An Approximation Algorithm for Stackelberg Network Pricing

Sébastien Roch*
Department of Statistics, University of California, Berkeley, Berkeley, California

Gilles Savard
Département de mathématiques et de génie industriel, École Polytechnique de Montréal, Montréal, Québec, Canada

Patrice Marcotte
Département d'informatique et de recherche opérationnelle, Université de Montréal, Montréal, Québec, Canada

We consider the problem of maximizing the revenue raised from tolls set on the arcs of a transportation network, under the constraint that users are assigned to toll-compatible shortest paths. We first prove that this problem is strongly NP-hard. We then provide a polynomial time algorithm with a worst-case precision guarantee of $\frac{1}{2}\log_2 m_T + 1$, where m_T denotes the number of toll arcs. Finally, we show that the approximation is tight with respect to a natural relaxation by constructing a family of instances for which the relaxation gap is reached. **NETWORKS, Vol. 46(1), 57–67 2005**

Keywords: network pricing; approximation algorithms; Stackelberg games; combinatorial optimization; NP-hard problems

1. INTRODUCTION

This article focuses on a bilevel problem that arises naturally when tariffs, tolls, or devious taxes are to be determined over a network, and is relevant to applications encountered in the transportation, telecommunication, and airline industries. Our aim in this work is twofold. First, we show that the problem is NP-hard. Next, we present a polynomial time approximation algorithm, and prove the tightness of the approximation factor.

Received March 2003; accepted April 2005
Correspondence to: P. Marcotte; e-mail: marcotte@iro.umontreal.ca
Contract grant sponsor: NSERC
Contract grant sponsor: NATEQ
*This work was done while the first author was at École Polytechnique, Montréal.
DOI 10.1002/net.20074
Published online in Wiley InterScience (www.interscience.wiley.com).

Bilevel programming is the relevant framework for modeling situations where one player (the "leader") integrates within its optimization schedule the reaction of a second player (the "follower") to its own course of action. These problems are closely related to static Stackelberg games and mathematical programs with equilibrium constraints (or MPECs (see [12]), in which the lower level solution characterizes the equilibrium state of a physical or social system. Various problems that arise in operations research, economics, and finance can be modeled as bilevel programs. For instance, one may consider the maximization of social welfare, taking into account the selfish behavior of consumers. Indeed, it is well known that the taxation of resources and services at marginal cost (Pigovian taxes, see [15]) maximizes global welfare. However, when some resources fall outside the control of the leader, the social optimum might not be reachable, yielding a "second-best" problem of true Stackelberg nature (see [5, 9, 20] for traffic examples). In constrast with these studies, we adopt the point of view of a firm involved in the management of the network but oblivious to social welfare; the firm's only goal is to maximize its own revenue.

Bilevel programs are generally nonconvex and nondifferentiable, that is, to all practical extent, intractable. In particular, it has been shown by Jeroslow [7] that linear bilevel programming is NP-hard. This result has been refined by Vicente et al. [21], who proved that obtaining a mere certificate of local optimality is strongly NP-hard. Actually, Audet et al. [1] unveiled a close relationship between bilevel programming and integer programming. This "intractability" has prompted the development of heuristics that are adapted to the specific nature of the instance under consideration, together with their worst-case analysis. Such analysis

was first performed for a network design problem with user-optimized flows by Marcotte [13], who proved worst-case bounds for convex optimization based heuristics. More recently, worst-case analysis of Stackelberg problems has been applied to job scheduling and to network design by Roughgarden [17, 18], to network routing by Korilis et al. [8] and to pricing of computer networks by Cocchi et al. [3]. All these works focus on "soft" Stackelberg games, where the objectives of both players are nonconflicting, and where heuristics are expected to perform well in practice, although their worst-case behavior may turn out to be bad.

In this article, we analyze an approximation algorithm for the toll optimization problem (MAXTOLL in this work) formulated and analyzed by Labbé et al., [10]. In this game, which is antagonistic, a leader sets tolls on a subset of arcs of a transportation network, while network users travel on shortest paths with respect to the cost structure induced by the tolls. Labbé et al. [10] proved that the Hamiltonian path problem can be reduced to a version of MAXTOLL involving *negative* arc costs and *positive* lower bounds on tolls. (It was also shown recently by Marcotte et al., [14] that the TSP is a special case of MAXTOLL.) In this article, we improve this result by showing that MAXTOLL, without lower bound constraints on tolls, is strongly NP-hard. Next, in the single-commodity case, we provide a polynomial time algorithm with a performance guarantee of $\frac{1}{2}\log_2 m_T + 1$, where m_T denotes the number of toll arcs in the network. We then use this result and specially constructed instances to prove the tightness of our analysis, as well as the optimality of the approximation factor obtained with respect to a natural upper bound.

The rest of the article is organized as follows. In Section 2, we state the problem and prove that it is NP-hard. In Section 3 we propose an approximation algorithm whose performance is analyzed in Section 4.

2. THE MODEL AND ITS COMPLEXITY

2.1. The Model

The generic *bilevel toll problem* can be expressed as

$$\max_{T} Tx$$

where x is the partial solution of the parametric linear program

$$\begin{aligned} \min_{x,y} \quad & (c_1 + T)x + c_2 y \\ \text{s.t.} \quad & A_1 x + A_2 y = b \\ & x, y \geq 0. \end{aligned}$$

In the above, T represents a *toll vector*, x the vector of *toll commodities* and y the vector of *toll-free commodities*.

We shall consider a combinatorial version of this problem. Let $G = (V, A)$ be a directed multigraph with two distinguished vertices: the origin $s \in V$ and the destination $t \in V$. The arc set A is partitioned into subsets A_T and A_U of *toll* and *toll-free* arcs, of respective cardinalities m_T and m_U. Arcs are assigned *fixed costs* $c : A_T \to \mathbf{N}^{m_T}$ and $d : A_U \to \mathbf{N}^{m_U}$ (in the sequel, $\mathbf{N} = \{0, 1, \ldots\}$). Once *tolls* are added to the fixed costs of A_T, we obtain a *toll network* $\mathcal{N}_T = (G, c + T, d, s, t)$. Denoting by $\mathcal{SP}[\mathcal{N}_T]$ the set of shortest paths from s to t, we can then formulate MAXTOLL as the combinatorial optimization problem (see also Fig. 1):

INSTANCE: – a directed graph $G = (V, A_T \cup A_U)$
– fixed cost vectors $c : A_T \to \mathbf{N}^{m_T}$ and $d : A_U \to \mathbf{N}^{m_U}$
– distinguished vertices $s, t \in V$ such that there exists a simple path from s to t in (V, A_U)

SOLUTION: – a nonnegative toll vector T on A_T
– a simple $s - t$ path P of minimal length w.r.t. the cost structure $(c + T, d)$

MEASURE: – maximize $\sum_{e \in A_T \cap P} T(e)$

FIG. 1. MAXTOLL.

$$\max_{\substack{T \geq 0 \\ P \in \mathcal{SP}[\mathcal{N}_T]}} \sum_{e \in A_T \cap P} T(e). \tag{1}$$

This is a single-commodity instance of the toll setting problem analyzed by Labbé et al. [10].

In this framework, the leader must strike the right balance between low toll levels, which generate low revenue, and high levels, which could also result in low revenue, as the follower would select a path with few toll arcs, or even none. An instance of MAXTOLL is illustrated in Figure 2.

Several remarks are in order. Firstly, to avoid a trivial situation, we posit the existence of at least one toll-free path from s to t. Second, our formulation implies that, given ties at the lower level (the user's level), the leader can choose, among the toll-compatible shortest paths, the one to be travelled by the follower. Indeed, the leader could always force the use of the most profitable path by subtracting a small amount from every toll on that path. Third, once a path P has been selected by the leader, toll arcs outside P become irrelevant. In practice, the removal of these arcs can be achieved by setting tolls to an arbitrarily large value on toll arcs outside P. We denote by $\mathcal{N}_T(P)$ the network where these arcs have been removed. Finally, our central results also hold in a version of MAXTOLL where T is unconstrained; in this case, negative tolls can be interpreted as subsidies. Actually, Labbé et al. [11] have constructed instances where the optimal solution involves negative tolls. Nevertheless, throughout most of the article we focus on nonnegative tolls because (1) this case is interesting in its own sake, (2) intermediate results are easier to interpret when tolls are thought to be nonnegative.

A natural upper bound on the leader's revenue has been derived by Labbé et al. [10] using duality arguments from linear programming theory. It also follows from Theorem 4 of Section 3.1. Let $\mathcal{L}_\infty$ be the length of a shortest toll-free path

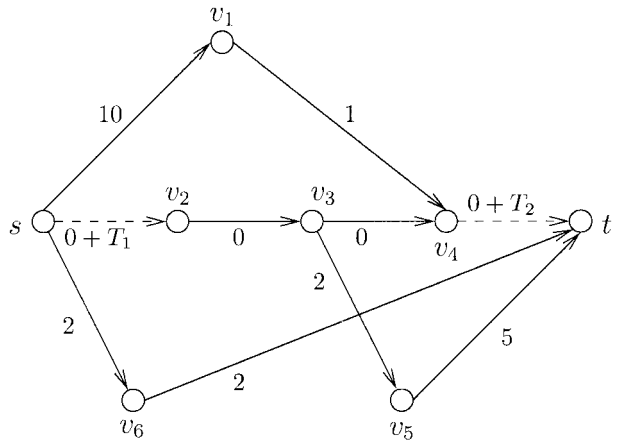

FIG. 2. This network contains two toll arcs (represented by dashed arcs). Fixed costs are given by numbers close to each arc. The optimal path is (s, v_2, v_3, v_4, t), with a revenue of 4 when $T_1 = 2$ and $T_2 = 2$.

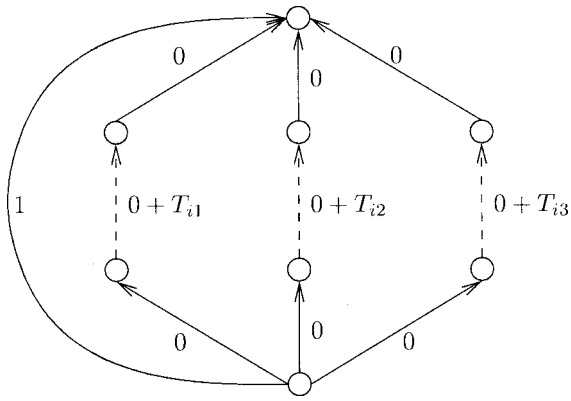

FIG. 3. Subnetwork for clause $(l_{i1} \vee l_{i2} \vee l_{i3})$.

and let $\mathcal{L}(P)$ be the length of a given path P with $T(e) \equiv 0$ for all toll arcs e.

Theorem 1. *Let P be a path. Then the optimal revenue associated with P is bounded above by*

$$B(P) \equiv \mathcal{L}_\infty - \mathcal{L}(P). \tag{2}$$

Because $\mathcal{L}_\infty$ does not depend on P, it follows that the largest upper bound corresponds to the path with smallest value of $\mathcal{L}(P)$; that is, P is a shortest path when tolls are set to 0. We denote the length of such a path by $\mathcal{L}_0$ and by

$$LP = \mathcal{L}_\infty - \mathcal{L}_0 \tag{3}$$

the value of a path-independent upper bound. This bound is simply the difference between the costs of shortest paths corresponding to infinite and null tolls, respectively. Note that, if the set of toll arcs is a singleton, the upper bound can always be achieved.

2.2. *NP-Hardness of* MAXTOLL

The purpose of this section is to show that MAXTOLL is strongly NP-hard. We also prove that a version of MAXTOLL where the toll vector is unconstrained shares this property, thus settling a conjecture of Labbé et al. [10] about the complexity status of the generic toll setting problem.

Theorem 2. MAXTOLL *is strongly NP-hard.*

Proof. Let C denote the sum of all fixed costs. It is not difficult to show that there exists an optimal toll vector T that is integer-valued and less than $C + 1$; in particular, optimal solutions are of polynomial size.

Now, consider a reduction from 3-SAT to MAXTOLL (see [4]). Let $x_1, \ldots, x_n$ be n Boolean variables and

$$F = \bigwedge_{i=1}^{m} (l_{i1} \vee l_{i2} \vee l_{i3}) \tag{4}$$

be a 3-CNF formula consisting of m clauses with literals (variables or their negations) l_{ij}. For each clause, we construct a subnetwork comprising one toll arc for each literal as shown in Figure 3.

The idea is as follows: if the optimal path goes through toll arc T_{ij}, then the corresponding literal l_{ij} is TRUE (note: if $l_{ij} = \bar{x}_k$, then $x_k =$ FALSE). The subnetworks are connected by two arcs, a toll-free arc of cost 2 and a toll arc of cost 0, as shown in Figure 4.

If F is satisfiable, we want the optimal path to go through a single toll arc per subnetwork (i.e., one TRUE literal per clause) and simultaneously want to make sure that the corresponding assignment of variables is consistent; that is, paths that include a variable and its negation must be ruled out. For that purpose, we assign to every pair of literals corresponding to a variable and its negation an interclause toll-free arc between the corresponding toll arcs (see Fig. 4). As we will see, this implies that *inconsistent* paths, involving a variable and its negation, are suboptimal.

Because the length of a shortest toll-free path is $m + 2(m-1) = 3m - 2$ and that of a shortest path with zero tolls is 0, $3m-2$ is an upper bound on the revenue. We claim that F is satisfiable if and only if the optimal revenue is equal to that bound.

Assume that the optimal revenue is equal to $3m-2$. Obviously, the length of the optimal path when tolls are set to 0 must be 0, otherwise the upper bound cannot be reached. To reach the upper bound, the optimal path has to go through one toll arc per subnetwork (it cannot use interclause arcs) and tolls have to be set to 1 on selected literals, $C+1$ on other literals and 2 on tolls T_k, $\forall\, k$. We claim that the optimal path does not include a variable and its negation. Indeed, if that were the case, the interclause arc joining the corresponding toll arcs would impose a constraint on the tolls between its endpoints. In particular, because the fixed cost of the interclause arc is 1, the toll T_k immediately following the initial vertex of this interclause arc would have to be set at most to 1, instead of 2. This yields a contradiction. Therefore, the optimal path must correspond to a consistent assignment, and F is satisfiable (note: if a variable and its negation do not appear on the optimal path, this variable can be set to any value).

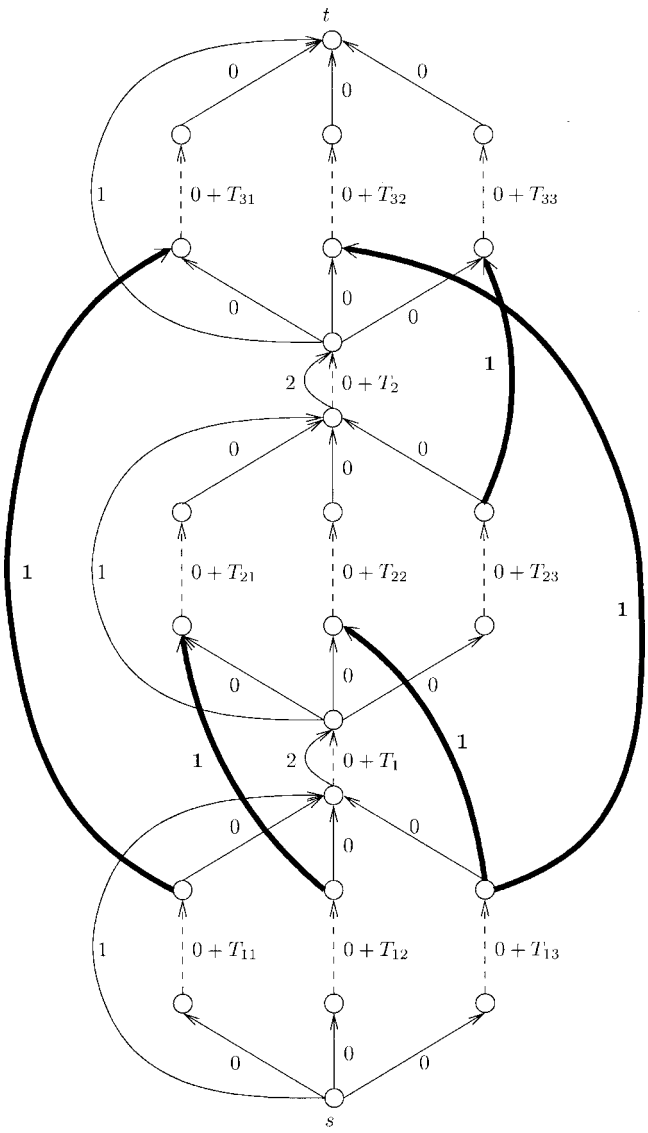

FIG. 4. Network for the formula $(x_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_2 \vee x_3 \vee \bar{x}_4) \wedge (\bar{x}_1 \vee x_3 \vee x_4)$. Interclause arcs are bold. Path through T_{12}, T_{22}, T_{32} is optimal ($x_2 = x_3 =$ TRUE).

Conversely, if F is satisfiable, at least one literal per clause is TRUE in a satisfying assignment. Consider the path going through the toll arcs corresponding to these literals. Because the assignment is consistent, the path does not simultaneously include a variable and its negation, and no interclause arc limits the revenue. Thus, the upper bound of $3m-2$ is reached on this path.

Finally, note that the number of arcs in the reduction is less than

$$10m + 2(m-1) + (3m)^2$$

and that all constants are polynomially bounded in m. ∎

It is not difficult to prove that the same NP-hardness reduction works when negative tolls are allowed.

Theorem 3. MAXTOLL *is still strongly NP-hard when negative tolls are allowed.*

Proof. We use the same reduction as in the nonnegative case. The proof rests on two results proved in [10]. First, the upper bound is valid when T is unrestricted. Second, there exists optimal solutions of polynomial size. The latter result follows from a polyhedral characterization of the feasible set.

From the first result, we know that the $3m-2$ upper bound on the revenue is unchanged. On the other hand, if F is satisfiable, the feasible solution considered in the nonnegative case still yields a $3m-2$ revenue. We only have to make sure that negative tolls cannot produce a $3m-2$ revenue when F is not satisfiable. Again, to reach the upper bound, one has to use a path of length 0 when tolls are set to 0. Consequently, the optimal path comprises exactly one literal per clause. Now the toll-free arcs of length 1 and 2 limit the T_{ij}s on the path to 1 and the T_ks to 2, so negative tolls are useless in this case. Indeed, inconsistent paths will see their revenue limited by interclause arcs without any possibility to make up for the loss incurred from negative tolls. ∎

3. ALGORITHM

In this section, we devise a polynomial-time approximation algorithm for MAXTOLL. Such an algorithm is guaranteed to compute a feasible solution with objective at least OPT/α, where OPT is the optimal revenue and α, which depends on the number m_T of toll arcs, denotes the approximation factor. For a survey of recent results on approximation algorithms, the reader is referred to Hochbaum [6], Vazirani [19], and Ausiello et al. [2].

3.1. Preliminaries

The leader is only interested in paths that have the potential of generating positive revenue. This remark warrants the following definitions. Recall that $\mathcal{N}_T(P)$ is the network $\mathcal{N}_T$ in which toll arcs outside the path P have been removed.

Definition 1. *Let m_T^P denote the number of toll arcs in a path P from s to t. We say that P is* **valid** *if $m_T^P \geq 1$ and P is a shortest path with respect to a null toll vector T, i.e., $P \in \mathcal{SP}[\mathcal{N}_0(P)]$.*

It is clear that nonvalid paths cannot generate revenue. Any valid path P can be expressed as a sequence

$$P = \left(\upsilon_{0,1},\ \tau_1,\ \upsilon_{1,2},\ \tau_2, \ldots, \tau_{m_T^P},\ \upsilon_{m_T^P, m_T^P+1}\right) \qquad (5)$$

where τ_i is the i-th toll arc of P (in the order of traversal) and $\upsilon_{i,i+1}$ is the toll-free subpath of P from the terminal vertex TERM(τ_i) of τ_i to the initial vertex INIT(τ_{i+1}) of τ_{i+1}. According to this notation, $\upsilon_{0,1}$ starts in s and $\upsilon_{m_T^P, m_T^P+1}$ ends in t. Because $P \in \mathcal{SP}[\mathcal{N}_0(P)]$, $\upsilon_{i,i+1}$ is a shortest toll-free path from TERM(τ_i) to INIT(τ_{i+1}). We extend this notation to $\upsilon_{i,j}$, a shortest toll-free path from TERM(τ_i) to INIT(τ_j), with length $\mathcal{U}_{i,j}$. For $k < l$, let $\mathcal{L}_{k,l}$ be the length of P (i.e., the sum of costs) from TERM(τ_k) to INIT(τ_l) with tolls set to 0, and $\mathcal{T}_{k,l}$

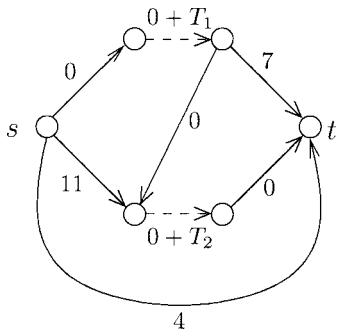

FIG. 5. Equivalent representation of the example of Figure 2. All arcs have been replaced by shortest paths between toll arcs and origin/destination.

be the sum of tolls between $\text{TERM}(\tau_k)$ and $\text{INIT}(\tau_l)$ on P,

$$\mathcal{L}_{k,l} = \sum_{i=k}^{l-1} \mathcal{U}_{i,i+1} + \sum_{i=k+1}^{l-1} c(\tau_i) \qquad \mathcal{T}_{k,l} = \sum_{i=k+1}^{l-1} T(\tau_i), \quad (6)$$

with the convention that $\sum_{i=k}^{l} x_i = 0$ if $l < k$.

Definition 2. *Let P be a valid path. The toll vector T is* **consistent** *with P if $P \in \mathcal{SP}[\mathcal{N}_T(P)]$; that is, P remains a shortest path when tolls outside P are removed and tolls on P are set according to the vector T.*

The following result, which characterizes consistent tolls, is the starting point of our algorithm.

Theorem 4. *Let P be a valid path. Then, the toll vector T is consistent with P if and only if*

$$\mathcal{L}_{i,j} + \mathcal{T}_{i,j} \le \mathcal{U}_{i,j} \qquad \forall\, 0 \le i < j \le m_T^P + 1. \quad (7)$$

Before providing the proof of this theorem, we give an example of its application. Consider again the network of Figure 2. Take the path $P = (s, v_2, v_3, v_4, t)$. Using the renumbering system introduced above, we have that $\tau_1 = (s, v_2)$, $\tau_2 = (v_4, t)$, $v_{0,1}$ is empty, $v_{1,2} = (v_2, v_3, v_4)$ and $v_{2,3}$ is also empty. Other toll-free subpaths that play a role in Theorem 4 are $v_{0,2} = (s, v_1, v_4)$, $v_{1,3} = (v_2, v_3, v_5, t)$, and $v_{0,3} = (s, v_6, t)$. The corresponding lengths are $\mathcal{U}_{0,1} = \mathcal{U}_{1,2} = \mathcal{U}_{2,3} = 0$, $\mathcal{U}_{0,2} = 11$, $\mathcal{U}_{1,3} = 7$, and $\mathcal{U}_{0,3} = 4$. Thus, the inequalities in Theorem 4 are

$$T(\tau_1) \le 11, \qquad T(\tau_2) \le 7, \qquad T(\tau_1) + T(\tau_2) \le 4.$$

From this we conclude that one possible set of optimal taxes is $T(\tau_1) = T(\tau_2) = 2$ as claimed in Figure 2. Figure 5 provides an equivalent representation of this example according to Theorem 4.

Proof of Theorem 4. [$\Longrightarrow$] Obvious from the very definition of consistent tolls.

[$\Longleftarrow$] We need to check that the length of P remains smaller than or equal to the length of any other path when tolls satisfy (7). This looks rather obvious. However, one must be careful about paths that borrow a subset of the toll arcs of P, possibly in a *different* sequence (note that toll-free paths are taken care of by setting $i = 0$ and $j = m_T^P + 1$ above). Assume, by contradiction, that such a path $\widetilde{P}$ is strictly shorter than P in $\mathcal{N}_T(P)$, and that conditions (7) are satisfied. We note

$$\widetilde{P} = \left(\tilde{v}_{0,1},\, \tilde{\tau}_1,\, \tilde{v}_{1,2},\, \ldots,\, \tilde{\tau}_{m_T^{\widetilde{P}}},\, \tilde{v}_{m_T^{\widetilde{P}}, m_T^{\widetilde{P}}+1}\right) \quad (8)$$

with corresponding $\widetilde{\mathcal{U}}_{i,j}$, $\widetilde{\mathcal{L}}_{k,l}$, and $\widetilde{\mathcal{T}}_{k,l}$ for all i, j, k, l, with $k < l$. For all i, $\tilde{\tau}_i = \tau_{\delta(i)}$ for some injective function δ (toll arcs outside P are irrelevant). To derive a contradiction, we will construct a path that is shorter than $\widetilde{P}$ by modifying the "backward" toll-free subpaths of $\widetilde{P}$. This improved path will happen to be P. See Figure 6 for an illustration.

Let $j_1 = 1$ and $j_k = \min\{j > j_{k-1} : \delta(j) > \delta(j_{k-1})\}$, as long as such j_k exists, the last one being denoted j_K. We further define $\delta(0) = j_0 = 0$ and $\delta(m_T^{\widetilde{P}} + 1) = j_{K+1} = m_T^{\widetilde{P}} + 1$. We claim that

$$\mathcal{L}_{\delta(j_{k-1}),\delta(j_k)} + \mathcal{T}_{\delta(j_{k-1}),\delta(j_k)} \le \widetilde{\mathcal{L}}_{j_{k-1},j_k} + \widetilde{\mathcal{T}}_{j_{k-1},j_k},$$

for all k. This implies that we can replace the subpath of $\widetilde{P}$ between $\text{TERM}(\tilde{\tau}_{j_{k-1}})$ and $\text{INIT}(\tilde{\tau}_{j_k})$, by the subpath of P between $\text{TERM}(\tau_{\delta(j_{k-1})})$ and $\text{INIT}(\tau_{\delta(j_k)})$ without increasing its length. But after doing this for all k, we obtain P, which is a contradiction. We divide the claim above in two cases. First, if $j_k - 1 = j_{k-1}$ (i.e. $\widetilde{P}$ "goes in the direction of P", e.g. $\tilde{v}_{0,1}$ in Fig. 6), then

$$\widetilde{\mathcal{L}}_{j_{k-1},j_k} + \widetilde{\mathcal{T}}_{j_{k-1},j_k} = \mathcal{U}_{\delta(j_{k-1}),\delta(j_k)} \ge \mathcal{L}_{\delta(j_{k-1}),\delta(j_k)} + \mathcal{T}_{\delta(j_{k-1}),\delta(j_k)},$$

where we have used (7) (note that in this case, there are no toll arcs on $\widetilde{P}$ between $\text{TERM}(\tilde{\tau}_{j_{k-1}})$ and $\text{INIT}(\tilde{\tau}_{j_k})$). Otherwise, $j_k - 1 \ne j_{k-1}$ (i.e. $\widetilde{P}$ "goes in the direction opposite to P", e.g. $\tilde{v}_{1,2} \to \tilde{\tau}_2 \to \tilde{v}_{2,3}$ in Fig. 6). By definition of j_k, we have $\delta(j_k - 1) \le \delta(j_{k-1})$ for all $1 \le k \le K + 1$. For example, in Figure 6, we have $\tau_{\delta(j_2-1)} = \tilde{\tau}_2$, which clearly comes before $\tau_{\delta(j_1)} = \tilde{\tau}_1$ on P. From this and (7), we get

$$\begin{aligned}
\widetilde{\mathcal{L}}_{j_{k-1},j_k} + \widetilde{\mathcal{T}}_{j_{k-1},j_k} &= \widetilde{\mathcal{L}}_{j_{k-1},j_k-1} + \widetilde{\mathcal{T}}_{j_{k-1},j_k-1} + c(\tau_{\delta(j_k-1)}) \\
&\quad + T(\tau_{\delta(j_k-1)}) + \mathcal{U}_{\delta(j_k-1),\delta(j_k)} \\
&\ge \mathcal{U}_{\delta(j_k-1),\delta(j_k)} \ge \mathcal{L}_{\delta(j_k-1),\delta(j_k)} + \mathcal{T}_{\delta(j_k-1),\delta(j_k)} \\
&\ge \mathcal{L}_{\delta(j_{k-1}),\delta(j_k)} + \mathcal{T}_{\delta(j_{k-1}),\delta(j_k)}.
\end{aligned}$$

■

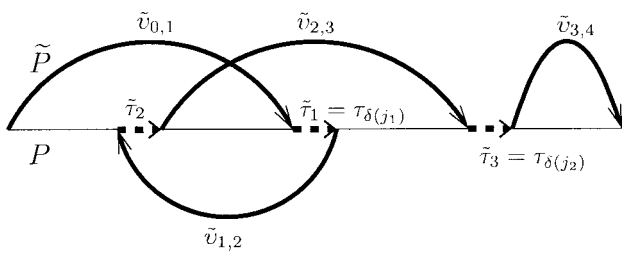

FIG. 6. The thick path is $\tilde{P}$, while the straight thin path is P. An example of the "backward" toll-free subpaths referred to in the proof of Theorem 4 is $\tilde{v}_{1,2} \to \tilde{\tau}_2 \to \tilde{v}_{2,3}$. This gets replaced by the section of P between $\tau_{\delta(j_1)}$ and $\tau_{\delta(j_2)}$, which is clearly shorter.

Algorithm EXPLOREDESCENDANTS

Input: a path P

Output: a path $\overline{P}$, tolls $\overline{T}$ and objective value $\overline{V}$

- Compute maximum revenue V_P achievable on P and corresponding toll vector T_P: $\left(V_P, T_P, \{(i'(k), j'(k))\}_{k=1}^{m_T^P}\right) :=$ MAXREV(P)
- If $V_P < B(P)$ then
  - — Derive new paths from P :

$$(P_1, P_2) := \text{TOLLPARTITION}\left(P, \{(i'(k), j'(k))\}_{k=1}^{m_T^P}\right)$$

  - — For $i = 1, 2$: $(\overline{V}_i, \overline{T}_i, \overline{P}_i) :=$ EXPLOREDESCENDANTS(P_i)
  - — Return $\overline{V} := \max\{V_P, \overline{V}_1, \overline{V}_2\}$ and corresponding toll vector $\overline{T}$ and path $\overline{P}$
- Else return $(\overline{V}, \overline{T}, \overline{P}) := (V_P, T_P, P)$

FIG. 7. Algorithm EXPLOREDESCENDANTS.

Remark. There always exists an optimal solution with tolls less than $C + 1$ (see Theorem 2 for a definition of C). Indeed, $\mathcal{U}_{i,j} < C + 1,\ \forall\, i, j$ implies that optimal tolls on P have to be lower than $C + 1$. Therefore, fixing tolls to $C + 1$ outside P generates no additional active constraints on the tolls of P.

3.2. Description of the Algorithm

EXPLOREDESCENDANTS is an approximation algorithm motivated by the previous characterization of consistent tolls. Initially, the algorithm computes the optimal revenue associated with a shortest path in $\mathcal{SP}[\mathcal{N}_0]$, that is, a path with the largest upper bound $B(P)$. (As shown in the next section, this can be achieved in polynomial time.) If revenue is smaller than the upper bound LP, Theorem 4 implies that there exists a toll-free subpath $\upsilon_{k,l}$ $k < l$ whose short length forces some tolls in P to be small. To relax this constraint, it makes sense to skip the subpath of P between TERM(τ_k) and INIT(τ_l) and to replace it by $\upsilon_{k,l}$. This yields a new path whose length will not be much larger than the length of P, and for which some constraints (7) have been removed. This path is a natural candidate for improved revenue. By repeating this process, we will show that Algorithm EXPLOREDESCENDANTS can find, in polynomial time, a path with good approximation properties.

Algorithm EXPLOREDESCENDANTS is presented in Figure 7. It comprises two subroutines, MAXREV and TOLLPARTITION. MAXREV computes the largest revenue compatible with the shortest path P. Starting from P, TOLLPARTITION generates two descendants of path P. The algorithm is initialized with P_0 in $\mathcal{N}_0$, the shortest path of length $\mathcal{L}_0$.

3.3. Maximizing Path-Compatible Revenue

Let P be a valid path in $\mathcal{N}$ denoted by

$$P = \left(\upsilon_{0,1},\ \tau_1,\ \upsilon_{1,2},\ \ldots,\ \tau_{m_T^P},\ \upsilon_{m_T^P, m_T^P+1}\right) \tag{9}$$

Algorithm MAXREV

Input: a path P

Output: tolls T_P, objective value V_P, and sequence of indices $\{(i'(k), j'(k))\}_{k=1}^{m_T^P}$

- Denote $P = \left(\upsilon_{0,1},\ \tau_1,\ \upsilon_{1,2},\ \tau_2, \ldots, \tau_{m_T^P},\ \upsilon_{m_T^P, m_T^P+1}\right)$, and set $k := 1$
- While $k < m_T^P + 1$, do
  - — Compute

$$T_P(\tau_k) := t_k \equiv \min_{0 \le i < k < j \le m_T^P + 1} \left\{ \mathcal{U}_{i,j} - \mathcal{L}_{i,j} - \sum_{l=i+1}^{k-1} t_l \right\}$$

  - — Let $(i'(k), j'(k))$ be the pair of indices for which the minimum above is attained, breaking ties by selecting the largest $j'(k)$ and corresponding smallest $i'(k)$
  - — Set $(i'(l), j'(l)) := (i'(k), j'(k))$ for all $k < l < j'(k)$
  - — Set $k := j'(k)$
- Let T_P be $+\infty$ outside P and $V_P := \sum_{k=1}^{m_T^P} t_k$
- Return T_P, V_P and $\{(i'(k), j'(k))\}_{k=1}^{m_T^P}$

FIG. 8. Algorithm MAXREV.

with corresponding values of $\mathcal{U}_{i,j}$, $\mathcal{L}_{k,l}$, and $\mathcal{T}_{k,l}$ for all i, j, k, l, with $k < l$. The optimal tolls compatible with P's shortest path status can be obtained from a simple greedy algorithm (see Fig. 8): consider each toll arc in order of traversal, and fix its value to the largest value allowed by Theorem 4, taking into account the tolls previously set. This leads to the recursion

$$T(\tau_k) := t_k \equiv \min_{0 \le i < k < j \le m_T^P + 1} \left\{ \mathcal{U}_{i,j} - \mathcal{L}_{i,j} - \sum_{l=i+1}^{k-1} t_l \right\}. \tag{10}$$

The validity of the above formula rests on Theorem 5 where we construct a sequence of toll-free subpaths such that (1) every toll of P is bounded from above by at least one path in the sequence [condition (13) below], (2) the sum of the tolls defined by (10) is equal to the sum of the bounds imposed by the paths of the sequence [condition (14) below].

Labbé et al. [10] give another polynomial time algorithm for this task, but it does not provide the information we need regarding the active toll-free subpaths. This information will be instrumental in generating new paths from P and in obtaining an approximation guarantee.

Theorem 5. *Let $\upsilon_{i,j}$, $0 \le i < j \le m_T^P + 1$, be the toll-free subpaths defined in Section 3.1 and let t_k, $1 \le k \le m_T^P$ be as in (10). Then, there exists a sequence of paths*

$$\upsilon_{i(1),j(1)},\ \upsilon_{i(2),j(2)}, \ldots,\ \upsilon_{i(q),j(q)} \tag{11}$$

with $i(1) = 0$ and $j(q) = m_T^P + 1$ such that for all h (see Fig. 9)

$$i(h+1) < j(h) \le i(h+2) < j(h+1) \tag{12}$$

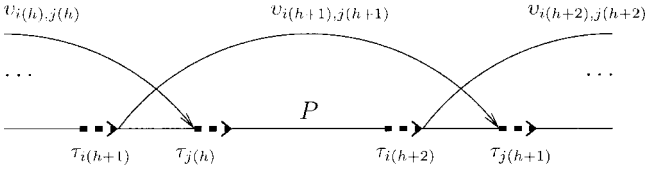

FIG. 9. A section of a path P, showing toll arcs τ_k with $k = i(h+1), j(h), i(h+2), j(h+1)$.

and for all k

$$|\{h \,:\, i(h)+1 \le k \le j(h)-1\}| \ge 1 \tag{13}$$

with equality if $t_k \neq 0$. *This implies that for all* h', h'' *with* $h' \le h''$

$$\sum_{k=i(h')+1}^{j(h'')-1} t_k = \sum_{h=h'}^{h''} \left[\mathcal{U}_{i(h),j(h)} - \mathcal{L}_{i(h),j(h)}\right]. \tag{14}$$

Proof of Theorem 5. This proof is rather cumbersome, so we start by giving an intuitive description. By Theorem 4, we know that tolls are constrained by toll-free subpaths. To obtain the corollary above—which is the ultimate purpose of this theorem—we need to find a subset of constraints such that each toll arc is covered by at least one constraint and such that the sum of the slacks of the constraints is the same as the sum of the t_ks. Natural candidates for saturated constraints are derived from (10): when we fix t_k, at least one constraint becomes saturated. These constraints will be denoted by the sequence of indices $\{(i'(k),j'(k))\}_{k=1}^{m_T^P}$ below. The problem with them is that they might be redundant. For example, in setting t_1 we might saturate the path $v_{0,3}$ and then in setting t_3 we might saturate the path $v_{0,5}$ which "supersedes" $v_{0,3}$. This situation makes it impossible to obtain equality in (14). To remove redundant constraints, a natural strategy is to first compute the sequence $\{(i'(k),j'(k))\}_{k=1}^{m_T^P}$, then to start from t_1 and choose among all constraints in this subset the one that "goes furthest" (say $v_{0,5}$ above). Then go to t_5 and do the same; and so on. This allows the redundant constraints to be skipped. Still, this is not quite what we want because there might be nonzero t_ks in the "overlaps" of the chosen constraints (in Fig. 9, this would correspond to nonzero t_ks between $\tau_{i(h+1)}$ and $\tau_{j(h)}$), which again would give strict inequality in (14). It turns out that using the above strategy but *starting from the end* gives the desired properties. The result of this will be denoted by $\{(i(k),j(k))\}_{k=1}^{q}$ below. To see why this works, consider Figure 9. Say that at some point in our choice of saturated constraints we get to toll arc $\tau_{i(h+2)}$. We then choose the constraint covering $\tau_{i(h+2)}$ [among $\{(i'(k),j'(k))\}_{k=1}^{m_T^P}$] which reaches furthest in the direction of s, here denoted $v_{i(h+1),j(h+1)}$. All arcs between $\tau_{i(h+2)}$ and $\tau_{i(h+1)}$ are now covered (some of those are not represented in Fig. 9), so we go to $\tau_{i(h+1)}$. Likewise, we choose the path $v_{i(h),j(h)}$. Now note that when we set $t_{i(h+1)}$ in the recursion (10), the constraint corresponding to $v_{i(h),j(h)}$ became saturated (or it had been saturated by a previous toll), so that all toll arcs between $\tau_{i(h+1)}$ and $\tau_{j(h)}$ (not represented here) have been set to 0, as claimed. The heart of the proof that follows is to check that this is indeed the case.

Consider the following construction. Start with $k = 1$. In the recursive formula (10), let $(i'(k),j'(k))$ denote the index for which the minimum is attained when setting t_k. This makes sense, because we assumed that there exists a toll-free path from s to t. In case of nonuniqueness, select the largest index j and the corresponding smallest index i. Note that there holds

$$t_l = 0, \quad \forall\, k+1 \le l \le j'(k)-1. \tag{15}$$

Then rather than evaluating (10) for t_{k+1}, we jump from t_k to $t_{j'(k)}$ and set

$$(i'(l),\, j'(l)) = (i'(k),\, j'(k)), \quad \forall\, k+1 \le l \le j'(k)-1. \tag{16}$$

Recursing gives a sequence of indices $\{(i'(k),j'(k))\}_{k=1}^{m_T^P}$.

In view of Theorem 4, which implies that

$$\sum_{k=i(h)+1}^{j(h)-1} T(\tau_k) \le \mathcal{U}_{i(h),j(h)} - \mathcal{L}_{i(h),j(h)}, \tag{17}$$

we say that $v_{i(h),j(h)}$ *covers* the toll arcs τ_k for $i(h)+1 \le k \le j(h)-1$. To derive (14), we look for a subset of $\{v_{i'(k),j'(k)}\}_{k=1}^{m_T^P}$ that covers all toll arcs of P and such that $t_k = 0$ for all arcs τ_k covered by more than one subpath.

We proceed backwards. Select $l_1 = m_T^P$ and recursively compute $l_k = i'(l_{k-1})$ until $l_{q+1} = 0$ for some index q. There follows:

$$j'(l_1) = m_T^P + 1 \quad \text{and} \quad i'(l_q) = 0. \tag{18}$$

Now, reverse the sequence by setting $(i(k), j(k)) = (i'(l_{q+1-k}), j'(l_{q+1-k}))$, $1 \le k \le q$, to obtain a sequence that satisfies the assumptions of Theorem 4:

1. $i(1) = 0, j(q) = m_T^P + 1$: This follows from (18).
2. $i(h) < i(h+1)$: This follows from the construction of the backwards sequence $\{l_i\}_{i=1}^{q}$.
3. $i(h+1) < j(h)$: By contradiction, assume that $j(h) \le i(h+1)$. This implies that $\tau_{j(h)}$ is not covered by a toll-free subpath. This is impossible by the very construction of $\{(i(k),j(k))\}_{k=1}^{q}$.
4. $j(h) \le i(h+2)$: by contradiction, assume that $j(h) > i(h+2)$. Then $(i'(l_{q+1-h}),\, j'(l_{q+1-h})) = (i(h), j(h))$ implies that $i'(l) = i(h)$ for all indices $l_{q+1-h} < l < j'(l_{q+1-h}) = j(h)$ and, in particular, for index $l = l_{q+1-h-1} = i'(l_{q+1-h-2}) = i(h+2)$ [$< j(h)$ by assumption]. This implies the contradiction $i'(l_{q+1-h-1}) = i(h+1) = i(h)$.
5. (13) and (14): by construction, every arc is covered by at least one path. By the preceding inequalities, the only

arcs covered by more than one path must belong to the interval $k = i(h+1)+1, \ldots, j(h)-1$, for some index h. By (15), their tolls must be zero, and (13) is satisfied; (14) then follows from (13).

∎

Corollary 1. *The toll assignment defined by (10) is optimal for P.*

Proof of Corollary 1. For every toll assignment T' consistent with P

$$\sum_{k=1}^{m_T^P} T'(\tau_k) \le \sum_{h=1}^{q} \left[\mathcal{U}_{i(h),j(h)} - \mathcal{L}_{i(h),j(h)}\right], \qquad \text{by (7) and (13)}$$

$$= \sum_{k=1}^{m_T^P} t_k, \qquad \text{by (14).}$$

∎

3.4. Partitioning the Set of Toll Arcs

The approximation algorithm progressively removes toll arcs from the network. It makes use of the following definition.

Definition 3. *A* **descendant** *P' of a valid path $P \in \mathcal{N}_0$ is a simple path from s to t in $\mathcal{N}_0(P)$ that traverses a subset of the toll arcs of P in the same order as does P.*

Let P be a valid path in $\mathcal{N}_0$. Theorem 5 suggests a way of constructing a descendant of P that stands a chance of achieving a high revenue, whenever P's revenue is low. Let us consider the set

$$\upsilon_{i(1),j(1)},\ \upsilon_{i(2),j(2)}, \ldots,\ \upsilon_{i(q),j(q)} \tag{19}$$

of toll-free paths such that

$$0 = i(1) < i(2) < j(1) \le i(3) < j(2) \le i(4) < j(3) \le \cdots \le i(q) < j(q-1) < j(q) = m_T^P + 1 \tag{20}$$

and equality (14) holds. If the maximum revenue on P is $B(P)$ then descendants need not be considered, because their upper bounds are smaller than or equal to $B(P)$. This happens in particular if $q = 1$. We can therefore assume that q is larger than 1.

We now consider two descendants: P_1 contains all $\upsilon_{i(h),j(h)}$ with h odd and is composed of arcs of P between them; P_2 is constructed in a similar manner, with even values of the index h. For instance, P_1 may start in s, borrow $\upsilon_{i(1),j(1)}$, take path P between $\text{INIT}(\tau_{j(1)})$ and $\text{TERM}(\tau_{i(3)})$, bifurcate on $\upsilon_{i(3),j(3)}$, return to P, and so on. Such a pattern, which is allowed by (20), is performed by procedure TOLLPARTITION (see Fig. 10). Note that P_1 and P_2 have no toll arcs in common, and that some toll arcs of P belong neither to P_1 nor to P_2.

Algorithm TOLLPARTITION

Input: a path P and sequence of indices $\{(i'(k), j'(k))\}_{k=1}^{m_T^P}$

Output: two paths P_1, P_2

- Set $l := m_T^P$, $r := 1$
- (Backwards selection of constraints) While $l > 0$, do
  - — Set $(i''(r), j''(r)) := (i'(l), j'(l))$
  - — Set $l := i'(l)$, $r := r + 1$
- Let q be the last index in the loop above
- (Reversing the order of the sequence) For all $1 \le k \le q$, set

$$(i(k), j(k)) := (i''(q-k+1), j''(q-k+1))$$

- (Constructing the descendants) Let P_1 contain all $\upsilon_{i(h),j(h)}$ with h odd and be composed of arcs of P between them; let P_2 be constructed in a similar manner, with even values of the index h
- Return P_1, P_2

FIG. 10. Algorithm TOLLPARTITION.

The rationale behind this construction is the relationship between the maximum revenue achievable on P and the upper bounds on the tolls of P_1 and P_2 given by Theorem 5.

Both these descendants are valid paths. If this were not the case for P_1, there would exist a subpath $\upsilon_{i(h),j(h')}$ ($h < h'$ both odd) such that $\mathcal{U}_{i(h),j(h')}$ is strictly smaller than the length of P_1 between $\text{TERM}(\tau_{i(h)})$ and $\text{INIT}(\tau_{j(h')})$ (indeed, a path is valid if and only if the null toll vector is consistent with it). Because this length is equal to $\mathcal{L}_{i(h),j(h')} + \mathcal{T}_{i(h),j(h')}$ by (14) we obtain that the toll assignment on P is not consistent, a contradiction. A similar argument applies to P_2.

A detailed example of the workings of the algorithm can be found in the online version of the article available on the arXiv e-print archive [16].

4. ANALYSIS OF THE ALGORITHM

4.1. Approximation Bound

We shall prove that EXPLOREDESCENDANTS is an $\frac{1}{2}\log_2 m_T + 1$-approximation algorithm for MAXTOLL. The *exact* approximation factor is given by the recursion

$$\alpha(k) = \frac{1}{2} \max_{\substack{i+j \le k \\ 0 < i \le j < k}} \{1 + \alpha(i) + \alpha(j)\}, \tag{21}$$

with $\alpha(1) = 1$. It can be shown by induction that for all k,

$$\alpha(k) \le \frac{1}{2}\log_2 k + 1$$

Definition 4. *The maximum revenue V_P induced by path P is* **sufficient** *if*

$$V_P \ge \frac{1}{\alpha(m_T^P)} B(P). \tag{22}$$

Theorem 6. *Let P be a valid path on $\mathcal{N}_0$. If the maximum revenue achievable on P is not sufficient, then either path P_1 or P_2 (say P') returned by* TOLLPARTITION *satisfies*

$$\frac{1}{\alpha(m_T^{P'})}B(P') \geq \frac{1}{\alpha(m_T^P)}B(P) \tag{23}$$

Proof. Let

$$P = \left(\upsilon_{0,1},\, \tau_1,\, \upsilon_{1,2},\, \ldots,\, \tau_{m_T^P},\, \upsilon_{m_T^P, m_T^P+1}\right)$$

with corresponding values of $\mathcal{U}_{i,j}$, $\mathcal{L}_{k,l}$, and $\mathcal{T}_{k,l}$ for all $i, j, k, l, k, < l$. Similarly, for $r = 1, 2$,

$$P_r = \left(\upsilon^r_{0,1},\, \tau^r_1,\, \upsilon^r_{1,2},\, \ldots,\, \tau^r_{m_T^{P_r}},\, \upsilon^r_{m_T^{P_r}, m_T^{P_r}+1}\right)$$

with corresponding values of $\mathcal{U}^r_{i,j}$, $\mathcal{L}^r_{k,l}$ and $\mathcal{T}^r_{k,l}$ for all $i, j, k, l, k < l$. Let

$$\upsilon_{i(1),j(1)},\, \upsilon_{i(2),j(2)}, \ldots,\, \upsilon_{i(q),j(q)}$$

be the sequence of toll-free paths obtained from Theorem 5.

For all $h > 1$, one of the paths P_r contains $\upsilon_{i(h-1),j(h-1)}$, while the other contains $\upsilon_{i(h),j(h)}$. Therefore, the subpath of P between TERM($\tau_{i(h)}$) and INIT($\tau_{j(h-1)}$) is in neither P_1 nor P_2. The remaining arcs of P belong to either P_1 or P_2. This implies that

$$\begin{aligned}\mathcal{L}^1_{0,m_T^{P_1}+1} + \mathcal{L}^2_{0,m_T^{P_2}+1} &= \sum_{h=1}^{q} \mathcal{U}_{i(h),j(h)} + \mathcal{L}_{0,m_T^P+1} \\ &\quad - \sum_{h=2}^{q} \mathcal{L}_{i(h),j(h-1)} \\ &= \sum_{h=1}^{q} \left[\mathcal{U}_{i(h),j(h)} - \mathcal{L}_{i(h),j(h)}\right] + 2\mathcal{L}_{0,m_T^P+1}\end{aligned}$$

By Theorem 5, the first term on the right-hand side is equal to $\sum_{k=1}^{m_T^P} t_k$, and is strictly smaller than $B(P)/\alpha(m_T^P)$ by hypothesis. Multiplying by (-1) and adding $2\mathcal{U}_{0,m_T^P+1} = \mathcal{U}_{0,m_T^{P_1}+1} + \mathcal{U}_{0,m_T^{P_2}+1}$ on both sides, we get

$$B(P_1) + B(P_2) > \left(2 - \frac{1}{\alpha\left(m_T^P\right)}\right)B(P).$$

By definition of α:

$$\begin{aligned}\alpha(m_T^P) \geq \frac{1}{2}\left[1 + \alpha(m_T^{P_1}) + \alpha(m_T^{P_2})\right] &\Longrightarrow \left(2 - \frac{1}{\alpha\left(m_T^P\right)}\right) \\ &\geq \frac{\alpha(m_T^{P_1}) + \alpha(m_T^{P_2})}{\alpha(m_T^P)}.\end{aligned}$$

By substituting in the preceding inequality, we obtain

$$\begin{aligned}&\alpha(m_T^{P_1})\left(\frac{1}{\alpha(m_T^{P_1})}B(P_1)\right) + \alpha(m_T^{P_2})\left(\frac{1}{\alpha(m_T^{P_2})}B(P_2)\right) \\ &\qquad \geq (\alpha(m_T^{P_1}) + \alpha(m_T^{P_2}))\left(\frac{1}{\alpha(m_T^P)}B(P)\right),\end{aligned}$$

which yields the desired result. Indeed, if we had

$$\frac{1}{\alpha(m_T^{P_r})}B(P_r) < \frac{1}{\alpha(m_T^P)}B(P) \text{ for } r = 1, 2,$$

this would imply the opposite inequality. ∎

As a corollary, we obtain the main result of this section.

Corollary 2. *Let APP denote the revenue obtained from the application of the procedure* EXPLOREDESCENDANTS *to a path P_0 of shortest length $\mathcal{L}_0$ in $\mathcal{N}_0$. Then,*

$$APP \geq \frac{1}{\alpha(m_T^{P_0})}LP \geq \frac{1}{\alpha(m_T)}OPT. \tag{24}$$

Proof. Let P be a valid path and $(\overline{V}, \overline{T}, \overline{P})$ the output of EXPLOREDESCENDANTS(P). It is sufficient to show, by induction on m_T^P that

$$\overline{V} \geq \frac{1}{\alpha(m_T^P)}B(P). \tag{25}$$

This statement is true if $m_T^P = 1$, because the upper bound $B(P)$ is always achievable on a path with a single toll arc. Now assume that the property holds when the number of toll arcs is less than $m_T^P > 1$. Let $(V_P, T_P) :=$ MAXREV(P). If V_P is sufficient, (25) is satisfied. If V_P is not sufficient then, by Theorem 6, TOLLPARTITION returns a path P' with

$$\frac{1}{\alpha(m_T^{P'})}B(P') \geq \frac{1}{\alpha(m_T^P)}B(P).$$

Because the number of toll arcs in P' is less than that in P, property (25) is satisfied for P', and the preceding inequality implies that (25) is satisfied for P as well. ∎

Note that this result applies to the case of negative tolls as well, because the upper bound (2) is the same. Indeed, EXPLOREDESCENDANTS allows only nonnegative tolls, but it computes a feasible solution with a revenue at least LP/α, where LP is unchanged in the unbounded case.

The approximation algorithm determines, in a constructive manner, an upper bound $\alpha(m_T)$ on the ratio OPT/LP. It can be shown through a family of instances that this bound is tight. A proof of the following theorem can be found in the online version of the article available on the arXiv e-print archive [16].

Theorem 7. *Let $\mathcal{I}(m_T)$ denote the set of instances of* MAX-TOLL *corresponding to a fixed number of toll arcs m_T. Then for all $m_T \geq 1$, the relaxation gap on $\mathcal{I}(m_T)$ is $\alpha(m_T)$, that is*

$$\alpha(m_T) = \max_{I \in \mathcal{I}(m_T)} \left\{\frac{LP[I]}{OPT[I]}\right\}, \tag{26}$$

where LP[I] and OPT[I] denote respectively the upper bound and optimal value for instance I.

4.2. Running Time Analysis

MAXTOLL is initialized with a shortest path P_0, which can be computed in $O(n^2)$ time. The toll arcs of the descendants constitute a subset of the toll arcs in P_0, and their traversal order is the same. Therefore, the values $\mathcal{U}_{i,j}$, $i < j$ computed for P_0 can be reused for all descendants, under an appropriate renumbering. This operation is achieved in $O(m_T^{P_0} n^2) = O(m_T n^2)$ time. The time required to evaluate $\mathcal{L}_{i,j}$, $i < j$, as the algorithm proceeds, is much smaller.

Within EXPLOREDESCENDANTS, MAXREV requires at most $O((m_T^P)^3)$ to compute the maximal revenue induced by path P. Based on the indices $\{(i'(k), j'(k))\}_{k=1}^{m_T^P}$ obtained from MAXREV, TOLLPARTITION generates two descendants in $O(m_T^P)$ time. It follows that the running time of EXPLOREDESCENDANTS on a path P is determined by the recursion

$$\mathbb{T}(m_T^P) = \mathbb{T}(m_T^{P_1}) + \mathbb{T}(m_T^{P_2}) + O((m_T^P)^3) \qquad (27)$$

where $m_T^{P_1} + m_T^{P_2} \leq m_T^P$. Therefore, the worst-case complexity, achieved when $m_T^{P_1}$ is always equal to $m_T^P - 1$, is $O((m_T^P)^4)$. The worst-case running-time of the entire algorithm is $O(m_T(m_T^3 + n^2))$.

5. CONCLUSION

Our algorithm can also be applied to the multicommodity extension of MAXTOLL considered in [10], where each commodity $k \in \mathcal{K}$ is associated with an origin-destination pair. Given a demand matrix, users solve shortest path problems parameterized by the toll vector T. If distinct tolls T^k could be assigned to distinct commodities, the multicommodity extension would reduce to a $|\mathcal{K}|$-fold version of the basic problem. Otherwise, the interaction between commodity flows on the arcs of a common transportation network complicates the problem, both from a theoretical and algorithmical point of view. Of course, we can obtain an $O(|\mathcal{K}| \log m_T)$ guarantee by applying MAXTOLL to each commodity separately and then selecting, among the $|\mathcal{K}|$ commodity toll vectors, the one that generates the highest revenue. However bad this bound is, we conjecture that it is tight with respect to the relaxation, which is the sum of the single-commodity bounds, weighted by their respective demands.

Other generalizations of MAXTOLL involve capacity constraints and lower bounds on tolls. In the latter case, the relaxation gap becomes infinite for any value of m_T, and our approach fails, as procedure EXPLOREDESCENDANTS becomes irrelevant. A completely different line of attack is then required.

Finally, we raise the following important issue: Can our $\frac{1}{2}\log_2 m_T + 1$ guarantee be improved? Such results would obviously require a tighter upper bound than the one used in this article.

Acknowledgments

This research was partially supported by NSERC, MITACS and NATEQ.

REFERENCES

[1] Audet, C., Hansen, P., Jaumard, B., and Savard, G., Links between linear bilevel and mixed 0-1 programming problems, J Optimizat Theory Appl 93 (1997), 273–300.

[2] Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A., and Protasi, M., Complexity and approximation, Springer, Berlin, 1999.

[3] Cocchi, R., Shenker, S., Estrin, D., and Zhang, L., Pricing in computer networks: Motivation, formulation and example, IEEE/ACM Trans Networking 1 (1993), 614–627.

[4] Garey, M.R. and Johnson, D.S., Computers and intractability: A guide to the theory of NP-completeness, W.H. Freeman, New York, 1979.

[5] Hearn, D.W. and Ramana, M.V., "Solving congestion toll pricing models," Equilibrium and advanced transportation modelling, Marcotte and Nguyen (Editors), Kluwer, New York, 1998, pp. 109–124.

[6] Hochbaum, D.S. (Editor), Approximation algorithms for NP-hard problems, PWS Publishing Company, Boston, 1997.

[7] Jeroslow, R.G., The polynomial hierarchy and a simple model for competitive analysis, Math Program 32 (1985), 146–164.

[8] Korilis, Y.A., Lazar, A.A., and Orda, A., Achieving network optima using Stackelberg routing strategies, IEEE/ACM Trans Networking 5 (1997), 161–173.

[9] Larsson, T. and Patriksson, M., "Side constrained traffic equilibrium models—Traffic management through link tolls," Equilibrium and advanced transportation modelling, Marcotte, P. and Nguyen, S. (Editors), Kluwer, New York, 1998, pp. 125–151.

[10] Labbé, M., Marcotte, P., and Savard, G., A bilevel model of taxation and its application to optimal highway pricing, Manage Sci 44 (1998), 1608–1622.

[11] Labbé, M., Marcotte, P., and Savard, G., "On a class of bilevel programs," Nonlinear optimization and related topics, Di Pillo, G. and Giannessi, F. (Editors), Kluwer Academic Publishers, New York, 1999, pp. 183–206.

[12] Luo, Z.-Q., Pang, J.-S., and Ralph, D., Mathematical programs with equilibrium constraints, Cambridge University Press, UK, 1996.

[13] Marcotte, P., Network design problem with congestion effects: A case of bilevel programming, Mathe Program 34 (1986), 142–162.

[14] Marcotte, P., Savard, G., and Semet, F., A bilevel programming approach to the travelling salesman problem, Oper Res Lett 32 (2004), 240–248.

[15] Pigou, A.C., The economics of welfare, Macmillan and Co., London, 1920.

[16] Roch, S., Marcotte, P., and Savard, G., An approximation Algorithm for Stackelberg network pricing, e-print, available at `http://arxiv.org/abs/cs.GT/0409054`.

[17] Roughgarden, T., Stackelberg scheduling strategies, Proceedings of the 33rd ACM Symposium on Theory of Computing, Crete, 2001, pp. 104–113.

[18] Roughgarden, T., Designing networks for selfish users is hard, Proceedings of the the 42nd Annual Symposium on Foundations of Computer Science, Las Vegas, 2001, pp. 472–481.

[19] Vazirani, V.V., Approximation algorithms, Springer, Berlin, 2001.

[20] Verhoef, E.T., P. Nijkamp, and P. Rietveld, Second-best congestion pricing: The case of an untolled alternative, J Urban Econ 40 (1996), 279–302.

[21] Vicente, L., Savard, G., and Júdice, J., Descent approaches for quadratic bilevel programming, J Optimizat Theory Appl 81 (1994), 379–399.