alpha(w_i - w_j)t_1 + (1-\alpha)(d_j - d_i) \leq \alpha(w_i - w_j)t_2 + (1-\alpha)(d_j - d_i)$$

This is automatically ensured as the $(1-\alpha)(d_j - d_i)$ is constant at both t_1 and t_2.

4. $t_1 \leq d_j$ and $d_j < t_2 \leq d_i$: We consider $t_1 = d_j$ as if we show that $(f_i - f_j)t_2 \geq (f_i - f_j)d_j$ then from case 1, $(f_i - f_j)t_2 \geq (f_i - f_j)t_1$ for all $t_1 < d_j$. We therefore need,

$$\alpha(w_i - w_j)d_j \leq \alpha(w_i - w_j)t_2 - (1-\alpha)(t_2 - d_j)$$

which leads us to:

$$(1-\alpha)/\alpha \leq (w_i - w_j)$$

5. $d_j < t_1 \leq d_i$ and $t_2 > d_i$: For $f_i - f_j$ to be monotonous,

$$\alpha(w_i - w_j)t_1 - (1-\alpha)(t_1 - d_j) \leq \alpha(w_i - w_j)t_2 + (1-\alpha)(d_j - d_i)$$

which leads us to:

$$(1-\alpha)(d_i - t_1) \leq \alpha(w_i - w_j)(t_2 - t_1)$$

Since $(t_2 - t_1)$ will tend to equal $(d_i - t_1)$ when t_2 is close to d_i we need:

$$(1 - \alpha)/\alpha \leq (w_i - w_j)$$

6. $t_1 \leq d_j$ and $t_2 > d_i$: We consider $t_1 = d_j$ for the same reason explained in case 4. For $f_i - f_j$ to be monotonous, we need to satisfy,

$$\alpha(w_i - w_j)d_j \leq \alpha(w_i - w_j)t_2 + (1 - \alpha)(d_j - d_i)$$

which leads us to:

$$(1 - \alpha)(d_i - d_j) \leq \alpha(w_i - w_j)(t_2 - d_j)$$

Since $(t_2 - d_j)$ will tend to equal $(d_i - d_j)$ when t_2 is close to d_i we end up once more with the equation that gives us values of α that ensure that $f_i - f_j$ is monotonous:

$$(1 - \alpha)/\alpha \leq (w_i - w_j)$$

The result means that even if the $w_{min} = 1$, DPB can be used to optimally solve the $\alpha * \sum w_j C_j + (1 - \alpha) * \sum T_j$ as long as α is 0.5 or higher. This conclusion is quite interesting and useful as it implies that the $\sum w_j C_j + \sum T_j$ function can be solved in polynomial time given the number of machines, while the closely related objective $\sum w_j T_j$ (which is not monotonous for all $f_i - f_j$ functions) is still an open problem.

4. Generating Extreme Schedules

Aneja and Nair (1979) generate the extreme points for a bicriteria transportation problem modeled as assignment problem. Though our optimization technique is a dynamic program it is still possible for us to use their algorithm to generate the extreme schedules on the efficient frontier for the bicriteria problem involving total completion time and total tardiness. The steps of the algorithm are given below. For notational convenience we use z_1 for $\sum C_j$ and z_2 for $\sum T_j$. $z_1^{(k)}$ and $z_2^{(k)}$ are used to denote the objective values of z_1 and z_2 and point k.

Step 0: Use DPB to find the optimal value $z_1^{(1)}$ for the z_1 objective. Record the value of $z_2^{(1)}$ for this solution. Set $k = 1$. Similarly use DPB to find the optimal value $z_2^{(2)}$ for the z_2 objective and record the value of $z_1^{(2)}$ for this solution. If $(z_1^{(1)}, z_2^{(1)}) = (z_1^{(2)}, z_2^{(2)})$, stop. Set $k = k + 1$. Define sets $L = (1, 2)$ and $E = \emptyset$ and go to step 1.

Step 1: Choose an element $(r, s) \in L$ and set $a_1^{r,s} = |z_2^s - z_2^r|$ and $a_2^{r,s} = |z_1^s - z_1^r|$. Use the DPB now to minimize: $a_1^{r,s} * z_1 + a_2^{r,s} * z_2$. Let z_1^* and z_2^* be the values of objectives z_1 and z_2 respectively. If (z_1^*, z_2^*) is equal either to $((z_1^{(r)}, z_2^{(r)})$ or $(z_1^{(s)}, z_2^{(s)})$, set $E = E \cup \{(r, s)\}$ and go Step 2. Else record, $((z_1^{(k)}, z_2^{(k)})$ such that $z_1^{(k)} = z_1^*$ and $z_2^{(k)} = z_2^*$. Set $k = k + 1$, $L = L \cup (r, k), (k, s)$ and go to step 2.

Step 2: Set $L = L - (r, s)$. If $L = \emptyset$, stop. Otherwise go to Step 1.

Insert figures 2 and 3 here.

The algorithm differs from the one given in Aneja and Nair (1979) in the following feature: the points 1 and 2 with which we start the algorithm may not always be extreme points representing $Lex(\sum C_j, \sum T_j)$ and $Lex(\sum T_j, \sum C_j)$. In other words, they may be weakly Pareto optimal solutions. However, it is easy to see that we will still end up with all the undiscovered extreme points including the lexicographic ones. This is because all the undiscovered efficient extreme will lie in halfspace beneath the line joining the two initial points (Figure 2), which represents the new objective function. The proof of correctness and finiteness will follow the same steps as the one described in Aneja and Nair (1979).

For the problem with $\sum w_j C_j$ and $\sum T_j$, however, the same algorithm can only be used to generate a subset of the extreme schedules. The only modification in the algorithm will involve changing the 2nd point in Step 0 of the algorithm to the one obtained by solving $\alpha * \sum w_j C_j + (1 - \alpha) \sum T_j$, where $\alpha = 1/(1 + w_{min})$. Thus the procedure will generate all the extreme points that lie in halfspace below the line joining the two initial points (Figure 3).

The DPB has a complexity of O(n^{3m+4}) (Baptiste 2000). If the number of extreme points is k, the complexity of this procedure is O(kn^{3m+4}).

5. Bicriteria Problems involving C_{max}

One of the main reasons for the polynomial solvability of release date problems with equal processing times is that the number of time points at which jobs can start or finish in a schedule is bounded by O(n^2). The following lemma from Baptiste (2000) expresses this more formally. If ζ denotes the schedule, among optimal ones then the following lemma characterizes the time points at which jobs start and end on the schedule ζ.

LEMMA 3 *The time points at which jobs start and end on the schedule ζ belong to $T = \{t : \exists r_i, \exists l \in \{0, ..., n\}, t = r_i + lp\}$*

This property is particularly useful in bicriteria optimization when one of the criteria is C_{max}. The optimal value of C_{max} can be obtained using the algorithm in Desousky et al. (1990). DPB is structured so that the optimal value function $F_n = ((min_{t\in T}t, ..., min_{t\in T}t), (max_{t\in T}t, ..., max_{t\in T}t)$ is calculated between a least possible time points in T on the machines and greatest possible time points in T. To achieve $Lex(C_{max}, \sum w_jC_j)$ and $Lex(C_{max}, \sum T_j)$, we merely need to rewrite the optimal value function as
$F_n = ((min_{t\in T}t, ..., min_{t\in T}t), (C_{max}, ..., C_{max}))$.

In order to generate the set of pareto optimal points, we set the optimal value functions to

$F_n = ((min_{t\in T}t, ..., min_{t\in T}t), (t^{'}, ..., t^{'})), \forall t^{'}$ such that $C_{max} \leq t^{'} \leq max_{t\in T}t$.

Since the upperbound on the number of values $t^{'}$ can take is O(n^2), the overall complexity of the algorithm is O(n^2n^{3m+4}) = O(n^{3m+6}).

6. Conclusions and Future Research

In this paper, we consider bicriteria scheduling problems in an identical parallel machine environment where jobs have equal processing times and unequal release dates. We use the framework provided by DPB of Baptiste (2000) and the algorithm of Aneja and Nair (1979) to generate the efficient frontier for $\sum C_j$ and $\sum T_j$, and a subset of the efficient schedules for $\sum w_jC_j$ and $\sum T_j$. The efficient schedules are generated in running time that is polynomial given the number of machines.

We also use DPB to generate the pareto optimal set for bicriteria problems involving C_{max} and either one of $\sum w_jC_j$ or $\sum T_j$. This algorithm is also polynomial given the number of machines.

Though we happen to discover the lexicographic points while generating the efficient frontier for $\sum C_j$ and $\sum T_j$, we are working to propose computationally more efficient algorithms, especially in cases where $\sum C_j$ is a primary measure.

Our research is also focussed on bicriteria scheduling problems in the same environment involving other criteria such as L_{max} and $\sum U_j$ and $\sum w_jU_j$ in terms of lexicographic optimization, generation of the efficient frontier and the set of pareto optimal schedules.

References

Aneja Y.P. and Nair K.P.K. (1979), Bicriteria Transportation Problem, *Management Science*, 25, 73-78.

Angel E., Bampis E. and Kononov A. (2003), On the approximate tradeoff for bicriteria batching and parallel machine scheduling problems, *Theoretical Computer Science* 306, 319-338.

Baptiste P. (2000), Scheduling Equal Length jobs on identical parallel machines, *Discrete Applied Mathematics*, 103, 21-32.

Baptiste P. , Brucker, P., Knust,S. Timkovsky, V.G. (2004), Ten notes on equal-processing-time scheduling; At the frontiers of solvability in polynomial time,*Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, 2, 111-127.

Dessousky M.I., Lageweg B.J., Lenstra J.K., van de Velde S.L. (1989), Scheduling identical jobs on uniform parallel machines, *Statist. Neerlandica*, 44, 115-123.

Gupta J.N.D. and Ruiz-Torrez A.J. , Generating efficient schedules for identical parallel machines involving flow time and tardy jobs, *European Journal of Operational Research*, Article in Press.

Gupta J.N.D. , Ruiz-Torrez A.J. and Webster S.(2003) ,Minimizing maximum tardiness and number of tardy jobs on parallel machines subject to minimum flow time, *Journal of Operational Research Society*, 54, 1263-1274.

Hoogeveen H. (2004) , Multicriteria Scheduling, *European Journal of Operational Research*, Article in Press.

Lin C-H. and Liao C-J. (2004), Makespan minimization subject to flowtime optimality on identical parallel machines, *Computers and Operations Research*, 31, 1655-1666.

Sarin S. and Prakash, D. (2004), Equal processing time Bicriteria Scheduling on Parallel Machines, *Journal of Combinatorial Optimization*, 8, 227-240.

T'kindt V., Billaut J-C. and Proust C. (2001), Solving a bicriteria scheduling problem on unrelated parallel machines occurring in the glass bottle industry, *European Journal of Operational Research*, 135, 42-49.

T'kindt V. and Billaut J-C. (2002), Multicriteria Scheduling, Springer, 2002.

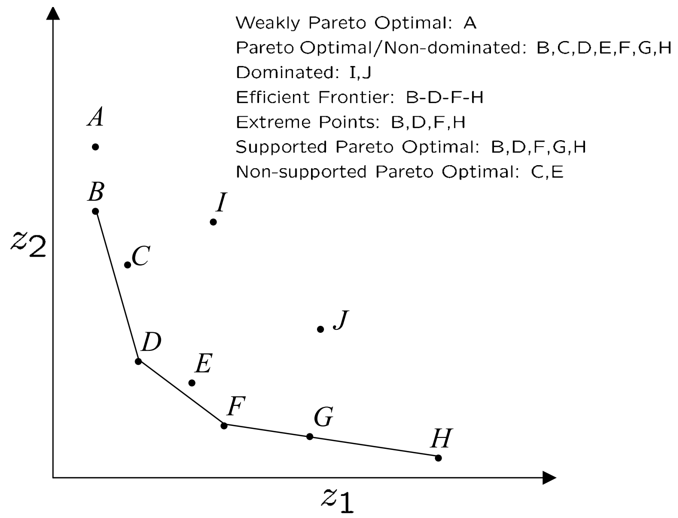

Figure 1. Solutions in criteria space

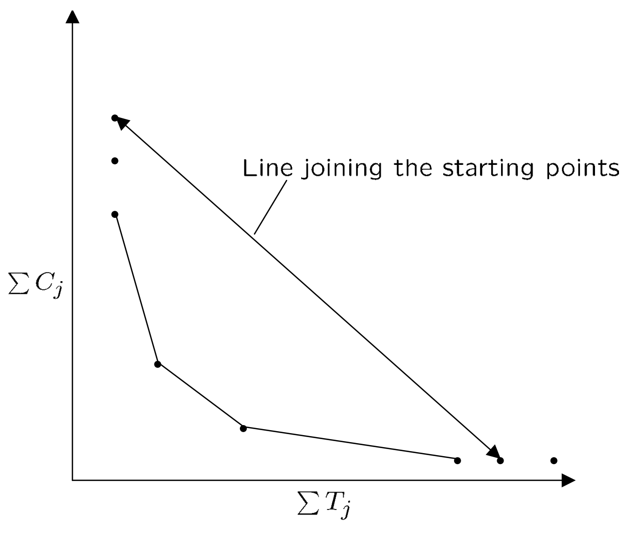

Figure 2. Generating extreme schedules for $\sum C_j$ and $\sum T_j$

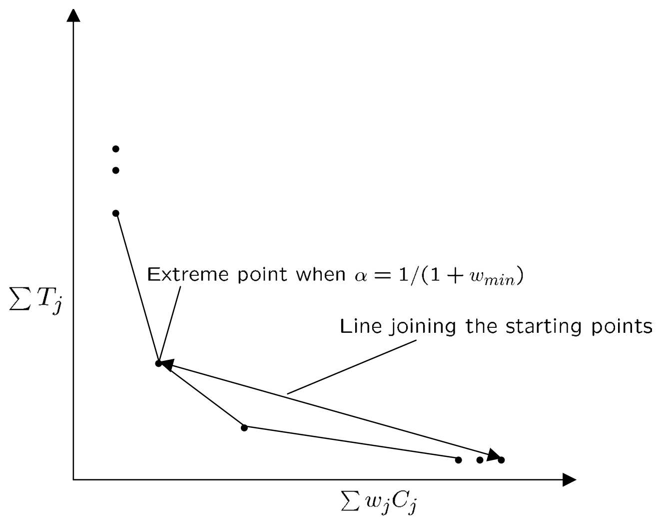

Figure 3. Generating a subset of extreme schedules for $\sum w_j C_j$ and $\sum T_j$