Journal of Global Optimization **19:** 103–120, 2001.
© 2001 *Kluwer Academic Publishers. Printed in the Netherlands.*

Minimum Spanning Trees with Sums of Ratios

CHRISTOPHER C. SKISCIM[1] and SUSAN W. PALOCSAY[2]
[1]*Megisto Systems, Inc., Dickerson, Maryland 20842, USA (e-mail: cskiscim@megisto.com)*
[2]*Computer Information Systems/Operations Management Program, James Madison University, Harrisonburg, Virginia, 22087, USA (e-mail: palocssw@jmu.edu)*

(Received 10 December 1998; accepted in revised form 21 June 2000)

Abstract. We present an algorithm for finding a minimum spanning tree where the costs are the sum of two linear ratios. We show how upper and lower bounds may be quickly generated. By associating each ratio value with a new variable in 'image space,' we show how to tighten these bounds by optimally solving a sequence of constrained minimum spanning tree problems. The resulting iterative algorithm then finds the globally optimal solution. Two procedures are presented to speed up the basic algorithm. One relies on the structure of the problem to find a locally optimal solution while the other is independent of the problem structure. Both are shown to be effective in reducing the computational effort. Numerical results are presented.

Key words: Fractional programming, Sums of ratios, Minimum spanning tree, Combinatorial optimization

1. Introduction

The Minimum Spanning Tree problem is one of the fundamental problems in combinatorial optimization. It is the problem of finding a tree in a network that connects every pair of nodes by one and only one elementary path such that some cost is minimized. The applications are well known (see Lawler (1976) and references therein).

Under a linear cost model, individual costs are imposed on every edge connecting a pair of nodes and the cost of the tree is simply the sum of its $n - 1$ edge costs where n is the number of nodes in the network. Under this cost model, the greedy algorithm finds the MST in polynomial time. Camerini et al. (1988) review several efficient implementations of the greedy algorithm. If, in addition to costs, there are also weights associated with each edge, then we are faced with the problem of finding a spanning tree that minimizes the cost of the tree divided by its weight. This problem, termed the Minimum Ratio Spanning Tree (MRST) problem, was first considered by Chandrasekaran (1977) who presented a polynomial-time algorithm for its solution.

Extending this idea, Meggido (1979) showed that if a combinatorial optimization problem admitted a polynomial-time algorithm under a linear cost model, then the minimum cost to weight ratio problem could also be solved in polynomial time.

Hashizume et al. (1987) applied Megiddo's results to hard combinatorial problems. They demonstrated that if the original (linear) problem had an ϵ-approximate algorithm, then the cost to weight ratio problem could also be solved to within the same accuracy. Later, Radzik (1992) showed that Newton's Method applied to linear combinatorial optimization problems with a single ratio was strongly polynomial when the underlying linear problem admitted a polynomial time algorithm. More recently, Nagih (1996) investigated the use of Lagrangian decomposition in solving general $[0, 1]$ cost-to-weight ratio problems with a specific application to the knapsack problem.

In this paper, we generalize the MRST by considering an objective function that is the sum of two ratios. More precisely, if we let $n_i(x)$ and $d_i(x)$, for $i = 1, 2$, be the linear functions, then we wish to find a spanning tree, x that minimizes $z = \sum_{i=1}^{2} n_i(x)/d_i(x)$. We term this problem the *Two Ratio Minimum Spanning Tree* (TRMST) problem.

The applications of this type of cost model are detailed in Schaible (1995) and his extensive bibliography. Obviously, such a cost model arises when several rates are to be simultaneously optimized. For example, Almogy and Levin (1969) used multiple ratios in formulating a multistage stochastic optimization shipping problem with a generalization given in Almogy and Levin (1971). Multiple rates can also be used in a weighted average or when a compromise between two rates is sought. For instance, if we let $d_2(x) = 1$ then we are trading off relative and absolute quantities. In this case it would be useful to consider a cost model such as $z = n_1(x)/d_1(x) + \alpha n_1(x)$ with $\alpha \neq 0$. The current state of the art in this area is reviewed in Schaible (1996).

For the TRMST, we make use of the algorithm developed by Falk and Palocsay (1992) to solve our problem. We develop their method within the context of combinatorial problems. We show how to solve the subproblems used to bound the optimal solution and develop an iterative algorithm for solving this problem. In addition, we also present a new method for overcoming the problem of stalling. Finally, computational results are presented.

Despite the success of solving combinatorial problems with a single ratio objective function, instances where two (or more) ratios form the objective function have not been addressed. We believe that the minimum spanning tree problem with more than one ratio in the objective function is at least NP-complete. We base this belief on the fact that we require the exact solution to NP-hard subproblems to improve upper and lower bounds on the problem. This we will show. Unlike the single ratio version of the problem, there appears to be no necessary and sufficient condition for optimality although a sufficient condition does exist. We do not address the complexity issue here, but will explore this subject in depth in a later paper.

The paper is organized as follows. Section 2 presents the notation and introduces the minimal ratio spanning tree problem and the extension to two ratios. We also show how upper and lower bounds can be quickly generated as well as presenting an obvious heuristic for finding a local optimum. Section 3 shows how these bounds can be improved. The algorithm is detailed in Section 4 and applied to an example problem. Computational results are then presented. Section 5 concludes the paper.

2. Preliminaries

The MRST problem is a network optimization problem defined on an undirected network $\mathcal{G}(\mathcal{V}, \mathcal{A})$ where $\mathcal{V}$ denotes the set of vertices or nodes and $\mathcal{A}$ denotes the set of edges or arcs connecting the nodes. The node set is indexed as $\mathcal{V} = (1, \ldots, n)$ and each of the $m = |\mathcal{A}|$ edges is represented by a pair of nodes (i, j). Let $\mathcal{T}$ denote the set of spanning trees in the network. Our decision variables are $x_{ij} = 1$ if (i, j) is an edge of a particular tree and zero otherwise. The decision variables will be represented by a tree's incidence vector $x \equiv \{ x_{ij} = 1 \mid (i, j) \in \tau_x \}$ where $\tau_x \in \mathcal{T}$ is the tree corresponding to x.

2.1. THE MINIMUM RATIO SPANNING TREE PROBLEM

For each edge (i, j) in the network, we have a cost a_{ij}, and a weight b_{ij}. The MRST is then defined as

$$\min_{\tau_x \in \mathcal{T}} \frac{n(x)}{d(x)} = \frac{\sum_{(i,j)\in\tau_x} a_{ij}x_{ij}}{\sum_{(i,j)\in\tau_x} b_{ij}x_{ij}}. \tag{1}$$

We assume that $\sum_{(i,j)\in\tau_x} b_{ij}x_{ij} > 0$ for all $\tau_x \in \mathcal{T}$ and $\sum_{(i,j)\in\tau_x} a_{ij}x_{ij} > 0$ for some $\tau_x \in \mathcal{T}$. As shown by Chandrasekaran (1977), when the denominator of the MRST objective function is allowed to take on negative values, the problem is NP-complete. One commonly used approach to solving ratio optimization problems is to define a parametric version of the problem,

$$\min_{\tau_x \in \mathcal{T}} [\, n(x) - \delta d(x) \,], \tag{2}$$

then pick a real number $\overline{\delta}$ and evaluate

$$H(\overline{\delta}) = \min\Big\{ \big[\, n(x) - \overline{\delta} d(x) \big] \;\Big|\; \tau_x \in \mathcal{T} \Big\}. \tag{3}$$

For a given $\overline{\delta}$, this amounts to finding a minimal spanning tree $\overline{x}$ with edge costs $n(x) - \overline{\delta} d(x)$. By a theorem due to Dinklebach (1967), if $H(\overline{\delta}) = 0$ for some $\tau_{\overline{x}} \in \mathcal{T}$, then $r^* = \overline{\delta}$ is the optimal value of the ratio objective function in the MRST and $x^* = \overline{x}$ is the optimal solution to the MRST. Newton's method, outlined in Figure 1 is used for solving the MRST. Radzik (1992) analyzes this algorithm and shows that it requires $O(m^2 \log^2 m)$ iterations to find the optimum.

```
Algorithm N:
begin:
  $\overline{\delta} \leftarrow 0$;
  repeat
    $H(\overline{\delta}) \leftarrow \min_{\tau_x \in \mathcal{T}} \left[ n(x) - \overline{\delta} d(x) \right]$;
    $\overline{\delta} \leftarrow n(\overline{x})/d(\overline{x})$;
  until $H(\overline{\delta}) = 0$;
  $r^* \leftarrow \overline{\delta}$;
  $x^* \leftarrow \overline{x}$;
end;
```

Figure 1. Algorithm N: Newton's algorithm for the minimum ratio spanning tree problem.

2.2. MINIMUM SPANNING TREES WITH TWO RATIOS

In considering the problem with two ratios in the objective function, we introduce an additional cost and weight assigned to each edge, c_{ij} and d_{ij}. Again, we assume that $\sum_{(i,j)\in\tau_x} d_{ij}x_{ij} > 0$ for all $\tau_x \in \mathcal{T}$, and $\sum_{(i,j)\in\tau_x} c_{ij}x_{ij} > 0$ for some $\tau_x \in \mathcal{T}$. The Two Ratio Minimum Spanning Tree problem is defined as

$$\min_{\tau_x \in \mathcal{T}} \left(\frac{n_1(x)}{d_1(x)} + \frac{n_2(x)}{d_2(x)} \right) \tag{4}$$

$$= \min_{\tau_x \in \mathcal{T}} \left(\frac{\sum_{(i,j)\in\tau_x} a_{ij}x_{ij}}{\sum_{(i,j)\in\tau_x} b_{ij}x_{ij}} + \frac{\sum_{(i,j)\in\tau_x} c_{ij}x_{ij}}{\sum_{(i,j)\in\tau_x} d_{ij}x_{ij}} \right). \tag{5}$$

To solve this problem it seems natural to form a parametric function to optimize similar to the single ratio case, i.e.,

$$H(r) = \min_{\tau_x \in \mathcal{T}} \sum_{i=1}^{2} \left(n_i(x) - r_i d_i(x) \right) \tag{6}$$

where $r = (r_1, r_2)$ are the values of the two ratios. The challenge in using the parametric form (6) is to find necessary and sufficient conditions for optimality perhaps relying only on the value of the parametric equation, as attempted by Almogy and Levin, or to supplement the parametric equation with additional information about the solution space, as did Falk and Palocsay.

In the former case, it was posited that x^* is the optimal solution of, in this case the TRMST, if and only if $r_i = n_i(x^*)/d_i(x^*)$, for $i = 1, 2$, solves $H(r) = 0$. By way of a counter-example presented by Falk and Palocsay, we know this is not necessarily true. In the latter case, a sufficient condition for establishing whether a solution satisfying $H(r) = 0$ is optimal was found in Falk and Palocsay (1992) and relies on knowledge of the surrounding solution space. Thus, the function $H(r)$ cannot be used to guide the optimization to a globally optimal solution, but it can be used to check a solution for optimality if lower and upper bounds are available.

In the next section we show how to quickly generate upper and lower bounds on the optimal value of the TRMST. Subsequently, we show how Lagrangian relaxation can be used to construct subproblems that tighten these bounds.

2.3. EASY BOUNDS AND A HEURISTIC FOR THE TRMST

Consider finding the MRST with respect to some ratio i, for $i = 1, 2$ in the TRMST. That is, find

$$u_{ii} = \min_{\tau_x \in \mathcal{T}} \frac{n_i(x)}{d_i(x)}, \text{ for } i = 1,2. \tag{7}$$

The value u_{ii} can be no larger than its associated ratio's value in the optimal solution of the TRMST. Thus, we have

$$f_{\text{lo}} = \sum_{i=1}^{2} u_{ii} \leqslant \min_{\tau_x \in \mathcal{T}} \sum_{i=1}^{2} \left(\frac{n_i(x)}{d_i(x)} \right) \tag{8}$$

and f_{lo} is a lower bound on the optimal value of the TRMST. If τ_x^* is an optimal solution of the TRMST, then we also have

$$u_{ii} \leqslant \frac{n_i(x^*)}{d_i(x^*)}, \quad \text{for } i = 1,2. \tag{9}$$

For each solution of (7) we also recover a feasible solution, x^i, from which we compute upper bounds

$$f_{\text{up}}^i = u_{i1} + u_{i2} = \sum_{j=1}^{2} \left(\frac{n_j(x^i)}{d_j(x^i)} \right), \text{ for } i = 1,2. \tag{10}$$

For the best upper bound, we select

$$f_{\text{up}} = \min \left[f_{\text{up}}^1, f_{\text{up}}^2 \right] \tag{11}$$

and record the tree associated with f_{up} as T_{up}.

A simple exchange heuristic can be used in hopes of improving f_{up}. Consider the list of edges not in T_{up}. Repeatedly, swap each of these $m - n + 1$ edges into T_{up}. This creates a cycle of at most n edges. Break this cycle by choosing an edge to remove that yields the greatest reduction in the value of f_{up} and record the new value of f_{up} and its associated tree as T_{up}. If no such edge exists, the swapped-in edge is removed and the procedure continues. There are $O(m)$ potential replacement edges for each of the $n - 1$ edges in T_{up}. Thus, the exchange heuristic takes $O\left(m(n-1)\right)$ time. We call this Heuristic $\mathcal{S}$.

3. Improved Bounds

If the previously computed upper and lower bounds coincide, then we know the solution associated with the upper bound is an optimal solution to the TRMST. More than likely, this will not be the case and a method is needed to improve the bounds. If we can generate a sequence of non-decreasing lower bounds and non-increasing upper bounds then we can drive to the optimal solution.

To develop this idea, suppose we have an optimal solution to the TRMST denoted by x^* with $r_1^* = n_1(x^*)/d_1(x^*)$ and $r_2^* = n_2(x^*)/d_2(x^*)$. Clearly, x^* is an optimal solution to the TRMST if and only if it is also the optimal solution to one of

$$\begin{array}{ll} \min_{\tau_x \in \mathcal{T}} & n_i(x)/d_i(x) \\ \text{s.t.} & n_j(x)/d_j(x) = r_j^* \\ & (i, j) \in [1, 2] \text{ and } i \neq j. \end{array} \tag{12}$$

Of course, we do not know the optimal solution to the TRMST. But suppose we relax the equality constraint in (12) to an inequality and replace its right hand side by an upper bound $\hat{r}_j \geqslant r_j^*$. Then the solution to

$$\begin{array}{ll} \min_{\tau_x \in \mathcal{T}} & n_i(x)/d_i(x) \\ \text{s.t.} & n_j(x)/d_j(x) \leqslant \hat{r}_j \\ & (i, j) \in [1, 2] \text{ and } i \neq j \end{array} \tag{13}$$

will yield a lower bound solution $\hat{x}^i$ such that

$$u_{ii} \leqslant n_i(\hat{x}^i)/d_i(\hat{x}^i) \leqslant r_i^* \quad \text{for } i = 1, 2. \tag{14}$$

Indeed, we can view (7) as the problem (13) with the right hand side set to $+\infty$. Because (13) does not exclude the optimal point in the TRMST and $\hat{x}^i$ is feasible to the TRMST, we have

$$r_1^* + r_2^* \leqslant \frac{n_1(\hat{x}^i)}{d_1(\hat{x}^i)} + \frac{n_2(\hat{x}^i)}{d_2(\hat{x}^i)} \leqslant f_{\text{up}} \quad \text{for } i = 1, 2. \tag{15}$$

Regrettably, the problem (13) is NP-hard (Aggarwal et al., 1982). Nevertheless, algorithms for solving these types of problems when the objective function is linear are well known (Handler and Zang, 1980; Aggarwal et al., 1982) and quite successful in practice. Our strategy is to fix the value of one ratio at a time and solve the resulting constrained problems to optimality.

Thus, we end up with two subproblems to solve:

$$\begin{array}{ll} p_1: & \min_{\tau_x \in \mathcal{T}} n_1(x)/d_1(x) \\ \text{s.t.}: & n_2(x)/d_2(x) \leqslant r_2 \end{array} \quad \text{and} \quad \begin{array}{ll} p_2: & \min_{\tau_x \in \mathcal{T}} n_2(x)/d_2(x) \\ \text{s.t}: & n_1(x)/d_1(x) \leqslant r_1 \end{array} \tag{16}$$

where r_1 and r_2 are fixed ahead of time at appropriate values. Upon solution, these two problems yield lower bounds on ratio 1 and ratio 2 which are not inferior to those given by (7) and their sum, a lower bound on the optimal solution of the TRMST. Denoting the solution of these subproblems by $x^1$and x^2 respectively, a potentially improved upper bound on the TRMST is

$$f_{\text{up}} = \min\left[\sum_{i=1}^{2} \frac{n_i(x^1)}{d_i(x^1)}, \sum_{i=1}^{2} \frac{n_i(x^2)}{d_i(x^2)}\right] \tag{17}$$

and a potentially improved lower bound on the TRMST is

$$f_{\text{lo}} = n_1(x^1)/d_1(x^1) + n_2(x^2)/d_2(x^2). \tag{18}$$

Clearly, if $f_{\text{up}} = f_{\text{lo}}$, the current solution is optimal.

We now need a systematic way of setting the right hand side values in the subproblems such that optimal solution point is not excluded yet we narrow the search space. We address this topic in the next section using the ideas from Falk and Palocsay (1992) and their image space algorithm to facilitate this process.

3.1. SETTING THE PARAMETERS

Suppose we enumerated all the spanning trees on $\mathcal{G}$ and recorded all the values for r_1 and r_2 thus forming a set $\mathcal{R}$. Formally, we have

$$\mathcal{R} = \left\{ (r_1, r_2) \mid r_1 = \frac{n_1(x)}{d_1(x)}, r_2 = \frac{n_2(x)}{d_2(x)}; \forall \tau_x \in \mathcal{T} \right\}. \tag{19}$$

Thus, each spanning tree τ_x on $\mathcal{G}$ maps non-uniquely to one point $r = (r_1, r_2)$. By plotting r_1 against r_2, we have a solution space in which the linear objective function $r_1 + r_2$ can be minimized. In essence, we wish to solve the optimization problem

$$\begin{array}{ll} \min_{\tau_x \in \mathcal{T}} & r_1 + r_2 \\ \text{s.t. :} & \begin{array}{l} n_1(x)/d_1(x) \leqslant r_1 \\ n_2(x)/d_2(x) \leqslant r_2 \end{array} \end{array} \tag{20}$$

by iteratively descending along each axis of $\mathcal{R}$ using the subproblems (16). How far we descend at each step is, in part, a function of right hand sides of the subproblems (16) and these must be chosen so that the subproblems are feasible and do not exclude the optimal solution to the TRMST. This is accomplished by constructing a triangle in $\mathcal{R}$ within which the optimal solution point must lie. The vertices of this triangle give us a way to iteratively set the right hand sides of (16) and thus improve the bounds.

To construct this triangle, start at the point corresponding to the initial lower bound (u_{11}, u_{22}) and draw a vertical line through the point u_{11}, followed by a horizontal line through the point u_{22}. The isovalue-contour $r_1 + r_2 = f_{\text{up}}$ intersects these two lines yielding a triangle in $\mathcal{R}$. The line $f_{\text{up}} = r_1 + r_2$ intersects the vertical line at the point

$$(l_1, l_2) = (u_{11}, f_{\text{up}} - u_{11}) \tag{21}$$

and the horizontal line at the point

$$(v_1, v_2) = (f_{\text{up}} - u_{22}, u_{22}). \tag{22}$$

By construction, the resulting triangular region is guaranteed to contain the optimal solution, and either v or l is feasible.

The feasible solution corresponding to f_{up} forms one acute vertex of the triangle. The other acute vertex is generally not feasible but we will use this point to set the right hand side in the constrained problems. In particular, choose

$$r_1 = v_1 = f_{\text{up}} - u_{22} \quad \text{and} \quad r_2 = l_2 = f_{\text{up}} - u_{11}. \tag{23}$$

To see how this works, assume without loss of generality, that the current upper bound is ratio 1, that is, $u_{11} + u_{12} < u_{21} + u_{22}$. This means that problem p_1 is immediately solved since l_1 was generated by minimizing the first ratio. Solving problem p_2 excludes the point (u_{21}, u_{22}) and so we must have

$$u_{22} \leqslant z^*(p_2) \leqslant r_2^*. \tag{24}$$

Thus, the lower bound on ratio 2 (and hence, on the objective function of the TRMST) does not deteriorate and in general, the search space is narrowed. Note that the upper bound may also improve. Figure 2 illustrates these ideas.

Given the three points of the initial triangle, we can check whether or not the feasible point is optimal by invoking the sufficient condition given in Falk and Palocsay (1992). Informally, the condition is as follows: Without loss of generality, let $l = (u_{11}, u_{12})$ be our feasible solution so that $f_{\text{up}} = u_{11} + u_{12}$. The lower bound is the point $u = (u_{11}, u_{22})$ with $f_{\text{lo}} = u_{11} + u_{22}$. The point v is as previously defined. Then if $H(l) = 0$ and $H(u) > 0$ and $H(v) > 0$, we know that $r^* = l = (u_{11}, u_{12})$ is the optimal solution since it is the only feasible point in the triangle (due to the concavity of H).

3.2. SOLVING THE CONSTRAINED SUBPROBLEMS

As proved in Aggarwal et al. (1982), the MST problem with a single side constraint is NP-hard, hence in theory, difficult to solve. We wish to solve the subproblems using Lagrangian relaxation and this requires us to form the Lagrangian Dual for the subproblems (16). The use of this strategy here is justified by the results of Bitran and Magnanti (1976).

Bitran and Magnanti considered the following parametric version of an arbitrary fractional program:

$$\min\big[\, n(x) - \delta d(x) \;\mid\; x \in F \,\big] \tag{25}$$

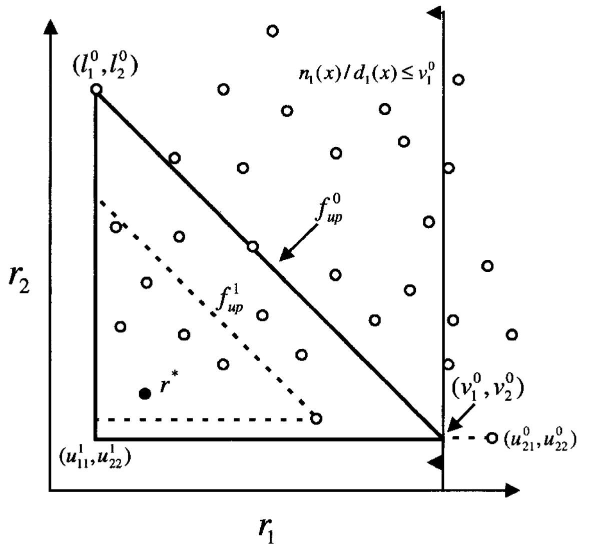

Figure 2. Bounding the solution space. Dashed triangle shows the narrowed search space resulting from solving a constrained subproblem. Superscripts indicate successive iterations.

where $F = \{x \in \mathbb{X} \subseteq \mathbb{R}^m, g(x) \leqslant 0\}$. This included the case where $n(x)$, $d(x)$ are linear, F is polyhedral and $g(x)$ is a real–valued function defined on $\mathbb{R}^m$. Their results guarantee the existence of a dual variable λ such that

$$\min_{x \in F}\Big[\, n(x) - \delta d(x) \,\Big] = \max_{\lambda \geqslant 0} \min_{x \in \mathbb{X}}\Big[n(x) - \delta d(x) + \lambda g(x)\Big] \tag{26}$$

when, for instance, (25) is a linear program and $d(x) > 0$. Furthermore, their analysis does not assume differentiability, thus it is valid for combinatorial problems.

For our subproblems (16), we consider the ratio constraints in the form $g(x) = n_i(x) - r_i d_i(x)$ for a fixed value of r_i. Then, for example, the Lagrangian dual for p_1 is

$$\max_{\lambda \geqslant 0} \min_{\tau_x \in \mathcal{T}}\Big[n_1(x) - \delta d_1(x) + \lambda\big(n_2(x) - r_2 d_2(x)\big)\Big]. \tag{27}$$

Thus for some fixed $\lambda \geqslant 0$, we need to solve

$$L_{p_1}(\lambda) = \min_{\tau_x \in \mathcal{T}} \left[(n_1(x) + \lambda\,(n_2(x) - r_2 d_2(x))) - \delta d_1(x)\right]. \tag{28}$$

This is the parametric version of the MRST problem with new problem coefficients which we solve over δ using Newton's method. By the duality results in Bitran and Magnanti (1976), we know $L_{p_1}(\lambda) \leqslant z^*(p_1)$ and the best lower bound is given by $\mathcal{L} = \max_{\lambda \geqslant 0} L_{p_1}(\lambda)$.

To maximize the Lagrangian dual, we adapted the Handler–Zang algorithm which was originally used to solve the constrained shortest path problem. We substituted solution of the MRST problem based on (28) for each of p_1 and p_2 where they required solution of a shortest path problem.

Briefly, their algorithm relies on Lagrangian relaxation of the side constraint to solve a series of, in this case, MRST problems. Beginning with an estimate of the optimal multiplier, the Handler–Zang procedure iteratively builds up the lower envelope of the concave Lagrangian dual objective function until this function is maximized. If a duality gap existed, they employed an enumeration procedure which produced non-decreasing solution values with respect to the Lagrangian objective function, thus closing the duality gap. We used the simple branch and bound procedure of Aggarwal et al. (1982) to close any existing duality gap.

3.3. STALLING

In general, the algorithm can stall. At some iteration i, we could have v^i and l^i both feasible and lying on the same isovalue contour $f^i_{\rm up}$ thus causing the procedure to cycle endlessly between these two points. What is required is to find a improved upper bound so the iterations can progress.

Fortunately we can take advantage of the problem structure and invoke the exchange heuristic outlined in Section 2.3 in the hopes of finding a better solution. This heuristic only needs to run until an improved solution is found and, while there is no guarantee that such a solution will be found, this approach worked well in our experiments. If the heuristic step fails to produce a better upper bound, an alternate method is available.

A general approach for re-starting the procedure with a new solution is explained in Falk and Palocsay (1992). Their approach gives rise to subproblems with a side constraint having upper *and* lower bounds. These are hard constraints in the combinatorial setting. Instead, we propose an alternate strategy that closely resembles their approach but only requires subproblems with a single side constraint.

Consider a point $r^- = (r_1^-, r_2^-)$ lying strictly within the current triangle. This has an upper bound $f^-_{\rm up} < f^i_{\rm up} = (v^i_1 + v^i_2) = (l^i_1 + l^i_2)$. Consequently, $r_1^- > l^i_1$ implies that $r_2^- < l^i_2$, and $r_2^- > v^i_2$ implies that $r_1^- < v^i_1$. This suggests that systematic search of the interior of the current triangle will uncover this point if it exists thus allowing the better upper bound to be used in restarting the procedure. Our approach is to subdivide the rectangle defined by the three vertices of the current triangle and the unlabeled point (v^i_1, l^i_2) along each of the axes of $\mathcal{R}$, as follows.

Consider the two subproblems

$$p_1^s : \min_{\tau_x \in \mathcal{T}} \; n_1(x)/d_1(x) \quad \text{s.t.} \quad n_2(x)/d_2(x) \leqslant t_2 \qquad \text{and} \qquad p_2^s : \min_{\tau_x \in \mathcal{T}} \; n_2(x)/d_2(x) \quad \text{s.t.} \quad n_1(x)/d_1(x) \leqslant t_1$$

with $t_1 = \frac{1}{2}\left(u^i_{11} + v^i_1\right)$and $t_2 = \frac{1}{2}\left(u^i_{22} + l^i_2\right)$. The constraints are illustrated in Figure 3.

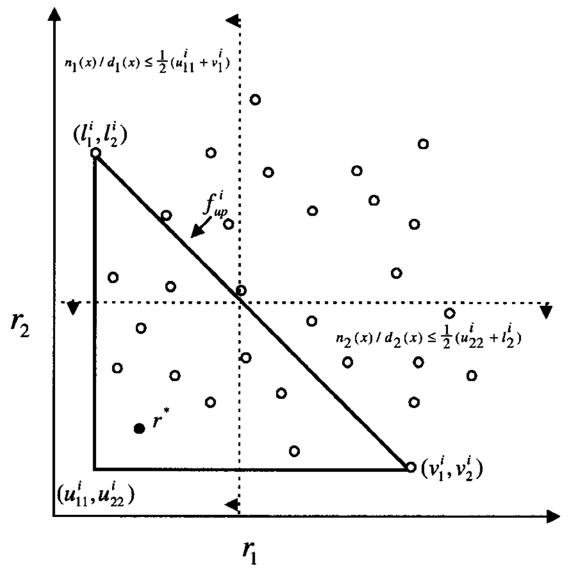
$n_1(x)/d_1(x) \leq \frac{1}{2}(u^i_{11} + v^i_1)$
(l^i_1, l^i_2)
f^i_{up}
r_2
$n_2(x)/d_2(x) \leq \frac{1}{2}(u^i_{22} + l^i_2)$
r^*
(v^i_1, v^i_2)
(u^i_{11}, u^i_{22})
r_1

Figure 3. Subdividing the current triangle to overcome stalling.

Note that a feasible solution to each of these subproblems exists. Because there may be points outside the current triangle which are feasible to these subproblems, it is likely that the subproblems will solve at those points. Consider a point outside the current triangle $r^+ = (r_1^+, r_2^+)$ and subproblem p_2^s. If $r_2^+ < r_2^-$, then p_2^s solves at this point with $(r_1^+ + r_2^+) = f_{\text{up}}^+ > f_{\text{up}}^i$. But if a point r^- exists, then we must have $r_1^- < r_1^+$ in which case p_1^s will find it eventually. An analogous situation holds in the case of subproblem p_1^s.

If, at the current subdivision, the solution to both these subproblems fails to produce an improved upper bound, then we can conclude that there are no feasible points within the intersection of $n_2(x)/d_2(x) \leqslant t_2$ and $n_1(x)/d_1(x) \leqslant t_1$. This further allows us to conclude that a lower bound lies along the iso-value contour defined by $t_1 + t_2$, thus we can take $t = (t_1 + t_2)$ as a lower bound. Consequently, the current solution must be within $\epsilon = (f_{\text{up}}^i - t)/t$ of being optimal. So, we continue the subdivision by moving both constraints half again the remaining distance toward the points v^i and l^i, respectively and solve the subproblems with the new constraints. This process continues until we find an improved upper bound or until the remaining gap falls below some pre-specified tolerance ϵ in which case we stop with an ϵ–approximate solution.

If we recover an improved upper bound, we establish a new triangle using this upper bound and the existing lower bound (u_{11}^i, u_{22}^i). The acute vertices of this triangle will, in all likelihood, not correspond to a feasible solution. To re-establish a feasible solution as a vertex of the triangle, we recompute the points l^i and v^i based on the new upper bound and solve the subproblems p_1 and p_2. We then take the minimum of the upper bounds resulting from their solution as the point at which to continue the iterations.

4. An Algorithm for the Two Ratio MST Problem

Putting the previous ideas together, we can now state the algorithm for finding an MST when the objective function is the sum of two linear ratios. We start with an initial upper and lower bound by first solving an (unconstrained) MRST for ratio 1 and ratio 2. Then we iteratively solve constrained MRST problems to narrow the bounds checking for optimality using the sufficiency condition described in Section 3.1 if $f_{\mathrm{up}}^i > f_{\mathrm{lo}}^i$, or terminating when the bounds coincide. The superscript i is used to index the iterations.

ALGORITHM 2-RATIO-MST:

Step 1. Set $i \leftarrow 1$ and α a predefined tolerance such that $\alpha < 1$. Find $u^i = (u_{11}^i, u_{22}^i)$ using Algorithm **N** from Figure 1. Compute f_{lo}^i and f_{up}^i using expressions (9) and (10). If $f_{\mathrm{lo}}^i = f_{\mathrm{up}}^i$ then stop, the optimal solution is $r^* = f_{\mathrm{up}}^i$ with the optimal spanning tree given by that associated f_{up}^i. Otherwise, set

$$\begin{aligned} l^i &= \left(l_1^i, l_2^i\right) \leftarrow \left(u_{11}^i, f_{\mathrm{up}}^i - u_{11}^i\right) \\ v^i &= \left(v_1^i, v_2^i\right) \leftarrow \left(f_{\mathrm{up}}^i - u_{22}^i, u_{22}^i\right) \end{aligned}$$

and proceed to Step 2.

Step 2. If $f_{\mathrm{lo}}^i > f_{\mathrm{lo}}^{i-1}$ or $f_{\mathrm{up}}^i < f_{\mathrm{up}}^{i-1}$ then proceed to Step 3; Otherwise, go to Step 6.

Step 3. Compute $H(u^i)$, $H(l^i)$ and $H(v^i)$. If $H(v^i) = 0$ and $H(l^i) > 0$ and $H(u^i) > 0$, then stop with $r^* = v^i$. Else, if $H(v^i) > 0$ and $H(l^i) = 0$ and $H(u^i) > 0$, then stop with $r^* = l^i$. The optimal objective function value is $r^* = r_1^* + r_2^*$, with the incidence vector x^* such that

$$\frac{n_1(x^*)}{d_1(x^*)} = r_1^* \quad \text{and} \quad \frac{n_2(x^*)}{d_2(x^*)} = r_2^*.$$

Otherwise, proceed to Step 4.

Step 4. Set $i \leftarrow i+1$. Solve the following constrained MRST problems optimally:

$$\begin{array}{ll} u_{11}^i & \leftarrow \min_{\tau_x \in \mathcal{T}} n_1(x)/d_1(x) \\ \text{s.t.} & n_2(x)/d_2(x) \leqslant l_2^{i-1} \end{array} \quad \text{and} \quad \begin{array}{ll} u_{22}^i & \leftarrow \min_{\tau_x \in \mathcal{T}} n_2(x)/d_2(x) \\ \text{s.t.} & n_1(x)/d_1(x) \leqslant v_1^{i-1} \end{array}$$

recalling that one of these is already solved. Set

$$\begin{aligned} f_{\rm lo}^i &\leftarrow u_{11}^i + u_{22}^i \\ f_{\rm up}^k &\leftarrow \min\left[u_{11}^i + u_{12}^i, u_{21}^i + u_{22}^i\right] \quad \text{(cf. eqs (9) and (10))} \\ l^i &\leftarrow \left(u_{11}^i, f_{\rm up}^i - u_{11}^i\right) \\ v^i &\leftarrow \left(f_{\rm up}^i - u_{22}^i, u_{22}^i\right). \end{aligned}$$

Step 5. If $f_{\rm lo}^i = f_{\rm up}^i$ then stop, the optimal solution is $r^* = f_{\rm up}^i$ with the optimal spanning tree given by that associated $f_{\rm up}^i$. Otherwise, return to Step 2.

Step 6. The algorithm has stalled. Execute the exchange procedure (cf. Section 2.3) terminating with an upper bound z. If $z < f_{\rm up}^i$ then set $f_{\rm up}^i \leftarrow z$ and go to Step 2, otherwise go to Step 7.

Step 7. The algorithm remains stalled. Set

$$\begin{aligned} \widetilde{u}_{11} &\leftarrow u_{11}^i \\ \widetilde{u}_{22} &\leftarrow u_{22}^i \\ t_1 &\leftarrow \frac{1}{2}\left(\widetilde{u}_{11} + v_1^i\right) \\ t_2 &\leftarrow \frac{1}{2}\left(\widetilde{u}_{22} + l_2^i\right) \end{aligned}$$

and solve the two constrained MRST problems of Step 4 with right-hand sides t_1 and t_2 to yield an upper bound $f_{\rm up}^s$. If $f_{\rm up}^s < f_{\rm up}^i$ then return to Step 2. If $\epsilon > \alpha$ (cf. Section 3), then set

$$\begin{aligned} t_1 &\leftarrow \frac{1}{2}(t_1 + v_1^i) \\ t_2 &\leftarrow \frac{1}{2}(t_2 + l_2^i) \end{aligned}$$

and repeat this step; otherwise stop with the approximate solution $f_{\rm up}^s$.

Note that *any* upper bound will suffice at the start of the algorithm. In particular, the simple exchange procedure of Section 2.3 could be used to improve the upper bound of Step 2 and decrease the number of constrained MST problems requiring solution.

4.1. EXAMPLE PROBLEM

The following problem will illustrate the algorithm and was adapted from the example network in Handler and Zang and is shown in Figure 4. The cost and weights for each edge were chosen such that the first ratio was negatively correlated with the second ratio. This was intended to produce initial upper and lower bounds that were far apart.

To begin, the MRSTs with respect to ratio 1 and ratio 2 are found yielding two feasible solutions:

$$\begin{aligned} (u_{11}^1, u_{12}^1) &= (3.56, 28.20) \\ (u_{21}^1, u_{22}^1) &= (24.74, 3.46) \end{aligned}$$

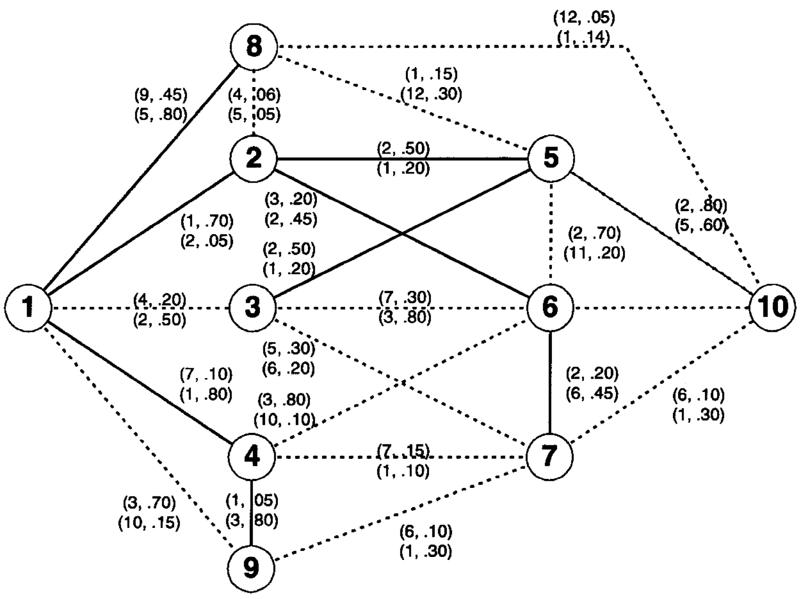

Figure 4. Example 10-node network. Bold lines depict the optimal tree.

with an initial upper bound $f^1_{\rm up} = \min[31.76, 28.20] = 28.20$ and an initial lower bound of $f^1_{\rm lo} = 7.02$. The sufficient condition for optimality is not satisfied and the lower and upper bounds are not equal, so the algorithm continues. The vertices of the triangle are the points

$$(u^1_{11}, u^1_{22}) = (3.56, 3.46)$$
$$(l^1_1, l^1_2) = (3.56, 24.63)$$
$$(v^1_1, v^1_2) = (24.74, 3.46).$$

Note that problem $p_2$does not need to be solved since v is feasible. Problem p_1, the constrained MRST, with right-hand side 24.63 is solved yielding a new upper bound $f^1_{\rm up} = 21.22$ and a new lower bound of $f^0_{\rm lo} = 7.09$. The vertices of the triangle are now

$$(u^2_{11}, u^2_{22}) = (3.63, 3.46)$$
$$(l^2_1, l^2_2) = (3.63, 17.58)$$
$$(v^2_1, v^2_2) = (17.75, 3.46).$$

The algorithm continues in this fashion (without stalling) and terminates after 24 complete iterations of the algorithm with the globally optimal solution (8.286, 5.977). Figure 5 shows the convergence of the procedure in $\mathcal{R}$-space. For clarity, the entire $\mathcal{R}$-space is not enumerated and only a subset of the iterations are represented. The largest and smallest triangles depict the initial upper bound and the final triangle containing the single optimal solution point.

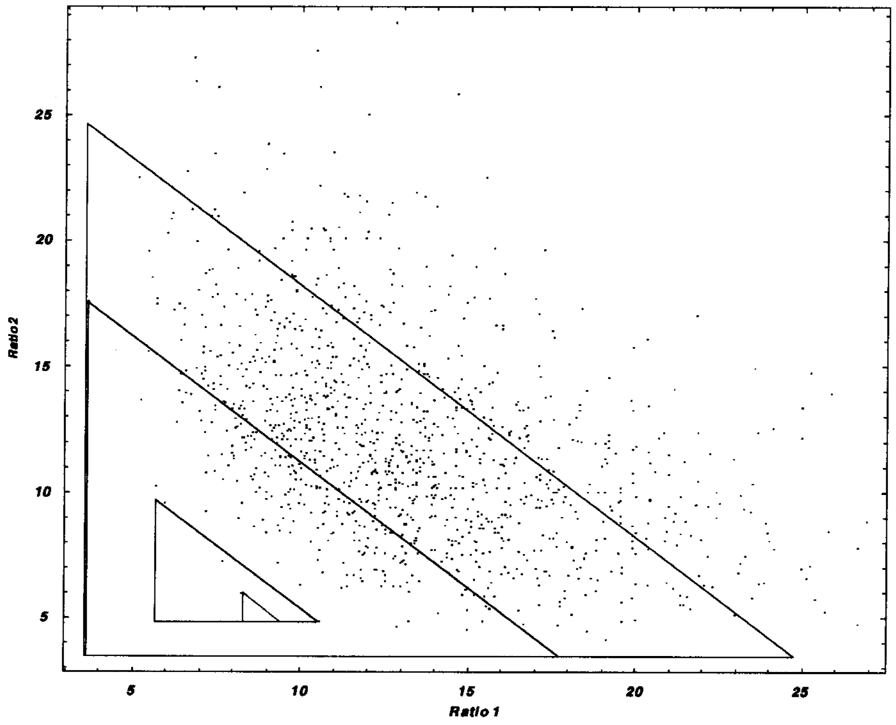

Figure 5. Convergence in $\mathcal{R}$-space for the 10-node example.

It should be obvious that a better starting point, either lower upper bounds or higher lower bounds, will reduce the number of iterations. In the example problem, we started the exchange procedure of Section 2.3 with the tree corresponding to the initial upper bound, f^1_{up} and the exchange procedure terminated with the optimal solution. After 5 iterations, the sufficient conditions for optimality were met and the algorithm terminated with the globally optimal solution.

4.2. SPEEDING UP THE ALGORITHM

During the course of our computational experiments a second heuristic step, one based on the sufficient condition for optimality, revealed itself. The appeal of this heuristic is that it does not depend on the problem structure and can be easily incorporated into the above algorithm with little additional computational effort.

While checking the current solution for optimality using the sufficienct condition, we supply the current lower bound u^i as well as v^i and l^i as arguments to $H(\cdot)$. Computing the value of $H(\cdot)$ requires the solution of an MST, thus it will return an upper bound. We discovered that one of these upper bounds was often superior to the incumbent solution. If there has been improvement in the upper bound, we replace the incumbent upper bound with this new upper bound. This further shrinks the solution space thus reducing the number of iterations required to find the globally optimal solution. For instance, on the example problem we computed $H(u^i)$ using the initial lower bound and it returned an upper bound of 15.4 which is within 8% of the optimal solution. On the second iteration, this checking further improved the upper bound to the value of the optimal solution. At the 4th iteration, the sufficient condition for optimality identified this upper bound as the globally optimal solution.

To incorporate this into the above algorithm, we replace the instruction to proceed to Step 4 in Step 3. We call this variant of the algorithm, Heuristic $\mathscr{P}$.

> Check the upper bound returned as a result of computing $H(u^i)$, $H(v^i)$, $H(l^i)$. If the upper bound resulting from these computations is superior to f^i_{up} then set f^i_{up} equal to the improved upper bound. Recompute l^i and v^i using the expressions in Step 4 and return Step 4.

4.3. COMPUTATIONAL RESULTS

The algorithm was implemented in C++ to run in the DOS/ix86 environment. We used a heap implementation of Kruskal's algorithm for the MST computations. The goals in our computational experiment were the following.

- Gain an understanding of how quickly the basic algorithm converges to the optimal solution;
- Determine how effective the swapping heuristic performed in generating high quality upper bounds thus improving the speed of convergence;
- Examine the effectivness of Heuristic $\mathscr{P}$ in improving the speed of convergence.

As our goal here is to gain an understanding of how this algorithm behaves, our computational experiments are modest, yet sufficiently expansive for purposes of this study.

For our test set, we choose randomly generated complete networks with up to 50 nodes. We choose complete networks as this generates an $\mathscr{R}$-space of maximum size. Ratio values for each arc were generated uniformly at random with the two ratios for each arc negatively correlated. This tended to generate upper and lower bounds relatively far apart. Since the bulk of the computational effort is in solving the MRST, our measure of computational effort is the number of spanning tree problems solved. We also report the number of iterations taken by the algorithm.

In looking at the table in Figure 6, it is clear that for even small problems the computational effort is large, with the basic algorithm taking many small but steadily improving steps. This is also evident in the example problems reported in Falk and Palocsay (1992). Two factors seem to affect the speed of convergence. First

	No Heuristic Start		Heuristic $\mathcal{S}$		Heuristic $\mathcal{P}$	
Nodes	Iters	MSTs	Iters	MSTs	Iters	MSTs
5	3	24	2	8	2	8
10	17	190	6	78	13	149
15	21	197	8	81	16	149
25	31	356	15	179	22	264
50	58	741	28	379	44	557

Figure 6. Computational Results on Complete Networks.

(and obviously), the size of the initial gap between the upper and lower bounds. For our problems, this gap was several orders of magnitude, thus requiring many iterations to close in on the optimal solution. Notice that using the results of the swapping heuristic as the starting point approximately halved the computational effort. In fact, for this test set, the heuristic always produced the optimal solution. Second, even if the initial upper bound corresponds to the optimal solution, the algorithm must close the remaining gap by raising the lower bound. Heuristic $\mathcal{P}$, though not as effective as the swapping heuristic, produced good results by reducing the computational effort by about 25% on average. None of the problems stalled and the sufficient condition for optimality did not terminate any of the problems early.

In general, we observed that the iterates produced by the algorithm trace the lower envelope of the convex hull of $\mathcal{R}$. Thus, the more points there are on the lower envelope of the convex hull, the more iterations required. Solving the constrained subproblems posed no problems. Using the best upper bound branching strategy as recommended in Aggarwal et al. (1982), the optimal solution was always resolved at the first level of the branch and bound tree.

5. Summary

We have presented an algorithm for solving the minimum spanning tree problem when the objective function is the sum of two linear ratios. By optimally solving the NP-hard subproblems the algorithm finds a globally optimal solution. The swapping heuristic (Heuristic $\mathcal{S}$) was shown to be effective in producing good upper bounds thus reducing the computational effort as was Heuristic $\mathcal{P}$. The basic algorithm with the latter heuristic step should be effective in solving other combinatorial optimzation problems that admit a polynomial time algorithm.

Acknowledgement

My thanks to Dr. Melinda Wolf who unwittingly suggested such a project. Her ongoing support has been invaluable.

References

Aggarwal, V., Aneja, Y.P. and Nair, K.P.K. (1982), ‘Minimal Spanning Tree Subject To A Side Constraint’. *Computers and Operations Research* 9(4): 287–296.

Almogy, Y. and Levin, O. (1969), ‘Parametric Analysis of a Multi-Stage Stochastic Shipping Problem’. In: *Proceedings of the Fifth IFORS Conference*. Venice, Italy, pp. 359–370.

Almogy, Y. and Levin, O. (1971), ‘A Class of Fractional Programming Problems’. *Operations Research* 19, 57–67.

Bitran, G.R. and Magnanti, T.L. (1976), ‘Duality and Sensitivity Analysis for Fractional Programs’. *Operations Research* 24(4): 675–699.

Camerini, P.M., Galbiati, G. and Maffioli, F. (1988), ‘Algorithms for Finding Optimum Trees: Description, Use and Evaluation’. *Annals of Operations Research* 13(1–4): 265–397.

Chandrasekaran, R. (1977), ‘Minimal Ratio Spanning Trees’. *Networks* 7: 355–342.

Dinklebach, W. (1967), ‘On Nonlinear Fractional Programming’. *Management Science* 13(7): 492–498.

Falk, J.E. and Palocsay, S.W. (1992), ‘Optimizing the Sum of Linear Fractional Functions’. In: C. Floudas and M. Pardalos (eds.), *Recent Advances in Global Optimization.* Princeton, New Jersey: Princeton University Press, pp. 221–258.

Handler, G. and Zang, I. (1980), ‘A Dual Algorithm for the Constrained Shortest Path Problem’. *Networks* 10(4): 293–310.

Hashizume, S., Fukushima, M., Katoh, N. and Ibaraki, T. (1987), ‘Approximation Algorithms for Combinatorial Fractional Programming Problems’. *Mathematical Programming* 37(3), 255–267.

Lawler, E.L.: 1976, *Combinatorial Optimization: Networks and Matroids*. New York: Hold, Reinhart and Winston.

Meggido, N. (1979), ‘Combinatorial optimization with rational objective functions’. *Mathematics of Operations Research* 4(4): 414–424.

Nagih, A. (1996), ‘Sur La Résolution Des Programmes Fractionnaires En Variables 0–1’. Ph.D. thesis, Université Paris 13, Institut Galilée, Paris, France.

Radzik, T. (1992), ‘Newton’s method for fractional combinatorial optimization’. In: *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science*. Pittsburg, Pennsylvania, pp. 659–669.

Schaible, S. (1995), ‘Fractional Programming’. In: R. Horst and P. Pardalos (eds.), *Handbook of Global Optimization Vol. 2*. Dordrecht: Kluwer Academic Publishers, pp. 495–608. Also available as University of California, Riverside, A. Gary Anderson Graduate School of Management, Working Paper Series 94-08.

Schaible, S. (1996), ‘Fractional Programming with Sums of Ratios’. Working Paper Series 96–04, University of California, Riverside, The A. Gary Anderson Graduate School of Management.