Open access • Journal Article • DOI:10.1007/BF02032304

On spanning tree problems with multiple objectives — Source link

Horst W. Hamacher, Günther Ruhe

Published on: 01 Dec 1994 - Annals of Operations Research (Baltzer Science Publishers, Baarn/Kluwer Academic Publishers)

Topics: Spanning tree, Minimum spanning tree, Distributed minimum spanning tree, Euclidean minimum spanning tree and Kruskal's algorithm

UNIVERSITÄT KAISERSLAUTERN

ON SPANNING TREE PROBLEMS WITH MULTIPLE OBJECTIVES

Horst W. Hamacher and Günther Ruhe

Preprint Nr. 239

FACHBEREICH MATHEMATIK

ON SPANNING TREE PROBLEMS WITH MULTIPLE OBJECTIVES

Horst W. Hamacher and Günther Ruhe

Preprint Nr. 239

UNIVERSITÄT KAISERSLAUTERN
Fachbereich Mathematik
Erwin–Schrödinger–Straße
6750 Kaiserslautern

März 1993

On Spanning Tree Problems with Multiple Objectives

by

Horst W. Hamacher[*)]

and

Günther Ruhe[&)]

Universität Kaiserslautern

Fachbereich Mathematik

and

Zentrum für Techno- und Wirtschaftsmathematik

Abstract

We investigate two versions of multiple objective minimum spanning tree problems defined on a network with vectorial weights. First, we want to minimize the maximum of Q linear objective functions taken over the set of all spanning trees (*max linear spanning tree problem* **ML-ST**). Secondly, we look for efficient spanning trees (*multi criteria spanning tree problem* **MC-ST**).

Problem ML-ST is shown to be NP-complete. An exact algorithm which is based on ranking is presented. The procedure can also be used as an approximation scheme. For solving the bicriterion MC-ST, which in the worst case may have an exponential number of efficient trees, a two-phase procedure is presented. Based on the computation of extremal efficient spanning trees we use neighbourhood search to determine a sequence of solutions with the property that the distance between two consecutive solutions is less than a given accuracy.

Keywords: Multiple objective, max-linear, multi-criteria, networks, spanning trees, algorithms

---

[*)] Partially supported by Deutsche Forschungsgemeinschaft

[&)] Partially supported by Alexander von Humboldt-Stiftung

Contents:

1. Introduction

The minimum spanning tree problem is one of the simplest, and one of the most central models in the field of combinatorial optimization. A minimum spanning tree connects nodes of a network at minimum total cost and has applications in the planning of efficient distribution systems such as pipelines, transmission lines or in the design of leased-line telephone networks and other telecommunication problems. In the context of network reliability, the weight of a minimum spanning tree represents the minimum probability that the tree will fail at one or more edges. Gomory and Hu (1961) used minimum spanning tree evaluations as subproblems for solving multiterminal flow problems. Held and Karp (1970) used 1-trees for solving traveling salesman problems.

Three basic algorithms for solving the minimum spanning tree problem have been developed. These are the routines of Kruskal (1956), Prim (1957) and Sollin (not published) all of which are based on the greedy approach. The running time of the algorithms are $O(m + n \cdot \log n)$ plus the time needed to sort m edge weights, $O(m + n \cdot \log n)$, and $O(m \cdot \log n)$, respectively. Glover et al. (1992) investigated several variants of non-greedy approaches. Computational testing proved them to be quite successful in reoptimization, where they dominated greedy approaches on all topologies and node degrees.

The appearance of multiple criteria is generally accepted in real-world problem solving. While multi-criteria linear programming with continuous variables is studied extensively, not so much is known for the integer case. In more detail, there is a deficit in practical procedures for multi-criteria integer and network optimization problems. We consider here the computation of minimum spanning trees in the context of vector weighted graphs.

Let $\boldsymbol{T}$ be the set of all spanning trees $T = (V,E(T))$ of a given graph $G = (V,E)$. With each edge $e \in E$ is associated a vector of integer weights $\mathbf{w}(e) = (w_1(e),\dots,w_Q(e))$. Correspondingly, the vector of weights $\mathbf{w}(T)$ of a tree $T \in \boldsymbol{T}$ is defined as

$\mathbf{w}(T) = (w_1(T),\ldots,w_Q(T))$ with

$$w_q(T) \ := \sum_{e \in E(T)} w_q(e) \qquad \text{for} \qquad q = 1,\ldots,Q.$$

We will say that a tree $T \in \boldsymbol{T}$ *dominates* $T' \in \boldsymbol{T}$ if $\mathbf{w}(T) \leq \mathbf{w}(T')$ but $\mathbf{w}(T) \neq \mathbf{w}(T')$. Here and in the following, the ordering relation between vectors is the component-wise ordering. An *efficient spanning tree* is a tree which is not dominated by another one. The set of efficient spanning trees is abbreviated by $\boldsymbol{T}_{\text{eff}}$.

We consider two problems:

Max-linear spanning tree problem (ML-ST)

$\min \{f(T): T \in \boldsymbol{T}\}$ where $f(T) \ := \ \max \{w_1(T),\ldots,w_Q(T)\}$

Multi-criteria spanning tree problem (MC-ST)

$\min^* \{(w_1(T),\ldots,w_Q(T)) \ : \ T \in \boldsymbol{T}\}$

where min* abbreviates the search for efficient spanning trees T.

Two problems related to ML-ST are bottleneck and balanced spanning tree problems. A spanning tree T is a *bottleneck spanning tree* if its maximum edge cost is minimum among all spanning trees. T is called a *balanced spanning tree* if the difference between its maximum and minimum edge cost is as small as possible among all spanning trees. Bottleneck and balanced spanning trees can be determined in $0(m \log n)$ and $0(m^2)$ time, respectively (see e.g. Ahuja et al. [1992]).

Max-linear versions of other combinatorial optimization problems occur in the modelling of printed circuit boards assembly (Drezner and Nof [1984], Lebrecht [1991]).

Most practical applications that require the use of the minimum spanning tree model can be extended naturally to become potential applications of MC-ST. We mention the design of physical systems with different objectives such as throughput, reliablility or design costs. Furthermore, there are many indirect applications such as optimal message passing, the all pairs minimax path problem or cluster analysis where the (multi-criteria) minimum-tree problem accurs as a subproblem.

The rest of the paper is organized as follows. In the second section, the complexity and algorithms for solving ML-ST are presented. In Section 3, the relation between max-linear and multi-criteria minimum spanning trees is discussed. Some theoretical results of MC-ST and corresponding solution algorithms are studied in the last sections.

2. Max-Linear Spanning Trees - Complexity and Algorithms

Before proposing a solution algorithm for ML-ST we investigate its complexity status.

Theorem 2.1.

The max-linear spanning tree problem ML-ST is NP-complete.

Proof:

We consider the unconstrained max-linear combinatorial optimization problem ML-CO introduced and shown to be NP-complete in Chung et al. [1990]:

Input: Q cost vectors $\mathbf{c}_1,\ldots,\mathbf{c}_Q \in Z^n$ where $\mathbf{c}_q = (c_{q1},\ldots,c_{qn})$ for all $q = 1,\ldots,Q$ and an integer B.

Question: Is there a vector $\mathbf{x} \in \{0,1\}^n$ such that $g(\mathbf{x}) := \max \ \{\mathbf{c}_1^T\mathbf{x},\ldots,\mathbf{c}_Q^T\mathbf{x}\} \le B$?

We polynomially transform ML-CO to ML-ST by defining a graph G = (V,E) with

(1) $V = \{1,2,\ldots,n,n+1\}$ and

(2) $E = E_1 \cup E_2$ where

$E_1 := \{(i,i+1)\ ;\ i = 1,\ldots,n\}$

$E_2 := \{(1,i)\ ;\ i = 2,\ldots,n+1\}$

Additionally, weights $w_q(e)$ are introduced as

$$(3) \quad w_q(e) := \begin{cases} c_{qi} & \text{if } e = (i, i+1) \in E_1 \\ 0 & \text{otherwise} \end{cases}$$

for all $e \in E$ and $q = 1, \ldots, Q$.

Let T^* be an optimal solution of the ML-ST problem defined by (1) - (3). For any $T \in \mathcal{T}$ define $\mathbf{x} = \mathbf{x}(T) \in \{0,1\}^n$ with $x_i = 1$ iff $(i,i+1) \in T$. Then (3) implies

$$\begin{aligned} (4) \qquad g(\mathbf{x}) &= \max\{\mathbf{c}_1{}^T\mathbf{x}, \ldots, \mathbf{c}_Q{}^T\mathbf{x}\} \\ &= \max\{w_1(T), \ldots, w_Q(T)\} = f(T). \end{aligned}$$

We claim that $\mathbf{x}^* = \mathbf{x}(T^*)$ is an optimal solution of the original problem ML-CO. To prove this, suppose there is a solution $\mathbf{y} \in \{0,1\}^n$ with $g(\mathbf{y}) < g(\mathbf{x})$. The set

$$E_1(\mathbf{y}) := \{(i,i+1) : y_i = 1; i = 1, \ldots, n\}$$

can be extended by

$$E_2(\mathbf{y}) := \{(1,i+1) : y_i = 0; i = 1, \ldots, n\}$$

forming a spanning tree $T(\mathbf{y})$ of G. Using (4) results in $f(T(\mathbf{y})) = g(\mathbf{y}) < g(\mathbf{x}^*) = f(T^*)$, but this contradicts the assumed optimality of T^*. Finally, we remark that ML-ST is in NP because there exists a polynomial algorithm verifying $f(T) \leq B$ for given T and B.

□

In the following we propose an exact solution procedure for ML-ST which is based on ranking solutions of a single criterion spanning tree problem.

Let $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_Q)$ be a vector of nonnegative real numbers with $\lambda_1 + \ldots + \lambda_Q = 1$ and let

$$w(\boldsymbol{\lambda}, e) = \lambda_1 w_1(e) + \ldots + \lambda_Q w_Q(e)$$

be a convex combination of the weights on each edge $e \in E$. Correspondingly,

$$w(\boldsymbol{\lambda},T) = \sum_{e\in E(T)} w(\boldsymbol{\lambda},e) = \lambda_1 w_1(T) + \ldots + \lambda_Q w_Q(T)$$

is a convex combination of the weights $w_q(T)$ of the tree T - *the combined weight of T (with respect to λ).*

Lemma 2.2.

For all $T \in \boldsymbol{T}$ and for all vectors $\boldsymbol{\lambda} = (\lambda_1,\ldots,\lambda_Q)$ of nonnegative real numbers satisfying $\lambda_1 + \ldots + \lambda_Q = 1$ we get $w(\lambda,T) \le f(T)$.

Proof:

$$w(\boldsymbol{\lambda},T) = \sum_{e\in E(T)} w(\boldsymbol{\lambda},e)$$

$$\le \left(\sum_{q=1}^{Q}\lambda_q\right) \max \{ w_1(T),\ldots,w_Q(T) \} = f(T).$$

□

Lemma 2.3.

Let $T^1 = T^1(\boldsymbol{\lambda})$ be a minimum spanning tree with respect to combined weight $w(\boldsymbol{\lambda},T)$.

Then $w(\boldsymbol{\lambda},T^1) \le f(T^*)$, where T^* denotes an optimal solution of ML-ST.

Proof:

Lemma 2.2 and the minimality of T^1 with respect to $w(\boldsymbol{\lambda})$ imply

$$w(\boldsymbol{\lambda},T^1) \le w(\boldsymbol{\lambda},T^*) \le f(T^*).$$

□

Next, we consider $T^1,\ldots,T^k$, the k best spanning trees with respect to combined weight, i.e.,

$$w(\boldsymbol{\lambda},T^1) \le w(\boldsymbol{\lambda},T^2) \le\ldots\le w(\boldsymbol{\lambda},T^k) \le w(\boldsymbol{\lambda},T)$$

for all spanning trees T different from $T^1,\ldots,T^k$. Ranking of spanning trees can be done by applying a procedure of GABOW (1977). Improvements of this procedure were developed by KATOH et al (1981). We used a binary search procedure of HAMACHER & QUEYRANNE (1985). Its complexity is $O(C(m)+(k-1)B(m))$, where $C(m)$ and $B(m)$ is the complexity to compute the best solution and the (restricted) second best solution, respectively.

As long as $w(\lambda,T^k) < \min\{f(T^i) : i = 1,\ldots,k\}$, the validity of $w(\lambda,T^k) \le w(\lambda,T) \le f(T)$ for all spanning trees T different from $T^1,\ldots,T^k$ implies that $w(\lambda,T^k)$ is a lower bound for the optimal objective value $f(T^*)$ of ML-ST. Note that this lower bound gets larger with increasing k. But if this inequality is violated, an optimum solution for ML-ST is found among the trees $T^1,\ldots,T^k$.

Theorem 2.4.

Let k be a positive integer such that

(5) $w(\lambda,T^{k-1}) < \min\{ f(T^i) : i=1,\ldots,k-1\}$ and

(6) $w(\lambda,T^k) \ge \min\{ f(T^i) : i=1,\ldots,k\}$.

Then

$$T^* \in \arg\min\{ f(T^i) : i=1,\ldots,k\}$$ is an optimal solution for ML-ST [1].

□

Proof:

By definition of the k best solutions of the minimum spanning tree problem with respect to the combined weights and using Lemma 2.2. we get

$$\begin{aligned} f(T^*) &= \min\{f(T^i) : i = 1,\ldots,k\} & \\ &\le w(\lambda,T^k) & \text{(by assumption (6))} \\ &\le w(\lambda,T) & \text{(by definition of k best solutions)} \\ &\le f(T) & \text{(by Lemma 2.2.).} \end{aligned}$$

for all spanning trees T which are different from $T^1,\ldots,T^k$. Since T^* is the best of the spanning trees $T^1,\ldots,T^k$, it solves ML-ST.

□

The above theorem is the foundation of an exact algorithm for the max-linear problem:

---

[1] arg min denotes here and in the following the set of all trees in which the minimum is attained.

Procedure ML-ST

(S1)	Choose $\lambda = (\lambda_1,\ldots,\lambda_Q)$ such that λ_q is nonnegative ($q=1,\ldots,Q$) and $\lambda_1 + \ldots + \lambda_Q = 1$.
(S2)	Compute the combined weights $$w(\lambda,e) := \lambda_1 w_1(e) + \ldots + \lambda_Q w_Q(e) \qquad \text{for all } e \in E.$$
(S3)	Apply a ranking algorithm to find the k best spanning trees with respect to weights $w(\lambda,e)$ until $w(\boldsymbol{\lambda},T^k) \geq \min\{f(T^i) : i=1,\ldots,k\}$.
(S4)	Define T^* due to $T^* \in \arg\min \{f(T^i) : i=1,\ldots,k\}$.

We can stop the procedure when k best spanning trees have been computed, even if the optimality criterion of Theorem 2.4. is not satisfied. In this situation $w(\boldsymbol{\lambda},T^k)$ is a lower bound and $\min\{f(T^i) : i=1,\ldots,k\}$ is an upper bound for $f(T^*)$. Hence the relative accuracy of the current best solution

$$T' \in \operatorname{argmin} \{f(T^i) : i=1,\ldots,k\}$$

is bounded by

$$\frac{f(T')-f(T^*)}{f(T^*)} \leq \frac{f(T')-w(\boldsymbol{\lambda},T^k)}{f(T^*)}$$

$$\leq \frac{f(T')-w(\boldsymbol{\lambda},T^k)}{w(\boldsymbol{\lambda},T^k)} =: \delta$$

We can therefore use the preceding procedure as approximation scheme by specifying the relative accuracy ε and stopping if $\varepsilon \leq \delta$.

A crucial part of the procedure is the choice of $\boldsymbol{\lambda} = (\lambda_1,\ldots,\lambda_Q)$. Consider for example the complete graph K_3 with vector weights $(w_1(e),w_2(e)) = (1,2)$ for all edges e in K_3. If we choose $\boldsymbol{\lambda}=(1,0)$, then all trees T of K_3 have weight $w(\boldsymbol{\lambda},T)=2$. On the other hand $f(T) = w_2(T) = 4$ for all $T \in \mathbf{T}$, such that the stopping criterion of Step S3 is never satisfied. However, we can always guarantee that the algorithm will stop by using the following choice for $\boldsymbol{\lambda}$:

Let T be an arbitrary spanning tree and let $f(T) = w_q(T)$ for some $q \in \{1,...,Q\}$. Then $\boldsymbol{\lambda} = e_q$, where e_q is the q-th unit vector, will always produce a tree T^k satisfying $w(\boldsymbol{\lambda}, T^k) \geq \min \{ f(T^i) : i=1,...,k\}$.

Numerical experience for Q=2 indicated that good results were obtained for $\boldsymbol{\lambda} = (\tau, 1 - \tau)$, where

$$\tau := \arg\min \{h_{max}(\rho,T) - h_{min}(\rho,T) : \rho \in \text{finite subset of } (0,1)\}$$

with

$$h_{max}(\rho,T) := \max \{\rho \cdot w_1(T) + (1-\rho)\, w_2(T) : T \in \boldsymbol{T}\}$$
$$h_{min}(\rho,T) := \min \{\rho \cdot w_1(T) + (1-\rho)\, w_2(T) : T \in \boldsymbol{T}\}.$$

The resulting heuristic is due to Nickel [1992]. Another heuristic choice of $\boldsymbol{\lambda}$ is one which provides the largest lower bound $w(\boldsymbol{\lambda}, T^1)$. This rule was implemented by Lebrecht [1991].

3. Relation between Max-Linear and Multi-Criteria Minimum Spanning Trees

In what follows we investigate some relationships between the set of solutions of ML-ST and MC-ST. Let f_{max} be the optimal objective function value of ML-ST and define

(7) $\boldsymbol{T}_{max} := \{T \in \boldsymbol{T} : f(T) = f_{max}\}$.

A solution $T \in \boldsymbol{T}_{max}$ is said to be *locally non-dominated with respect to* $\boldsymbol{T}_{max}$ if there is no $T' \in \boldsymbol{T}_{max}$ such that $\mathbf{w}(T') \leq \mathbf{w}(T)$ and $\mathbf{w}(T') \neq \mathbf{w}(T)$.

Lemma 3.1.

All locally non-dominated solutions $T \in \boldsymbol{T}_{max}$ are efficient spanning trees.

Proof:

Suppose, there is $T' \in \boldsymbol{T}$ with $\mathbf{w}(T') \leq \mathbf{w}(T)$ and $\mathbf{w}(T') \neq \mathbf{w}(T)$. Then $f(T') \leq f(T) = f_{max}$. Consequently, $T' \in \boldsymbol{T}_{max}$ and hence T′ locally dominates T (contradiction).

$\square$

The following results are immediate consequences of Lemma 3.1

Corollary 1:

If T is the unique solution of ML-ST then T $\in$ $\boldsymbol{T}_{\text{eff}}$

Corollary 2:

Among all solutions of MC-ST there is a spanning tree solving ML-ST.

4. Multicriteria Minimum Spanning Trees - Some Theoretical Results

We consider the integer programming formulation of the minimum spanning tree problem:

(8) $\quad \min \sum_{(i,j)\in E} w_{ij}x_{ij}$ $\qquad$ subject to

(8.1) $\quad \sum_{(i,j)\in E} x_{ij} = n - 1$

(8.2) $\quad \sum_{(i,j)\in E(S)} x_{ij} \leq |S| - 1 \quad$ for all $S \subset V$

(8.3) $\quad x_{ij} \geq 0$ and integer

(8.2) with $|S|=2$ and (8.3) imply that x_{ij} are (0,1)-variables indicating whether we select edge (i,j) as part of the chosen spanning tree. (8.1) implies that exactly n-1 edges are taken, and (8.2.) ensures that the set of chosen edges contains no cycle. The polyhedron defined by the linear programming relaxation of (8.1.) - (8.3.) is denoted by **LP (MST)** and is known to have integer extreme points (see e.g. Ahuja et al. [1992]).

In the following, we consider the multi-criteria, non-integer linear problem related to LP (MST) with objective functions analogous to the one of MC-ST:

$$\min^{*} \{ (w_1^T x, \dots, w_Q^T x) : x \in LP(MST) \}$$

The discussion above implies that the integer efficient solutions of this continuous problem correspond to spanning trees, which are called *extremal efficient spanning trees*. The next result shows that this class of efficient spanning trees is easy to compute.

Theorem 4.1.

An extremal efficient spanning tree is a solution of the parametric problem

(9) $\min \quad w(\boldsymbol{\lambda}, T) := \sum_{q=1}^{Q} \lambda_q w_q(T)$ subject to

(9.1.) $T \in \mathcal{T}$

(9.2.) $\lambda_1 + \dots + \lambda_Q = 1$

(9.3.) $\lambda_q > 0 \quad$ for $q = 1, \dots, Q$.

Proof:

Geoffrion (1968) proved that a feasible solution $\mathbf{x} \in P$ of a multi-criteria linear program $\min^{*} \{ (\mathbf{c}_1^T\mathbf{x}, \dots, \mathbf{c}_Q^T\mathbf{x}) : \mathbf{x} \in P \}$ is an efficient solution if and only if there is a vector $\boldsymbol{\lambda}$ fulfilling (9.2.), (9.3.) such that $\mathbf{x}$ is a solution of the linear program $\min \sum_{q=1}^{Q} (\lambda_q \mathbf{c}_q)^T\mathbf{x}$. If applied to polyhedron LP(MST), the integer feasible solutions are the spanning trees of the given graph, such we obtain all extremal efficient spanning trees. (Notice that the difference between "proper efficient" solutions and "efficient" solutions in Geoffrion's paper is irrelevant in this context, since the number of solutions in (9) is finite.)

□

A special case of extremal spanning trees is obtained by considering the lexicographical ordering "≤" defined by

$\mathbf{y} := (y_1,\dots,y_Q) \leq (z_1,\dots,z_Q) =: \mathbf{z} \;:\Leftrightarrow\; \mathbf{y} = \mathbf{z}$ or $y_i < z_i$ for $i = \min\{j: y_j \neq z_j\}$.

For any permutation $\pi = (\pi(1),\dots,\pi(Q))$ we call T_π a *lexicographical minimum spanning tree* (with respect to π) iff $(w_{\pi(1)}(T_\pi),\dots,w_{\pi(Q)}(T_\pi))$ is lexicographical minimum among all vectors $(w_{\pi(1)}(T),\dots,w_{\pi(Q)}(T))$ of spanning trees $T \in \boldsymbol{T}$.

Corollary: Lexicographical minimum spanning trees are extremal efficient

Proof:

We assume without loss of generality that $\pi = (1,\dots,Q)$. Let $M \geq \max\{ w_q(T) : q=1,\dots,Q,\ T \in \boldsymbol{T}\}$ and define $\varepsilon := 1 / 2M$. We will apply Theorem 4.1 and show that T_π is a solution of (9) if we define $\boldsymbol{\lambda}$ by

(10.1) $\lambda_1 := 1 - \varepsilon\,(1-\varepsilon)$, and

(10.2) $\lambda_q := \varepsilon^{\,q-1}(1-\varepsilon)$ for $q=2,\dots,Q$

We will show that $w(\boldsymbol{\lambda},T_\pi) \leq w(\boldsymbol{\lambda},T)$ for any tree $T \in \boldsymbol{T}$ with $\mathbf{w}(T) \neq \mathbf{w}(T_\pi)$. If $w_q(T_\pi) \leq w_q(T)$ for all $q = 1,\dots,Q$, then the claim obviously holds, since all λ_q are positive. Consider therefore the case, where

$$i := \min\{ q :\ w_q(T) < w_q(T_\pi),\ 1 \leq q \leq Q \}$$

exists. Then the definition of $\boldsymbol{\lambda}$ in (10) implies

$$
\begin{aligned}
& w(\lambda, T_\pi) - w(\lambda, T) \\
& = \lambda_i(w_i(T_\pi) - w_i(T)) + \sum_{j=i+1}^{Q} \lambda_j(w_j(T_\pi) - w_j(T)) \\
& = \lambda_i(w_i(T_\pi) - w_i(T)) + \sum_{j=i+1}^{Q} \varepsilon^{j-1}(1-\varepsilon)(w_j(T_\pi) - w_j(T)) \\
& \le \lambda_i(-1) + (1-\varepsilon)\cdot\varepsilon^i\left(\frac{1-\varepsilon^{Q-i}}{1-\varepsilon}\right)\cdot M \\
& \qquad\qquad (\text{since } w_i(T_\pi) - w_i(T) \le -1 \text{ and } w_j(T_\pi) - w_j(T) \le M) \\
& = \varepsilon^{i-1}(1-\varepsilon)(-1) + \frac{1}{2\varepsilon}\varepsilon^i(1-\varepsilon^{Q-i}) \\
& = \varepsilon^{i-1}\left(-1 + \varepsilon + \frac{1}{2} - \frac{1}{2}\varepsilon^{Q-i}\right) \\
& = \varepsilon^{i-1}\left(-\frac{1}{2} - \frac{1}{2}\varepsilon^{Q-i} + \varepsilon\right) \;<\; 0, \quad \text{since } \varepsilon \le \frac{1}{2}
\end{aligned}
$$

Consequently, $w(\boldsymbol{\lambda}, T_\pi) < w(\boldsymbol{\lambda}, T)$ and T_π is extremal efficient.

□

It should be noted that lexicographical minimum spanning trees are particularly easy to find by using in Kruskal [1956] the lexicographic ordering of the vectors **w**(e) instead of the ordering of real numbers.

In general, however, there are efficient spanning trees $T \in \mathcal{T}_{\text{eff}}$ which can not be derived from the solution of an appropriate problem (9). This is illustrated by the following instance of MC-ST.

Example 1:

Consider the complete graph K_4 with two criteria defined by

edges e_i	**e_1=(1,2)**	**e_2=(1,4)**	**e_3=(1,3)**	**e_4=(2,3)**	**e_5=(3,4)**	**e_6=(2,4)**
$w_1(e_i)$	32	16	8	4	2	1
$w_2(e_i)$	1	20	30	40	41	42

This results in 16 spanning trees all of which are efficient. The location of the corresponding objective function vectors is shown in Figure 1. Note that only three of the 16 trees are extremal. These are the trees $T^1 = \{e_1, e_2, e_3\}$, $T^{10} = \{e_2, e_3, e_6\}$ and $T^{16} = \{e_3, e_5, e_6\}$.

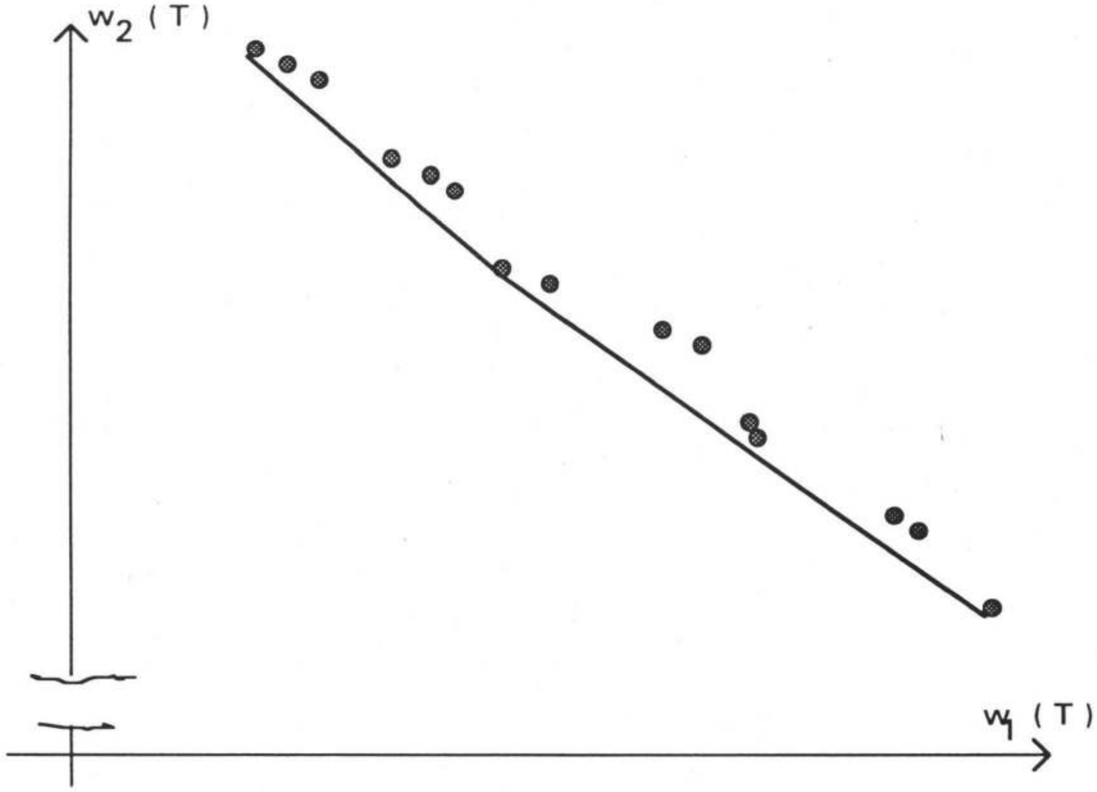

Figure 1 Objective function vectors of the trees of Example 1. All trees are efficient, but only 3 are extremal efficient. The efficient frontier of the corresponding continous multi-criterion linear program passing through the three points is shown as bold line.

Example 1 can be generalized to show that MC-ST may have an exponential number of efficient solutions.

Theorem 4.2:

The number # = | T_{eff} | of efficient spanning trees is in the worst case exponential in the number n = |V| of nodes.

Proof:

Consider $G = K_n$ with edge set $E := \{1,..., m\}$ where $m := n(n-1)/2$. For $k=1,...,m$ we define weights $w_1(k)$ and $w_2(k)$ by

$$w_1(k) := 2^{k-1}, \qquad \text{and}$$
$$w_2(k) := 2^m - 2^{k-1}$$

Consequently, $w_1(k) + w_2(k) = 2^m$ for all $k \in E$ and $w_1(T) + w_2(T) = (n-1)\, 2^m$ for all $T \in \boldsymbol{T}$. By the uniqueness of the number representation in the binary system, $T_1 \neq T_2$ for any two trees T_1, T_2 implies $w_1(T_1) \neq w_1(T_2)$.

Assume that all trees are ordered with respect to strictly increasing weights $w_1(T)$. Then $w_1(T) + w_2(T) = (n-1)\, 2^m$ implies that the ordering is stictly decreasing with respect to weights $w_2(T)$, i.e. all trees are pairwise non-comparable and are therefore efficient. Since the number of trees is $|\boldsymbol{T}| = n^{n-2}$ the claim follows.

□

A further indication of the difficulty of MC-ST is the characterization of efficient trees by a parametric integer program. This can be shown already for the case of two criteria. Using

(11.1) $\quad l_1 := \min \{ w_1(T) : T \in \mathbf{T} \}$,

(11.2) $\quad l_2 := \min \{ w_2(T) : T \in \mathbf{T} \}$

(11.3) $\quad u_1 := \min \{ w_1(T) : T \in \mathbf{T} \text{ and } w_2(T) = l_2 \}$

(11.4) $\quad u_2 := \min \{ w_2(T) : T \in \mathbf{T} \text{ and } w_1(T) = l_1 \}$.

efficient solutions can be characterized by solving an integer program with lexicographic objective function. This is shown in the next result.

Theorem 4.3:

$T' \in \mathbf{T}_{eff}$ if and only if

i) $T' \in \arg \operatorname{lexmin} \{(w_1(T), w_2(T)) : T \in \mathbf{T} \text{ and } w_2(T) \leq b_2 \}$ for some $b_2 \in [l_2, u_2]$

or

ii) $T' \in \arg \operatorname{lexmin} \{(w_1(T), w_2(T)) : T \in \mathbf{T} \text{ and } w_1(T) \leq b_1 \}$ for some

$b_1 \in [l_1,u_1]$

Proof:

Since (i) and (ii) are symmetric we only prove (i).

For $T' \in \mathbf{T}_{eff}$ define $b_2 := w_2(T')$. By definition of l_2 in (11.2) we get $b_2 \geq l_2$. Moreover $b_2 \leq u_2$, since otherwise T' is dominated by T^* with $w_1(T^*) = l_1$ and $w_2(T^*) = u_2 < b_2$. Hence T' satisfies $T' \in \mathbf{T}$ and $w_2(T) \leq b_2$ for $b_2 \in [l_2,u_2]$.

Now suppose $T' \notin \arg \text{lexmin} \{w_1(T) : T \in \mathbf{T} \text{ and } w_2(T) \leq b_2\}$. Then we choose $T'' \in \arg \text{lexmin} \{(w_1(T),w_2(T)) : T \in \mathbf{T} \text{ and } w_2(T) \leq b_2\}$ and get

$$w_1(T'') \leq w_1(T') \quad \text{and} \quad w_2(T'') \leq b_2 = w_2(T'),$$

where one of the inequalities is strict. This contradicts the efficiency of T':

Conversely, let $T' \in \arg \text{lexmin} \{(w_1(T),w_2(T)) : T \in \mathbf{T} \text{ and } w_2(T) \leq b_2\}$. Suppose T' is dominated by T''. Then $w_2(T'') \leq w_2(T') \leq b_2$.

If $w_1(T'') < w_1(T')$, T' cannot be a lexmin tree.

If $w_1(T'') = w_1(T')$, the domination of T' by T'' implies $w_2(T'') < w_2(T')$, again contradicting the fact that T' is a lexmin tree. Consequently, $T' \in \mathbf{T}_{eff}$.

□

In particular the two lexicographical minimum spanning trees T^1 and T^2 of the bicriterion problem are obtained by setting $b_2=u_2$ in (i) and $b_1=u_1$ in (ii). The resulting objective function vectors are $(w_1(T^1), w_2(T^1)) = (l_1,u_2)$ and $(w_2(T^2), w_1(T^2)) = (l_2,u_1)$. These lexicographic minimum spanning trees are obtained by using in Kruskal [1956] the lexicographic ordering of the vectors $(w_1(e), w_2(e))$ and $(w_2(e), w_1(e))$, respectively, instead of the ordering of integer numbers. But notice that this approach is only valid for the case where $b_1=u_1$ or $b_2=u_2$. In all other cases we have to solve an integer program where the constraints are the spanning tree constraints plus an additional linear constraint.

Moreover the result of Theorem 4.3 shows that the objective function vectors of all efficient trees T satisfy $(w_1(T),w_2(T)) \in [l_1,u_1] \times [l_2,u_2]$. The approximation algorithm of the next section will iteratively reduce this area of potential locations of efficient points in the objective space.

5. Bicriteria Minimum Spanning Trees - Algorithms

In Section 4 we have seen that an instance of MC-ST may have an exponential number of efficient trees. Moreover the computation of minimum spanning trees subject to an additional (additive) constraint as used in Theorem 4.3 is known to be NP-complete (CAMERINI et al. [1984]). We therefore present an approximation approach for solving MC-ST.

The idea of our approach is to determine a subset of efficient trees satisfying the following conditions:

(i) It represents the set of all efficient solutions by the fact that the (Euclidean) distance between two consecutive trees is bounded by a given number $\varepsilon > 0$.

(ii) We start the construction of the subset by using extremal efficient trees as long as they exist. After that, new trees are added which satisfy a local efficiency criterion.

(iii) The addition of each new tree to the subset is done by a fast algorithm which is polynomially bounded in $|V|$.

To be more specific, we call an ordered set of spanning trees $\{T^1,T^2,...,T^k\}$ *well-distributed* if for given $\varepsilon > 0$

(12) $\Delta(T^i,T^{i+1}) \leq \varepsilon$ for all $i = 1,...,k-1$,

where

(13) $\Delta(T^i,T^{i+1}) := \sqrt{(w_1(T^i)-w_1(T^{i+1}))^2 + (w_2(T^i)-w_2(T^{i+1}))^2}$

is the Euclidean distance of T^i and T^{i+1} in the objective space.

In the following we will construct a well-distributed set of spanning trees in two stages: In the first stage we will compute extremal efficient spanning trees based on the following Lemma 5.1 If the set of all these trees is not well-distributed we add trees by applying a local search procedure.

Lemma 5.1

Let T^i, T^j be two extremal efficient spanning trees and let

$$\alpha = |w_2(T^i) - w_2(T^j)|$$
$$\beta = |w_1(T^i) - w_1(T^j)|.$$

Then the solution of

(14) $$\min \{\alpha \cdot w_1(T) + \beta \cdot w_2(T) : T \in \mathbf{T}\}$$

is again an extremal efficient tree.

Proof:

With $\alpha' = \alpha/(\alpha + \beta)$ and $\beta' = \beta/(\alpha + \beta)$ the proposition follows from Theorem 4.1, since the constant $1/(\alpha + \beta)$ does not change the minimizers of (14)

□

Stage 1 of our algorithm for solving MC-ST consists of an iterative application of the above lemma. As starting point we use the solutions T^1 and T^2 of the two lexicographical minimum spanning tree problems (see remark after Theorem 4.3).

Procedure MC-ST

Stage 1: Computation of extremal efficient spanning trees

(S1) Compute $T^1 \in \arg \text{lexmin } \{(w_1(T), w_2(T)) : T \in \mathbf{T}\}$ and
$T^2 \in \arg \text{lexmin } \{(w_2(T), w_1(T)) : T \in \mathbf{T}\}$.

(S2) Define $\mathbf{T^*} := \{T^1, T^2\}$, $k := 2$; and $I^o := \varnothing$.

(S3) Let $\mathbf{T^*} := \{T^1,\dots,T^k\}$ with $w_1(T^1) < w_1(T^2) <\dots< w_1(T^k)$
<u>If</u> $\Delta(T^i, T^{i+1}) \le \varepsilon$ for all $i = 1,\dots,k-1$ then **stop**: $\mathbf{T^*}$ is a well distributed set.
<u>Else</u> define $I^1 := \{ i : \Delta(T^i,T^{i+1}) > \varepsilon \} \setminus I^o$

(S4) <u>If</u> $I^1 = \varnothing$ then goto Stage 2
<u>else</u> select $i \in \arg\max \{ \Delta(T^j,T^{j+1}) : j \in \{1,\dots,k-1\}\}$

(S5) Define $\alpha := w_2(T^i) - w_2(T^{i+1})$ and $\beta := w_1(T^{i+1}) - w_1(T^i)$ and
compute $T^k \in \arg\min \{\alpha \cdot w_1(T) + \beta \cdot w_2(T) : T \in \mathbf{T}\}$

(S6) if $w(T^k) \in \{w(T^i), w(T^{i+1})\}$ then $I^o := I^o \cup \{i\}$
else $\mathbf{T^*} := \mathbf{T^*} \cup \{T^k\}$; $k := k+1$
Goto (S3)

If $\#_1$ is the number of extremal efficient spanning trees we can show the following result:

Theorem 5.2

Stage 1 of Procedure MC-ST needs $O(\#_1 (m + n \log n))$ steps to determine extremal efficient spanning trees for all breakpoints in the objective space or to compute a well-distributed set $\mathbf{T^*}$ for given $\varepsilon > 0$.

Proof:

We first remark that the complexity to solve the lexicographical problems in (S1) is the same as the complexity of the minimum spanning tree problem (MST) in (S5) because we can replace the ´min´-operation by ´lexmin´ in the greedy algorithm. The procedure needs $2 \cdot k - 3$ calls of MST to determine k extremal efficient spanning trees $k = 2,\dots,\#_1$. To see that all breakpoints are investigated we assume the contrary,i.e., that there is a breakpoint $\mathbf{w}(T^k)$ corresponding to an extremal efficient spanning tree which is not found by the algorithm. Then consider the breakpoints $\mathbf{w}(T^i)$ and $\mathbf{w}(T^{i+1})$ which define the smallest rectangle

$$[w_1(T^i), w_1(T^{i+1})] \times [w_2(T^{i+1}), w_2(T^i)]$$

containing $\mathbf{w}(T^k)$. However, the solution of (14) with

$$\alpha = |w_2(T^i) - w_2(T^{i+1})|$$
$$\beta = |w_1(T^i) - w_1(T^{i+1})|.$$

results in an extremal efficient tree T^j. Since T^k is strictly smaller than T^i and T^{i+1} with respect to objective function $\alpha w_1(T)+\beta w_2(T)$, $T^j \notin \{T^i, T^{i+1}\}$. Therefore either $\mathbf{w}(T^j) = \mathbf{w}(T^k)$ (contradicting the choice of T^k) or T^j induces a smaller rectangle containing $\mathbf{w}(T^k)$ (contradicting the choice of T^i and T^{i+1}).

For solving the minimal spanning tree problem with scalar weights Prim's algorithm with complexity $0\,(m + n \log n)$ can be used resulting in $0\,(\#_1(m + n \log n))$ steps. If at any intermediate stage of the algorithm $\Delta(T^i, T^{i+1}) \leq \varepsilon$ for all $i = 1,\ldots,k-1$ then a well distributed set is computed and the procedure terminates.

□

In a series of numerical experiments we compared $\#_1$, the number of extremal efficient spanning trees with $\# = |\mathbf{T}_{\text{eff}}|$, the number of all efficient spanning trees. The results obtained for randomly generated graphs are shown in Table 1.

Example	$n = \lvert V \rvert$	$m = \lvert E \rvert$	$\#_1$	#
1	10	20	5	10
2	10	20	6	12
3	10	30	10	26
4	10	30	11	35
5	10	40	8	33
6	15	30	8	26
7	15	60	14	94
8	20	40	13	33

Table 1. Comparison of
$\#_1$ - number of extremal efficient spanning trees and
- number of all efficient spanning trees
for eight randomly generated test examples.

For the computation of # we used ideas of Corley's [1985] algorithm. The algorithm is of the Prim-Type and performs an iterative composition of spanning trees. In each iteration a new edge is added along a cut defined by the set of vertices already contained in a subtree. The main difference to the scalar case is that at each step all efficient extensions are considered, yielding a series of subtrees of increasing cardinality with respect to the set of vertices. The following result is of importance in this approach.

Lemma 5.3:

If $T \in \boldsymbol{T}$ is an efficient spanning tree then the following results hold:

(i) For all edges $e \in E(T)$ let $(X_e, V \backslash X_e)$ be the (unique) cut defined by eliminating e from T. Then no $f \in (X_e, V \backslash X_e)$ satisfies
$\mathbf{w}(f) \leq \mathbf{w}(e)$ and $\mathbf{w}(f) \neq \mathbf{w}(e)$.

(ii) For all edges $f \in E \backslash E(T)$ let P[f] be the (unique) path in T defined by connecting the two end nodes of f in T. Then no $e \in P[f]$ satisfies
$\mathbf{w}(f) \leq \mathbf{w}(e)$ and $\mathbf{w}(f) \neq \mathbf{w}(e)$.

Proof:

In both cases, assume the contrary. Then consider T´ with $E(T') := E(T) \setminus \{e\} \cup \{f\}$ which is a spanning tree again. But $\mathbf{w}(f) < \mathbf{w}(e)$ implies $\mathbf{w}(T') < \mathbf{w}(T)$ contradicting the efficiency of T.

□

Corley [1985] used the converse of Lemma 5.3 in his solution procedure to solve MC-ST. Unfortunately, the following example shows that conditions (i) and (ii) of Lemma 4.5. are necessary, but not sufficient for efficiency of a given minimum spanning tree.

Example 2:

For $G = K_4$ define weights by

edges e_i	**e_1=(1,2)**	**e_2=(1,4)**	**e_3=(1,3)**	**e_4=(2,3)**	**e_5=(3,4)**	**e_6=2,4)**
$w_1(e_i)$	32	16	8	4	2	1
$w_2(e_i)$	1	2	3	4	5	6

Consider the tree T given by $E(T) = \{ e_1, e_3, e_5 \}$. Since the weight vectors of all pairs of edges are non-comparable, conditions (i) and (ii) are satisfied. However $T \notin \mathbf{T}_{eff}$ because $\mathbf{w}(T) > \mathbf{w}(T')$ for T´ with $E(T') = \{ e_2, e_3, e_4 \}$

In our experiments we used a modified version of Corley's algorithm which excludes in each iteration subtrees which are non-efficient. Nevertheless, it generates an exponentially growing number of candidate sets which is prohibitively large even for small problems with $|V| = 30$ nodes. This is one more motivation for the search of approximative solution sets.

If Stage 1 of Procedure MC-ST stops with the set of all extremal efficient solutions, but this set is not well-distributed (Step S4), we need to find additional efficient trees. Figure 2 shows that we can restrict ourselves to investigate a set of triangles which are generated by the objective function vectors $\mathbf{w}(T) = (w_1(T), w_2(T))$ of the extremal efficient trees. This is true, since all objective function vectors are above the efficient frontier and the ones which are not in the triangles are dominated by the extremal efficient trees.

In Stage 2 we iteratively reduce the area of potential locations of efficient value points, until we have computed a well-distributed set.

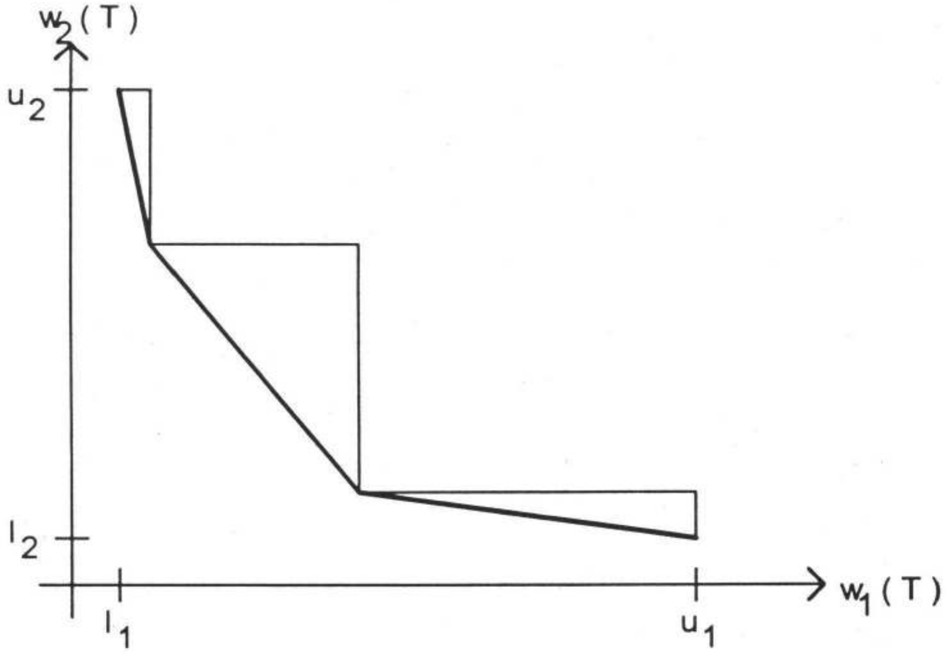

Figure 2 After finding the objective value vectors of the extremal efficient trees only the triangles above the efficient frontier (thick polygon) of the corresponding continous problem may contain objective function points $(w_1(T), w_2(T))$ of additonal efficient trees.

For this purpose the second stage of the solution algorithm performs neighbourhood search. The *neighbourhood Nh(T)* of a tree T is defined as

(15) $\quad Nh(T) := \{T' \in \mathbf{T} : E(T') = E(T) \setminus \{e\} \cup \{f\} \text{ for } e \in E(T), f \in E \setminus E(T)\}$

Neighbourhood search is applied whenever two consecutive solutions T^i, T^{i+1} of the current solution set do not fulfil $\Delta(T^i, T^{i+1}) \leq \varepsilon$. The aim is to find a new tree T^k

$\in Nh(T^i) \cup Nh(T^{i+1})$ such that

(16) $w_1(T^k) < maxw_1 := \max \{w_1(T^i), w_1(T^{i+1})\}$
$w_2(T^k) < maxw_2 := \max \{w_2(T^i), w_2(T^{i+1})\}$

and

(17) $A(j) := [maxw_1 - w_1(T^j)]\,[maxw_2 - w_2(T^j)]$
is maximized by k among all indices j such that T^j is satisfying (16).

The choice of k is illustrated in Figure 3. From (17) it follows that there is no k′ with the property that $\mathbf{w}(T^{k'}) < \mathbf{w}(T^k)$ and $\mathbf{w}(T^{k'}) \neq \mathbf{w}(T^k)$, i.e., the solution T^k is locally efficient with respect to the defined neighbourhood.

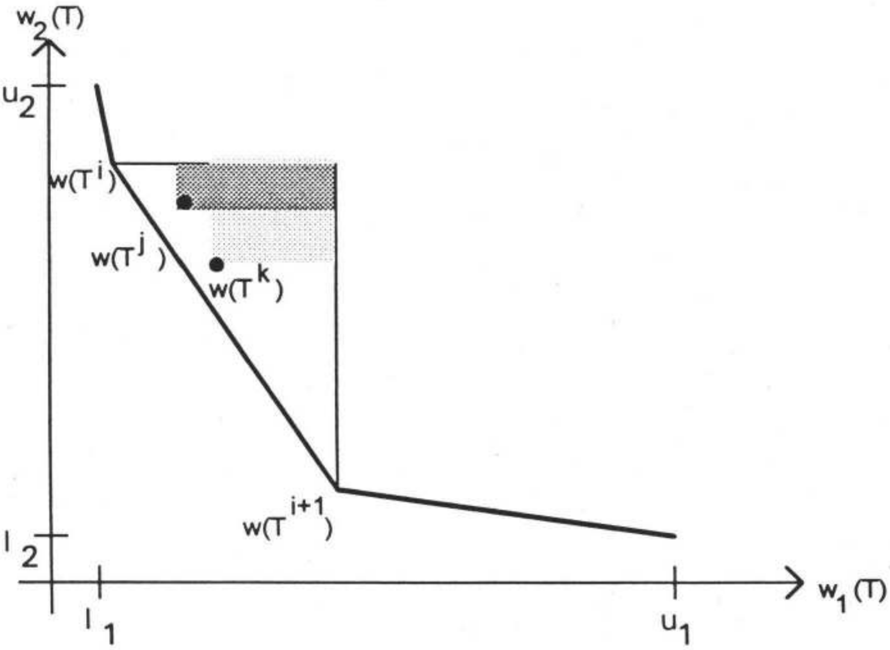

Figure 3: The new solution element T^k is characterized by cutting off a maximum area A(k) from the area containing additional efficient points. The alternative tree T^j cuts off a smaller area.

Procedure MC-ST

Stage 2: Neighbourhood search

(S1) Assume a set $\mathbf{T^*} = \{T^1,...,T^q\}$ with
$w_1(T^1) <...< w_1(T^q)$.
<u>If</u> $\Delta(T^i,T^{i+1}) \le \varepsilon$ for all $i=1,...,q-1$ then **stop:** $\mathbf{T^*}$ is a well-distributed set.

(S2) <u>Else</u> select $i \in \arg\max \{\Delta(T^j,T^{j+1}) : j \in \{1,...,q-1\}\}$
area := 0;
$maxw_1 := w_1(T^{i+1})$; $maxw_2 := w_2(T^i)$

(S3) <u>for all</u> j with $T^j \in Nh(T^i) \cup Nh(T^{i+1})$ <u>do</u>
$f(j) := [maxw_1 - w_1(T^j)]\,[maxw_2 - w_2(T^j)]$
if $w_1(T^j) < maxw_1$ and $w_2(T^j) < maxw_2$ and $f(j) >$ area
then k := j, area = f(j)

(S4) <u>If</u> area = 0 then define $\varepsilon := \Delta(T^i,T^{i+1})$ and **stop:**
$\mathbf{T^*}$ is a well-distributed set.
<u>else</u> $\mathbf{T^*} := \mathbf{T^*} \cup \{T^k\}$
q := q+1 and goto (S1).

We denote by $\#_2$ the number of (locally) efficient solutions calculated in Stage 2 of MC-ST.

Theorem 5.4

Procedure MC-ST needs $0(\#_1(m + n \log n) + \#_2 \cdot m \cdot n)$ steps to determine a well distributed set $\mathbf{T^*}$ of accuracy ε.

Proof:

In addition to the complexity of Stage 1, the procedure investigates $0(|Nh(T^i) \cup Nh(T^{i+1})|) = 0(n \cdot m)$ trees in the iteration caused by $\Delta(T^i,T^{i+1}) > \varepsilon$. Because the number of operations performed per tree is constant, the complexity of the second stage is $0(\#_2 \cdot m \cdot n)$. The algorithm terminates with a well distributed set of the accuracy ε.

□

References

R.K. AHUJA; T.L. MAGNANTI; J.B. ORLIN (1992):
Network Flows: Theory, Algorithms, and Applications.
Prentice Hall

P.M. CAMERINI; G. GALBIATI; F. MAFFIOLI (1984):
The complexity of weighted multi-constrained spanning tree problems.
Colloquium on the Theory of Algorithms,
Pecs, July 23 - 27, 1984.

S. CHUNG; H.W. HAMACHER; F. MAFFIOLI; K.G. MURTY (1990):
A note on combinatorial optimization with max-linear objective functions.
Universität Kaiserslautern, Fachbereich Mathematik, Preprint No. 189.
Accepted for Discrete Applied Mathematics.

H.W. CORLEY (1985):
Efficient spanning trees.
JOTA 45, 481 - 485.

Z. DREZNER; S.Y. NOF (1984):
On optimizing bin packing and insertion plans for assembly robots.
IEE Transactions 16, 262 - 270.

H.N. GABOW (1977):
Two algorithms for generating weighted spanning trees in order.
SIAM Journal on Computing 6, 139 - 150.

A.M. GEOFFRION (1968):
Proper efficiency and the theory of vector optimization.
Journal of Mathematical Analysis and Applications 22, 618 - 630.

F. GLOVER; D. KLINGMAN; R. KRISHNAN; R. PADMAN (1992):
An in-depth empirical investigation of non-greedy approaches for the minimum spanning tree problem.
EJOR 56, 343 - 356.

R.E. GOMORY; T.C. HU (1961):
Multiterminal network flows.
SIAM Journal of Applied Mathematics 9, 551 - 571.

H.W. HAMACHER; M. QUEYRANNE (1985):
K best solutions to combinatorial optimization problems.
Annals of Operations Research 4, 123 - 143.

M. HELD; R.M. KARP (1970):
The traveling salesman problem and minimum spanning trees.
Operations Research 18, 1138 - 1162.

N. KATOH; T. IBARAKI; H. MINE (1981):
An algorithm for finding k minimum spanning trees.
SIAM Journal on Computing 10, 247 - 255.

J.B. KRUSKAL (1956):
On the shortes spanning subtree of a graph and the traveling salesman problem.
Proceedings of the American Mathematical Society 7, 48 - 50.

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

S. NICKEL (1992):
Private communication.

R.C. PRIM (1957):
Shortes connection networks and some generalizations.
The Bell System Technical Journal 36/6, 1389 - 1401.

G. RIGHINI; A. COLORNI (1991):
Max-linear combinatorial optimization.
Politecnico di Milano, Dipartimento d. Elettronica, report n. 91.005.