UNIVERSITÄT KAISERSLAUTERN

RANKING APPROACH TO MAX–ORDERING COMBINATORIAL OPTIMIZATION AND NETWORK FLOWS

Claus Hüsselmann and Horst W. Hamacher

Preprint Nr. 246

ISSN 0943–8874

FACHBEREICH MATHEMATIK

Ranking Approach to Max-Ordering Combinatorial Optimization and Network Flows

Horst W. Hamacher*
and
Claus Hüsselmann

December 8, 1993

Abstract

Max ordering (MO) optimization is introduced as tool for modelling production planning with unknown lot sizes and in scenario modelling. In MO optimization a feasible solution set X and, for each $x \in X$, Q individual objective functions $f_1(x), ..., f_Q(x)$ are given. The max ordering objective $g(x) := \max\{f_1(x), ..., f_Q(x)\}$ is then minimized over all $x \in X$.
The paper discusses complexity results and describes exact and approximative algorithms for the case where X is the solution set of combinatorial optimization problems and network flow problems, respectively.

*partially supported by grants of the Deutsche Forschungsgemeinschaft and the Ministerium für Wissenschaft und Weiterbildung, Rheinland-Pfalz

1 Introduction

Deterministic combinatorial optimization and network flows are among the most powerful models used in the solution of real-world problems (Groetschel (1992), Ahuja et al. (1993)). Whereas the merit of these models is recognized by the practitioners it is - like in any other OR model- nevertheless claimed that the insufficent knowledge of the data leads to optimal solutions of the model which are far from optimal in the practical implementation.
The OR community has reacted to this - obviously meaningful - objection by considering various derivations from deterministic models such as stochastic (Andreatta et. al. (1987), Kall (1976)), fuzzy (Zimmermann (1992)) or multi-criteria optimization (Zeleny (1982)).
The present paper deals with a specific multi-criteria optimization model having strong interrelations with stochastic models which we will call max-ordering (MO) optimization . In other publications it is called min-max (Granot (1984), Warburton (1985)) or max linear optimization (Warburton (1985), Chung et al. (1993)). We will first formally introduce MO combinatorial optimization (MOCO) problems and then show the usefulness of this model.
Let E be a finite set of M elements and let $X \subseteq 2^E$ be a set of feasible solutions of a combinatorial optimization problem. Each $x \in X$ has associated with it Q individual objective values $f^q(x),\ q = 1, ..., Q$. Then the **MO-objective function** is

$$g(x) := \max_{q=1,\dots,Q} f^q(x) \tag{1.1}$$

and the **max ordering combinatorial optimization (MOCO)** problem is

$$\min_{x\in X} \ g(x) \tag{1.2}$$

An important special case of MOCO is the **linear MOCO** where each individual objective has the form $f^q(x) = \sum\limits_{e\in E} w^q(e)x(e) = w^q{\cdot}x$. The notion "max-ordering" is motivated by the fact, that the objective (1.2) compares vectors $(x_1, ..., x_Q)$ with respect to the max-ordering defined by

$$(x_1, ..., x_Q) \leq_{MO} (y_1, ..., y_Q)$$

$$iff$$

$$\max_{q=1,\dots,Q} \ x_q \leq \max_{q=1,\dots,Q} \ y_q$$

such that MOCO can be written as

$$\min_{x \in X} \begin{pmatrix} f^1(x) \\ \vdots \\ f^Q(x) \end{pmatrix} \quad \text{under the max ordering}$$

In spite of this "algebraic" formulation, MOCO cannot be solved using the methods of algebraic optimization as summarized for instance, in Burkard and Zimmermann (1984) or Zimmermann (1981). The reason for this is that the compatibility of vector addition and max ordering

$$x \leq_{MO} y \Rightarrow x + z \leq_{MO} y + z$$

for all $x, y, z \in \mathbb{R}^Q$ requires that we restrict ourselves to vectors with non-negative entries, i.e. $x, y, z \in \mathbb{R}^Q_+$. On the other hand the division rule

$$x \leq_{MO} y \Rightarrow \ \exists\, z : x + z = y$$

requires that $z = y - x$ is not sign constrained. Hence not even the simplest combinatorial optimization algorithm - like the Greedy algorithm for spanning tree - can be carried over to MO problems. This insight into MOCO problems is further enhanced by the complexity results of Section 2.

We illustrate MOCO in the following example.

Example 1.1: *MO-assignment problem*

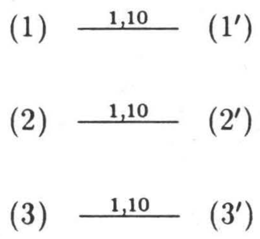
Consider an MO-assignment problem with two linear individual objectives

$$f^q(x) = \sum_{e \in E} w^q(e) \cdot x(e) \quad , \quad q = 1, 2$$

where $w^q(e)$, $q = 1, 2$, are for each $e \in E$ two weights given by

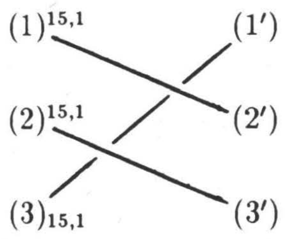
$$w^1 = (w^1(e)) = \begin{pmatrix} 1 & 15 & 2 \\ 4 & 1 & 15 \\ 15 & 2 & 1 \end{pmatrix} , \ w^2 = (w^2(e)) = \begin{pmatrix} 10 & 1 & 3 \\ 2 & 10 & 1 \\ 1 & 3 & 10 \end{pmatrix} .$$

Figure 1.1 shows three assignments including the optimal solution of the MO-assignment problem.



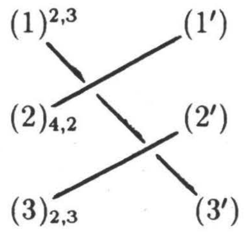
$g(x_1) = 30$

x_1 optimal for first objective



$g(x_2) = 45$

x_2 optimal for second objective



$g(x_3) = 8$

x_3 optimal MO-assignment

Figure 1.1: Three assignment x_1,x_2,x_3 which are optimal for the first and second individual objective, and for the MO assignment problem, respectively.

MO optimization models are useful whenever multiple objectives are considered and a conservative planning is desired .

Application 1: *Unknown lot size production*

We model the planning of a common set-up of a production environment for Q different product types with unknown lot sizes. Each of the possible set-ups (represented by a feasible solution x of a combinatorial optimization problem) implies individual costs $f^q(x)$ for the q-th product type. If we denote with ℓ_q the (unknown!) lot size of product q the overall cost incurred by the decision x is $\ell_1 f^1(x) + \ell_Q f^Q(x)$. But since we do not know the values $\ell_1, ..., \ell_Q$ a conservative planning will require, that none of the values $f^1(x), ..., f^Q(x)$ is particulary large - as this could just turn out to be a product type with a large lot-size, and thus lead to a large overall cost. Hence the minimization of $g(x) := \max\left\{f^1(x), ..., f^Q(x)\right\}$ is a suitable model to use in this context.

Application 2: *Scenario Modelling*

Let E be a set of (yes-no) decisions and let $s_1(e), ..., s_K(e)$ be k possible values (for instance, costs) for the yes-decision of $e \in E$. Each of Q experts chooses a scenario $\left(w^q(e_1), ..., w^q(e_M)\right)$where for all $e \in E$ $w^q(e)$ can be chosen from the set $\{s_1(e), ..., s_K(e)\}$. If the expert opinions, expressed by their commulative values (= individual objectives)

$$f^q(x) = \sum_{e \in E} w^q(e) x(e) \, ,$$

are equally important, then the MO objective $g(x) = \max\left\{f^1(x), ..., f^Q(x)\right\}$ is again a suitable measure for a conservative evaluation of the overall decision represented by $x \in \{0,1\}^M$.

Application 1 and 2 are special cases of a situation where Q different objective function are treated as equal. A discussion of such equity-measurements and their use in location applications can be found in Erkurt (1993) and Marsh and Schilling (1993).

Other applications include partition problems (Garey and Johnson (1979)), multiprocessor scheduling (Granot and Zang (1984)), and robotic assembly problems (Drezner and Nof (1984)).
Obviously, MOCO can be generalized to arbitrary, non-combinatorial optimization problems with a feasible solution set X and Q objective function $f^1, ..., f^Q$. In particular, we will deal in this paper with MO network flow (MONF) problems[1] defined in a (directed) graph $G = (V, E)$. Here

[1]Although network flow problems are almost always included in textbooks on combinatorial optimization, it should be noted that they are formally none. This will become important in the solution procedure of Section 5.

$$X := \left\{ \begin{array}{c} x \in \mathbb{Z}^{|E|} : \sum\limits_{j=1}^{|V|} x_{ij} - \sum\limits_{j=1}^{|V|} x_{ji} = b_i \text{ for all } i \in V \\ \text{and } \ell_{ij} \leq x_{ij} \leq \mu_{ij} \text{ for all } (i,j) \in E \end{array} \right\}$$

and

$$f^q(x) := \sum_{e \in E} w^q(e) \cdot x_e \ , \ q = 1, \dots, Q$$

are Q different linear cost objectives. Then the MO network flow problem looks for a flow such that its worst individual cost objective is as small as possible.

In the next section we will briefly review the complexity status of MOCO and MONF and discuss the relation between MO problems and pareto optimality. In Section 3 we introduce a ranking approach for solving MO problems. Section 4 and 5 show how this ranking approach can be applied to MOCO and MONF.

2 Complexity, Relation to Parateo Optimality and some Easily Solvable Special Cases

To the best of our knowledge Warburton, 1985, was the first to consider the complexity of MO problems and to state its NP completeness in the context of matroid theory. The following results are due to Chung et al., 1993.

Theorem 2.1

a) *The unconstrained MOCO problem with two linear individual objectives*

$$\min_{x \in \{0,1\}^n} \max \{f^1(x), f^2(x)\}$$

is NP-hard.

If the number Q of individual objectives is part of the input, MOCO is strongly NP-hard.

The proof is by reducing the subset sum and partitioning problem, respectively, to the unconstrained MOCO. Also, special combinatorial optimization problems like assignment problems (Lebrecht (1991)) and spanning tree problems (Hamacher and Ruhe (1992)) are known to be NP-hard.
A polynomial solution method to a MO location problem is presented in Hamacher and Nickel (1993). This method is based on the interrelation between MO and pareto solutions. Recall that a solution $x \in X$ is called *pareto solution* iff there exists no $y \in X$ dominating x, i.e. such that $f^q(y) \leq f^q(x)$ for all $q = 1, ..., Q$ where at least one inequality is strict.

Theorem 2.2:

There is at least one $x \in X$ which is both pareto and MO solution.

Proof:

Suppose that the MO solution x is dominated by the pareto solution $y \in X$. Since $f^q(y) \leq f^q(x)$, $\forall\, q = 1, ..., Q$

$$g(y) = \max_{q=1,\dots,Q} f^q(y) =: f^p(y) \leq f^p(x) \leq \max_{q=1,\dots,Q} f^q(x) = g(x),$$

such that y is both pareto and MO solution. □

In situations where X_{par}, the set of pareto solutions of an optimization problem with Q objectives, can be well-characterized, Theorem 2.2 implies that MO problems can be written as

$$\min_{x \in X_{par}} g(x).$$

This fact is used in Hamacher and Nickel (1993), to solve some MO planar 1-facility problems with two objectives with a polynomial algorithm.

If all of the individual objectives are of the linear bottleneck type, i.e.

$$f^q(x) = \max_{e \in E} w^q(e) \cdot x(e)$$

MOCO can be reduced to a single-objective bottleneck problem with cost coefficients $w'(e) = \max_{q=1,\dots,Q} w^q(e)$, since

$$\begin{aligned} g(x) &= \min_{x \in X} \max \left\{ f^q(x), \dots, f^Q(x) \right\} \\ &= \min_{x \in X} \max \left\{ \max_{e \in E} w^1(e) \cdot x(e), \dots, \max_{e \in E} w^Q(e) \cdot x(e) \right\} \\ &= \min_{x \in X} \max_{e \in E} \left(\max_{q=1,\dots,Q} w^q(e) \right) \cdot x(e) \end{aligned}$$

3 Bounds and the Ranking Approach

The first result of this section states some simple bounds for MO problems.

Lemma 3.1:

a) *For any $y \in X \quad g(y)$ is an upper bound.*

b) *For all $q = 1, ..., Q$ the optimal single-objective value*

$$\min_{x \in X} f^q(x)$$

is a lower bound.

Proof:

a) For any $y \in X : g(y) \geq \min_{x \in X} g(x)$

b) For any $y \in X$ and for any $q = 1, ..., Q$:

$$\min_{x \in X} f^q(x) \leq f^q(y) \leq \max\left\{f^1(y), ..., f^Q(y)\right\} = g(y)$$

Hence

$$\min_{x \in X} f^q(x) \leq \min_{y \in X} g(y)$$

$\square$

In the following we will consider linear MO problems, any $\lambda = (\lambda_1, ..., \lambda_Q)$ with $\lambda_1, ..., \lambda_Q \geq Q$ and $\sum_{q=1}^{Q} \lambda_q = 1$ and the resulting function $h(x) = (\lambda_1 w^1 + ... + \lambda_Q w^Q)x$.

Lemma 3.2:

a) $\min_{x \in X} h(x)$ *is a lower bound of the linear MO problem*

b) *For all $x \in X : h(x) \leq g(x)$*

Proof:

a) Rewrite the linear MO problem as

$$\begin{array}{ll} \min & z \\ \text{subject to} & z - w^1 x \geq 0 \\ & z - w^2 x \geq 0 \\ & \dots \\ & z - w^Q x \geq 0 \\ & \quad x \in X \\ & \quad z \text{ unrestricted} \end{array}$$

Langrangean relaxation yields

$$\max_{\lambda \geq 0} \min_{x \in X} (1 - \lambda_1 - \dots - \lambda_Q) z + (\lambda_1 w^1 + \dots + \lambda_Q w^Q) x$$

as lower bound. Since z is unrestricted in sign, we can restrict ourselves to λ with $\sum_{q=1}^{Q} \lambda_q = 1$, such that the first part in the summation vanishes, and the result follows.

b) $h(x) = (\lambda_1 w^1 + \dots + \lambda_Q w^Q) x \leq (\lambda_1 + \dots + \lambda_Q) \cdot \max_{q=1,\dots,Q} w^q x = g(x)$
(since $\lambda_1 + \dots + \lambda_Q = 1$). □

Using Lemmata 3.1 and 3.2 we obtain with

$$x_1 \in \arg\min_{x \in X} h(x)$$

a lower bound and with $g(x_1)$ an upper bound for the linear MO problem. Drezner and Nof (1984) proposed (in the context of linear assignment problems) a procedure which iteratively improves this upper and lower bounds until an optimality criterion is satisfied. We subsequently discuss this *ranking approach* for general linear MO problems.

Let $x_1, \dots, x_K$ be the K best (different) solutions of $h(x)$, i.e. $\{x_1, \dots, x_K\} \subseteq X$ and

(3.1) $$h(x_1) \leq h(x_2) \leq \dots \leq h(x_K) \leq h(x) \quad \forall x \notin \{x_1, \dots, x_K\}, \; x \in X$$

Theorem 3.3:

Let $K \in \mathbb{N}$ be the smallest index such that

(3.2) $$\min_{k=1,...,K-1} g(x_k) \leq h(x_K).$$

Then any

(3.3) $$x^* \in \operatorname*{arg\,min}_{k=1,...,K-1} g(x_k)$$

is a solution to the MO problem.

Proof: For all $x \in X$ with $x \neq x_1, ..., x_K$

$$\begin{array}{rcll} g(x^*) & \leq & h(x_K) & \text{by (3.2)} \\ & \leq & h(x) & \text{by (3.1)} \\ & \leq & g(x) & \text{by Lemma 3.2 (b)} \end{array}$$

For all $x \in \{x_1, ..., x_K\}$

$g(x^*) \leq g(x)$ by (3.2), Lemma 3.2(b), and the definition of x* in (3.3)

Hence g(x*) $\leq g(x) \quad \forall x \in X$ □

Theorem 3.3 implies the validity of the following algorithm for solving linear MO problems.

Ranking Algorithm for Linear MO-Problems

- **Input:**

finite X set of feasible solutions

$f^q(x) = \sum_{e \in E} w^q(e)x(e)$ individual, linear objectives

- **Output:**

x optimal solution of MO problem

(1) Compute for each $q = 1, ..., Q$ the single objective optimal values

$$z^q_{low} = \min_{x \in X} f^q(x) \text{ and } z^q_{up} = \max_{x \in X} f^q(x)$$

(2) Compute the set $P \subseteq \{1, ..., Q\}$ of indices p such that

$$z^q_{low} \geq z^p_{up} \text{ for some } q \in \{1, ..., Q\}$$

(3) **If** $\{1, ..., Q\} \setminus P$ contains only a single index q, then output

$$x^* \in \arg\min_{x \in X} \; f^q(x)$$

(4) **Else**

a) Choose $\lambda \in (0,1)^Q$ with $\lambda_1, ..., \lambda_Q \geq 0$; $\sum\limits_{q=1}^{Q} \lambda_q = 1$ and

$$\lambda_p = 0, \; \forall \, p \in P$$

b) Compute the K best solutions of the single-criterion optimization problem $\min\limits_{x \in X} \; h(x)$

where $h(x) = (\lambda_1 w^1 + ... + \lambda_Q w^Q)x$

until for the first time

(3.2) $$\min_{k=1,...,K-1} \; g(x_k) \leq h(x_K)$$

or define $k = |X| + 1$ if (3.2) is never satisfied.

c) Output $x^* \in \arg\min\limits_{k=1,...,K-1} \; g(x_k)$.

The validity of the algorithm is obvious. Its worst case complexity is exponential - as could be expected by the results of Section 2 - if we exclude the trivial case that $|X|$ is polynomial in $|E|$. It should be noted that the algorithm without Steps (1)-(3) is also valid, but may lead to practically inefficient procedures by a wrong choice of λ. Lebrecht (1991) investigated the impact of choosing various λ for MO-assignment problems, Hüsselmann (1993) developed codes for MO-network flows based on this approach (see Section 5).

It is also worth mentioning that the algorithms provides in each iteration lower and upper bounds if the optimality criterion (3.2) is not satisfied.

Theorem 3.4:

In each iteration of the Ranking Algorithm in which the optimality criterion is not satisfied

(3.4) $$h(x_K) \leq \min_{x \in X} \; g(x) \leq \min_{k=1,...,K} \; g(x_k).$$

Proof:

The second inequality of (3.4) follows from Lemma 3.1(1). The first one holds, since

$$h(x_K) < \min_{k=1,\dots,K-1} g(x_k) \qquad \text{(since (3.2) is not satisfied)}$$

and

$$h(x_K) \leq h(x) \leq g(x) \qquad \text{(by Lemma 3.2)(b))}$$

for all $x \in X$ with $x \neq x_1, \dots, x_k$.

□

We can therefore deduce a heuristic from the Ranking Algorithm by modifying the stopping criterion in Step 4(b)

(Step 4b′)

...

until for the first time

(3.2) $\min_{k=1,\dots,K-1} g(x_K) \leq h(x_K)$

or

(3.5) $\min_{k=1,\dots,K-1} g(x_K) - h(x_K) \leq \varepsilon.$

4 Solving Linear MOCO Problems by Ranking

In order to solve a given linear MOCO we need to know how to solve the ranking problem in Step 4. We summarize a general procedure which is due to Hamacher and Queyranne (1985). Other general procedures, which turn out to be special implementations of the one described here, are due to Murty (1968), and Lawler (1972).
Given a combinatorial optimization problem with feasible solution set X we denote with $Opt(X')$ the set of the best solutions in $X' \subseteq X$.
In the binary partition algorithm of Hamacher and Queyranne (1985) we start with the trivial partition $\{X\}$ of X and calculate a best solution $x_1 \in OPT(X)$ and a second best solution y_1 of X. In the k-th step of the algorithm we have a partition $PART$ of X into k sets L_1,...,L_k and $x_v \in OPT(L_v)$ $(v = 1, ..., k)$ such that $\{x_1, ..., x_k\}$ is the set of k best solutions of X. Furthermore we know a second best solution $y_v \in L_v$ for all L_v with $|L_v| > 1$ $(v = 1, ..., k)$. Thus, $y_k \in OPT\{y_v : |L_v| > 1,\ v = 1, ..., k\}$ is a (k+1)-best solution in X. Next, we partition L_j into two sets $L^{(1)}$ and $L^{(2)}$ with $x_j \in L^{(1)}$ and $y_j \in L^{(2)}$. Thus, x_j and y_j is the best solution in $L^{(1)}$ and $L^{(2)}$, respectively. For $i = 1, 2$ if $\left|L^{(i)}\right| > 1$ we calculate a second best solution, replace L_j by $L^{(1)}$ and $L^{(2)}$ and continue with the new partition (see Figure 4.1). Details of the algorithm are given below.

Binary Partition Algorithm for Finding the K Best Solutions of a Combinatorial Optimization Problem

Input: X set of feasible solution, $K > 1$ integer
Output: K best solutions $\{x_1, ..., x_K\}$

```
(1) $L := X$; $PART := \{L\}$; $k := 1$

(2) Calculate a best and second best solution $x_k$ and $y_k$, respectively, in $X$;

(3) WHILE $k < K$ DO

        BEGIN

(4) $Y := \{y_L : L \in PART;\ |L| > 1\}$

(5) IF $Y = \emptyset$ THEN STOP (Feasible set has less than $K$ elements)
    ELSE find $y_L \in OPT(Y)$;

(6) Find a partition $L^{(1)}, L^{(2)}$ of $L$ with $x_L \in L^{(1)}$ and $y_L \in L^{(2)}$;
```

```
(7) $PART := (PART\setminus\{L\}) \cup \left\{L^{(1)}, L^{(2)}\right\}$;
(8) $x_{L^{(1)}} := x_L;\ x_{L^{(2)}} := y_L$;
(9) FOR $j = 1,2$ DO IF $\left|L^{(j)}\right| > 1$ THEN
        find a second best solution $y_{L^{(k)}}$in $L^{(j)}$
(10) $k := k + 1$
        END
```

If we want to apply the Binary Partition Algorithm to specific combinatorial optimization problems we need to know how to compute a second best solution y_L in $L \subseteq X$ if a best solution $x_L \in OPT(X)$ is known. A general way to do this is due to Murty (1968) and Lawler (1972): We consider sets $L_1, ..., L_R$ which partition $L\setminus\{X_L\}$ and compute the best solution $X_r \in Opt(L_r)$, $r = 1, ..., R$. Then $Y_2 \in Opt\ (\{x_1, ..., x_R\})$ can be found by comparing the R objective values of $x_1, ..., x_R$.
In several instances this general approach of finding a second best solution can be improved by specialized algorithms (see, for instance, Chegireddy and Hamacher (1987) for matchings, Hamacher and Queyranne (1985), for bases of matroids, or Camerini and Hamacher (1990) for matroid intersections).
In the following section we will specialize the binary partition approach to network flow problems.

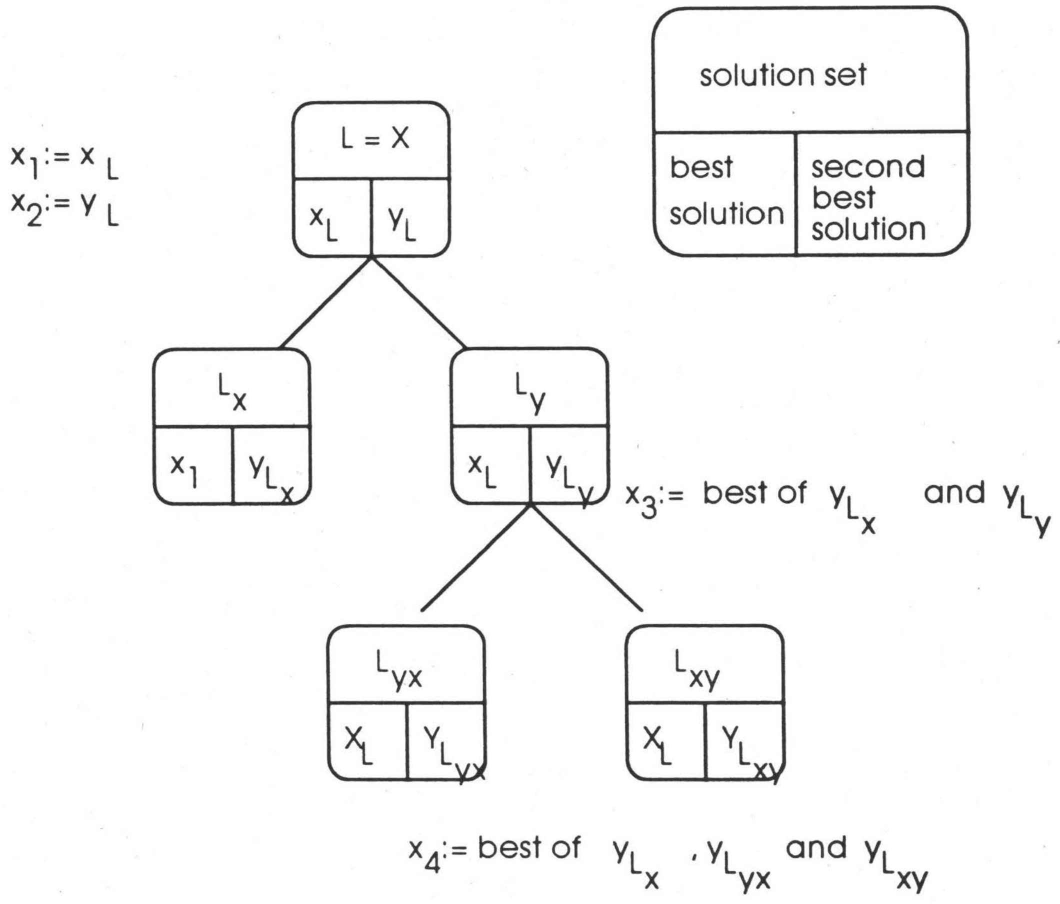


Figure 4.1:

In each step of the binary partition algorithm at most 2 second best solution are computed. The figure shows the binary tree for computing the 4 best solutions $x_1,\ldots,x_4$.

5 Solving MO Network Flow Problems

In this section

$$X := \left\{ \begin{array}{c} x \in \mathbb{Z}^{|E|} : \sum_{j=1}^{|V|} x_{ij} - \sum_{j=1}^{|V|} x_{ji} \text{ for all } i \in V \text{ and } 0 \le l_{ij} \le x_{ij} \le u_{ij} \\ \text{for all } (i,j) \in E \end{array} \right\}$$

is the set of all integer-valued flows in the given (di-)graph G = (V,E) with respect to supplies b_i, $i \in V$ and lower and upper capacities $l_{ij} \in \mathbb{Z}$ and $u_{ij} \in \mathbb{Z}$, respectively, $(i,j) \in E$. We assume that $X \neq \emptyset$, i.e., in particular, $\sum_{i \in V} b_i = 0$.

Since the flow variables are not (0,1) variables, network flow problems are no combinatorial optimization problems. But each X can be represented by (0,1) variables as follows: Duplicate each $(i,j) \in E$ $(u_{ij} - l_{ij})$–times and associate with each of the copies a (0,1)-variable y^t_{ij}, $t = 1, ..., (u_{ij} - l_{ij})$. Then each x_{ij} can be represented by a choice of variable y^t_{ij} such that

$$(5.1) \qquad \left(\sum_{t=1}^{u_{ij}-l_{ij}} y_{ij} \right) + l_{ij} = x_{ij}.$$

Notice however that this representation of an integer flow x by (0,1) variables is not unique, since (5.1) may be satisfied by $\begin{pmatrix} u_{ij} - l_{ij} \\ x_{ij} \end{pmatrix}$ many choices of (0,1) variables $y^t_{i,j}$, $t = 1, ..., (u_{ij} - l_{ij})$. In the single-criterion network flow problem $\min_{x \text{ flow}} f(x) = \sum_{e \in E} w(e) \cdot x(e)$ the distinction of these different representations is not important. But if we want to rank network flows, this phenomenon is in fact the reason why we cannot simply carry over the results of Section 4. A second best solution (which is different from the best one) is in general just another representation (5.1) of the same x_{ij}, such that we stay with the same network flow. Hence we have to develop a specific ranking procedure for network flows. This problem is interesting for its own sake and has so far not been considered in the literature. The closest problem to the ranking of integer network flows is the ranking of minimal cuts in a flow network which was solved in Hamacher (1982) and Hamacher, Picard and Queyranne (1984 a,b).

Let x be an optimal solution of the network flow problem and let G_x be the incremental graph of G with respect to x, i.e. $G_x = (V, E_x)$ where $E_x = E_+ \cup E_-$

$$E_+ := \{(i,j) : x_{ij} < u_{ij}\} ,$$

and

$$E_- := \{(j,i) : x_{ij} > l_{ij}\} .$$

We assume that the original graph $G = (V, E)$ is antisymmetric, i.e. $(i,j) \in E \Rightarrow (j,i) \notin E$, such that $E_+ \subseteq E$ and $E_- \cap E = \emptyset$. (If this is not the case, add a new node k and replace (j,i) by two arcs (j,k) and (k,i) with cost $w_{jk} = w_{ki} = \frac{1}{2}w_{ji}$.)
In G_x costs are defined by

$$\tilde{w}(i,j) = \begin{cases} w(i,j) & \text{if } (i,j) \in E^+ \\ -w(i,j) & \text{if } (j,i) \in E^- \end{cases} .$$

The following results are well-known and can be found in any text book on network flows (see, e.g., Ahuja et al. (1993)).

Lemma 5.1:

a) *If x is an optimal flow and C is a cycle in G_x, then* $\tilde{w}(C) := \sum\limits_{e \in C} \tilde{w}(e) \geq 0$

b) *If x and y are flows in G, then there exist cycles $C_1, ..., C_p$ in G_x and integers $\varepsilon_1, ..., \varepsilon_p > 0$ such that $f(y) - f(x) = \varepsilon_1 \tilde{w}(C_1) + ... + \varepsilon_p \tilde{w}(C_p)$.*

Using Lemma 5.1 obtaining a second best network flow seems to be very simple: Identify a cycle C in G_x with minimal cost $\tilde{w}(C)$ and change x along C by one flow unit, i.e. define the flow $y = x \oplus C$ by

$$y(i,j) := \begin{cases} x(i,j) + 1 & \text{if } (i,j) \in C \cap E^+ \\ x(i,j) - 1 & \text{if } (j,i) \in C \cap E^- \\ x(i,j) & \text{otherwise} \end{cases}$$

By Lemma 5.1 a) and b) $f(y) - f(x)$ is obviously as small as possible.
Unfortunately, this approach does not work without additional assumptions since it may result in a flow y which is equal to x. If C is, for instance, the cycle $C = \{(i,j),(j,i)\}$ with $(i,j) \in E^+$ and $(j,i) \in E^-$, then $\tilde{w}(C) = 0$ and is therefore by Lemma 5.1 a) minimal. But $x \oplus C = x$.

In order to avoid this situation introduce *proper minimal cycles* in G_x defined by

(5.2) $\quad (j,i) \notin C$ for at least one arc $(i,j) \in C$

and

(5.3) $\quad \tilde{w}(C)$ is minimum among all these cycles.

If (5.2) is satisfied for $(i,j) \in C$ we call C also a *proper minimal (i,j)-cycle.* (5.2) implies that $x \oplus C \neq x$ and (5.3) together with Lemma 5.1 yields that $y = x \oplus C$ is a second best integer flow.
In order to find a proper minimal cycle we use the following result.

Theorem 5.2:

Let (i,j) be an arc in G_x and let P_{ji} be a shortest path from j to i in $G_x \backslash \{(j,i)\}$. Then $C_{ij} = \{(i,j)\} \cup P_{ji}$ is a proper minimal (i,j)-cycle.

Proof:

Since any proper minimal (i,j)-cycle C must not contain (j,i), $C \backslash \{(i,j)\}$ is a path from j to i in $G_x \backslash \{(j,i)\}$. Hence

$$\tilde{w}(C) = \tilde{w}(C \backslash \{i,j\}) + \tilde{w}(i,j) \geq \tilde{w}(P_{ji}) + \tilde{w}(i,j) = \tilde{w}(C_{ij}) \qquad \square$$

In order to find a proper minimal cycle we hence solve for each $(i,j) \in E_x$ a shortest path problem in $\mathcal{O}(|V| \cdot |E|)$ time (see Ahuja et al. (1993), Section 5.4) in $G_x \backslash \{(i,j)\}$ resulting in the following $\mathcal{O}\left(|V| \cdot |E|^2\right)$ algorithm for finding the second best network flow.

Second Best Network Flow Algorithm

Input: x optimal solution of network flow problem in $G = (V, E)$

Output: y second best network flow

(1) Compute the incremental graph $G_x = (V, E_x)$.

(2) For all $(i,j) \in E_x$ compute a shortest path P_{ji} from j to i in $G_x \backslash \{(j,i)\}$ and compute

$$C = \arg\min \; \{\tilde{w}(C) : C = \{(i,j)\} \cup P_{ji}\}$$

(3) **Output**

$y := x \oplus C$ defined by

$$y(i,j) := \begin{cases} x(i,j) + 1 & \text{if } \; (i,j) \in C \cap E^+ \\ x(i,j) - 1 & \text{if } \; (j,i) \in C \cap E^- \\ x(i,j) & \quad \text{otherwise} \end{cases} .$$

The average complexity of the Second Best Network Flow Algorithm can be improved by the following observations.

Let $D = (d_{ij})$ be the distance table of G_x with respect to $\tilde{w}$, i.e. d_{ij} is the length of a shortest path P_{ij} from i to j in G_x with length $\tilde{w}(P_{ij}) = d_{ij}$. This can be done for instance with the Floyd-Warshall Algorithm (see Ahuja et al. (1993))

in $\mathcal{O}\left(|V|^3\right)$ time. Notice that the same algorithm allows the computation of P_{ij} as well. Let E_n be the set of nonsymmetric arcs in E_x, defined by $(i,j) \in E_n \subseteq E_x \Rightarrow (j,i) \notin E_n$.

Theorem 5.3:

For all $(i,j) \in E_n$ $\tilde{w}_{ij} + d_{ji}$ is the length of a proper minimal $(i,j)-$cycle.

Proof:

By definition of E_n, the shortest path P_{ji} in G_x does not contain (j,i), i.e., P_{ji} is the shortest path in $G_x \setminus \{(j,i)\}$. Hence Theorem 5.3 follows from Theorem 5.2. □

Due to Theorem 5.3 we can restrict ourselves in Step (2) of the Second Best Network Flow Algoirhtm to computing P_{ji} in $G_x \setminus \{(j,i)\}$ only for arcs $(i,j) \in E_x$ with $(j,i) \in E_x$.

Since we are able to compute a second best network flow y it remains to specify, how we decompose a set L of flows into $L^{(1)}$ and $L^{(2)}$, such that x and y is the best flow in $L^{(1)}$ and $L^{(2)}$, respectively. This can easily be done by identifying some $e \in E$ with $x(e) \neq y(e)$. (Such an arc must exist, since $x \neq y$.) Assume wlog $x(e) < y(e)$, then

$$L^{(1)} := \{z \in L : z(e) \leq x(e)\}$$

and

$$L^{(2)} := \{z \in L : z(e) \geq x(e) + 1\}$$

such that each of the new second best flow problems is defined in a modified flow network with a single change in the upper (for $L^{(1)}$) and lower capacity (for $L^{(2)}$). By using the resulting Binary Partition Algorithm for Network Flows we can solve the max ordering network flow problem by the Ranking Algorithm of Section 3.

Based on the results of this section, Hüsselmann 1993, developed a c-code for solving MO network flows. (This code is availabe to academic users upon request.)

6 Literature

Ahuja, R.K., Magnanti, T.L. and Orlin, J.B.: "Network Flows: Theory, Algorithms, and Applications", Prentice Hall (1992).
Andreatta, G., Mason, F. and Serofini, P.: "Advanced School on Stochastics in Combinatorial Optimization", CISM, Udine, Italy, World Scientific Publ. Co. (1987).
Brucker, P. and Hamacher, H.W. : "K-Optimal Solution Sets in Scheduling Problems", European Journal of Operations Research 41 (1989), 194-202.
Burkard, R.E., Cunningham-Green, R.A. and Zimmermann U. (eds.): "Algebraic and Combinatorial Methods in Operations Research", Annals of OR 19, North-Holland (1984).
Camerini, P.M. and Hamacher, H.W.: "Intersection of Two Matroids: (Condensed) Border Graphs and Ranking", SIAM Journal of Algebraic and Discrete Methods 2 (1989), 16-27.
Chegireddy, C.R. and Hamacher, H.W.: "Algorithms for Finding K Best Matchings in an Undirected Graph", Annals of Discrete Applied Mathematics, 18 (1987), 155-165.
Chung, S.-J., Hamacher, H.W,.Maffioli, F. and Murty, K.G.: "Combinatorial Optimization Problems with Max-Linear Objective Function", Discrete Applied Mathematics (1993).
Drezner, Z. and Nof, S.Y.: "On Optimizing Bin Picking and Insertion Plans for Assembly Robots", IEE Transactions 16,3 (1984), 262-270.
Erkurt, E.: "Inequality Measures for Location Problems", Research Report # 91-2, University of Alberta (1992).
Garey, M.R. and Johnson, D.S.: "Computers and Intractability: A Guide to the Theory of NP-Completeness", W.H. Freeman, San Francisco (1979).
Grötschel, M.: "Discrete Mathematics in Manufacturing", Report # SC 92-3, Konrad-Zuse-Zentrum für Informationstechnik Berlin (1992).
Granot, D.: "A New Exchange Property for Matroids and Its Application to Max-Min Problems", Zeitschrift für Operations Research 28 (1984), 41-45.
Hamacher, H.W. and Nickel, S.: "Multicriterial Planar Location Problems", Research Report # 243, Universität Kaiserslautern, Fachbereich Mathematik (1993).
Hamacher, H.W. and Ruhe, G.: "On Spanning Tree Problems with Multiple Objectives", Preprint # 239, Fachbereich Mathematik, Universität Kaiserslautern (1993) (eingereicht bei Annals of Operations Research).
Hamacher, H.W. and Queyranne, M.: "K Best Solutions to Combinatorial Optimization Problems", Annals of Operations Research 4 (1985), 123-143.

Hamacher, H.W., Picard, J.-C. and Queyranne, M.: "On Finding the K Best Cuts in a Network", Operations Research Letters 2 (1984).

Hamacher, H.W., Picard, J.-C. and Queyranne, M.: "Ranking the Cuts and Cut-Sets of a Network", Annals of Discrete Applied Mathematics 19 (1984), 183-200.

Hamacher, H.W.: "On 0(k n**4) Algorithm for Finding the K Best Cuts in a Network", Operations Research Letters 1 (1982), 186-189.

Hüsselmann, C.: "Das Max-Linear Flußproblem", Diplomarbeit (in German), Universität Kaiserslautern, Fachbereich Mathematik (1993).

Kall, P.: "Stochastic Linear Programming", Lecture Notes in Economics and Mathematical Systems, No. 21, Springer-Verlag, New York (1976).

Lebrecht, U.: "Max-Lineare Zuordnungsprobleme in der Roboteroptimierung", Diplomarbeit (in German), Universität Kaiserslautern, Fachbereich Mathematik (1991).

Marsh, M.T. and Schilling, D.A.: "Foundations of Equity Measurement in Facility Siting Decisions", Studies in Locational Analysis 4 (1993), 207-211.

Righini, G. and Colorni, A.: "Max-Linear Combinatorial Optimization", Politecnico di Milano, Dipartimento di Elettronica, report n. 91.005 (1991).

Warburton, A.: "Worst Case Analysis of Greedy and Related Heuristics for Some Min–Max Combinatorial Optimization Problems", Mathematical Programming 33 (1985), 234-241.

Zeleny, M.: "Multiple Criteria Decision Making", McGraw Hill, New York (1982).

Zimmermann, H.-J.: "Fuzzy Set Theory and its Applications", Kluwer, Boston (1992).

Zimmermann, U.: "Linear and Combinatorial Optimization in Ordered Algebraic Structures", Annals of OR 10, North-Holland (1981).