# ON A NONLINEAR-FRACTIONAL OPTIMIZATION PROBLEM IN GRAPHS

Eugenia IACOB[*]

Received: November 1, 1992
AMS Subject Classification: 05C75, 05C38, 90C35

**REZUMAT. Asupra unei probleme neliniar-fractionară de optimizare în grafe.** Fie $G(N,A)$ un graf finit şi neorientat. Asociem fiecărei muchii a grafului două ponderi pozitive: costul $c_{ij}$ şi capacitatea $k_{ij}$. Notăm cu $Y$ mulţimea tuturor arborilor de acoperire ai grafului $G$. Pentru $T \epsilon Y$ definim costul $c(T)$ şi capacitatea $k(T)$ a arborelui $T$. Scopul acestei lucrări este de a rezolva următoarea problemă de optimizare: Să se determine un arbore $T \epsilon Y$ care minimizează $c(T)/k(T)$ pe mulţimea $Y$. În articol se prezintă şi un algoritm aproximativ de determinare a soluţiei optimale.

**1. General problem.** Let $G(N,A)$ be a finite undirected graph, where $N$ is the vertex set and $A \subseteq \{(i,j): i,j \in N\} \setminus \{(i,i): i \in N\}$ is the edge set.

We associate to each edge $(i,j)$ in $A$ two positive integer pounds, that is a cost $c_{ij}$ and a capacity $k_{ij}$.

Let $Y$ be a given set of subgraphs of $G(N,A)$. For every subgraph $T$ in $Y$ we define:

$$c(T) = \sum_{(i,j) \in T} c_{ij} \quad \text{and} \quad k(T) = \min \{ k_{ij} : (i,j) \in T\},$$

representing, the cost and the capacity of the subgraph $T$, respectively.

We consider the following general fractional problem:

(P). Find a subgraph $T'$ in $Y$ minimizing the ratio $c(T)/k(T)$ over the set $Y$, namely:

$$\min \{ c(T)/k(T) : T \in Y \}.$$

The particular case of problem (P) , when $Y$ is the set of

[*] University "Babeş-Bolyai" Cluj-Napoca, Department of Computer Science, 3400 Cluj-Napoca, Romania

paths between two given vertex of the graph $G$ was studied by Martins [3]. In the paper [5], we gave a generalization of Martins algorithm.

The purpose of this paper is to apply this general algorithm for a particular case of the set $Y$. Namely, we take $Y$ to be the set of all spanning trees of the graph $G$. In this case, we solve problem (P) by the perturbation of its cost coefficients. We obtain an approximate optimal solution for the initial problem and we give an evaluation of the deviation from the optimal value.

**2. Basic properties.** First we introduce the "nondomination" relation on the set $Y$ of spanning trees.

DEFINITION 1. *Let $T,T' \in Y$ be two distinguished spanning trees of $G(N,A)$. $T$ dominates $T'$ if and only if $c(T) \le c(T')$ and $k(T) \ge k(T')$ and the strict inequality holds at least once.*

The fact that $T$ dominates $T'$ we denote by $T\ D\ T'$.

Let $Y_D = \{T \in Y : \exists\ T' \in Y$ such that $T'\ D\ T\}$ be the set of dominated spanning trees.

DEFINITION 2. $Y_N = Y - Y_D$ *is the set of nondominated spanning trees.*

From the above definition it results that $Y_N$ can be viewed as a set of optimal solutions for a bicriterion problem associated to $(P)$.

The algorithm that will be presented for solving problem $(P)$, is based on the concept of nondominated spanning tree. This concept is inspired by the procedures of solving the bicriterion problem.

Further on, we present without proof some theorems and propositions which represent the theoretic support for the algorithm. In the general case, when $Y$ is an arbitrary set of subgraphs of $G$, we allready proved all these results (see, [5]).

Let $Y_0$ be the set of optimal solutions of problem $(P)$. The following theorem establishes a relationship between $Y_0$ and $Y_N$.

THEOREM 1. ([5]) *Any optimal solution for* $(P)$ *is a non-dominated spanning tree, that is* $Y_0 \in Y_N$ .

DEFINITION 3. *The subset* $Y_N'$ *of* $Y_N$ *is a selection of* $Y_N$ *if and only if for any* $T$ *in* $Y_N$ *there exists a unique spanning tree* $T' \in Y_N'$ *such that* $c(T)=c(T')$ *and* $k(T)=k(T')$.

PROPOSITION 1. ([5]) *Let* $Y_N'$ *a selection of* $Y_N$. *If* $T'$ *and* $T''$ *are two distingueshed spanning trees such that* $T',T''\in Y_N'$ *then* $c(T')$ *is not equal with* $c(T'')$ *and* $k(T')$ *is not equal with* $k(T'')$.

From Proposition 1, it follows that the number of elements belonging to $Y_N'$ is bounded by the number $m'$ of edges with distinct capacities from $G(N,A)$. Hence, a selection $Y_N'$ can be computed in $O(m'C(m,n))$ time, where $m$ is the number of edges of $G(N,A)$ and $C(m,n)$ is the time needed for determining the minimum cost spanning tree.

It follows that the complete determination of a selection $Y'$ can be done in polynomial time. Therefore, the execution of an exhaustive search for $Y_N'$ (using Theorem 1), in order to compute an optimal solution of $(P)$ is not unrealistic. However, in the algorithm that we present, we need not to find an entire set $Y_N'$. As a consequence the number of executions of a minimum cost spanning tree algorithm is minimized.

PROPOSITION 2. ([5]) *Let $Y_N' = \{T_1, T_2, \ldots, T_r\}$ ($r < m$) be a selection of $Y_N$. Then $Y_N'$ can be ordered such that $c(T_i) < c(T_{i+1})$ and $k(T_i) < k(T_{i+1})$ for $i=1,2,\ldots,r-1$.*

We consider now the following set:

$Y_N'' = \{T_1, \ldots, T_h\} \in Y_N'$ *where* $T_1, \ldots, T_h$ are the first $h$ elements of $Y_N'$ and $Y_N'$ is a selection which has the elements ordered in sense of proposition 2.

THEOREM 2. ([5]) *Let $k' >$ max $\{k(T) : T \in Y\}$. If the element $T_j \in Y_N'$ verifies the following conditions:*

i) $c(T_j)/k(T_j) = \min \{ c(T)/k(T) : T \in Y_N' \}$,

ii) $c(T_h) < [\, c(T_j)/k(T_j)\,]\, k'$,

iii) *$T_j$ is not in $Y_0$,*

*then exists $T_s$ in $Y_0$ and $(Y_N' - Y_N'')$ such that:*

$k(T_s) > [k(T_j)/c(T_j)]\, c(T_h)$.

**3. Algorithm for the problem of the spanning tree with minimum cost/capacity ratio.** The basic scheme of the algorithm that we propose is the same as the algorithm for the MINSUM-MAXMIN bicriterion problem. Let us assume that $T_s \in Y_N'$ was just determined with such an algorithm. Then, as $k(T_{s+1}) > k(T_s)$, the edges $(i,j) \in A$, for which $k_{ij} < k(T_s)$, can be deleted from $G(N,A)$. In the resulting graph, the subgraph $T_{s+1}$ is determined as the spanning tree with minimum cost and maximal capacity relatively to the set of all spanning trees with minimum cost.

In fact, the difficulty is that we haven't an algorithm to determine the whole set of minimum cost spanning trees. The algorithm of Kruskal find only one of these spanning trees. These difficulty doesn't appear when $Y$ is the set of all paths between two given vertex of $G(N,A)$ or $Y$ is the set of assignements in a bipartite graph (see, e.g. [1], [2], [3]).

We avoid this dificulty by using the following method.

Let $C = \{c_1,c_2,\dots,c_m\}$ be the sequence of cost values for the edges of $G(N,A)$ in a nondecreasing order, and let $E=\{(i_1,j_1),\dots,(i_m,j_m)\}$ be the sequence of corresponding edges of $G(N,A)$. Further on we suppose that $c_1 < c_2$. Otherwise we have $c_1 = c_2 = \dots = c_m$ and (P) degenerates to the usual problem of determining the maximum capacity spanning tree of a graph $G(N,A)$.

The following algorithm generates some perturbation pounds $d_{ij}$ and perturbated pounds $c_{ij}'=c_{ij}+d_{ij}$, ordered in the sequences $D = \{ d_1,\dots,d_m \}$ and $C' = \{ c_1',\dots,c_m' \}$, respectively, corresponding to the same edges as the elements of $C$.

**Algorithm 1.**

Step 1. Take $k = 1$ and $p = 0$.

Step 2. If $k+1 > m$, then $d_m = 0$ and go to Step 11. Otherwise, go to Step 3.

Step 3. If $c_k = c_{k+1}$, go to Step 4. Otherwise, set $d_k = 0$, take $k+1$ instead of $k$ and go to Step 2.

Step 4. Set $d_k = 0$.

Step 5. Take $p+1$ instead of $p$ and go to the next step.

Step 6. If $k+p = m$, then go to Step 10. If $k+p < m$, then go to Step 7.

Step 7. If $c_k = c_{k+p}$, then go to Step 5. Otherwise, go to Step 8.

Step 8. For $q$ from 1 to $p$ do $d_{k+q} = q\ 10^{-L}\ (c_{k+p}-c_k)/p$, where $L$ is a natural number choosen such that $10^L > m$.

Step 9. Take $k+p$ instead of $k$ and go to Step 2.

Step 10. For $q$ from 1 to $m-k$ do

$$d_q = q\ 10^{-L}\ (m-k)^{-1} \max \{ c_{h+1}-c_h : h=1,\dots,m-1\}.$$

Step 11. For $q$ from 1 to $m$ do $c_q' = c_q + d_q$ .

Step 12. Stop.

This algorithm modifies the costs of those edges from $G(N,A)$ which have the same cost. More exactly, the algorithm adds to the costs of these edges a positive quantity in order to differentiates their costs. In this way, all values of costs associate to edges are different. In this case, the spanning tree of minimum cost is unique.

The following theorem estimates the maximum error, due to this method, in the determination of the spanning tree with minimum cost/capacity ratio.

Let $e = 10^{-L} \max \{ c_{h+1} - c_h : h = 1, \dots, m-1 \}$ and

$$g = \min \{ k_{ij} : (i,j) \in A\}.$$

THEOREM 4. *The maximum error caused by Algorithm* 1 *in finding the spanning tree with minimum cost/capacity ratio is* $(n-1)\ e/g$.

*Proof.* Let suppose that $T^*$ is the spanning tree with minimum cost/capacity ratio. Then, from the definition of e and g, it follows :

$$\frac{c'(T^*)}{k(T^*)} = \frac{\sum\limits_{(i,j)\in T^*} c_{ij}}{\min\{ k_{ij} : (i,j)\in T^* \}} = \frac{\sum\limits_{(i,j)\in T^*} ( c_{ij} + d_{ij} )}{\min\{ k_{ij} : (i,j)\in T^* \}} =$$

$$= \frac{\sum\limits_{(i,j)\in T^*} c_{ij}}{\min\{ k_{ij} : (i,j)\in T^* \}} + \frac{\sum\limits_{(i,j)\in T^*} d_{ij}}{\min\{ k_{ij} : (i,j)\in T^* \}} \le$$

$$\le \frac{c(T^*)}{k(T^*)} + \frac{(n-1)e}{g}$$

Since, for any $(i,j)$ in $A$, $d_{ij} > 0$, from the above relations, it results:

$$0 \le \frac{c'(T^*)}{k(T^*)} - \frac{c(T^*)}{k(T^*)} \le \frac{(n-1)e}{g}$$

which means that the conclusion of the theorem is true.

Further we will present an adjustement of Martins's algorithm. Theorems 2 and 3 are used as an attempt to decrease the number of spanning trees that have to be determined.

Three working variables are used in the algorithm: $T$, $c^*$ and $z$. $T'$ keeps the best spanning tree that was determined until the curent iteration, $c^*$ keeps the value of $(c(T')/k(T'))\ k'$ and $z$ keeps the value of $c(T')/k(T')$. Foregoing $k'$ is a parameter of the algorithm which is fixed such that $k' > \max\{k(T) : T \in Y\}$.

**Algorithm 2.**

Step 1. Apply the algorithm 1 for $G(N,A)$, and let denote by $c_{ij}'$ the perturbed costs.

Step 2. Set $c^*$=INF, $z$ = INF and the graph $H(N,B) = G(N,A)$, where INF is integer positive number such that:

INF > 2 max $\{c(T)/k(T) : T \in Y\}$.

Step 3. Find $T''$ the spanning tree with minimum cost from $Y(H) = \{ T \subset Y : T$ spanning tree for $H(N,B)\}$, that is :

$c'(T'') = \min \{ c'(T) : T \in Y(H) \}$.

Step 4. If $Y(H)$ is an empty set or $c'(T'') > c^*$, then finish the algorithm. In the opposite case go to Step 5.

Step 5. If $c'(T'')/k(T'') < z$, then perform the following operations:

i) Take $T' = T''$.

ii) Take $c^* = [c'(T'')/k(T'')]\ k'$ and $z = c'(T'')/k(T'')$.

iii) Take $x = k(T'')$ and go to Step 7.

If $c'(T'')/k(T'') > z$, then go to Step 6.

Step 6. Set $x = c'(T'')/z$ and go to Step 7.

Step 7. Eliminate from $H(N,B)$ all the edges $(i,j)$ which has $k_{ij} < x$. After this the new set of edges $B$ is :

$B := B - \{(i,j) \in A : k_{ij} < x \}$.

Go to Step 2.

In order to clarify the implications of theorems 2 and 3 we explain the algorithm step by step.

The first step modifies, if it is necessary, the costs associated to edges of $G(N,A)$ such that these became distinct.

In the second step the working variables $c^*$ and $z$ are initialized .

In the third step we determined a nondominated spanning tree $T''$. This step is the most complex step of the algorithm because each iteration requires the solving of an optimization problem on the spanning trees set $Y(H)$. We can use in this step Kruskal's algorithm (see, e.g., [4], [2]).

In the fourth step, we check a stop condition of the algorithm. Thus the algorithm stops with an optimal solution $T'$ when either in $Y(H)$ there is no elements or it is verified the condition of theorem 2.

Step 5 is performed when the spanning tree $T''$ determined in Step 3 is "beter" then the best spanning tree determined until that moment. Also in this step we actualize $T'$, $c^*$ and $z$. If $T''$ is worse than $T'$, Step 6 is performed. In Step 5 is used Theorem 3, to justify the elimination (in Step 6) of some edges from the auxiliary graph $H(N,B)$. In this way, it can be possible to avoid some nondominated spanning trees which not belong to $Y_0$ .

In Step 7 the edges having the capacities smaller than $x$ are eliminated from $H(N,B)$. We must note that $x=k(T')$ when the Step 4 is performed. But when Step 5 is executed the value of $x$ is determined by the Theorem 3.

## R E F E R E N C E S

1. Burkard, R.E., Hahn, W., Zimmerman, U.,: *An algebraic approach to assignement problem,* Math. Programming, 12 (1977), pp.219-232.
2. Gondran, M., Minoux, M., *Graphes et algorithmes,* Editions Eyrolles, Paris, 1979.
3. Kruskal, H.B., *On the shortest spanning subtree of a graph and the traveling salesman problem,* Proc. Am. Math. Soc., 71 (1956), pp.48-50.
4. Martins, E.Q.V., *An algorithm to determine a path with minimal cost/capacity ratio,* Discrete Applied Math., 8 (1984), pp. 189-194.
5. Tigan E., *Algorithms to minimize the cost/capacity ratio,* Econ. Comp. and Econ. Cyb. Studies and Res., 4 (1989), pp.53-59.