CAHIER DU LAMSADE

Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision
(Université Paris-Dauphine)
Unité de Recherche Associée au CNRS UMR 7024

A-

ON THE INTEGER BI-CRITERIA NETWORK FLOW PROBLEM: A BRANCH-AND-BOUND APPROACH

CAHIER N° 191
février 2002

José FIGUEIRA [1]

[1] School of Economics, University of Coimbra, Av. Dias da Silva, 165, 3004-512 Coimbra, Portugal (figueira@fe.uc.pt) and LAMSADE, Université Paris-Dauphine, Place du Maréchal De Lattre de Tassigny, 75775 Paris Cedex 16, France.

Contents

RÉSUMÉ	II
ABSTRACT	II
1. INTRODUCTION	1
2. STATEMENT OF THE PROBLEM: NOTATIONS AND DEFINITIONS	1
3. NETWORK SIMPLEX METHOD	4
4. A BRANCH-AND-BOUND APPROACH	5
5. OUTLINE OF THE METHOD	6
6. AN ILLUSTRATIVE EXAMPLE	8
7. CONCLUSIONS	12
REFERENCES	12
APPENDIX A: SET OF ALL FEASIBLE SOLUTIONS	14
APPENDIX B: CRITERIA SPACE FOR SUBPROBLEM B	16
APPENDIX C: CRITERIA SPACE FOR SUBPROBLEM C	17
APPENDIX D: CRITERIA SPACE FOR SUBPROBLEM D	18
APPENDIX E: CRITERIA SPACE FOR SUBPROBLEM E	19
APPENDIX F: CRITERIA SPACE FOR SUBPROBLEM F	20
APPENDIX G: CRITERIA SPACE FOR SUBPROBLEM G	21
APPENDIX H: CRITERIA SPACE FOR SUBPROBLEM H	22
APPENDIX I: CRITERIA SPACE FOR SUBPROBLEM I	23
APPENDIX J: CRITERIA SPACE FOR SUBPROBLEM J	24
APPENDIX K: CRITERIA SPACE FOR SUBPROBLEM K	25
APPENDIX L: ADJACENT STSs	26
APPENDIX M: PRIMAL AND DUAL SOLUTIONS	27

DÉTERMINATION DE L'ENSEMBLE DES SOLUTIONS NON-DOMINÉES DANS LE CADRE DES PROBLÈMES DE FLOT BI-CRITÈRES : UNE APPROCHE BASÉE SUR LES TECHNIQUES DE PARTITION ET ÉVALUATION SUCESSIVES

RÉSUMÉ

Cet article présente une nouvelle méthode pour identifier tous les vecteurs non-dominés concernant le problème de flot à "coût minimum" bi-critère dans le cas entier. L'approche combine trois techniques différentes : la méthode e-contraint, la méthode du simplexe en théorie des graphes et la méthode de la partition et évaluation sucessives. L'avantage principal de cette approche consite à identifier toutes les solutions efficaces sans détruire la structure de réseau inhérente aux problèmes de flot. L'article présente aussi une méthode fondée sur la programmation paramétrique ayant pour objectif la déterminaison de tous les vecteurs non-dominés supportés. Parfois, ces vecteurs nous donnent une approximation de l'ensemble complet de tous les vecteurs non-dominés. Cette information peut être utilisée dans le contexte des procédures interactives où le décideur définit une zone d'intérêt pour effectuer une recherche plus détaillée.

Mots-clés : Critères multiples, Analyse combinatoire, Graphes et réseaux, Programmation en nombres entiers.

ON THE INTEGER BI-CRITERIA NETWORK FLOW PROBLEM: A BRANCH-AND-BOUND APPROACH

ABSTRACT

This paper presents a new method for identifying all the non-dominated vectors for integer bi-criteria "minimum cost" network flow problems. The method combines a network simplex algorithm, the ε-constraint method and a branch-and-bound algorithm. The set of all non-dominated vectors in criteria space (or efficient solutions in decision variable space) is determined by solving an ε-constraint problem with branch-and-bound techniques. The main advantage of the proposed method concerns the identification of non-integer solutions exploiting only network structures. We suspect that it can be many times faster than general LP-algorithms. The paper also presents a method for finding the set of non-dominated supported vectors based on parametric programming techniques. Sometimes supported non-dominated vectors represent a rough approximation of the entire set of non-dominated vectors and allow us to define zones of potential interest for decision makers where a more detailed search must be done.

Key words: Multiple Criteria, Combinatorial Analysis, Networks and Graphs, Integer Programming.

1. INTRODUCTION

Network flow problems arise in all walks of life, whether in planning highway systems, or in planning daily production electricity, or in planning manufacturing and distribution systems. The field of network flows has a rich tradition in many other domains. In all of the areas of application of network flow methods, it is frequent to wish to "optimize" not only one but several criteria simultaneously. In general the criteria are mutually conflicting. The concept of optimal solution is thus replaced by the concept of efficient solution. In such solutions it is not possible to improve the evaluations of one criterion without incur in a degradation of the evaluations of at least one of the remaining criteria.

Multiple criteria analysis deals with the help of decision-makers to make "good" decisions in the presence of multiple but usually conflicting criteria. Multiple criteria combinatorial optimization is a field of multiple criteria analysis. It has not been very intensively investigated in spite of its importance from the practical standpoint. In the field of bi-criteria "minimum cost" network flows (both linear and integer cases) we can mention the following works:

- *Fruhwirth et al.* (1989), proposed two different rules (the angle and the slope bisections) in a method designed to approximate the set of all non-dominated vectors.
- *Malhotra* and *Puri* (1984), and *Lee* and *Pulat* (1993), generalize the out-of-kilter method.
- *Lee* and *Pulat* (1991), *Pulat et al.* (1992), and *Calvete* and *Mateo* (1996), generalize network simplex algorithm.

Multiple criteria "minimum cost" network flow problems are known to be *hard* to solve. *Ruhe* (1988) proves, for a particular instance with only two criteria, that the number of non-dominated extreme points (in criteria space) grows exponentially with the number of vertices of the network.

Our research is intended to reflect the concern with the design of new algorithms for multiple criteria combinatorial problems. So, this paper provides a new method for identifying all the non-dominated vectors for integer bi-criteria "minimum cost" network flow problems. The method combines a network simplex algorithm, the ε-constraint method, and a branch-and-bound algorithm. The set of all non-dominated vectors in criteria space (or efficient solutions in decision variable space) is determined by solving an ε-constraint problem with branch-and-bound techniques. The paper also presents a method for finding the set of non-dominated supported vectors based on parametric programming techniques.

The contents of the paper are as follows. In Section 2, the necessary mathematical background is reviewed and the ε-constraint method is presented. Section 3 contains a short presentation of the network simplex method. A branch-and-bound algorithm is given in Section 4. In Section 5 the proposed method is presented and is illustrated in Section 6. Finally, in Section 7, some concluding remarks are outlined.

2. STATEMENT OF THE PROBLEM: NOTATIONS AND DEFINITIONS

To better understand the proposed approach, definitions and notations regarding network flows optimization, graph theory and linear programming must be introduced in this section (for more details about network optimization, see also *Ahuja et al.*, 1993).

Let $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ be a *directed* and *connected graph*, where $\mathcal{S}$ is a finite set of *nodes* or *vertices* with cardinality $|\mathcal{S}| = n$, and $\mathcal{A}$ is a collection of ordered pairs of elements of $\mathcal{S}$ called *arcs*, with cardinality $|\mathcal{A}| = m$.

A graph $\mathcal{G}' = (\mathcal{S}', \mathcal{A}')$ is called a *subgraph* of $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ if $\mathcal{S}' \subseteq \mathcal{S}$ and $\mathcal{A}' \subset \mathcal{A}$. It is a spanning subgraph of $\mathcal{G}$ if $\mathcal{S}' = \mathcal{S}$. A *path* $\mathcal{P}$ is a sequence of vertices and arcs, $i_1 - a_1 - i_2 - a_2 - \ldots - i_{s-1} - a_{s-1} - i_s$, without repetition of vertices and where $1 \leq k \leq s-1$ for which either $a_k = (i_k, i_{k+1}) \in \mathcal{A}$, or $a_k = (i_{k+1}, i_k) \in \mathcal{A}$. A *directed path* is a path without backwards arcs. A *cycle* C is a closed path where the only repeated vertex is the starting and the end point that coincide. A *directed cycle* is a closed directed path. When in a given

graph $\mathcal{G}$ there is always a path linking any two different vertices of $\mathcal{G}$, the graph is called *connected*. A *tree* $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ is a subgraph without cycles where $\mathcal{V} \subseteq \mathcal{S}$ and $\mathcal{E} \subset \mathcal{A}$. A tree $\mathcal{T}$ is called a *spanning tree* when it spans the set of vertices $\mathcal{S}$ of $\mathcal{G}$, that is $\mathcal{V} = \mathcal{S}$. A spanning tree is denoted by $\mathcal{T} = (\mathcal{S}, \mathcal{E})$. Consider (k,l) a given arc belonging to the set $\mathcal{A}$ but not in $\mathcal{E}$. Then, there is a unique cycle C when the arc (k,l) is added to $\mathcal{E}$. The orientation of C is the same as (k,l). In a cycle C a partition of its vertices can be made by separating the arcs having the same orientation as C from the arcs in the opposite direction. The collection of all possible cycles of this type is called *fundamental cycle basis*. All these definitions are essential for a better understanding of the network simplex method, presented in Section 3.

A directed graph with numerical values assigned to its vertices and/or arcs is called *network*. Let $\mathcal{G} = (\mathcal{S}, \mathcal{A})$ be a network with two *"costs"* $c_1(i,j)$ and $c_2(i,j)$, a *lower bound* $l(i,j)$ and an *upper bound* or *capacity* $u(i,j)$ associated with every arc $(i,j) \in \mathcal{A}$. The numerical values $l(i,j)$ and $u(i,j)$ respectively denote the minimum and the maximum amount that must flow on the arc (i,j). Finally, let $x(i,j)$ be the *amount of flow* on the arc (i,j). A numerical value $b(i)$ is also associated with each vertex $i \in \mathcal{S}$ denoting its *supply* (if $b(i) > 0$) or its *demand* (if $b(i) < 0$). A vertex with $b(i) = 0$ is called a *transshipment vertex*. The bi-criteria "minimum cost" network flow problem can be stated as follows:

$$\begin{aligned} &\min f_1(x) = \sum_{\substack{(i,j)\in\mathcal{A} \\ i\neq j}} c_1(i,j)x(i,j) \\ &\min f_2(x) = \sum_{\substack{(i,j)\in\mathcal{A} \\ i\neq j}} c_2(i,j)x(i,j) \end{aligned}$$

$$\text{subject to :} \qquad \sum_{\substack{\{j \mid (i,j)\in\mathcal{A}\} \\ i\neq j}} x(i,j) - \sum_{\substack{\{j \mid (j,i)\in\mathcal{A}\} \\ i\neq j}} x(j,i) = b(i), \quad \forall i \in \mathcal{S} \tag{1}$$

$$l(i,j) \leq x(i,j) \leq u(i,j), \quad \forall (i,j) \in \mathcal{A}$$

In what follows, the assumptions below must be taken into account:

1. The graph is directed and connected.
2. All the numerical values for the costs, lower and upper bounds on the arcs and supplies/demands on the vertices are integral and finite.
3. The condition $\sum_{i\in\mathcal{S}} b(i) = 0$ must be fulfilled.
4. The integer bi-criteria "minimum cost" network flow problem has at least two feasible solutions and the minimum values for the individual objective functions are different.

Problem (1) can be presented in a more dense form as follows:

$$\begin{aligned} &\text{"min"} \qquad F(x) = \big(f_1(x), f_2(x)\big) \\ &\text{subject to :} \\ &\qquad\qquad x \in X \leftarrow \big\{x \in R^m | Ax = b, l \leq x \leq u\big\}, \end{aligned} \tag{2}$$

where: $x \in R^m$, $x = (x_1, \ldots, x_m)^T$, is the vector of decision variables; X is the set of feasible solutions of (1); A is the $n \times m$ *node-arc incidence matrix*; l is the vector of lower bounds; u is the vector of upper bounds; b is the vector of supplies/demands on vertices; $F(x) = \big(f_1(x), f_2(x)\big)^T$ is the vector of objectives to be "minimized"; $Y = F(X)$ is the set of all feasible vectors y in R^2, where $y = (y_1, y_2)^T$ with $y_q = f_q(x)$ for $q = 1,2$.

Dominance is a key concept in multiple criteria decision analysis. Let us define this concept for the general multiple criteria case with r criteria.

Definition of dominance. *Consider y^1 and y^2 two feasible solutions. Then, y^1 dominates y^2 iff $y^1 \leq y^2$ and $y^1 \neq y^2$, that is, ${y_q}^1 \leq {y_q}^2$ for all $q = 1,\ldots,r$ with at least one strict inequality.*

Definition of non-dominated vector. *A vector $y^{nd} \in Y$ is called non-dominated iff there does not exist another vector $y \in Y$ such that $y \leq y^{nd}$ and $y \neq y^{nd}$.*

A distinction between efficient solutions in decision variable space and non-dominated vectors in criteria space can be made. Efficient solutions are crucial for the usefulness of multiple criteria methods. This concept was first introduced by *Pareto* (1896). Thus, these solutions are called *Pareto optimal*, and also *non-inferior* or *functional efficient* solutions.

Definition of efficient solution. *A solution $x^e \in X$ is said to be efficient iff it is impossible to find another solution $x \in X$ with a better evaluation of a given criterion without deteriorating the evaluations of at least another criterion.*

In multiple criteria linear integer programming, two types of non-dominated vectors can be distinguished:

1. The *non-dominated supported vectors*, that is, those vectors of Y that can be obtained by solving the following parametric mathematical programming problem:

$$\min_{y \in Y} \sum_{q=1}^{r} \lambda_q y_q \quad \text{for} \quad \sum_{q=1}^{r} \lambda_q = 1, \quad \lambda_q > 0 \quad \text{for} \quad q = 1,\ldots,r.$$

2. The *non-dominated unsupported vectors*, that is, the non-dominated vectors of Y belonging to the *interior* of the *convex hull* of Y, $Conv(Y)$.

Let:

- Y^{nd} be the set of all the non-dominated vectors of Y;
- Y^{nds} be the set of all the supported non-dominated vectors of Y;
- Y^{ndu} be the set of all the unsupported non-dominated vectors of Y, that is, $Y^{ndu} = Y^{nd} - Y^{nds}$;
- X^e be the set of all the efficient solutions of X;
- X^{es} be the set of all the efficient solutions related to the set of all the supported non-dominated vectors, Y^{nds};
- X^{eu} be the set of all efficient solutions related to the set of all the unsupported non-dominated vectors, that is, $X^{eu} = X^e - X^{es}$.

Among the supported non-dominated vectors we may also distinguish the set of non-dominated supported *extreme* points of $Conv(Y)$, denoted by Y^{ndse}, and the set of non-dominated supported *intermediate* vectors that can be obtained by a linear convex combination of non-dominated supported extreme points, denoted by Y^{ndsi}. The same kind of distinction between the set of supported extreme efficient solutions X^{ese} and the set of supported intermediate efficient solutions X^{esi} can be made in the decision variable space.

In multiple criteria linear programming several techniques (scalar optimization problems) can be used in order to characterize efficient solutions (non-dominated vectors) like, for example, weighted-sum approaches, Tchebycheff metrics based methods, ε-constraint methods, and so on (see *Steuer*, 1985). Among the existing methods, the ε-constraint approach can be easily used in multiple criteria integer problems without any additional restrictions. Efficient solutions can be characterized as optimal solutions for the ε-constraint problem.

The ε-constraint problem associated with the bi-criteria (1) can be stated as follows:

$$\begin{array}{ll} \min & f_1(x) \\ \text{subject to :} & \\ & x \in X \\ & f_2(x) \leq \varepsilon, \end{array} \tag{3}$$

where ε is a scalar.

In the ε-constraint method, ε varies among all the values for which (3) remains feasible. So, in order to identify a set of efficient solutions, a sequence of problems (3) is solved for each different value of ε (*Chankong* and *Haimes*, 1983). For integer bi-criteria linear programming problems the entire non-dominated set Y^{nd} can be easily determined by solving a sequence of problems (3).

Theorem of equivalence [*Haimes et al.*, 1971]. *Consider* $\varepsilon \geq \min f_2(x)$. *If the solution* x^* *solves problem (3) and when* x^* *is not unique it leads to a minimal value for criterion* $f_2(x)$, *then* x^* *solves (1), that is,* x^* *is an efficient solution for (1).*

Proof.
Suppose now that x^* does not solve the problem. Another solution $\hat{x}$ can then be considered so that only one of the following two cases can occur:

- $f_1(\hat{x}) < f_1(x^*)$ and $f_2(\hat{x}) \leq f_2(x^*)$ which contradicts the fact that x^* solves (3), or
- $f_2(\hat{x}) < f_2(x^*)$ and $f_1(\hat{x}) \leq f_1(x^*)$, which contradicts the hypothesis that x^* is optimal for (3) with the smallest value for $f_2(x)$.

The theorem is proved by the two cases above.

Problem (3) will be used in the algorithm outlined in Section 5 to determine all the non-dominated vectors for problem (1).

3. NETWORK SIMPLEX ALGORITHM

Let us now succinctly recall the network simplex method on minimum cost network flow problems (see Figure 1). The basic idea for any variant of the network simplex method is a Spanning Tree Structure (STS), $(\mathcal{T}, L, U)$. Such a structure (or solution) is obtained when, for any arc not belonging to this tree, the flow value is fixed at its lower bound level or at its upper bound level. All the arcs fixed at their lower bound level belong to the set L, while all the arcs fixed at their upper bound level belong to the set U. The remaining arcs are those belonging to the spanning tree $\mathcal{T}$. A minimum cost network flow problem has always at least one STS optimal solution (see *Ahuja et al.*,, 1993). It is possible to find an optimal STS by shifting from one STS to another, successively. At each iteration, we exchange a pair of arcs (one arc entering STS and one arc coming out of STS). Any STS corresponds to one feasible basic solution in linear programming, and each shift from one STS to another coincides with one pivoting operation in the standard simplex method. The initialization of the algorithm consists of finding one feasible STS (or equivalently, a feasible basic solution in the standard simplex method). Two vectors are associated with this STS, the flow x (primal solution) and the potential π (dual solution). Each iteration of the method consists of: (1) identifying one eligible arc (k,l) with $(k,l) \notin \mathcal{T}$; (2) adding the arc (k,l) to $\mathcal{T}$ and finding an arc (p,q) coming out of $\mathcal{T}$; and, updating STS and the primal and dual solution (x,π).

An arc not belonging to $\mathcal{T}$ is said to be eligible if:

i) Its reduced cost is strictly negative and its flow is at its lower bound, that is, $\bar{c}(i,j) < 0$ and $(i,j) \in L$.

ii) Its reduced cost is strictly positive and its flow is at its upper bound, that is, $\bar{c}(i,j) > 0$ and $(i,j) \in U$.

The reduced cost of a given arc (i,j) is defined as follows:

$$\bar{c}(i,j) = c(i,j) - \pi(i) + \pi(j),$$

where, $\pi(i)$ and $\pi(j)$ are the dual variables associated with the vertices i and j, respectively. It should be noted that for all the arcs $(i,j) \in \mathcal{T}$ the reduced cost $\bar{c}(i,j) = 0$.

Simplex method.
{ Computing a minimum cost flow. }
(1) **begin**
(2) let $(\mathcal{T}, \mathcal{L}, \mathcal{U})$ be a starting feasible STS;
(3) let x be the flow and π the dual variable associated with $(\mathcal{T}, \mathcal{L}, \mathcal{U})$;
(4) **while** (not optimal solution) **do**
(5) **begin**
(6) select an entering arc (k,l) not in $\mathcal{T}$;
(7) add (k,l) to $\mathcal{T}$ and remove (p,q) from $\mathcal{T}$;
(8) update the STS and the solutions x and π;
(9) **end**
(10)**end**

Figure 1: *Network simplex algorithm.*

At each iteration, the network simplex method shown in Figure 1 always gives an integer solution for the minimum cost network flow problem. But it is possible to obtain non-integer solutions between two adjacent STSs. Let us recall that when moving from one STS to an adjacent one, an amount of flow, $\triangle$, must be sent along the orientation of cycle C. This quantity $\triangle$ is integer. But, what happens if a non-integer amount o flow is sent along C? It is obvious that a non-integer solution will be obtained. This solution has exactly $|C|$ non-integer variables, but it does not define a spanning tree structure. This idea is very important if we wish to obtain non-integer solutions for the LP-relaxation of problem (3).

4. A BRANCH-AND-BOUND APPROACH

This section deals with branch-and-bound techniques. The problem to be solved is problem (3) presented in Section 2. The branch-and-bound algorithm can be decomposed into three main steps: *branching*, *bounding* and *fathoming* (*Hillier* and *Lieberman*, 1995). Branching deals with the partition of a given subproblem into several (here we consider only the partition into two subproblems). Bounding consists of computing the optimal value of LP-relaxations. Finally, fathoming is concerned with a set of tests meant to discarding subproblems from further analysis. In what follows it will be supposed that problem (3) has at least one integer feasible solution.

Let W be a list of subproblems to be analyzed or of subproblems that remain unfathomed. The three main steps of the algorithm can be described as follows:

1. *Branching:* Select the last subproblem in W, identify the non-integer variable $x(k,l)$. Let us remark that variable $x(k,l)$ is associated with the incoming arc (k,l) that forms a cycle C allowing to move from one STS to an adjacent one. This variable is called *branching variable*. Let $x^*(k,l)$ be the optimal value of $x(k,l)$ for the LP-relaxation of problem (3). Create two subproblems by changing the bounds on the arc (k,l) as follows:

   - build P^1 such that $\lfloor x^*(k,l) \rfloor + 1 \leq x(k,l) \leq u(k,l)$, and
   - build P^2 such that $l(k,l) \leq x(k,l) \leq \lfloor x^*(k,l) \rfloor$,

   where $\lfloor x^*(k,l) \rfloor$ means the nearest integer value lower than $x^*(k,l)$. It is easy to see that when forming subproblems P^1 and P^2 in this way, one of the STS will remain feasible for P^1, while the adjacent one will be feasible for P^2.

2. *Bounding:* Compute the costs for the LP-relaxation of both subproblems, P^1 and P^2, by applying the network simplex algorithm of Figure 1. As it was explained before, this algorithm can be easily adapted to compute non-integer solutions.

3. *Fathoming:* Three tests are introduced in order to discard subproblems from further study. Let us consider the *incumbent problem* as the problem having an integer solution with the best value for the objective function to the current iteration. The tests to be implemented are the following:

   a) A subproblem P^i is discarded when it has no feasible solution, or

   b) when its cost is greater than the cost of the incumbent problem.

   c) If P^i has an integer solution with a cost lower than the cost of the incumbent problem, then remove from W all the subproblems with a cost greater than the cost of P^i. This means that P^i is now the incumbent problem.

Figure 2 shows the branch-and-bound algorithm described above.

```
Branch-and-bound.
{ Computing an integer optimal solution for problem (3). }
(1) begin
(2)     CurrentIncumbent ← anything;
(3)     CurrentCost ← ∞;
(4)     let PR be the LP-relaxation of the initial integer problem;
(5)     W ← {PR};
(6)     while (W ≠ {}) do
(7)     begin
(8)         let P be the last element of W and remove it from W;
(9)         create two new subproblems P^1 and P^2 { Branching step };
(10)        for (i = 1) to (i = 2) do
(11)        begin
(12)            compute the cost for P^i using the simplex algorithm { Bounding step };
(13)            if (P^i has no feasible solution) then { Fathoming steps }
(14)                discard P^i
(15)            else
(16)                if (cost of P^i ≥ CurrentCost) then
(17)                    discard P^i;
(18)                else
(19)                    if (P^i has an integer solution) then
(20)                    begin
(21)                        CurrentIncumbent ← P^i;
(22)                        CurrentCost ← cost of P^i;
(23)                        remove from W all the problems with a cost greater than CurrentCost;
(24)                    end
(25)                    else
(26)                        W ← W ∪ {P^i};
(27)        end
(28)    end
(29) end
```

Figure 2: *A branch-and-bound algorithm.*

5. OUTLINE OF THE METHOD

This section outlines an approach for the search of all the non-dominated vectors, Y^{nd}. The method solves a sequence of problems (3) and uses the branch-and-bound algorithm shown in Figure 2 to determine its integer optimal solutions. We also describe how to obtain sets Y^{nde} and Y^{ndi} using parametric programming

techniques. Sometimes supported non-dominated vectors represent a rough approximation of the entire set of non-dominated vectors and allow us to define zones of potential interest for decision makers where a more detailed search must be done.

Let us starting by presenting how to determine sets Y^{nde} and Y^{ndi}, and then we shall present the general algorithm that can be applied to obtain the entire set Y^{nd}:

1. Identify the set of all supported extreme non-dominated vectors, Y^{ndse}. The algorithm is based on parametric programming techniques (see *Steuer*, 1985) and it proceeds as follows:

   a) Compute the optimal solution for the first criterion, f_1^*, and the corresponding minimal value for the second criterion, $\hat{f}_2$.

   b) Compute the optimal solution for the second criterion, f_2^*, and the corresponding minimal value for the first criterion, $\hat{f}_1$.

   c) Compute all the remaining non-dominated extreme points by decreasing order of the costs for the second criterion and by increasing order of the costs for the first criterion until the solution found in 1*b*) is reached, as follows:

   - Consider a given STS corresponding to a non-dominated extreme point (suppose that no degeneracy phenomenon occurs) and compute the dual variables associated with the first and the second criteria, that is, $\pi_1(i)$ and $\pi_2(i)$ for all $i \in S$.
   - Determine both reduced costs $\bar{c}_1(i,j)$ and $\bar{c}_2(i,j)$, for all arcs (i,j) in L and U.
   - For the arcs in U, identify $\bar{c}_2(i,j) > 0$ and $\bar{c}_1(i,j) < 0$. Then, compute the ratio between these two values, $r(i,j) = \bar{c}_2(i,j)/\bar{c}_1(i,j)$.
   - For the arcs in L, identify $\bar{c}_2(i,j) < 0$ and $\bar{c}_1(i,j) > 0$. Then, compute the ratio between these two values, $r(i,j) = \bar{c}_2(i,j)/\bar{c}_1(i,j)$.
   - Finally, choose the lowest (negative) ratio, identify the corresponding arc and proceed to a "pivoting operation" in order to obtain the adjacent STS. The ratios give the slopes of the lines connecting two adjacent STSs and the lowest one leads to an adjacent extreme point of $Conv(Y)$.

2. Determine all the supported intermediate non-dominated vectors, Y^{ndsi}. A very simple technique can be used, selecting two adjacent STSs related to two adjacent non-dominated extreme vectors, and identifying the cycle allowing to move from the first STS to the second one, and then increasing the amount of flow along the cycle unit by unit. In this way all the intermediate solutions are obtained. These solutions are not STSs.

3. Find the entire set of all non-dominated vectors, Y^{nd}. In this case, a more sophisticated technique is needed. The approach is based on the ε-constraint problem (3) and on branch-and-bound algorithm presented in Figure 2. Non-dominated vectors are determined by decreasing order of the values for the second objective function. The potential zones for the search of non-dominated vectors are identified by a set of triangles built from the adjacent non-dominated extreme points of the convex hull of the feasible region in the criteria space. This procedure can be described as follows:

   a) Identify two extreme supported non-dominated vectors and the associated STSs as in 1*c*) and build the region (triangle) of potential non-dominated vectors.

   b) For each triangle, search all the non-dominated vectors using a sequence of problems (3).

   c) Proceed in the same way until no more triangles exist.

The method given in 3*a*) is also based on parametric techniques in order to identify the adjacent movements from one STS to another. The main drawback of this procedure is related to the fact that several STSs may correspond to the same extreme vector in criteria space. Some additional difficulties can appear due to degeneracy phenomena that unfortunately tend to occur very often in problems of this type (*Bradley et al.*, 1977). Degeneracy phenomena give rise to cycling and stalling (see *Bazaraa et al.*, 1990).

The main advantage of the algorithm is related to the way in which non-integer solutions are determined, exploiting only network structures and thus avoiding the need to solve these problems with LP-codes. Despite the drawbacks pointed out above (alternate solutions and degeneracy), network simplex codes are still very fast in practice and solutions can be computed in a few seconds.

The example presented in Section 6 shows, step by step, how the method works.

6. AN ILLUSTRATIVE EXAMPLE

Consider the following example of a bi-criteria "minimum cost" network flow problem:

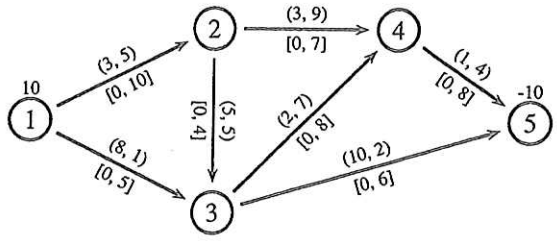


Figure 3: A bi-criteria example.

This example has 93 feasible solutions that are presented in Tables 2 and 3 in Appendix A. Among the 93 solutions only 10 are efficient. Figure 5 presents all the vectors in criteria space.

First, we shall show how the set Y^{nde} can be determined by parametric programming. In the example, $Y^{nde} = \{y^{48}, y^4, y^1, y^5\}$ as can be seen in Figure 5. These vectors are obtained by decreasing order of the cost for the second criterion and by increasing order of the cost for the first criterion. In order to obtain all the vectors, we first identify y^{48}, then y^4 and so on. Let us show how to determine the first two vectors of Y^{nde}, y^{48} and y^4:

- First, $f_1(x)$ is minimized. Its optimal value can be obtained at two different points y^{48} and y^{90}, where $f_1^* = 96$, but only y^{48} gives the minimal value for $f_2(x)$, $\hat{f}_2 = 144$. Vector y^{90} is thus discarded and y^{48} is saved. The STS corresponding to the vector y^{48} is presented in Figure 4. The arcs represented by the lines in bold are those belonging to the spanning tree, $\mathcal{T}$.

- Second, from y^{48} an adjacent STS leading to the next extreme non-dominated point of $Conv(Y)$ must be identified. STSs corresponding to the vectors y^4, y^{47} and y^{90} can be reached from y^{48} by identifying the fundamental cycle basis (see Figure 23 on Appendix L), but only vector y^4 is interesting. How can this vector be obtained? As we can see in Figure 5, the slope of the line connecting y^{48} and y^4 is the lowest. In order to identify this slope we may use the information given by the reduced costs on the arcs not belonging to $\mathcal{T}$. Let us recall that the slope of the line connecting y^{48} to y^{47} is $m_1 = \frac{132-144}{104-96} = \frac{-12}{8} = -1.5$, while the slope of the line connecting y^{48} to y^4 is $m_2 = \frac{135-144}{103-96} = \frac{-9}{7} = -1.286$. Both slopes can be obtained by the ratios $r(2,4) = 6/-4 = -1.5$ and $r(4,5) = 9/-7 = -1.286$, respectively. The lowest is the ratio $r(2,4)$. So, the arc $(2,4)$ forms a cycle allowing to move from y^{48} to y^4. Figure 4 presents the complete information concerning point y^{48} while Appendix M gives the remaining STSs until the last extreme non-dominated point of $Conv(Y)$, y^5, is reached.

Second, we must show how to determine all the intermediate points, Y^{ndi}. In our example $Y^{ndi} = \{y^{47}, y^3, y^2\}$. Vector y^{27} is obtained by sending one unit along the cycle $C = \{(1,2),(2,4),(3,4),(1,3)\}$ which allows to move from y^4 to y^{48} (see Figure 6). Vectors y^3 and y^2 are identified in the same way using the STSs associated with the extreme vectors of $Conv(Y)$, y^1 and y^4.

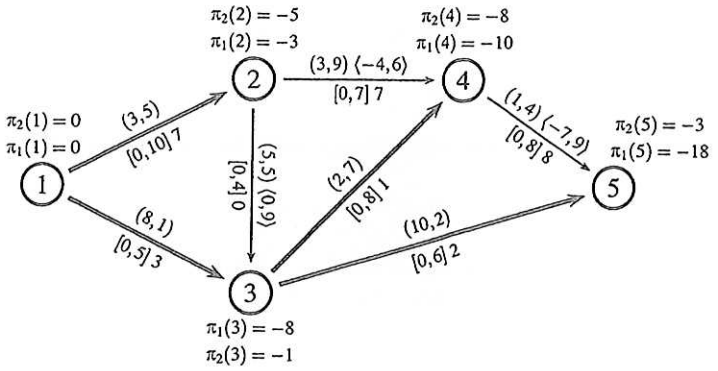


Figure 4: Primal and dual solutions for y^{48}.

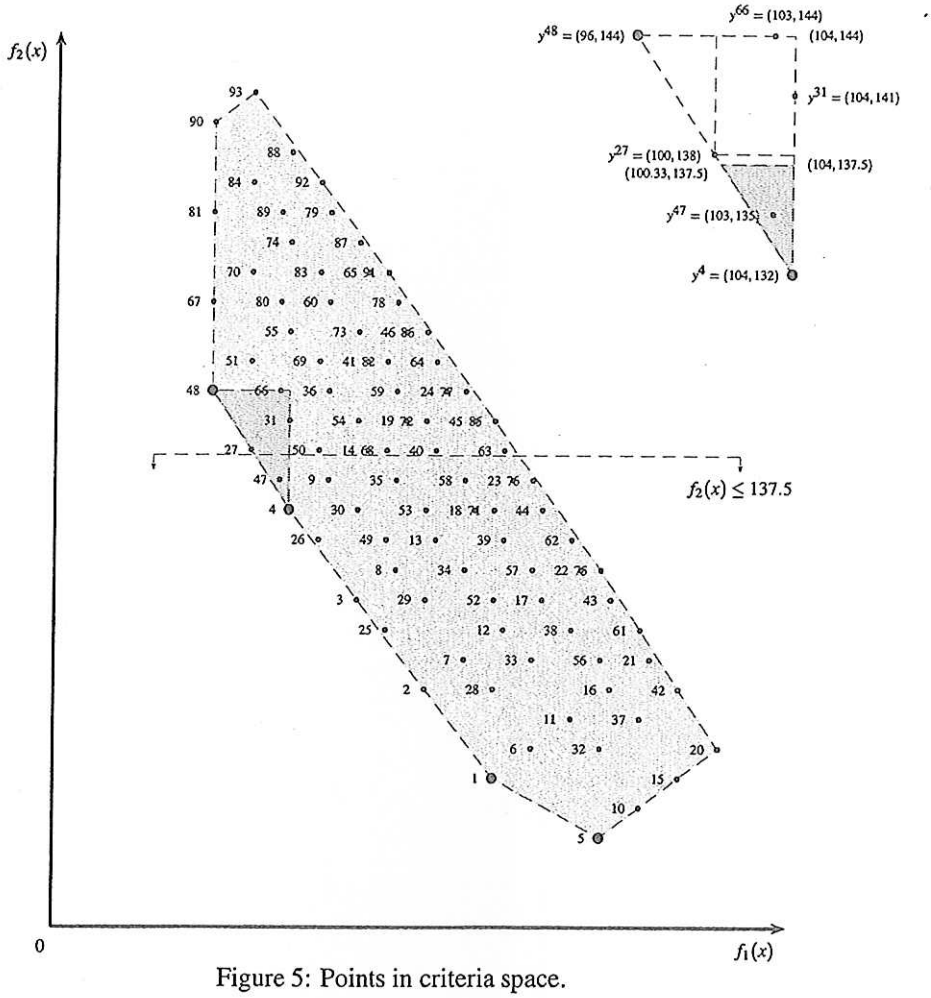
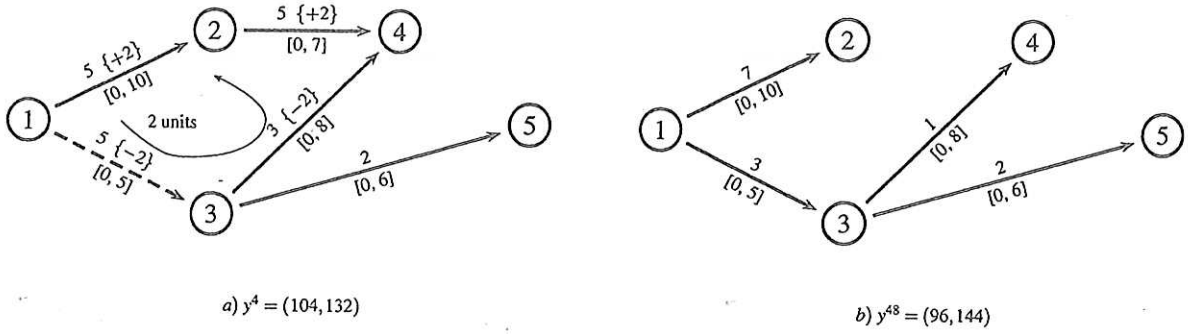


Figure 5: Points in criteria space.

Finally, to complete the illustration of the methods we need to show how the optimal integer solution for (3) can be determined by using the branch-and-bound technique of Figure 2 and the simplex algorithm of Figure 1.

This approach can be used to find all non-dominated solutions for problem (1) as well as a set of non-



a) $y^4 = (104, 132)$

b) $y^{48} = (96, 144)$

Figure 6: Adjacent spanning tree solutions 4 and 48.

dominated solutions in a zone of interest. In practice it is frequent for decision makers to define certain zones of interest for a local search. The proposed method is appropriate to situations of this kind.

Let us suppose that we need to determine all the non-dominated vectors inside the triangle formed by the points $(96, 144)$, $(104, 144)$ and $(104, 132)$. Figure 5 presents this triangle. The same triangle is represented in detail on the upper right corner of Figure 5. Imagine we have already computed vectors y^{48} and y^{27}. Next step is to identify vector y^{47} which is the optimal integer solution of (3), where $\varepsilon = 137.5$. Before reaching vector y^{47}, several steps were executed:

1. First, after obtaining point $y^{27} = (100, 138)$, the constraint $f_2(x) \leq 137.5$ must be introduced in problem (3). In our example this problem is denoted by A, and our list W is updated so that $W = \{A\}$. The feasible region is given by the dark area in the triangle placed on the upper right of Figure 5.

2. Second, the optimal non-integer solution of A must be determined while A is removed from W. In order to compute the optimal value of A, a simple technique can be used. We only need to identify the two nearest extreme points of the non-integer solution for A. These two extreme points correspond to the STSs 48 and 4 (see Figure 5). This means that the non-integer solution for A is between STSs 48 and 4. STS 48 is on the left of the optimal non-integer solution while STS 4 is on the right. The cycle allowing to move from 4 to 48 is $C = \{(1,2),(2,4),(3,4),(1,3)\}$, where the arc (k,l) is the arc $(1,3)$. Figure 6 presents these two adjacent STSs. The amount sent along C is $\Delta = 2$. This means that when we send 2 units from STS 4 along C, STS 48 is reached and the cost for the second criterion increases in the quantity $144 - 132 = 12$, that is, an increase of 6 for each unit sent along C. So, if we want an increase of $137.5 - 132 = 5.5$ in the second criterion, we must send $\Delta = 5.5/6 = 0.9167$ along C. In this case we obtain a non-integer solution with exactly $|C| = 4$ non-integer variables (see Table 1). The branch-and-bound tree is given in Figure 7.

3. Third, we proceed to a partition of A into two subproblems, B and C. The branching variable is $x(1,3) = 4.08333$. Subproblem B is defined by introducing constraint $x(1,3) = 5$, while subproblem C is defined with the help of constraint $0 \leq x(1,3) \leq 4$. Defining these two subproblems in this way, we can guarantee that STSs 48 and 4 are always feasible for B and C, respectively.

4. Fourth, we need to determine the bounds for both subproblems, B and C. Subproblem B has an integer solution. Now, B is the incumbent problem with cost equal to 104 (see Appendix B and Table 1).

5. Fifth, let us study now subproblem C. Appendix C contains the feasible region for C in criteria space. Let us recall that STS 48 remains feasible for subproblem C. So, we can start by using this solution and then move to 27 which is now a STS, but it is still on the left (or above the line for $f_2(x) = 137.5$) of the non-integer optimal solution for C. Therefore, we need to continue in order to obtain a STS

on the right of the non-integer optimal solution for C. When moving to the adjacent solution on the boundary of $Conv(Y)$, STS 25 is attained (see Figures 9 and 10 on Appendix C). Now, we proceed as in 2 and C is added to the list W for further analysis (see also Figure 7 and Table 1).

6. We proceed in the same manner until the optimal solution is obtained. The tree shown in Figure 7 gives us all the iterations needed to solve (3) where $\varepsilon = 137.5$. The corresponding solutions are presented in Table 1. Appendices B to K provide a set of Figures which represent the criteria space for each subproblem of the tree presented in Figure 7.

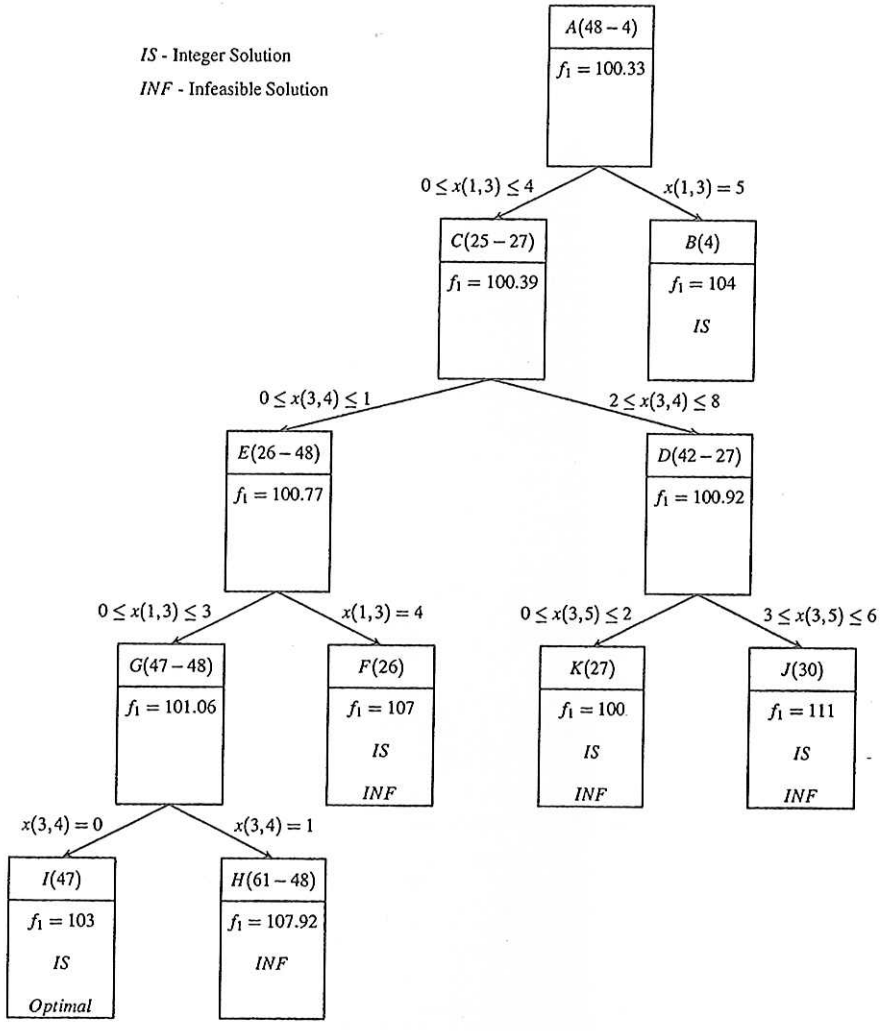


Figure 7: Branch-and-bound iterations.

$c_1(i,j)$ $c_2(i,j)$	3 5	8 1	5 5	3 9	2 7	10 2	1 4		
Solution	$x(1,2)$	$x(1,3)$	$x(2,3)$	$x(2,4)$	$x(3,4)$	$x(3,5)$	$x(4,5)$	$f_1(x)$	$f_2(x)$
A (4-48)	5.91667	4.08333	0	5.91667	1.08333	2	8	100.33331	137.5
B (4)	5	5	0	5	3	2	8	104	132
C (25-27)	6	4	0	6	1.94444	2.05556	7.94444	100.38892	137.5
D (42-27)	6	4	0.08333	5.91667	2	2.08333	7.91667	100.91663	137.5
E (26-48)	6.56667	3.43333	0	6.56667	1	2.43333	7.56667	100.76663	137.5
F (26)	6	4	0	6	1	3	7	107	129
G (47-48)	7	3	0	7	0.27778	2.72222	7.27778	101.05554	137.5
H (61-48)	7	3	1.08333	5.91667	1	3.08333	6.91667	107.91663	137.5
I (47)	7	3	0	7	0	3	7	103	135
J (30)	6	4	1	5	2	3	7	111	132
K (27)	6	4	0	6	2	2	8	100	138

Table 1: Solutions A to K.

7. CONCLUSIONS

Studies of the characterization of the efficient set and design of new approaches for multiple criteria combinatorial problems are scarce. There are, of course, many questions which remain open in this field. In this paper a method for identifying the efficient solution set for the bi-criteria "minimum cost" network flow problem was presented. The method is based on ε-constraint and branch-and-bound techniques. This method is also valid for more general linear integer bi-criteria problems, but in such cases LP-relaxation must be solved by linear programming algorithms that may not be very efficient. The present paper shows how the LP-relaxation can be easily computed exploiting the particular structure of networks. This method has not yet been implemented and tested, but we have good reasons to suspect that it can be very efficient in practice. The next step of our research is to implement the algorithm.

The proposed method can also be of great interest in the area of interactive methods when a database with all the non-dominated solutions must be built before starting the interactive protocol between a user and a software. We also claim to build an interactive Decision Support System for bi-criteria "minimum cost" network flow problems using ideas developed in this paper.

REFERENCES

AHUJA, R.K., MAGNANTI, T.M. and ORLIN, J.B. (1993): *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

BAZARAA, M., JARVIS, J. and SHERALI, H. (1990): *Linear Programing and Network Flows*, 2nd ed, John Wiley and Sons, New York

BRADLEY, G., BROWN, G. and GRAVES, G. (1977): "Design and implementation of large scale primal transshipment algorithms", *Management Science*, 24, 1-34.

CALVETE, H. and MATEO, M. (1996): "A sequential network-based approach for the multiobjective network flow problem with preemptive priorities", In M. Tamiz, ed. *Multi-objective Programming and Goal Programming: Theory and Applications*, LNEMS, Vol. 432, Springer-Verlag, Berlin, 74-86.

CHANKONG, V. and HAIMES, V.V. (1983): *Multiobjective Decision Making: Theory and Methodology*, Elsevier, North-Holland, New York.

FRUHWIRTH, B., BURKARD, R., and ROTE, G. (1989): "Approximation of convex curves with application to the bicriterial minimum cost flow problem", *European Journal of Operational Research*, 42, 326-338.

HAIMES, Y.Y., LASDON, L.S. and WISMER, D.A. (1971): "On a bricriterion formulation of the problems of integrated system identification and system optimization", *IEEE Transactions on Systems, Man, and Cybernetics*, July, 296-297.

HILLIER, F. and LIEBERMAN, G. (1995): *Introduction to Operations Research*, 6th ed, McGraw-Hill International Editions, New York.

LEE, H. and PULAT, P. (1991): "Bicriteria network flow problems: Continuos case", *European Journal of Operational Research*, 51, 119-126.

LEE, H. and PULAT, P. (1993): "Bicriteria network flow problems: Integer case", *European Journal of Operational Research*, 66, 148-157.

MALHOTRA, R. and PURI, M. (1984): "Bicriteria network problem", *Cahiers du Centre d'Etudes de Recherche Opérationnelle*, 26(1-2), 95-102.

PARETO, V. (1896): *Cours d'Economie Politique*, Rouge, Lausanne.

PULAT, S., HUARNG, F. and LEE, H. (1992): "Efficient solutions for bicriteria network flow problem", *Computers and Operations Research*, 19(7), 649-655.

RUHE, G. (1988): "Complexity results for multicriteria and parametric network flows using a pathological graph of Zadeh", *Zeitschrift für Operations Research*, 32, 9-27.

STEUER, R.E. (1985): *Multiple Criteria Optimization: Theory, Computation, and Application*, John Wiley and Sons, New York.

APPENDIX A: SET OF ALL FEASIBLE SOLUTIONS

This appendix contains all the solutions concerning the example presented in Section 6. Efficient solutions (non-dominated vectors) are in bold.

Sol.	$x(1,2)$	$x(1,3)$	$x(2,3)$	$x(2,4)$	$x(3,4)$	$x(3,5)$	$x(4,5)$	$f_1(x)$	$f_2(x)$
1	**5**	**5**	**0**	**5**	**0**	**5**	**5**	**125**	**105**
2	**5**	**5**	**0**	**5**	**1**	**4**	**6**	**118**	**114**
3	**5**	**5**	**0**	**5**	**2**	**3**	**7**	**111**	**123**
4	**5**	**5**	**0**	**5**	**3**	**2**	**8**	**104**	**132**
5	**5**	**5**	**1**	**4**	**0**	**6**	**4**	**136**	**99**
6	5	5	1	4	1	5	5	129	108
7	5	5	1	4	2	4	6	122	117
8	5	5	1	4	3	3	7	115	126
9	5	5	1	4	4	2	8	108	135
10	5	5	2	3	1	6	4	140	102
11	5	5	2	3	2	5	5	133	111
12	5	5	2	3	3	4	6	126	120
13	5	5	2	3	4	3	7	119	129
14	5	5	2	3	5	2	8	112	138
15	5	5	3	2	2	6	4	144	105
16	5	5	3	2	3	5	5	137	114
17	5	5	3	2	4	4	6	130	123
18	5	5	3	2	5	3	7	123	132
19	5	5	3	2	6	2	8	116	141
20	5	5	4	1	3	6	4	148	108
21	5	5	4	1	4	5	5	141	117
22	5	5	4	1	5	4	6	134	126
23	5	5	4	1	6	3	7	127	135
24	5	5	4	1	7	2	8	120	144
25	**6**	**4**	**0**	**6**	**0**	**4**	**6**	**114**	**120**
26	**6**	**4**	**0**	**6**	**1**	**3**	**7**	**107**	**129**
27	**6**	**4**	**0**	**6**	**2**	**2**	**8**	**100**	**138**
28	6	4	1	5	0	5	5	125	114
29	6	4	1	5	1	4	6	118	123
30	6	4	1	5	2	3	7	111	132
31	6	4	1	5	3	2	8	104	141
32	6	4	2	4	0	6	4	136	108
33	6	4	2	4	1	5	5	129	117
34	6	4	2	4	2	4	6	122	126
35	6	4	2	4	3	3	7	115	135
36	6	4	2	4	4	2	8	108	144
37	6	4	3	3	1	6	4	140	111
38	6	4	3	3	2	5	5	133	120
39	6	4	3	3	3	4	6	126	129
40	6	4	3	3	4	3	7	119	138
41	6	4	3	3	5	2	8	112	147
42	6	4	4	2	2	6	4	144	114
43	6	4	4	2	3	5	5	137	123
44	6	4	4	2	4	4	6	130	132
45	6	4	4	2	5	3	7	123	141

Table 2: Solutions 1 to 45.

Sol.	$x(1,2)$	$x(1,3)$	$x(2,3)$	$x(2,4)$	$x(3,4)$	$x(3,5)$	$x(4,5)$	$f_1(x)$	$f_2(x)$
46	6	4	4	2	6	2	8	116	150
47	7	3	0	7	0	3	7	103	135
48	7	3	0	7	1	2	8	96	144
49	7	3	1	6	0	4	6	114	129
50	7	3	1	6	1	3	7	107	138
51	7	3	1	6	2	2	8	100	147
52	7	3	2	5	0	5	5	125	123
53	7	3	2	5	1	4	6	118	132
54	7	3	2	5	2	3	7	111	141
55	7	3	2	5	3	2	8	104	150
56	7	3	3	4	0	6	4	136	117
57	7	3	3	4	1	5	5	129	126
58	7	3	3	4	2	4	6	122	135
59	7	3	3	4	3	3	7	115	144
60	7	3	3	4	4	2	8	108	153
61	7	3	4	3	1	6	4	140	120
62	7	3	4	3	2	5	5	133	129
63	7	3	4	3	3	4	6	126	138
64	7	3	4	3	4	3	7	119	147
65	7	3	4	3	5	2	8	112	156
66	8	2	1	7	0	3	7	103	144
67	8	2	1	7	1	2	8	96	153
68	8	2	2	6	0	4	6	114	138
69	8	2	2	6	1	3	7	107	147
70	8	2	2	6	2	2	8	100	156
71	8	2	3	5	0	5	5	125	132
72	8	2	3	5	1	4	6	118	141
73	8	2	3	5	2	3	7	111	150
74	8	2	3	5	3	2	8	104	159
75	8	2	4	4	0	6	4	136	126
76	8	2	4	4	1	5	5	129	135
77	8	2	4	4	2	4	6	122	144
78	8	2	4	4	3	3	7	115	153
79	8	2	4	4	4	2	8	108	162
80	9	1	2	7	0	3	7	103	153
81	9	1	2	7	1	2	8	96	162
82	9	1	3	6	0	4	6	114	147
83	9	1	3	6	1	3	7	107	156
84	9	1	3	6	2	2	8	100	165
85	9	1	4	5	0	5	5	125	141
86	9	1	4	5	1	4	6	118	150
87	9	1	4	5	2	3	7	111	159
88	9	1	4	5	3	2	8	104	168
89	10	0	3	7	0	3	7	103	162
90	10	0	3	7	1	2	8	96	171
91	10	0	4	6	0	4	6	114	156
92	10	0	4	6	1	3	7	107	165
93	10	0	4	6	2	2	8	100	174

Table 3: Solutions 46 to 93.

APPENDIX B: CRITERIA SPACE FOR SUBPROBLEM B

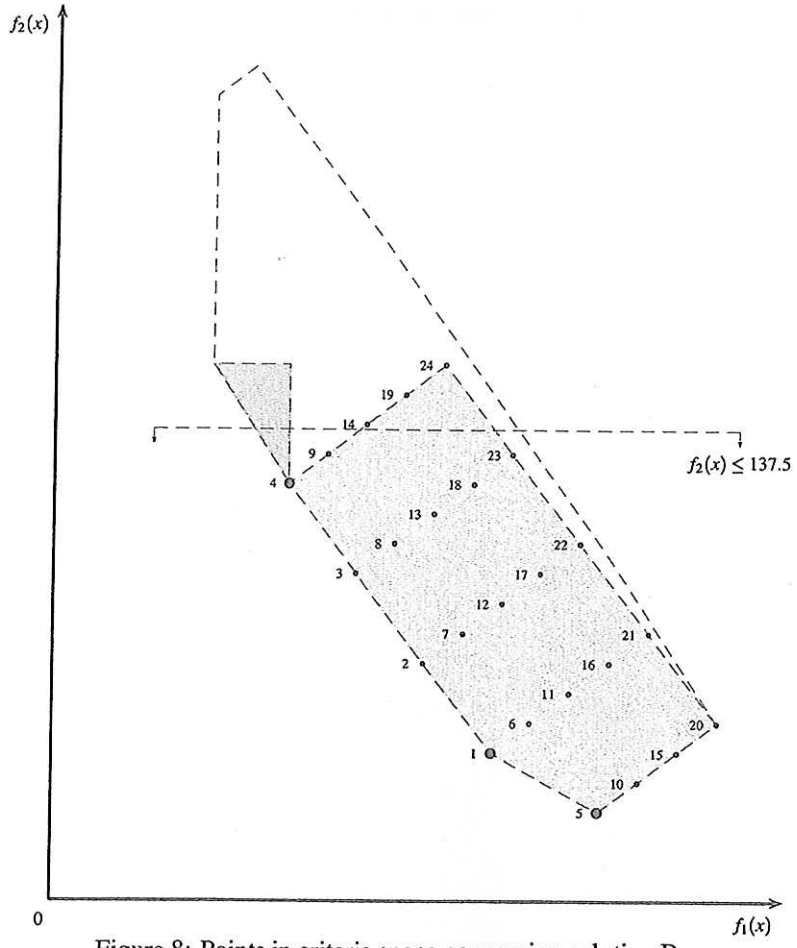


Figure 8: Points in criteria space concerning solution B.

APPENDIX C: CRITERIA SPACE FOR SUBPROBLEM C

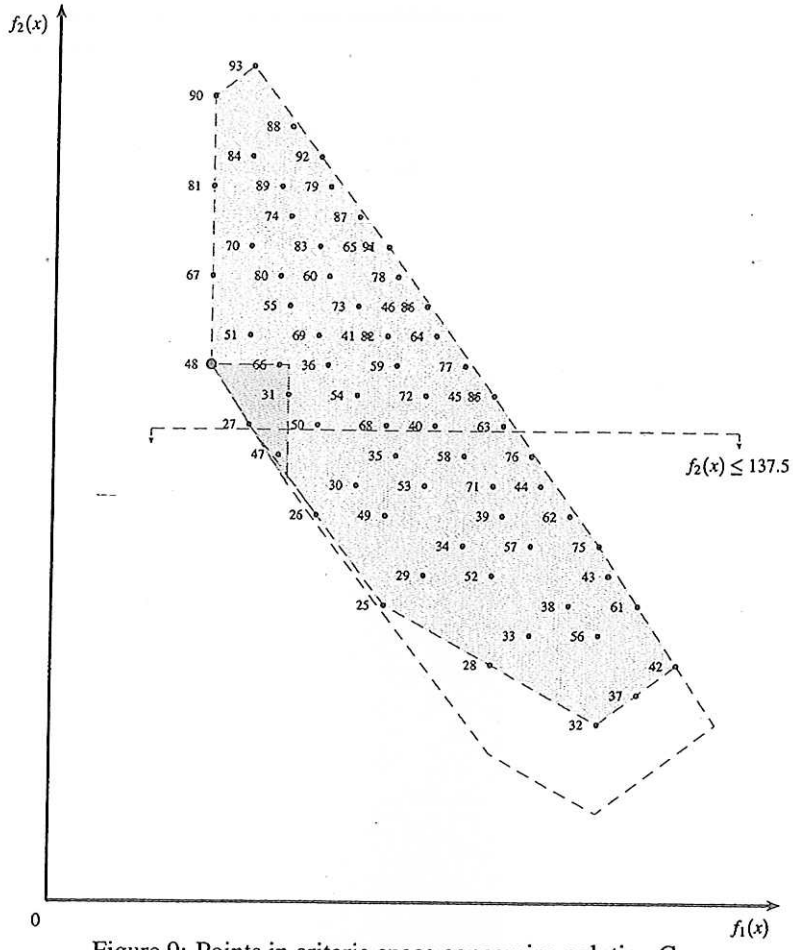


Figure 9: Points in criteria space concerning solution C.

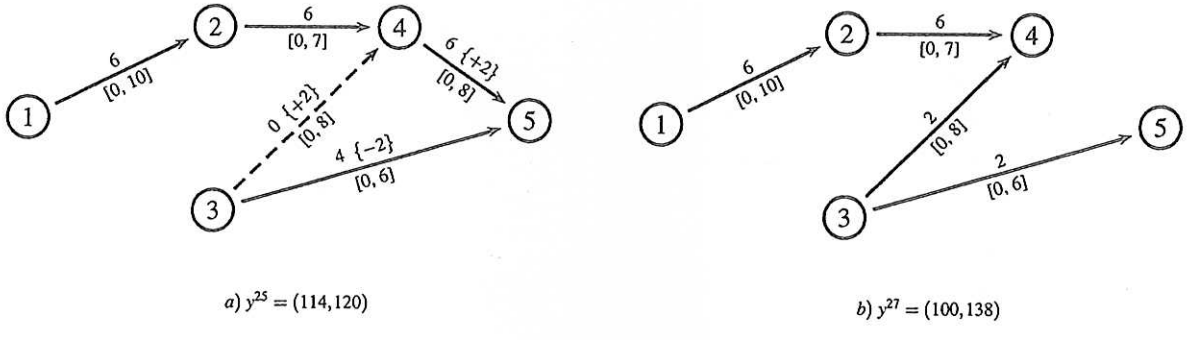


Figure 10: Adjacent spanning tree solutions 25 and 27.

APPENDIX D: CRITERIA SPACE FOR SUBPROBLEM D

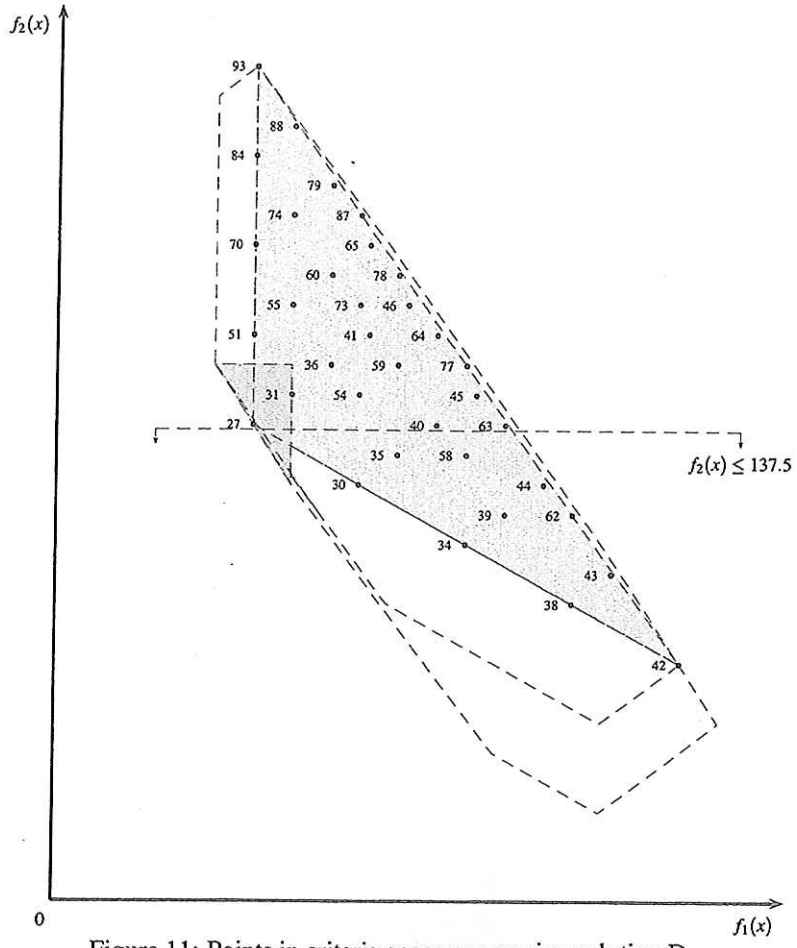


Figure 11: Points in criteria space concerning solution D.

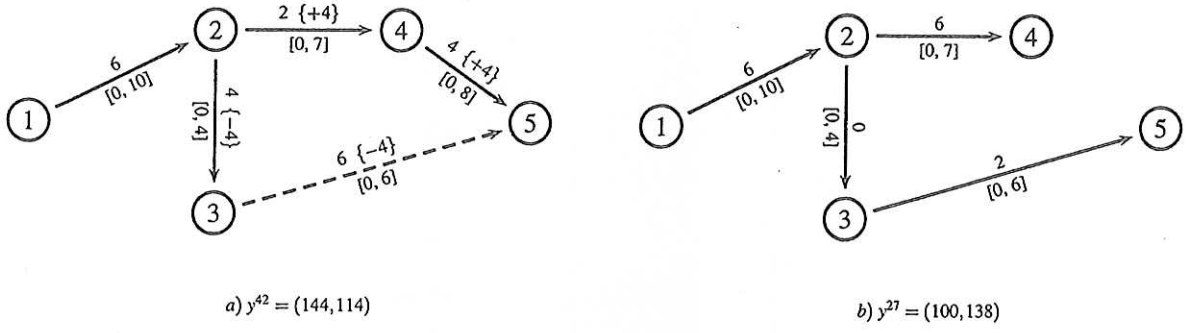


Figure 12: Adjacent spanning tree solutions 42 and 27.

APPENDIX E: CRITERIA SPACE FOR SUBPROBLEM E

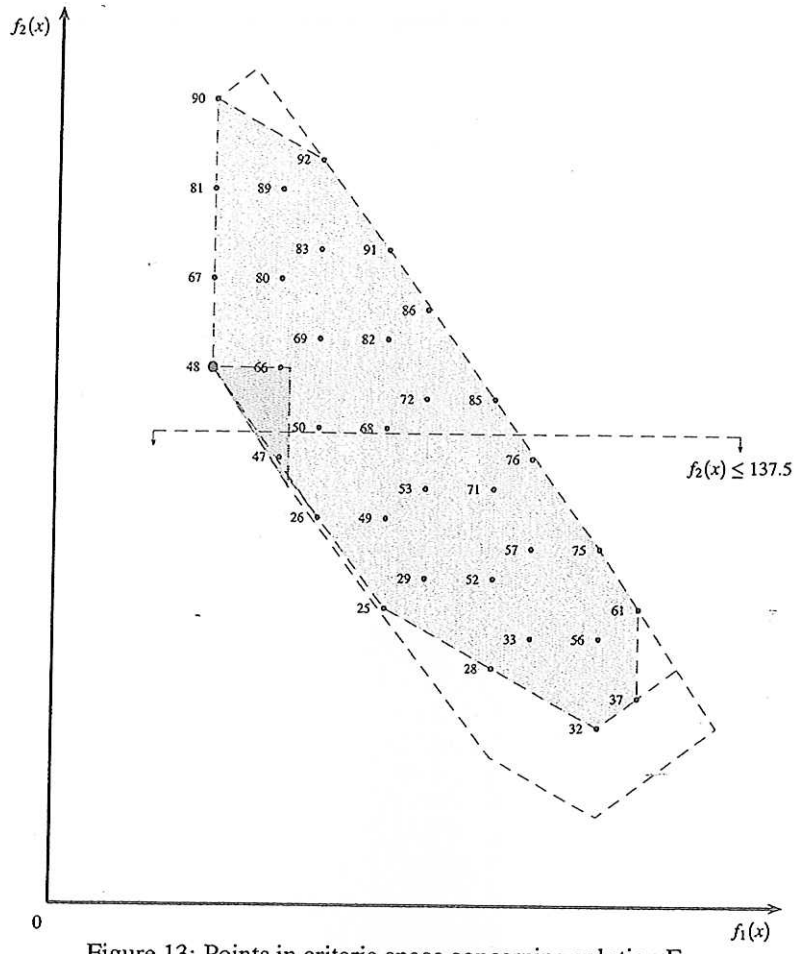
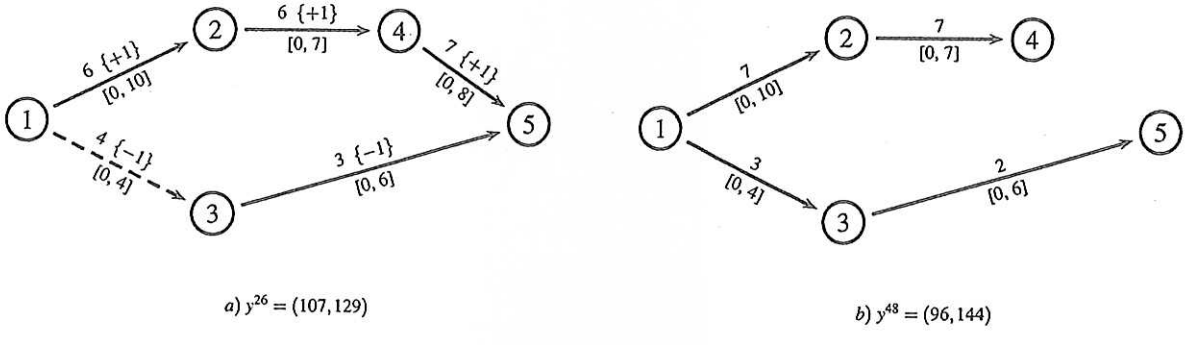


Figure 13: Points in criteria space concerning solution E.



a) $y^{26} = (107, 129)$

b) $y^{48} = (96, 144)$

Figure 14: Adjacent spanning tree solutions 26 and 48.

APPENDIX F: CRITERIA SPACE FOR SUBPROBLEM F

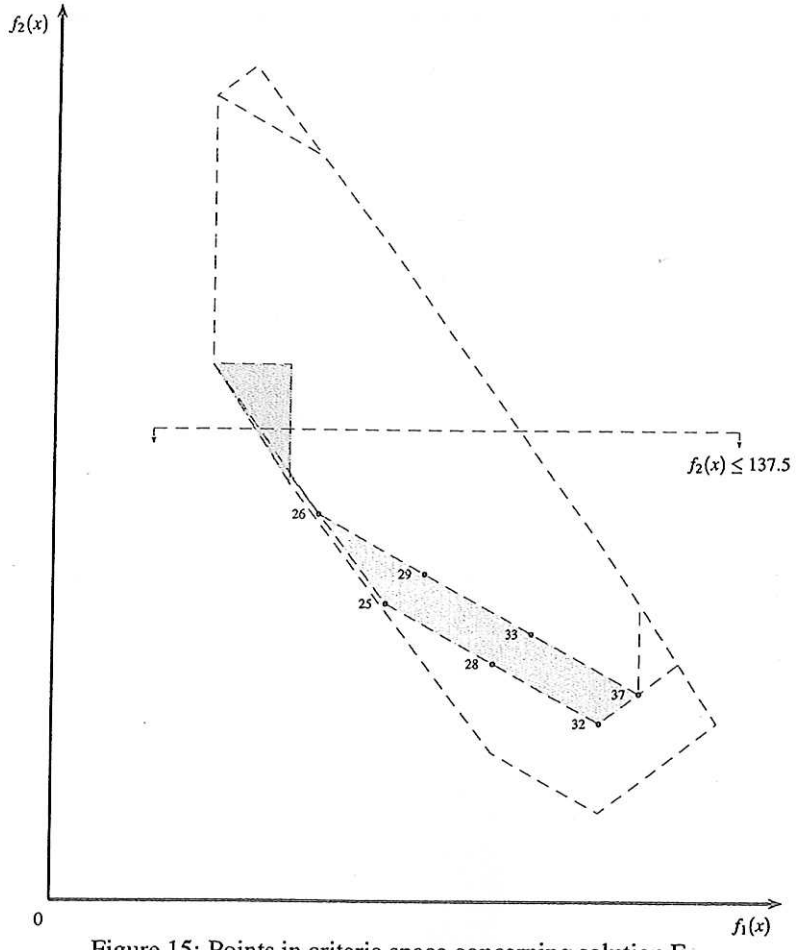


Figure 15: Points in criteria space concerning solution F.

APPENDIX G: CRITERIA SPACE FOR SUBPROBLEM G

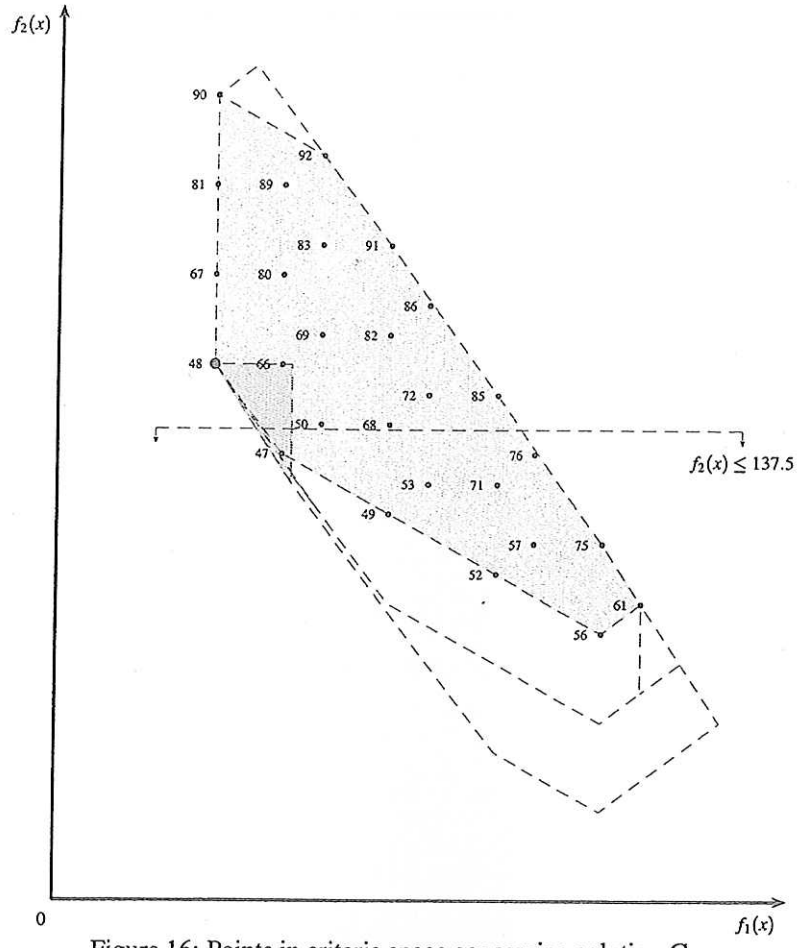


Figure 16: Points in criteria space concerning solution G.

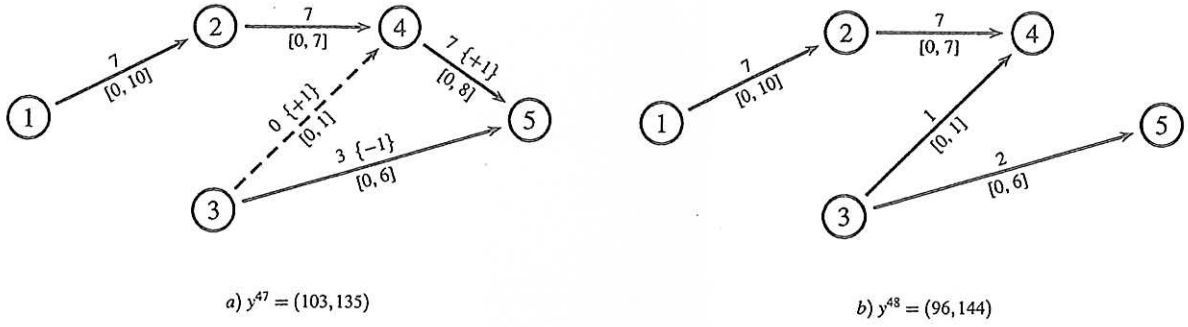


Figure 17: Adjacent spanning tree solutions 47 and 48.

APPENDIX H: CRITERIA SPACE FOR SUBPROBLEM H

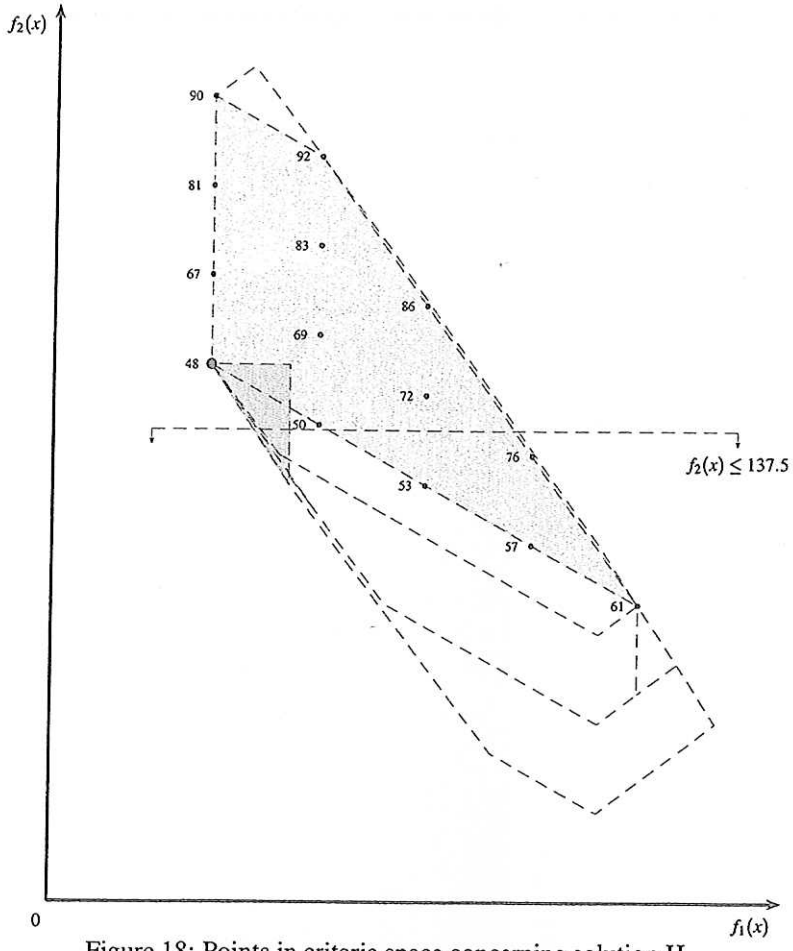


Figure 18: Points in criteria space concerning solution H.

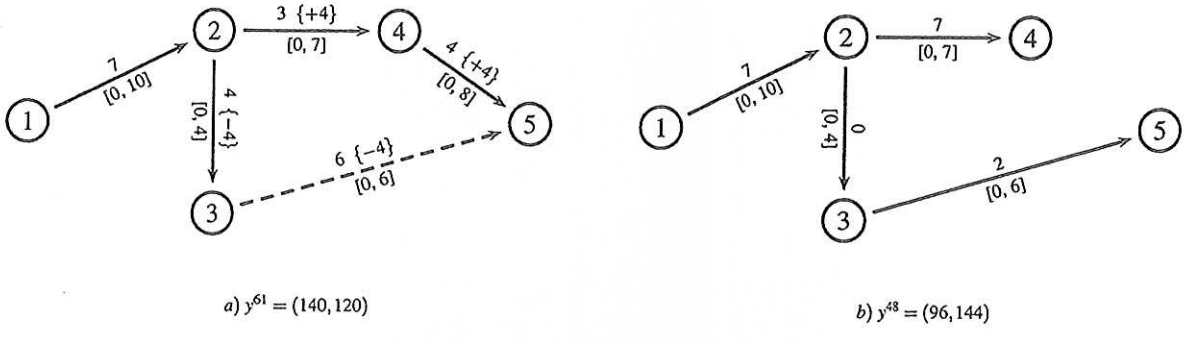


Figure 19: Adjacent spanning tree solutions 61 and 48.

APPENDIX I: CRITERIA SPACE FOR SUBPROBLEM I

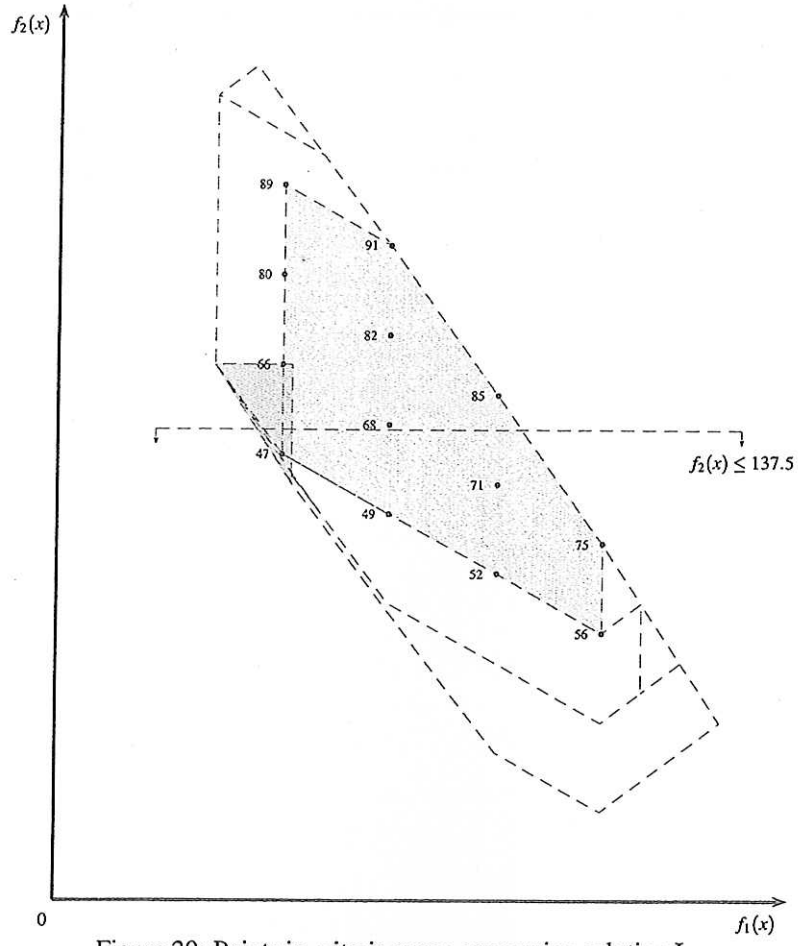


Figure 20: Points in criteria space concerning solution I.

APPENDIX J: CRITERIA SPACE FOR SUBPROBLEM J

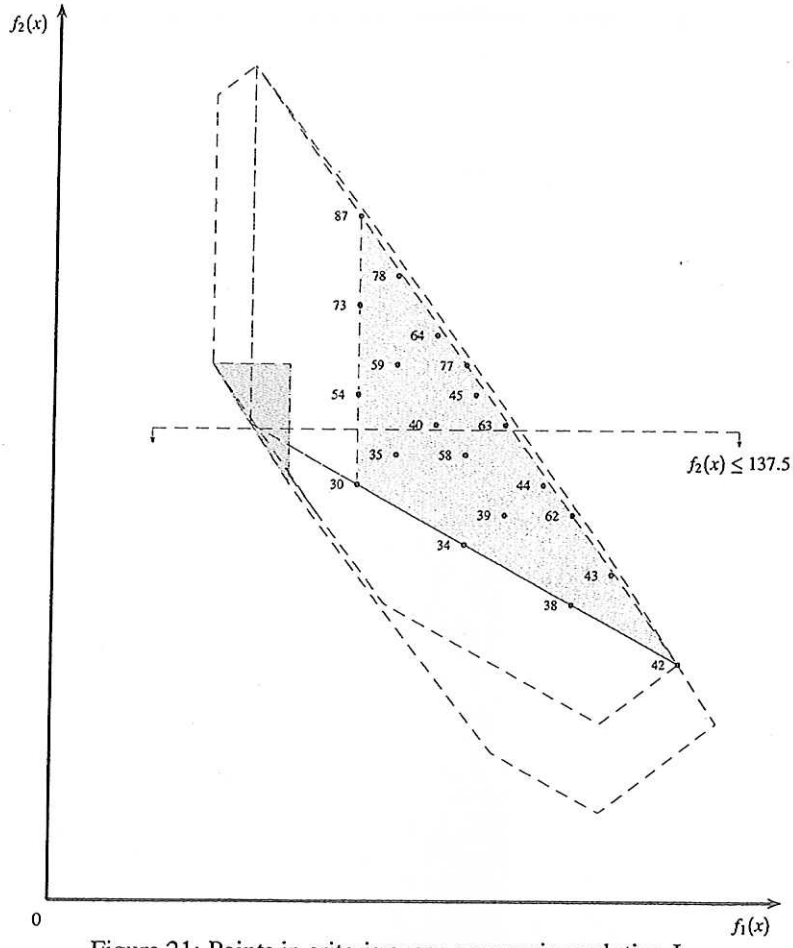


Figure 21: Points in criteria space concerning solution J.

APPENDIX K: CRITERIA SPACE FOR SUBPROBLEM K

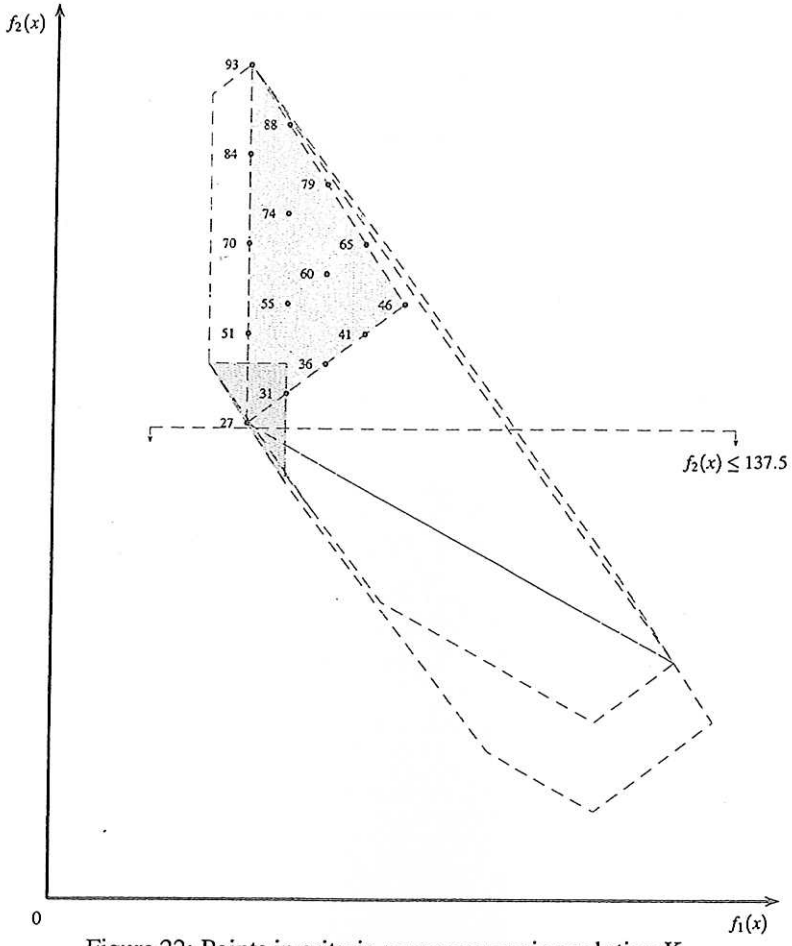


Figure 22: Points in criteria space concerning solution K.

APPENDIX L: ADJACENT STSs

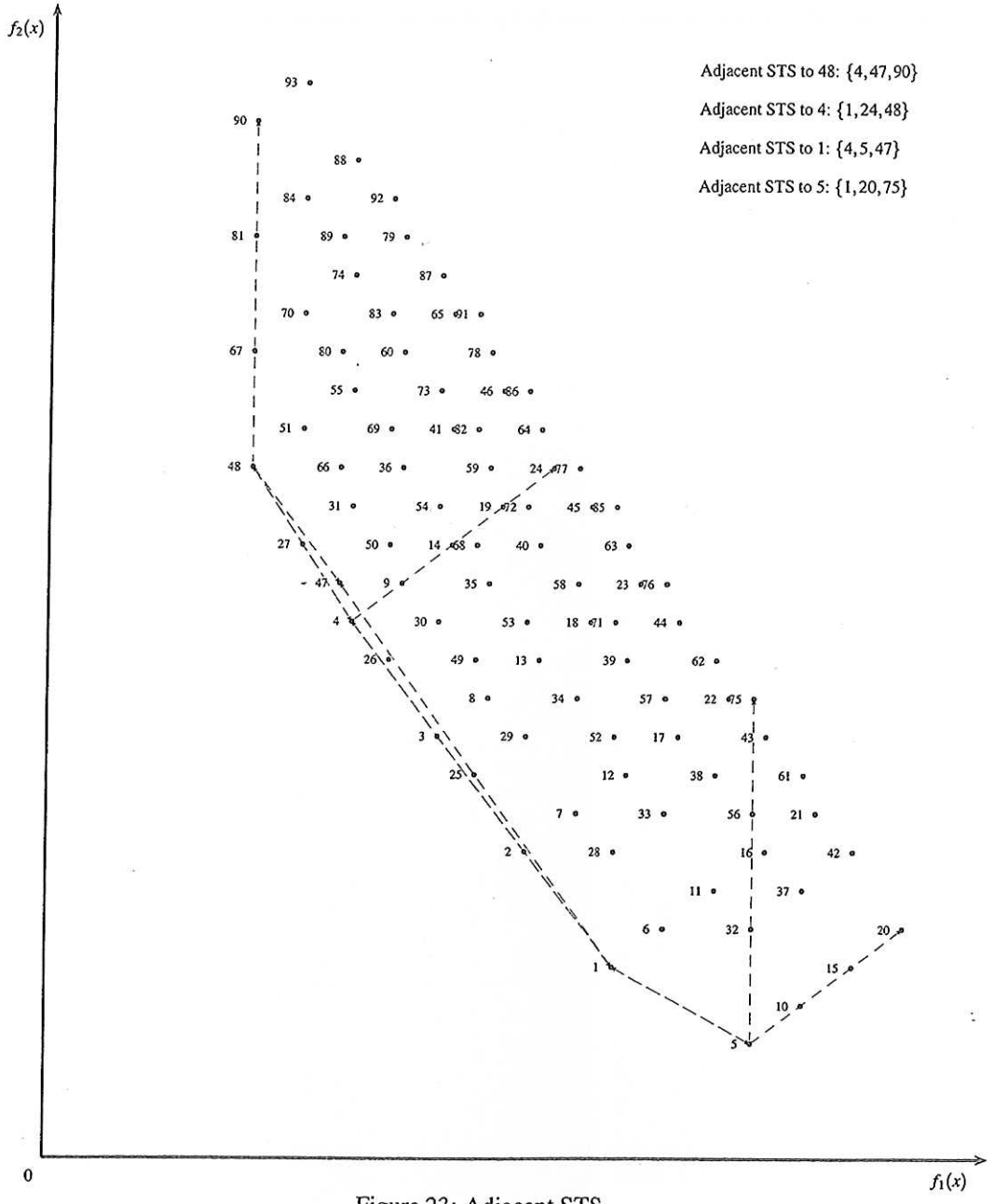


Figure 23: Adjacent STS.

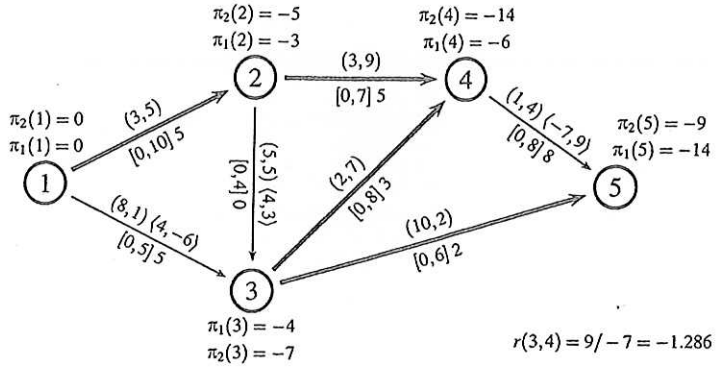


Figure 24: Primal and dual solutions for $y^4 = (104, 132)$.

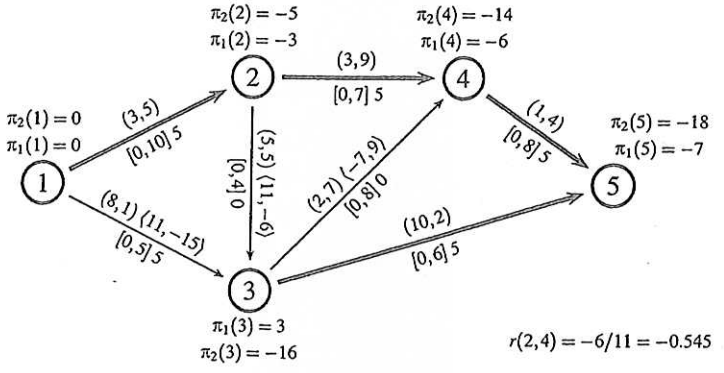


Figure 25: Primal and dual solutions for $y^1 = (125, 105)$.

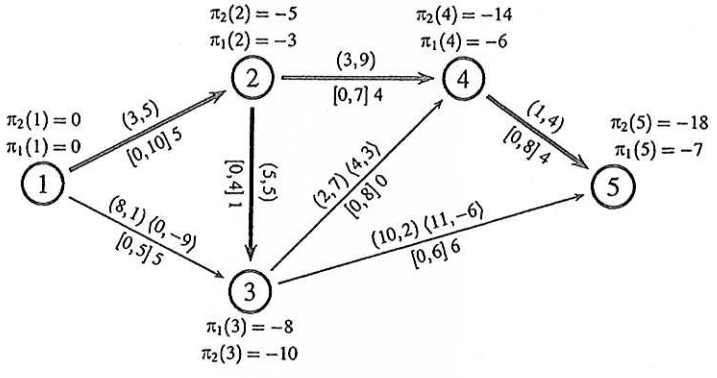


Figure 26: Primal and dual solutions for $y^5 = (136, 99)$.