Tariff Optimization in Networks

Mustapha Bouhtou [*] Stan van Hoesel [†] Anton F. van der Kraaij [‡]
Jean-Luc Lutton [§]

25 February 2003

Abstract

We consider the problem of determining a set of optimal tariffs for an agent in the network, who owns a subset of all the arcs, and who receives revenue by setting the tariffs on the arcs he owns. Multiple rational clients are active in the network, who route their demands on the cheapest paths from source to destination. The cost of a path is determined by fixed costs and tariffs on the arcs of the path.

We introduce a remodeling of the network, using shortest paths. We develop three algorithms based on this shortest paths graph model, namely a combinatorial branch and bound algorithm, a path oriented mixed integer program and a known arc oriented mixed integer program. Combined with reduction methods this remodeling enables us to solve the problem to optimality, for quite large instances. We provide computational results for the methods developped and compare them with the results of the arc oriented mixed integer programming formulation of the problem, applied to the original network.

Subject Classification: Networks/Graphs: Applications. Programming: Integer Algorithms. Mathematics: Combinatorics.

1. Introduction

The tariff-setting problem in a network involves two non-cooperative groups, tariff setting agents and tariff following clients. Each arc in the network is owned by (at most) one agent. Being the owner of an arc an agent can set the price for renting capacity on the arc freely, in order to maximize his revenues. The clients wish to route a certain demand for flow capacity on a path connecting two vertices (a commodity). A selected route can involve connections belonging to different agents. Clearly, each client will select a route with minimum cost to satisfy the demand for their commodity. This problem is essentially a game theoretic problem, see Fisk [3], where the agents decide on their prices based on (partial) knowledge of the prices of competing agents. We restrict the problem to a single agent who knows the tariffs of his competitors and intends to charge revenue-maximizing tariffs on the subset of the network arcs owned by himself. This agent is generally referred to as the leader, while the clients are referred to as followers. Labbé et al. [7] showed that the single agent problem is $\mathcal{NP}$-hard, already with only one commodity, given bounds on the tariffs. They also identify two polynomially solvable special cases: the case with multiple commodities, but one tariff arc; and the case with one commodity where the path in the network taken by the client in the optimal solution is known a priori.

[*] France Télécom R&D, 39-40 rue du Général Leclerc, F-92131 Issy-Les-Moulineaux, France. E-mail: mustapha.bouhtou@rd.francetelecom.fr

[†] Dept of Quantitative Economics, Maastricht University, P.O.Box 616, 6200 MD Maastricht, The Netherlands. E-mail: s.vanhoesel@ke.unimaas.nl

[‡] Dept of Quantitative Economics, Maastricht University, P.O.Box 616, 6200 MD Maastricht, The Netherlands. E-mail: a.vanderkraaij@ke.unimaas.nl

[§] France Télécom R&D, 39-40 rue du Général Leclerc, F-92131 Issy-Les-Moulineaux, France. E-mail: jean-luc.lutton@francetelecom.fr

The tarification problem has a wide range of applications such as tariff setting in freight transportation, and highway toll optimization, see Brotcorne et al. [1] and Labbé et al. [7], respectively. Recently, the problem has become of interest in the telecommunications market. Here, many operators are active, who rent their capacity on connections to customers. In general, a single operator does not own complete connections between all pairs of vertices in a network. Thus, a customer has to rent connections from different operators to establish a complete line.

An interesting variant of the tarification problem occurs in highway traffic routing, see Jahn et al. [5], Roughgarden et al. [8], and IP (Internet etc.) traffic routing, see Fortz et al. [4]. Here a single operator is active, and tariffs are introduced with the objective to divert traffic from areas of congestion.

A linear bilevel model is a very natural and elegant formulation of the problem. The upper level relates to the leader and fixes the tariffs, the lower level belongs to the clients, who given the tariffs set by the leader, can determine their best (shortest) path. This model was proposed first in Labbé et al. [7]. For the single-commodity case, primal-dual heuristics based on a penalization of the lower level objective function are proposed by Brotcorne et al. [1]. These heuristics were extended to multiple commodities in Brotcorne et al. [2]. Furthermore, in Labbé et al. [7], the problem was reformulated as a mixed integer program. This program solves up to medium sized instances within reasonable time.

The solution methods we propose here are all based on a remodeling of the network, that generally reduces the problem. Moreover, the essential structure of solutions to the problem is simultaneously captured. Concretely, we consider shortest paths that can be used by the clients in the network. These shortest paths consist of fixed cost arcs and tariff arcs, but between each pair of consecutive tariff arcs the shortest path, using only fixed cost arcs, will always be selected, independent from the tariffs. Such paths are replaced by arcs in the so called *shortest paths graph model*, which is equivalent to the original problem. This simple, but elegant, reformulation works very well on almost any instance. For this shortest paths graph model, the still huge number of possible paths for each commodity can be reduced enormously with processing techniques using dominance criteria among paths. These ideas are exploited in three solution methods: a combinatorial branch and bound algorithm, a path oriented mixed integer program, and a known arc oriented mixed integer program (Labbé et al. [7]). All three methods allow us to solve fairly large instances of the problem in a small amount of time.

In section 2, we define the tarification problem, address its complexity and introduce the necessary notation. Then, in section 3, the remodeling of the network is described. In section 4, model specific graph reduction methods are given. In section 5 these concepts are used in a branch and bound algorithm, while section 6 gives a path and arc oriented mixed integer program. We present some computational results in section 7, where we compare the three methods developed to the arc oriented mixed integer program of Labbé et al. [7] on the original network, to illustrate the efficiency of our methods and remodeling technique.

2. The Tariff-Setting Problem

Consider a network represented by a directed graph $G = (N, A)$ with nodes N and arcs A. The arc set A is partitioned into two sets: the set of *tariff* arcs T, and the set of *fixed* cost arcs F. The tariff arcs belong to the leader in the network and incur a revenue generating toll for routing a unit of a client's demand. The fixed arcs are owned by other agents in the network, whose tariffs are known a priori and hence can be viewed as fixed per unit costs. The tariffs on the arcs of T are determined such that the total revenue of the leader is maximized. Both the tariffs and the fixed costs are assumed to be nonnegative. The clients on the network route their demands from source to destination according to the shortest path with respect to total cost, where the total cost of a path is defined as the sum of all the tariffs and fixed costs on the arcs of the path. Whenever the client has a choice among multiple shortest paths with the same total cost but with different revenues for the leader, we suppose the client takes the shortest path that is most profitable to the leader. This tie-breaking rule is justified by noting that we can always decrease the tariff on one of the arcs of the path with highest revenue for the leader by ϵ.

Labbé et al. [7] studied the tarification problem on a transportation network and have shown that the related decision problem is $\mathcal{NP}$-complete for a single commodity when lower bounds on the tariffs are given. Some variants of the tarification problem have been shown to be solvable by polynomial-time algorithms. Among these is the class of problems with only a single tariff arc and multiple clients and the class of problems where the leader is dealing with multiple tariff arcs and a single client for which the path taken at optimality is known a priori. The reader is referred to Labbé et al. [7] for a description of these algorithms.

We denote by c_a the cost of routing a unit demand on a fixed cost arc $a \in F$ and by t_a, to be determined by the leader, the cost of routing a unit demand on a tariff arc $a \in T$. The commodities are denoted by the set K. The demand of a commodity $k \in K$ is given by d_k. The source and destination of commodity k are given by the pair (s_k, t_k). The set of paths that connect s_k and t_k is given by P_k. For each path $p \in P_k$ we introduce T_p for its set of tariff arcs, and F_p for its set of fixed cost arcs. Furthermore, the cost of routing a unit demand on p is denoted by its length $l_p(t)$, which is a function of the tariffs t. The length of p is determined by the sum of the costs on the fixed arcs of the path, denoted by c_p, and the costs on the tariff arcs of the path, represented by $\pi_p(t)$. Thus, $l_p(t) = c_p + \pi_p(t)$, where $c_p = \sum_{a \in F_p} c_a$, and $\pi_p(t) = \sum_{a \in T_p} t_a$. Note that our model implicitly incorporates arcs with both fixed and tariff costs since we can divide such an arc a with cost c_a and tariff t_a into two arcs: a fixed arc with cost c_a and a tariff arc with tariff t_a.

To ensure that the problem is bounded, we assume that for each commodity there exists a path from source to destination which uses only fixed cost arcs. Otherwise, the leader can set the tariffs on the arcs in T arbitrarily high.

The following formulation of the tarification problem is a direct implementation of the above description.

$$\begin{array}{ll} \max\limits_{t \geq 0} & \sum\limits_{k \in K} d_k \pi_{p_k^*}(t) \\ s.t. & p_k^* = \arg\min\limits_{p \in P_k} l_p(t) \quad \forall k \in K \end{array} \tag{1}$$

The formulation given by (1) is a bilevel problem where at the upper level the leader strives to maximize his revenue, while at the lower level the clients (followers) seek to minimize the cost of routing their demands. Both objective functions are linear and hence we are dealing with a linear-linear bilevel program. The general linear-linear bilevel program has been shown to be $\mathcal{NP}$-hard by Jeroslow [6]. For a reference on bilevel programming, we refer the reader to Vicente and Calamai [9] who have compiled an annotated bibliography on this subject containing more than one hundred references. Notice furthermore that the bilevel program given by (1) is not polynomial in its input data, since the set of all possible paths for each client $k \in K$ may be exponential.

Labbé et al. [7] rewrite the path oriented formulation (1) to the following arc oriented bilevel model. Let the vector b^k be the demand/supply vector for each commodity where each element of the vector represents the demand/supply for a commodity at each node in the graph.

$$\begin{array}{lll} \max\limits_{t \geq 0} & \sum\limits_{k \in K} \sum\limits_{a \in T} t_a y_a^k & \\ \min\limits_{y^k \geq 0} & \sum\limits_{k \in K} \left\{ \sum\limits_{a \in T} t_a y_a^k + \sum\limits_{a \in F} c_a y_a^k \right\} & \\ s.t. & A y^k = b^k & \forall k \in K \end{array} \tag{2}$$

In this bilevel model, $y^k \in \mathbb{R}^{|A|}$ represents the flow on the arcs, in vector notation of commodity k. Furthermore, A represents the node-arc incidence matrix of the network. This formulation has

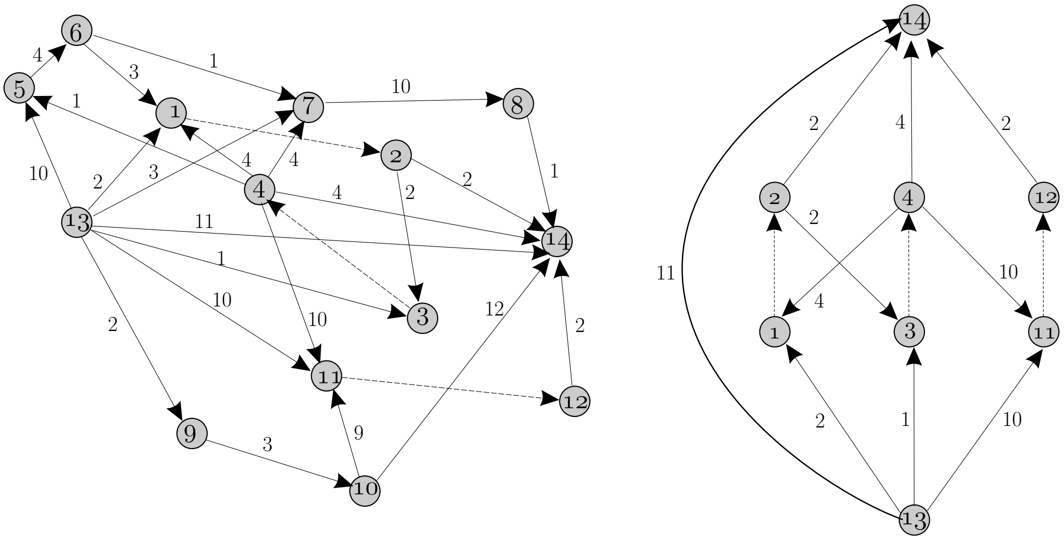

Figure 1: Shortest Paths Graph Model (right) for $|T| = 3$.

been used in Brotcorne et al. [1] and Brotcorne et al. [2] for the development of primal-dual heuristics in case of a single-commodity and multiple commodities, respectively.

Labbé et al. [7] have reformulated the bilevel model (2) as a mixed integer program. To this end the lower problem is separated into $|K|$ problems, one for each commodity. This is done by introducing for each commodity its own tariff t^k. Next, the LP for each commodity is restricted to its optimal solution(s) by adding the variables and constraints of the dual LP, and a constraint that sets the primal and dual objective at the same value. Later, constraints are added to force equality of tariffs for all commodities. We will refer to the single-level mixed integer program of Labbé et al [7] in the remainder of this paper as AMIP.

3. The Shortest Paths Graph Model

If for given tariffs t_a for all $a \in T$ a client will select the shortest path, say p, between the two end nodes of his commodity, then clearly, the subpaths of p are also shortest paths. Consider two tariff arcs $a_1 = (i_1, j_1)$ and $a_2 = (i_2, j_2)$ that appear consecutively on p. Then the subpath between j_1 and i_2 is a shortest path that only contains fixed arcs. Since such paths can be computed using the original data, we constructed a new graph model, for each commodity, in which this is actually done: the shortest paths graph model (SPGM). We will define this graph model for a single customer first. Consider the original graph $G = (N, A)$ with the tariff arcs in $T \subseteq A$. For a client with a demand d from s to t, we define the graph $G^* = (N^*, A^*)$ and the tariff arcs $T^* \subseteq A^*$. In this graph, the tariff arcs are copied from G as a matching. So, arcs with a common vertex are separated. Next, we construct the following fixed cost arcs. For two tariff arcs $a_1 = (i_1, j_1)$ and $a_2 = (i_2, j_2)$ we connect j_1 with i_2, if there is a path in G that uses fixed arcs only. Similarly, we connect j_2 with i_1. From the source s we construct arcs to all the tail nodes of the tariff arcs, and from all the head nodes we construct an arc to the destination t, again only if paths exist using only fixed arcs in G. Any fixed arc in A^* has a cost equal to the length of the shortest path between its end vertices in G, using only fixed cost arcs in G. The new network is called the shortest paths graph model (SPGM).

Example 1. Figure 1 shows the shortest paths graph model (right) of a network (left) containing three tariff arcs for a commodity from node 13 to 14. The tariff arcs are given by the arcs $(1, 2), (3, 4)$ and $(11, 12)$. All other arcs in the shortest paths graph model are representations of the shortest path using no tariff arcs between each node. The cost of the arc is the cost of

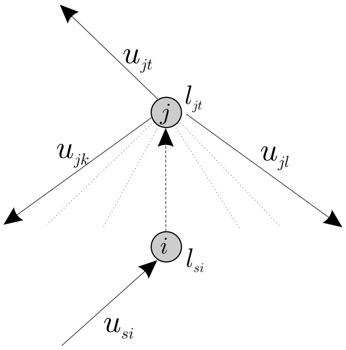

Figure 2: Detailed view of tariff arc

the corresponding shortest path in the original network between the two nodes. If no path exists between two nodes in the original network, the corresponding arc in the shortest paths graph model is not present or has infinite cost.

The shortest paths graph model can easily be extended to multiple commodities. For each commodity, we create a SPGM. The *inner graph* (consisting of the end vertices of the tariff arcs, and the arcs between them) is equal for all commodities and hence needs to be determined only once. Additional shortest path calculations are necessary only for the arcs leaving the source and/or entering the target of each commodity. The shortest path graph model is equivalent to the original graph in the sense that both have an optimal solution of the same value: if a path exists in the original graph, then there exists a path in the shortest path graph model which is at least as good. Alternately, if a path exists in the shortest path graph model, then a path with the same cost exists in the original graph.

4. Reduction Methods

We can decrease the size of the shortest paths graph model considerably by using reduction methods. The aim of these methods is to limit the amount of potentially shortest paths for a commodity to an acceptable number. First, we describe some techniques with which we can remove arcs from the network. Second, we describe a dominance relationship for paths that removes paths from the set of potentially optimal paths explicitly. The aim is to end with a manageable set of potentially optimal paths.

4.1 Arc Reductions

Denote by u_{ij}, the cost of the shortest path using only fixed arcs from node i to node j in G, i.e., u_{ij} is the length of the arc (i, j) in G^*. Let l_{ij} denote the cost of the shortest path from i to j in G, possibly using tariff arcs, when the tariffs are set to zero. We restrict ourselves to a single commodity, where node s represents the source node and node t the destination node. In figure 2 we depict the values defined here: the u_{ij} are arc values, and the l_{ij} are node values.

Note that l_{ij} is a lower bound for the cost of a path from i to j, and u_{ij} is an upper bound for the cost of a path from i to j taken by the client.

Proposition 1. *If $l_{jt} = u_{jt}$, then any optimal path from s to t using node j can use arc (j, t): all other arcs leaving j can be removed.*

Proof. A lower bound on the cost of a path, possibly using tariff arcs, from j to t is l_{jt}. An upper bound is given by u_{jt}. If $l_{jt} = u_{jt}$, there is no room for taxation on any path from j to t. Thus, (j, t) is an optimal choice. □

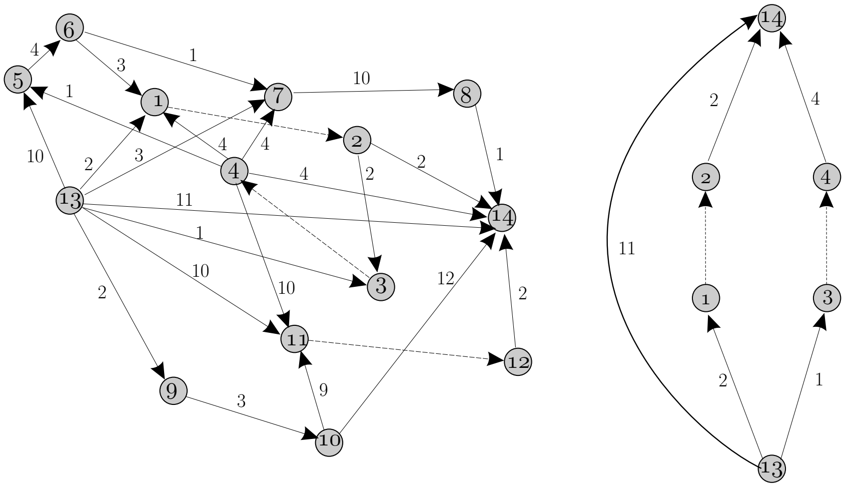

Figure 3: Original graph (left) and final SPGM graph (right).

Proposition 2. *If $l_{si} = u_{si}$, then any optimal path from s to t using node i can use arc (s, i): all other arcs entering i can be removed.*

Proof. Analogous to proposition 1. □

Proposition 3. *Consider two tariff arcs: (i_1, j_1) and (i_2, j_2). If $u_{j_1t} \leq u_{j_1i_2} + l_{j_2t}$, we can delete arc (j_1, i_2).*

Proof. The lower bound on the cost of a path from j_1 to t taking the arc (j_1, i_2) is equal to $u_{j_1i_2} + l_{j_2t}$. An upper bound on the cost of a shortest path is given by u_{j_1t}. Hence, there is no room for taxation on a path from j_1 to t using the arc (j_1, i_2). Thus, (j_1, t) is an optimal choice. □

Proposition 4. *Consider two tariff arcs: (i_1, j_1) and (i_2, j_2). If $u_{si_1} \leq u_{j_2i_1} + l_{si_2}$, we can delete arc (j_2, i_1).*

Proof. Analogous to proposition 3. □

Proposition 5. *If $u_{st} \leq l_{si_1} + l_{j_1t}$, we can delete the tariff arc (i_1, j_1).*

Proof. The lower bound on the cost of a path going through the tariff arc (i_1, j_1) is $l_{si_1} + l_{j_1t}$. Hence, there is no room for taxation on the tariff arc (i_1, j_1), and thus (s, t) is always at least as good. □

Proposition 6. *Consider two tariff arcs: (i_1, j_1) and (i_2, j_2). If $u_{st} \leq l_{si_1} + u_{j_1i_2} + l_{j_2t}$, we can delete the arc (j_1, i_2).*

Proof. The upper bound on the cost of a shortest path from s to t is u_{st}. A lower bound on the cost of a path using the tariff arc (i_1, j_1) and going to the tariff arc (i_2, j_2) is at $l_{si_1} + u_{j_1i_2} + l_{j_2t}$. Hence, there is no room for taxation on a path from s to t using the arc (j_1, i_2). Thus, this arc need not be used in an optimal solution. □

Example 2. We illustrate some of the reduction methods proposed in this section on the graph shown in figure 3 on the left. For this instance, we are dealing with one client who has a unit demand from node 13 to node 14. The tariff arcs in the network are given by the arcs $(1, 2)$, $(3, 4)$ and $(11, 12)$, also indicated by the dotted arcs in the figure. In this graph we can delete the arcs $(4, 1)$, $(2, 3)$ and $(4, 11)$ of the shortest paths graph model by applying for example proposition 3

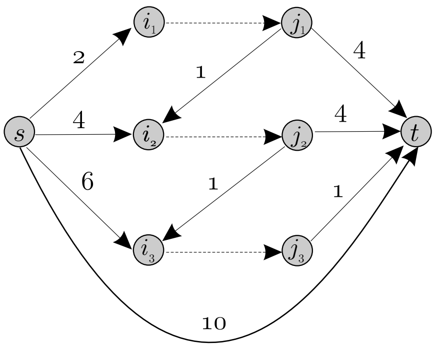

Figure 4: Dominated paths example.

or proposition 4. Furthermore, proposition 5 allows us to delete the tariff arc $(11, 12)$ from the graph.

The final graph is shown on the right in figure 3. For this graph it is easy to see that the optimal solution is to set the tariff on the arc $(1, 2)$ to 7, while setting all the other tariffs to a suitable large value, i.e. 11, yielding a revenue of 7 for the leader.

4.2 Path Reduction

By applying the propositions described in section 4.1 and finding all paths in the graph, we obtain a reduced set of paths that remains relevant for the commodity. Hopefully, this set of paths is small. It is however possible to reduce the size of this set in some instances even more by eliminating paths which are always dominated by others. The notion of path dominance is given by the following definition.

Definition 1. *If we can replace in all feasible solutions the path q by the path p without violating the feasibility constraints or decreasing the value of the objective function, then path p dominates path q.*

The following proposition allows us to eliminate dominated paths. Recall that T_p is the set of tariff arcs from path p, and that c_p is the total cost of the fixed arcs from p.

Proposition 7. *Consider a path p, resp. q. If $T_p \subseteq T_q$ and $c_q \geq c_p$, then path p dominates path q for all tariff values.*

Proof. Suppose that path q is the shortest path taken by the client. Then

$$c_q + \sum_{a \in T_q} t_a \leq c_p + \sum_{a \in T_p} t_a \quad \text{or equivalently} \quad c_q + \sum_{a \in T_q \setminus T_p} t_a \leq c_p \tag{3}$$

Since $c_q \geq c_p$ and $t_a \geq 0$ for all tariff arcs $a \in T$, equation (3) only holds when $c_q = c_p$ and $t_a = 0$ for all $a \in T_q \setminus T_p$. Hence, we can replace path q by path p without violating the feasibility constraints or changing the value of the objective function since path p has the same revenue for the leader and the same cost for the client as path q. Equation (3) shows furthermore that whenever $c_q > c_p$, path q can never be the path taken by a client in a feasible solution, since path p will always be cheaper. □

Example 3. An instance where this dominance of paths occurs is given in figure 4. The tariff arcs are the arcs $(i_1, j_1), (i_2, j_2)$ and (i_3, j_3). The leader is dealing with one client who wants to route his demand from node s to node t. For this graph, the path $\{s, i_1, j_1, i_2, j_2, t\}$ is dominated by the path $\{s, i_1, j_1, t\}$.

The shortest paths graph model combined with the arc and path reductions mentioned in this section allow us to create a sparse graph for each commodity, which in turn yields a small set of relevant paths for each commodity, as will be shown by the numerical results in section 7.

5. Branch and Bound Algorithm

In this section we describe a branch and bound algorithm for our tariff-setting problem which uses the shortest paths graph model from section 3 and the reduction methods from section 4. This algorithm consists of two steps. In the first step we create for each client a shortest paths graph model and apply to it the reduction methods of section 4. For each client we thus find the relevant shortest paths. In step two we solve the problem to optimality by a classical branch and bound method.

As stated in section 2, denote the clients by the set K and the set of paths a client $k \in K$ can take by P_k. The reduction methods applied to the shortest paths graph model allow us to determine the set of relevant paths for each commodity. We suppose that P_k is reduced to contain the relevant paths only. Recall furthermore from section 2 the linear function $l_p(t) = c_p + \pi_p(t)$ denoting the cost of a path p as a function of all tariff values. Let p_k^l be the path for client $k \in K$ with the smallest fixed cost, i.e., $p_k^l = \arg\min_{p \in P_k} c_p$ and p_k^u the path with the largest fixed cost, i.e., $p_k^u = \arg\max_{p \in P_k} c_p$. Note that p_k^u has no revenues for the leader, since it denotes the path with fixed cost arcs only. Clearly, $c_{p_k^u} - c_{p_k^l}$ is an upper bound on the revenues that can be generated from client k. This is an important measure in the branch and bound algorithm.

5.1 Branching Rules

In each node of the branch and bound tree, we select a client, and we create a branch for each of the relevant paths of the client. The selection method of the clients is based on the upper bound $c_{p_k^u} - c_{p_k^l}$ on the revenue generated by each client for the leader: the client for which this upper bound is highest, is selected first. Next, we walk through the search tree in a depth-first manner.

5.2 Node Processing

Due to our branching rules, in each node of the branch and bound tree for some clients the path taken in the solution is fixed, whereas for other clients this choice is still to be made. In each node, we denote by the set $K_f \subseteq K$ the set of clients for which we have fixed the path taken in the solution. Suppose that for any client $k \in K_f$, we have fixed the path p_k^*. We can find the optimal, revenue maximizing tariffs for the problem restricted to the clients in K_f by solving the following linear problem.

$$\begin{array}{lll} \max & \sum\limits_{k \in K_f} d_k \pi_{p_k^*}(t) & \\ s.t. & l_p(t) \geq l_{p_k^*}(t) & \forall k \in K_f, \forall p \in P_k \\ & t_a \geq 0 & \forall a \in T \end{array} \tag{4}$$

The linear program described in (4) forces the path p_k^* to be the shortest path in P_k, while maximizing the leader's revenue.

We generate lower bounds in each node of the branch and bound tree by computing a feasible solution. Such a feasible solution can be created by solving (4) and then adding the revenues from the tariffs of (4) for the clients in $K \backslash K_f$. A better lower bound may be generated by fixing for all clients the path taken in the solution. For each client $k \in K_f$ we already know which path is taken in the solution and we denote it by p_k^*. For the clients $k \in K \backslash K_f$, we fix the path to the one for which the possible revenue is highest. This is the path with smallest fixed cost, i.e. the path p_k^l:

$$\begin{array}{lll} \max & \sum\limits_{k \in K_f} d_k \pi_{p_k^*}(t) + \sum\limits_{k \in K \backslash K_f} d_k \pi_{p_k^l}(t) & \\ s.t. & l_p(t) \geq l_{p_k^*}(t) & \forall k \in K_f, \forall p \in P_k \\ & l_p(t) \geq l_{p_k^l}(t) & \forall k \in K \backslash K_f, \forall p \in P_k \\ & t_a \geq 0 & \forall a \in T \end{array} \tag{5}$$

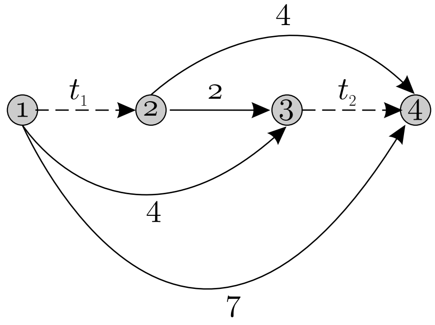

Figure 5: Upper bound on revenue or cost is not reached.

Note that this linear program may be infeasible in which case no sensible lower bound is generated.

For a client $k \in K$ an upper bound for the revenue generated by that client is given by $c_{p_k^u} - c_{p_k^l}$. As is shown by Labbé et al. [7], this upper bound is not necessarily reached. Even the upper bound on the cost of the path, $c_{p_k^u}$, is not tight. This is shown by the example given in figure 5. For a single client with a unit demand from node 1 to node 4, the optimal tarification scheme is to set the tariffs on the tariff arcs to $t_1 = t_2 = 2$. Hence, the cost of the path taken by the client is 6, yielding a revenue of 4 for the leader. The upper bound on the cost of the path is however 7, while the upper bound on the revenue is given by $7 - 2 = 5$.

In each node of the branch and bound tree, let Π^* be the optimal value of (4), i.e., the optimal revenue obtained from the commodities with fixed paths. The remaining customers can contribute no more per unit than $\sum_{k \in K \setminus K_f} \left\{ c_{p_k^u} - c_{p_k^l} \right\}$. An upper bound on the total revenue for the leader in a node is thus given by:

$$\Pi^* + \sum_{k \in K \setminus K_f} d_k \left\{ c_{p_k^u} - c_{p_k^l} \right\} \tag{6}$$

This upper bound can be tightened by using the information we can retrieve from the constraints of the linear program given by (4). These constraints must be satisfied by any feasible solution, for all clients and their relevant paths. Thus, the constraints of (4) also hold for the commodities for which a path is not fixed yet, as can be seen from the constraints in (5). Each of the constraints in (4) involves two paths, say p and q, and states that the total cost of one path (say p) is at most the total cost of the other path (say q): $c_p(t) \leq c_q(t)$. Now, let T_p be the set of tariff arcs on path p. The constraint reads in more detail

$$c_p + \sum_{a \in T_p} t_a \leq c_q + \sum_{a \in T_q} t_a \tag{7}$$

or equivalently,

$$\sum_{a \in T_p \setminus T_q} t_a - \sum_{a \in T_q \setminus T_p} t_a \leq c_q - c_p \tag{8}$$

If $\sum_{a \in T_q \setminus T_p} t_a$ can be bounded from above, then we have an upper bound on the sum of a set of tariffs. For instance, if $T_q \setminus T_p$ is empty, then we get zero as a trivial upper bound. Thus, this way we find a number of constraints of the type

$$\sum_{a \in T_i \subseteq T} t_a \leq b_i \quad (i \in \Phi) \tag{9}$$

for index set Φ.

Next, consider a path $p \in P_k$ from commodity $k \in K \backslash K_f$, i.e., a commodity for which no path has been fixed yet. An upper bound on the total cost per unit demand was previously given by the path using only fixed arcs, i.e. the path $c_{p_k^u}$. With the constraints of type (9) obtained from the commodities in K_f and by defining $b_i = 0$ for $T_i = \emptyset$, the upper bound on the total cost of a path $p \in P_k$, denoted by $\gamma(p)$, is given by:

$$\gamma(p) = \min_{i \in \Phi : T_p \subseteq T_i} (c_p + b_i) \quad \forall k \in K \backslash K_f, \forall p \in P_k \tag{10}$$

For each commodity $k \in K$, the total cost of each path taken in a solution can be at most the minimum cost of all other possible paths for that commodity. Hence, an upper bound on the total cost of *any* path in P_k is given by:

$$\min_{p \in P_k} \gamma(p) \tag{11}$$

For each commodity $k \in K$, the upper bound on the revenue is still highest for the path with smallest fixed cost, i.e. the path p_k^l. We can thus improve the upper bound computation given in (6) by using the following upper bound:

$$\Pi^* + \sum_{k \in K \backslash K_f} d_k \left\{ \min_{p \in P_k} \gamma(p) - c_{p_k^l} \right\} \tag{12}$$

We will use the upper bound calculation of (12) for our branch and bound algorithm.

6. Mixed Integer Programming Formulations

In this section, we introduce two mixed integer programs. The first is a formulation using path variables, where the paths are generated by the shortest paths graph model. The second is an application of the mixed integer program formulation of Labbé et al. [7], which uses the arcs of the shortest paths graph model instead of the original arcs in the network.

We rewrite the bilevel formulation (1) to the following single-level program, using path variables. To this end, we introduce the binary variable h_p which indicates whether or not a given path is taken in the optimal solution.

$$\begin{array}{lll}
\max \sum_{k \in K} \sum_{p \in P_k} d_k \sum_{a \in T_p} h_p t_a & & \\
s.t. \quad \quad \sum_{p \in P_k} h_p = 1 & \forall k \in K & \\
\quad \quad \quad l_p(t) \geq \sum_{q \in P_k} h_q l_q(t) & \forall k \in K, \forall p \in P_k & \\
\quad \quad \quad h_p \in \{0,1\} & \forall k \in K, \forall p \in P_k & \\
\quad \quad \quad t_a \geq 0 & \forall a \in T &
\end{array} \tag{13}$$

The first constraint in this formulation indicates that a client selects only one path. The second constraint ensures that the path taken by the client is indeed the cheapest available path. Note that this formulation is not only nonlinear in its objective function, but also in this constraint. It is however possible to use standard linearization techniques for (13) by introducing the variable r_{pa} for each path $p \in P_k$, and tariff arc $a \in T$. The variable r_{pa} is equal to t_a if and only if client $k \in K$ takes the path $p \in P_k$ going through the tariff arc $a \in T$ and is equal to 0 otherwise. After adding the constraints enforcing this relationship, we can write the following mixed integer programming formulation. We will refer to this formulation as PMIP.

$$
\begin{array}{lll}
\max \sum\limits_{k\in K}\sum\limits_{p\in P_k} d_k\left(\sum\limits_{a\in T_p} r_{pa}\right) & & \\
s.t. & \sum\limits_{p\in P_k} h_p = 1 & \forall k\in K \\
 & l_p \geq \sum\limits_{q\in P_k}\sum\limits_{a\in F_q} h_q c_a + \sum\limits_{q\in P_k}\sum\limits_{a\in T_q} r_{qa} & \forall k\in K, \forall p\in P_k \\
 & l_p = \sum\limits_{a\in F_p} c_a + \sum\limits_{a\in T_p} t_a & \forall k\in K, \forall p\in P_k \\
 & r_{pa} - t_a \leq (1-h_p)M & \forall k\in K, \forall p\in P_k, \forall a\in T_p \\
 & r_{pa} - t_a \geq -(1-h_p)M & \forall k\in K, \forall p\in P_k, \forall a\in T_p \\
 & r_{pa} \leq h_p M & \forall k\in K, \forall p\in P_k, \forall a\in T \\
 & h_p \in \{0,1\} & \forall k\in K, \forall p\in P_k \\
 & t_a \geq 0 & \forall a\in T \\
 & r_{pa} \geq 0 & \forall k\in K, \forall p\in P_k, \forall a\in T
\end{array}
\tag{14}
$$

Note that for the PMIP, we are in theory still left with an exponential number of constraints. However, the remodeling of the network proposed in this paper, combined with the model specific graph reduction methods, will allow us to generate the relevant paths for each commodity and thus reduce the necessary variables and constraints to a manageable amount.

We can use the shortest paths graph model in an arc oriented model by applying the formulation referred to as AMIP of Labbé et al. [7]. The idea is to create a multiple commodity mixed integer program by first allowing the tariffs on each tariff arc of the shortest path graph model to be different for each commodity, i.e. to allow for price discrimination and maximize the revenue of the leader over all commodities, and then prohibit price discrimination by setting tariffs on the arcs of each SPGM equal to each other if they represent the same tariff arc in the original network. We will refer to the adaptation of the formulation AMIP to the shortest paths graph model as AMIP+.

The results of AMIP+ and PMIP, together with the numerical results of the branch and bound algorithm developed in this paper, will show the efficiency of the shortest paths graph model when compared to AMIP on the original network.

7. Numerical Results

The branch and bound algorithm (PBB) was implemented in C++, using CPLEX 7.5 to solve the linear programs as described in (4) in each node of the tree. The formulations AMIP, AMIP+ and PMIP were also implemented in C++, using CPLEX 7.5. All tables concerning the numerical results are found in the appendix.

The solution methods developed in this paper were tested on two type of data sets. The first type are data sets provided by France Télécom Research and Development and represent real life instances of the tarification problem as occurring for France Télécom at the international level. In these instances, the graph represents a telecommunications network and the tariff arcs are the interconnections between the different operators. The clients are large corporations who wish to route their (international) demand on the network. A description of these real life instances is given in table 1, where, for each data set, we describe the number of nodes and arcs in the network, the number of tariff arcs and the number of clients.

The second type of data set consists of randomly generated graphs (subgraphs of grid graphs), where the arc between two nodes is created with a given probability. To each arc, we assign a uniformly distributed random cost. The tariff arcs are also assigned randomly, just as the source and destination and demand of each commodity. These instances are much larger than the real life instances, and are used to illustrate what happens with each algorithm when the number of (tariff) arcs or the number of clients in the network increases. Each of these data sets has a name of the form $'xTyD'$, where x represents the number of tariff arcs and y the number of clients in the network. Moreover, for the randomly created instances, each data set referred to as $'xTyD'$ actually consists of 5 instances. Statistics or execution times of a data set $'xTyD'$ are averages over all 5 instances created. Hence, each formulation developed in this paper has been tested on a total of 165 random instances.

To illustrate the effectiveness of the shortest paths graph model, table 3 and table 4 show some statistics on the number of paths generated for each data set using the shortest paths graph model and its reduction methods for both type of data sets. In these two tables, the column MIN, resp. MAX, indicates the minimum, resp. maximum, number of paths generated over all commodities. The column AVRG gives the average number of paths generated for each commodity for the whole data set. As stated previously, in table 4 the value of MIN, MAX and AVRG for each data set is an average over the 5 instances used. We can see from these two tables the competitiveness of the shortest paths graph model and its reduction methods by the small number of paths generated on average for the given data sets. With this information we can expect the PBB algorithm and the PMIP formulation to benefit from this small number of paths for each commodity.

The performance of the shortest paths graph model was tested using the PBB algorithm, the PMIP and AMIP+ formulations and compared to the mixed integer programming formulation AMIP of Labbé et al. [7], which uses the original arcs in the network and all tariff arcs available.

The computational results for the PBB algorithm, AMIP, AMIP+ and PMIP were established on an AMD Athlon 2400XP+ with 1 Gb RAM, running Debian GNU/Linux 3.0 with kernel 2.4.18. For the AMIP+ formulation, the arcs generated by the shortest paths graph model were used instead of the original arcs in the network. For the PMIP and the PBB algorithm, the size of the set P_k for each commodity $k \in K$ was determined by the number of paths generated by the shortest paths graph model and its reduction techniques. As can be seen from tables 3 and 4, for all data sets this resulted on average in a small number of constraints for each commodity.

Table 2 gives an overview of the results for all algorithms for the real life instances. In this table, the column OPT indicates the value of the optimal solution for the given data set. The column CPU(s), resp. Nodes, indicates the execution time in seconds, resp. the number of nodes in the B&B tree for each algorithm. For the PBB algorithm, the PMIP and the AMIP+, the CPU time includes the time needed for the generation of the shortest paths graph model and all relevant paths. As can be seen in table 2, the PBB algorithm efficiently uses the few relevant paths generated to find the optimal solution for each data set. When we compare its execution to the AMIP, we see that the execution time of the PBB algorithm is less or equal to the running time of CPLEX for the AMIP for each data set considered. The difference is especially large for the data sets D2, D3, D6, D7 and D8. The most striking difference is for the data set D3, for which the execution time of the AMIP is 113423 seconds, whereas the PBB algorithm takes only 30 seconds. The PMIP and AMIP+ columns show the efficiency and power of the shortest paths graph model used in this paper. The execution time of the PMIP is 3 seconds or less for all data sets, except the data set D6. For this data set the execution time is however still much less than the execution time of the AMIP and less than the time needed for the PBB algorithm to execute. The AMIP+ column shows furthermore that the AMIP formulation can benefit very much from the shortest paths graph model: for all data sets, the time needed for the AMIP+ formulation is a little more than the time needed for the PMIP formulation and much less than the time needed for the same formulation on the original network (AMIP).

Table 5 and table 6 illustrate the behaviour of all formulations developed for larger networks and show what happens when we increase the number of tariff arcs or clients in the network. For these instances, the CPU time of the PMIP, AMIP and AMIP+ does *not* include the time needed for the shortest paths graph model. This value is given in the column SPGM. The maximal execution time for each solution method was set to 3600 seconds. If for a certain formulation or algorithm an instance could not be solved within this time, the column CPU indicates the number of instances where the time limit was exceeded, while the column Nodes gives the average percentual gap between the best (optimal) solution as found by any of the other algorithm within the time limit. All instances were solved to optimality by both the PMIP or AMIP+ formulation.

The numerical results for the random instances show that the performance of the algorithms developed is consistent with the numerical results for the real life instances: the PMIP and AMIP+ are the fastest, followed by respectively the PBB and AMIP. For the PMIP, AMIP+ and PBB algorithms, the numerical results show also that the time needed for the generation of the SPGM is the largest part of the execution time of the algorithm. When the size of the network increases, the shortest paths graph model takes more time to generate, but its execution time is more dependent on the number of shortest paths that need to be calculated (depending on the number of tariff arcs or clients in the network) than on the size of the network each shortest path calculation is executed on. The execution time of the SPGM increases when the number of clients is increased, but is especially sensitive to the number of tariff arcs in the network. We can furthermore conclude that although the time needed for the SPGM generation increases when the number of tariff arcs or clients increases, the AMIP+ and PMIP formulation still use the information generated by the shortest paths graph model very efficiently when compared to the PBB algorithm or AMIP formulation, where an increase in the number of clients in the network is soon a problem. For example, for the networks with 100 nodes and 2000 arcs, the data set $90T50D$, consisting thus of 90 tariff arcs and 50 clients, cannot be solved in 4 out of 5 instances for the PBB algorithm and 5 out of 5 instances for the AMIP formulation.

We can also note that although the PMIP and AMIP+ are much faster than the PBB algorithm and the AMIP, the advantage of the PBB algorithm over the PMIP, AMIP+ and AMIP is that it does not need a powerful mixed integer programming solver. Since very basic techniques have been used both with respect to concept and implementation, there is still room for improvement on this algorithm.

8. Summary

In this paper we studied the tariff-setting problem. After introducing the problem and its complexity we propose in a first part of the paper a remodeling of the network referred to as shortest paths graph model, which in our opinion, reduces the problem and captures the essential structure of solutions to the problem. The available arcs and paths in the shortest paths graph model are furthermore reduced by processing techniques using dominance criteria among paths.

In the second part of the paper, we use this remodeling of the network in three solution methods: a combinatorial branch and bound (PBB), a path oriented mixed integer programming formulation (PMIP) and an adaptation of the arc oriented mixed integer programming formulation described by Labbé et al. [7] and referred to as AMIP+. These three solution methods are compared to the general arc oriented mixed integer program (AMIP) as described by Labbé et al. [7] and tested on two type of instances: real life instances provided by France Télécom Research and Development and randomly generated grid graphs.

It is shown that the shortest paths graph model is very helpful in remodeling the network and that formulations based on this remodeled network are very efficient, as demonstrated by the numerical results of the PMIP, AMIP+ and PBB algorithms. The execution time of the AMIP+ and AMIP formulations illustrate that existing formulations (AMIP) can also benefit from the information provided by the shortest paths graph model.

As a possible future extension, the paths generated by the shortest paths graph model and its reduction methods could be used in other models based on a path formulation of the tarification problem. Based on the numerical results given in section 7, such a path formulation could benefit from the small number of paths generated.

Acknowledgment

This research was supported by France Télécom Research & Development. The authors would like to thank DONET for additional financial support. Furthermore, the authors are grateful to M. Labbé , P. Marcotte and G. Savard for their suggestions and comments.

References

[1] L. Brotcorne, M. Labbé, P. Marcotte, and G. Savard. A bilevel model and solution algorithm for a freight tariff-setting problem. *Transportation Science*, 34(3):289 – 302, 2000.

[2] L. Brotcorne, M. Labbé, P. Marcotte, and G. Savard. A bilevel model for toll optimization on a multicommodity transportation network. *Transportation Science*, 35(4):345–358, 2001.

[3] C. Fisk. A conceptual framework for optimal transportation systems planning with integrated supply and demand models. *Transportation Science*, 20:37–47, 1986.

[4] B. Fortz and M. Thorup. Internet traffic engineering by optimizing ospf weights. In *Proc. 19th IEEE Conf. on Computer Communications (INFOCOM)*, pages 519–528, 2000.

[5] O. Jahn, R. Möhring, and A. Schulz. Optimal routing of traffic flows with length restrictions. Working paper 658, TU Berlin, 1999.

[6] R.G. Jeroslow. The polynomial hierarchy and a simple model for competitive analysis. *Mathematical Programming*, 32:146–164, 1985.

[7] M. Labbé, P. Marcotte, and G. Savard. A bilevel model of taxation and its application to optimal highway pricing. *Management Science*, 44:1608–1622, 1998.

[8] T. Roughgarden and Tardos E. How bad is selfish routing? In *Proceedings of the 41st Annual IEEE Symposium on the Foundations of Computer Science*, 2000.

[9] L.N. Vicente and P. H. Calamai. Bilevel and multilevel programming: A bibliography review. *Journal of Global Optimization*, 5:291–306, 1994.

Appendix

Table 1: Description of France Télécom instances.

Data	#Nodes	#Arcs	#Tariff Arcs	#Clients
D1	29	92	6	13
D2	29	98	11	13
D3	43	176	10	20
D4	60	212	9	22
D5	60	212	10	8
D6	60	212	17	8
D7	60	212	21	18
D8	49	116	9	23
D9	33	116	15	30

Table 2: CPU times for France Télécom instances.

		PMIP		**PBB**		**AMIP+**		**AMIP**	
Data	**OPT**	*CPU(s)*	*Nodes*	*CPU(s)*	*Nodes*	*CPU(s)*	*Nodes*	*CPU(s)*	*Nodes*
D1	6.6225	0	1	0	52	1	8	2	187
D2	7.7928	0	10	1	201	0	49	8	1060
D3	728.435	1	55	30	14856	2	786	113423	13619345
D4	1321	1	78	37	21523	6	2445	208	10305
D5	995	1	6	0	185	1	16	1	35
D6	1426	35	134	39	1195	39	1246	399	71013
D7	1565	3	257	25	9689	8	1254	1426	76003
D8	664406	0	110	23	16709	1	32	317	29509
D9	189180	1	112	11	6129	1	132	56	3141

Table 3: SPGM statistics for France Télécom instances.

Data	MIN	MAX	AVRG
D1	2	3	2.5
D2	2	5	3.1
D3	2	8	5.5
D4	2	15	4.2
D5	2	20	5.0
D6	2	30	12.0
D7	2	9	4.8
D8	2	10	4.0
D9	2	8	3.1

Table 4: SPGM statistics for random instances.

<table>
<tr><th colspan="8"><b>Networks with 100 nodes and 2000 arcs</b></th></tr>
<tr><th><b>Data</b></th><th><i>MIN</i></th><th><i>MAX</i></th><th><i>AVRG</i></th><th><b>Data</b></th><th><i>MIN</i></th><th><i>MAX</i></th><th><i>AVRG</i></th></tr>
<tr><td>20T15D</td><td>2</td><td>2.8</td><td>2.1</td><td>90T20D</td><td>2</td><td>5.8</td><td>2.8</td></tr>
<tr><td>30T15D</td><td>2</td><td>3.4</td><td>2.2</td><td>90T30D</td><td>2</td><td>6.4</td><td>2.8</td></tr>
<tr><td>40T15D</td><td>2</td><td>3.6</td><td>2.3</td><td>90T40D</td><td>2</td><td>6.4</td><td>2.8</td></tr>
<tr><td>50T15D</td><td>2</td><td>4.0</td><td>2.6</td><td>90T50D</td><td>2</td><td>6.4</td><td>2.8</td></tr>
<tr><td>60T15D</td><td>2</td><td>5.0</td><td>2.8</td><td>90T60D</td><td>2</td><td>6.8</td><td>2.8</td></tr>
<tr><td>70T15D</td><td>2</td><td>5.4</td><td>3.1</td><td>90T70D</td><td>2</td><td>7.2</td><td>2.8</td></tr>
<tr><td>80T15D</td><td>2</td><td>5.4</td><td>3.1</td><td>90T80D</td><td>2</td><td>7.8</td><td>2.8</td></tr>
<tr><td>90T15D</td><td>2</td><td>7.2</td><td>3.4</td><td>90T90D</td><td>2</td><td>7.8</td><td>2.8</td></tr>
<tr><td>100T15D</td><td>2</td><td>7.2</td><td>3.5</td><td>90T100D</td><td>2</td><td>8.0</td><td>2.8</td></tr>
<tr><th colspan="8"><b>Networks with 75 nodes and 4000 arcs</b></th></tr>
<tr><th><b>Data</b></th><th><i>MIN</i></th><th><i>MAX</i></th><th><i>AVRG</i></th><th><b>Data</b></th><th><i>MIN</i></th><th><i>MAX</i></th><th><i>AVRG</i></th></tr>
<tr><td>20T20D</td><td>2</td><td>3.6</td><td>2.2</td><td>100T30D</td><td>2</td><td>4.6</td><td>2.5</td></tr>
<tr><td>30T20D</td><td>2</td><td>4.0</td><td>2.3</td><td>100T40D</td><td>2</td><td>4.8</td><td>2.6</td></tr>
<tr><td>40T20D</td><td>2</td><td>4.2</td><td>2.4</td><td>100T50D</td><td>2</td><td>6.2</td><td>2.6</td></tr>
<tr><td>50T20D</td><td>2</td><td>4.4</td><td>2.5</td><td>100T60D</td><td>2</td><td>6.2</td><td>2.6</td></tr>
<tr><td>60T20D</td><td>2</td><td>5.0</td><td>2.6</td><td>100T70D</td><td>2</td><td>6.4</td><td>2.6</td></tr>
<tr><td>70T20D</td><td>2</td><td>5.4</td><td>2.8</td><td>100T80D</td><td>2</td><td>6.8</td><td>2.6</td></tr>
<tr><td>80T20D</td><td>2</td><td>5.8</td><td>2.9</td><td></td><td></td><td></td><td></td></tr>
<tr><td>90T20D</td><td>2</td><td>6.0</td><td>3.0</td><td></td><td></td><td></td><td></td></tr>
<tr><td>100T20D</td><td>2</td><td>6.8</td><td>3.1</td><td></td><td></td><td></td><td></td></tr>
</table>

Table 5: CPU times for networks with 100 nodes and 2000 arcs

	SPGM	**PMIP**		**PBB**		**AMIP+**		**AMIP**	
Data	*CPU(s)*	*CPU(s)*	*Nodes*	*CPU(s)*	*Nodes*	*CPU(s)*	*Nodes*	*CPU(s)*	*Nodes*
20T15D	3.4	0.2	2	0.2	164.8	0.4	5	38.6	121.2
30T15D	7	0.6	4.2	0.1	206.4	0.2	11.8	51.2	163.8
40T15D	11.6	0.2	2.2	0.4	215.6	0.4	3	142.2	934.2
50T15D	17.8	0.1	3.4	0.8	305.4	0.6	4.2	182.6	1105.8
60T15D	26.2	0.2	5	1.6	720.2	0.8	7.6	203.8	1096
70T15D	37	0.4	9.2	1.8	710.4	0.8	35.8	452.8	2868.4
80T15D	51.4	0.4	9.6	2.4	786	0.6	29.4	772.6	5763.6
90T15D	69	0.8	9	3.4	988.8	1.4	19.2	445.6	2876.4
100T15D	90	0.8	9	4.2	1038.4	1.2	29.2	754.2	5457.4
90T20D	90.4	0.6	2.8	2.6	555.2	1.6	9.4	856.6	3208.2
90T30D	134.2	0.8	12	69	12545.6	2.4	47.8	(3/5)	13.09%
90T40D	177.8	1.6	11	1243.6	206068.2	2.8	50.8	(4/5)	27.50%
90T50D	221.2	2.8	58.2	(4/5)	0.48%	3.6	137.4	(5/5)	26.14%
90T60D	264.6	3.6	54	(4/5)	3.77%	5.2	163	(5/5)	31.48%
90T70D	309	4.4	241.6	(5/5)	8.31%	7	386.4	(5/5)	28.47%
90T80D	352.8	5.2	378.4	(5/5)	10.85%	10	705.4	(5/5)	34.55%
90T90D	396.2	6.6	702.4	(5/5)	13.58%	22.6	2506.6	(5/5)	33.78%
90T100D	441	13	3168.6	(5/5)	16.39%	51.8	7460	(5/5)	35.44%

Table 6: CPU times for networks with 75 nodes and 4000 arcs

	SPGM	**PMIP**		**PBB**		**AMIP+**		**AMIP**	
Data	*CPU(s)*	*CPU(s)*	*Nodes*	*CPU(s)*	*Nodes*	*CPU(s)*	*Nodes*	*CPU(s)*	*Nodes*
100T30D	316.6	3.2	8.6	46.8	9717	4.8	17.4	332.33	179.33
100T40D	420.4	4.6	27.4	706.4	114278.6	6.4	44	(2/5)	35.58%
100T50D	523.8	5.8	118.8	(3/5)	1.82%	7.4	166.6	(5/5)	22.18%
100T60D	627.8	7	1647.6	(5/5)	5.86%	8.6	192.8	(5/5)	26.26%
100T70D	732.4	12.8	3672.8	(5/5)	9.72%	12.2	1281.6	(5/5)	17.04%
100T80D	835.6	12.4	2142	(5/5)	15.21%	16.8	2369.6	(5/5)	17.38%
20T20D	9.4	0.4	8.2	0.8	477.4	0.4	6.2	466	960
30T20D	17.8	0.6	16.4	1.4	959.8	0.6	16.8	672.2	1245.2
40T20D	28.4	1	10.4	2	1128.8	1.2	14.8	1267.6	2909.2
50T20D	44.2	1	10	2.2	1143.2	1.4	15.4	1425.4	3330
60T20D	64.2	1.4	21.6	3.6	1385.4	1.4	20.4	(1/5)	4.41%
70T20D	90.6	1	23.4	4.8	1622.8	1.6	30.4	(3/5)	41.81%
80T20D	123.8	1.2	23	5.6	1759.8	1.8	32.6	(2/5)	21.30%
90T20D	163.8	1.6	20.6	6.4	1844.8	2.8	32.4	(3/5)	40.83%
100T20D	214.4	2.6	45.2	10.4	2499.2	3.6	25.8	(3/5)	39.40%