Design and analysis of an approximation algorithm for Stackelberg network pricing

Sébastien Roch and 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

17th March 2003

Abstract

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 $\frac{1}{2} \log m_T + 1$, 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.

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

1

1 Introduction

This paper focuses on a class of bilevel problems that arise naturally when tariffs, tolls, or devious taxes are to be determined over a network. This class of problems encompasses several important optimization problems encountered in the transportation, telecommunication, and airline industries. Our aim is twofold: first, we show that the problem is NP-hard; then we present a polynomial time algorithm with a tight worst-case guarantee of performance.

Bilevel programming is a modelling framework for 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 Luo, Pang and Ralph [12]), in which the lower level solution characterizes the equilibrium state of a physical or social system. Bilevel programs allow the modelling of a variety of situations that occur in operations research, economics, finance, etc. For instance, one may consider the maximization of social welfare, taking into account the selfish behavior of consumers. It is well-known that the taxation of resources and services at marginal cost (Pigovian taxes [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 Verhoef, Nijkamp and Rietveld [19], Hearn and Ramana [5], and Larsson and Patriksson [9] 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.

2

Bilevel programs are generically nonconvex and nondifferentiable, i.e., 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, Savard and Júdice [20], who proved that obtaining a mere certificate of local optimality is strongly NP-hard. Actually, Audet et al. [1] unveiled the 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 [16, 17], to network routing by Korilis, Lazar and Orda [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 non-conflicting, and where heuristics are expected to perform well in practice, although their worst-case behavior may turn out to be bad.

In this paper, we analyze an approximation algorithm for the toll optimization problem (MAXTOLL in the sequel) formulated and analyzed by Labbé, Marcotte and Savard [10]. In this game, which is almost zero-sum, 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[1]. In this paper, 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 $\frac{1}{2}\log m_T + 1$, where m_T denotes the number of toll arcs in the network. We then use this result as well as 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 paper is organized as follows. In Section 2, we state the problem and prove that it is NP-hard. In Section 3 we introduce 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

$$\min_{x,y} \quad (c_1 + T)x_1 + c_2 x_2$$

$$\text{s.t.} \quad A_1 x_1 + A_2 x_2 = b$$

$$x, y \geq 0.$$

---

[1] It was also shown recently by Marcotte et al.[14] that the TSP is a special case of MaxToll.

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 mathematical program (see also Figure 1):

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

This is a single-commodity instance of the toll setting problem analyzed in [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. First, 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 chooses among the toll-compatible shortest paths the one travelled by the follower. Note that a risk-averse leader could always force the use of the most profitable path by subtracting a small amount from every toll on that path, thus yielding a revenue as close as desired to the revenue generated by the "cooperative" solution. Thirdly, once a path P has been selected by the leader, toll arcs

```
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 $\sum_{e \in A_T \cap P} T(e)$
```
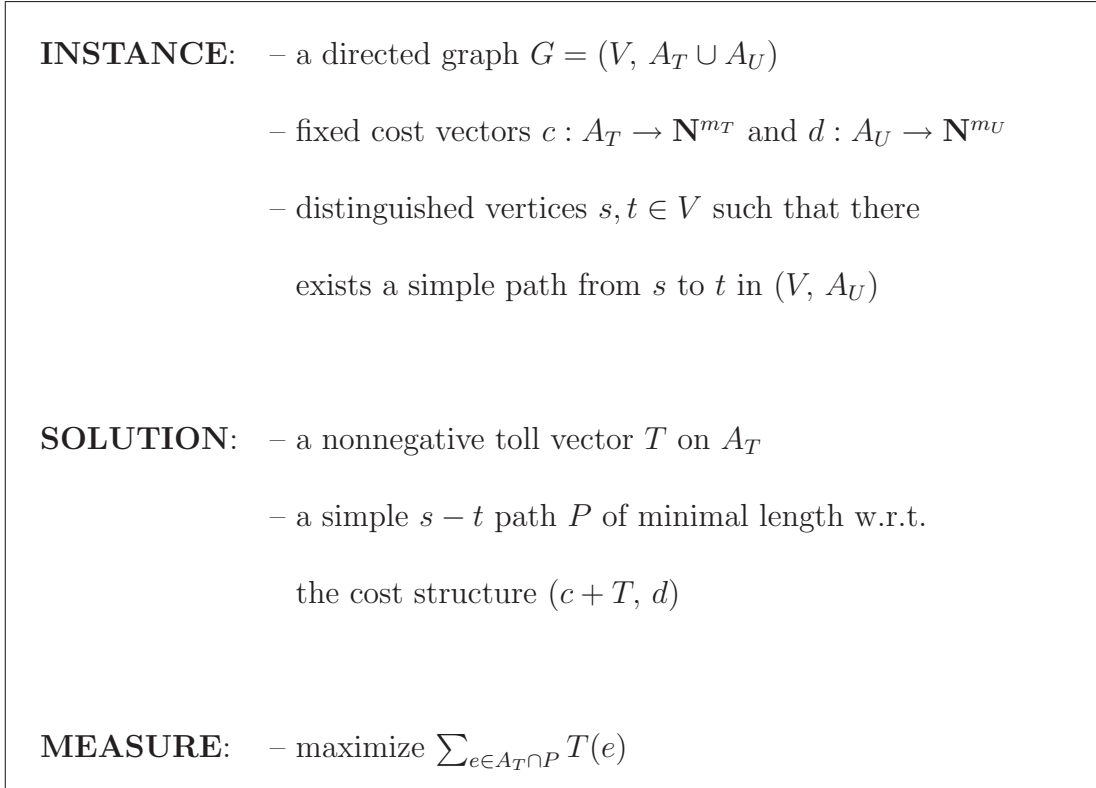
Figure 1: MaxToll

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 hold also in a version of MaxToll where T is unconstrained; in this case, negative tolls can be interpreted as subsidies. Actually, Labbé, Marcotte and Savard [11] have constructed instances where the optimal solution involves negative tolls. Nevertheless, throughout most of the paper we focus on nonnegative tolls because (i) this case is interesting in its own sake, (ii) intermediate results are easier to interpret when tolls are thought to be nonnegative.

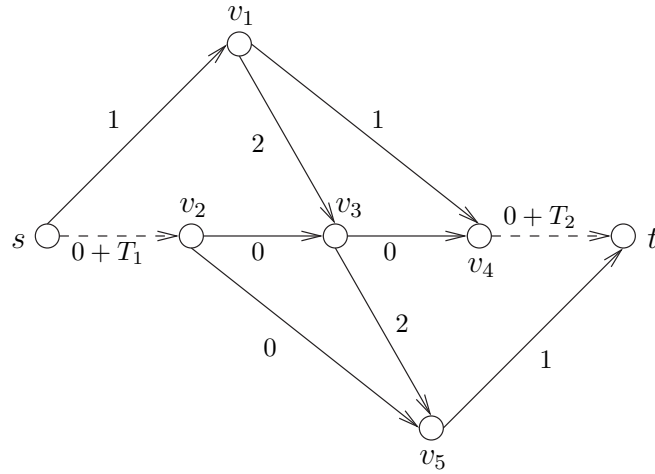A natural upper bound on the leader's revenue has been derived by Labbé et al. [10]

Figure 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_5, t) with a revenue of 5 when $T_1 = 5$ and $T_2 = 1000$.

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 and let $\mathcal{L}(P)$ be the length of a given path P with $T = 0$.

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

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

Since $\mathcal{L}_\infty$ 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

7

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 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 sub-network comprising one toll arc for each literal as shown in Figure 3.

The idea is the following: if the optimal path goes through toll arc T_{ij}, then the corresponding literal l_{ij} is TRUE (note: if $l_{ij} = \overline{x}_k$, then $x_k =$FALSE). The sub-networks 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 sub-network (i.e., one TRUE literal per clause) and simultaneously want to make sure
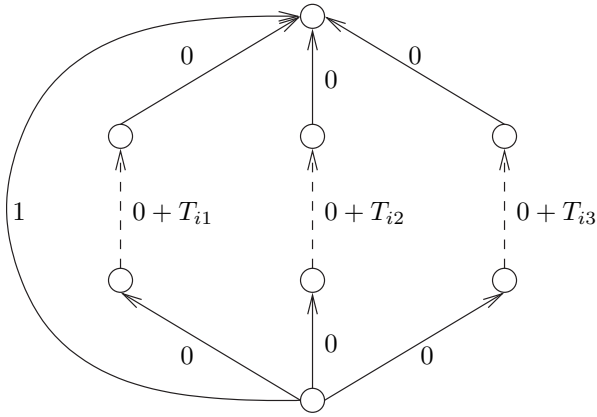
8

Figure 3: Sub-network for clause $(l_{i1} \lor l_{i2} \lor l_{i3})$.

that the corresponding assignment of variables is consistent; i.e., 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 inter-clause toll-free arc between the corresponding toll arcs (see Figure 4). As we will see, this implies that *inconsistent* paths, involving a variable and its negation, are suboptimal.

Since 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 achieve this, the optimal path has to go through one toll arc per sub-network (it cannot use inter-clause 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 inter-clause arc joining the corresponding toll arcs would impose a constraint on the tolls between its endpoints.
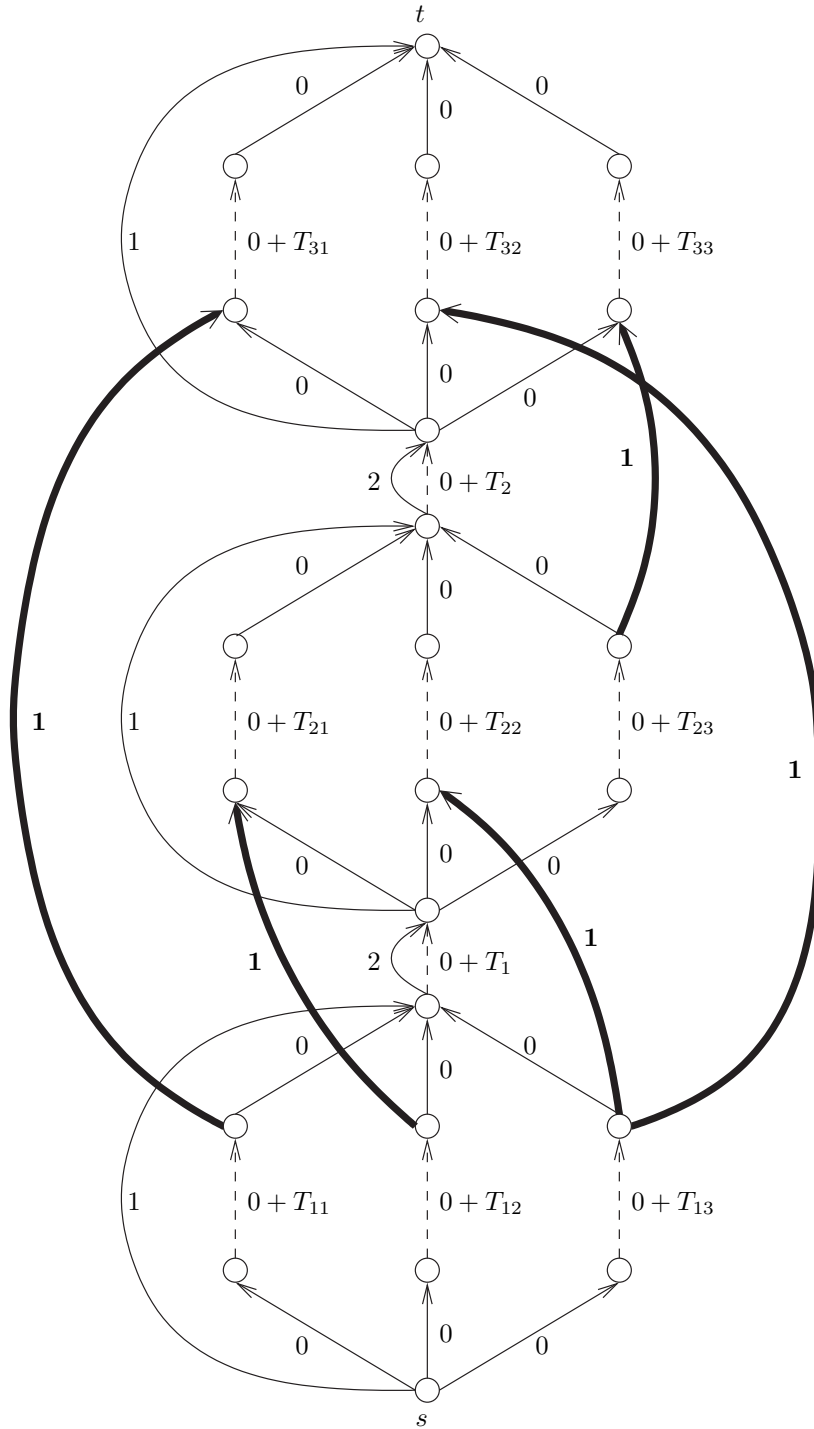
9

Figure 4: Network for the formula $(x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_2 \vee x_3 \vee \overline{x}_4) \wedge (\overline{x}_1 \vee x_3 \vee x_4)$.

Inter-clause arcs are bold. Path through T_{12}, T_{22}, T_{32} is optimal ($x_2 = x_3 =$TRUE).

10

In particular, the toll T_k immediately following the initial vertex of this inter-clause 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).

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. Since the assignment is consistent, the path does not simultaneously include a variable and its negation, and no inter-clause 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.

$\square$

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_k's to 2, so negative tolls are useless in this case. Indeed, inconsistent paths will see their revenue limited by inter-clause arcs without any possibility to make for the loss incurred from negative tolls.

$\square$

3   An approximation algorithm

In this section, we devise a polynomial time approximation algorithm for MAXTOLL. Such 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 [6] and [18].

3.1   Preliminaries: characterizing consistent tolls

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 non-valid paths cannot generate revenue. Any valid path P can be expressed as a sequence

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

where τ_i is the i-th toll arc of P (in the order of traversal) and $v_{i,i+1}$ is the toll-free subpath of P from the terminal vertex $\textsc{term}(\tau_i)$ of τ_i to the initial vertex $\textsc{init}(\tau_{i+1})$ of τ_{i+1}. According to this notation, $v_{0,1}$ starts in s and $v_{m_T^P, m_T^P + 1}$ ends in t. Since $P \in \mathcal{SP}[\mathcal{N}_0(P)]$, $v_{i,i+1}$ is a shortest toll-free path from $\textsc{term}(\tau_i)$ to $\textsc{init}(\tau_{i+1})$. We extend this notation to $v_{i,j}$, a shortest toll-free path from $\textsc{term}(\tau_i)$ to $\textsc{init}(\tau_j)$, with length $\mathcal{U}_{i,j}$. For $k < l$, let $\mathcal{L}_{k,l}$ be the length of P from $\textsc{term}(\tau_k)$ to $\textsc{init}(\tau_l)$ with tolls set to 0, and $\mathcal{T}_{k,l}$ be the sum of tolls between $\textsc{term}(\tau_k)$ and $\textsc{init}(\tau_l)$ on P,

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

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 the 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.

13

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

$$\mathcal{L}_{i,j} + \mathcal{T}_{i,j} \leq \mathcal{U}_{i,j} \qquad \forall\, 0 \leq i < j \leq m_T^P + 1. \tag{7}$$

Proof : $\Longrightarrow$) Obvious from the very definition of consistent tolls. $\Longleftarrow$) The converse looks equally obvious. However, one must be careful about paths that borrow the toll arcs of P in a *different* sequence. 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

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

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 $\widetilde{P}$ to obtain a new path that is shorter than $\widetilde{P}$. This improved path will happen to be P.

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 $\delta(m_T^{\widetilde{P}}+1) = j_{K+1} = m_T^{\widetilde{P}}+1$. By the definition of this increasing subsequence, we have $\delta(j_k-1) \leq \delta(j_{k-1})$ for all $k > 1$. Thus, if $j_k - 1 = j_{k-1}$,

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

14

(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

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

and 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.

$\square$

Remark. There always exists an optimal solution with tolls less than $C+1$. 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.

Figure 5 provides an equivalent representation of the example of Figure 2 according to Theorem 4.

3.2 The general idea underlying the algorithm

EXPLOREDESCENDANTS is an approximation algorithm motivated by the characterization of consistent tolls. Initially, the algorithm computes the optimal revenue associated with a shortest path in $\mathcal{SP}[\mathcal{N}_0]$, i.e, a path with the largest upper bound $B(P)^2$. If revenue is smaller than the upper bound LP, Theorem 4 implies that there exists a

---

[2]As shown in the next section, this can be achieved in polynomial time.
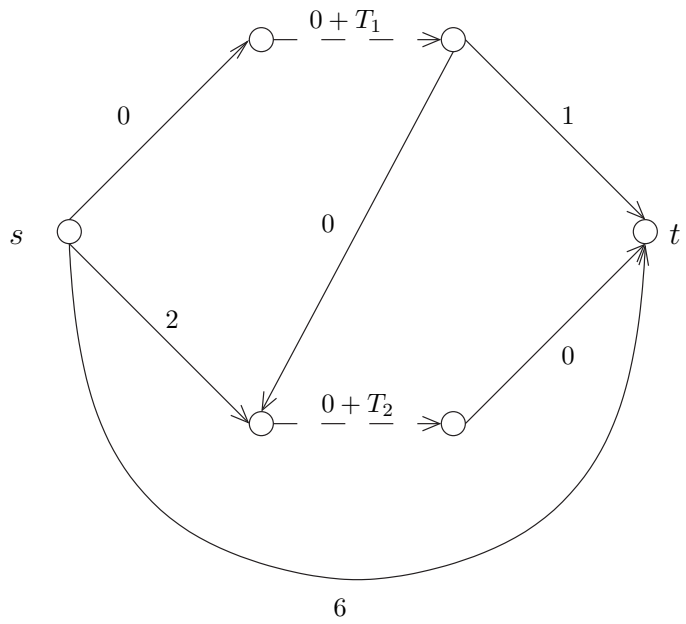
Figure 5: Equivalent representation of the example of Figure 2. All arcs have been replaced by shortest paths between toll arcs and origin/destination.

toll-free subpath $v_{k,l}$ $k < l$ 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 $v_{k,l}$. 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 uncover, in polynomial time, a path with good approximation properties.

Algorithm EXPLOREDESCENDANTS is streamlined in Figure 6. It comprises two subroutines, MAXREV and TOLLPARTITION. MAXREV computes the largest revenue compatible with the shortest path status of P. Starting from P, TOLLPARTITION generates two descendants of path P. The algorithm is initialized with P_0 in $\mathcal{N}_0$, a shortest path of length $\mathcal{L}_0$.

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: $(V_P, T_P) \leftarrow$ MAXREV(P)

- If $V_P < B(P)$ then

    - Derive new paths from P : $(P_1, P_2) \leftarrow$ TOLLPARTITION(P)

    - For $i = 1, 2$: $(\overline{V}_i, \overline{T}_i, \overline{P}_i) \leftarrow$ EXPLOREDESCENDANTS(P_i)

    - Return $\overline{V} \leftarrow \max\{V_P, \overline{V}_1, \overline{V}_2\}$ and corresponding toll vector $\overline{T}$ and path $\overline{P}$

- Else return $(\overline{V}, \overline{T}, \overline{P}) \leftarrow (V_P, T_P, P)$

Figure 6: Algorithm EXPLOREDESCENDANTS

3.3 Maximizing path-compatible revenue

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

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

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 are obtained from the recursion

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

where $T(\tau_1) = t_1, \ldots, T(\tau_{k-1}) = t_{k-1}$. The validity of the above formula rests on Theorem 5 where we construct a sequence of toll-free paths such that (i) every toll of P is bounded from above by at least one path in the sequence (condition (13) below), (ii) 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 $\upsilon_{i,j}$, $0 \leq i < j \leq m_T^P + 1$, be the toll-free subpaths defined in Section 3.1 and let t_k, $1 \leq k \leq m_T^P$ 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 Figure 7)

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

19

and for all k

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

with equality if $t_k \ne 0$. 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: In the recursive formula (10), let $\{(i'(k), j'(k))\}_{k=1}^{m_T^P}$ denote the indices for which minima are attained. These indices exist for all k since 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. Actually, we will introduce a slightly different construction for $\{(i'(k), j'(k))\}_{k=1}^{m_T^P}$. Note that for all k, there holds

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

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

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

where $(i'(k), j'(k))$ is chosen according to the rule stated above. This modification is necessary in order to ensure that $j(h) \le i(h + 2)$, as will become apparent shortly.
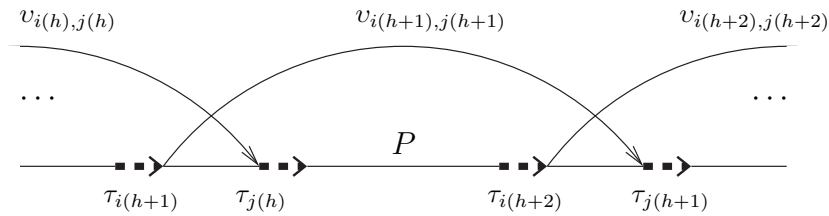

Figure 7: A section of a path P, showing toll arcs τ_k with $k = i(h + 1), j(h), i(h + 2), j(h + 1)$.

20

In view of Theorem 4, which implies that

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

we say that $v_{i(h),j(h)}$ *covers* the toll arcs τ_k for $i(h) + 1 \leq k \leq j(h) - 1$. 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 $(i(k), j(k)) = (i'(l_{q+1-k}), j'(l_{q+1-k}))$, $1 \leq k \leq q$, 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 $j(h) \leq i(h + 1)$. 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. $j(h) \leq i(h + 2)$: 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 $i'(l_{q+1-h-1}) = i(h + 1) = i(h)$.

21

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).

$\square$

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

Proof: For every toll assignment T' consistent with P

$$
\begin{aligned}
\sum_{k=1}^{m_T^P} T'(\tau_k) &\leq \sum_{h=1}^{q} \left[\mathcal{U}_{i(h),j(h)} - \mathcal{L}_{i(h),j(h)} \right], && \text{by (7) and (13)} \\
&= \sum_{k=1}^{m_T^P} t_k, && \text{by (14)}.
\end{aligned}
$$

$\square$

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 the toll arcs of P (actually only a subset of them) 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

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

22

of toll-free paths such that

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

and equality (14) holds. If the maximum revenue on P is $B(P)$ then descendants need not be considered, their upper bounds are smaller 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 $v_{i(h),j(h)}$ 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 $v_{i(1),j(1)}$, take path P between $\text{INIT}(\tau_{j(1)})$ and $\text{TERM}(\tau_{i(3)})$, bifurcate on $v_{i(3),j(3)}$, return to P, and so on. Such pattern, which is allowed by (20), is performed by procedure TOLLPARTITION. 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. 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, there would exist a subpath $v_{i(h),j(h')}$ ($h < h'$ 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). Since 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.

23

3.5 A detailed example

We apply the algorithm to the example of Figure 2. Here $LP = 6$. We start with the shortest path when tolls are set to 0

$$P_0 = (s, v_2, v_3, v_4, t).$$

Because of the toll-free subpaths (s, v_1, v_4) and (v_2, v_5, t), optimal tolls on P_0 are $T_1 = 2$ and $T_2 = 1$. Since $V_{P_0} = 3 < B(P_0) = 6$, the algorithm splits the tolls of P_0 into two subsets, forming the paths

$$P_1 = (s, v_2, v_5, t)$$

and

$$P_2 = (s, v_1, v_4, t).$$

The optimal revenue on P_1 is 5 and that on P_2 is 4. EXPLOREDESCENDANTS finally returns path P_1. Note that in this example, the algorithm returns the optimal path, this is not always the case.

4 Analysis of the approximation factor and running time

4.1 Worst-case guarantee

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

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

24

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

$$\alpha(k) \leq \frac{1}{2} \log k + 1$$

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

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

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

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

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 $i = 1,\, 2$,

$$P_i = \left(v_{0,1}^i,\, \tau_1^i,\, v_{1,2}^i,\, \ldots, \tau_{m_T^{P_i}}^i,\, v_{m_T^{P_i}, m_T^{P_i}+1}^i \right)$$

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

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

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

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

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

25

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 $2\mathcal{U}_{0,m_T^P+1} = 2\mathcal{U}_{0,m_T^{P_i}+1}$ 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:

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

By substituting in the preceding inequality, we obtain

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

which yields the desired result. Indeed, if we had $\frac{1}{\alpha(m_T^{P_i})}B(P_i) < \frac{1}{\alpha(m_T^P)}B(P)$ for $i = 1, 2$, this would imply the opposite inequality.

$\square$

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 $m_T^P = 1$ since 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 $(V_P, T_P) := \text{MAXREV}(P)$. If V_P is sufficient, (25) is satisfied. If V_P is not sufficient then, by Theorem 6, TOLLPARTITION returns a path P' with $\frac{1}{\alpha(m_T^{P'})} B(P') \geq \frac{1}{\alpha(m_T^P)} B(P)$. Since 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.

$\square$

Note that this result applies to the case of negative tolls as well since 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.

4.2 Tightness of the approximation

The approximation algorithm determines, in a constructive manner, an upper bound $\alpha(m_T)$ on the ratio OPT/LP. In this section, we show through a family of instances that this bound is tight.

Theorem 7 *Let $\mathcal{I}(m_T)$ denote the set of instances of MAXTOLL correponding 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*

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

Proof: Let us consider a two-node and two-arc network $Z(1)$, with origin s_1 and destination t_1. The first arc (a toll arc) has fixed cost 0 and the second arc (a toll-free arc) has fixed cost 2. We recursively construct a network $Z(k)$ made up of a copy of $Z(\lceil k/2 \rceil)$

27

and a copy of $Z(\lfloor k/2 \rfloor)$ (if k is even, there are two copies of $Z(k/2)$), and two distinguished nodes, the origin s_k and the destination t_k. These are linked by five toll-free arcs, as illustrated in Figure 8. Strictly speaking, the vertices $s_{\lceil k/2 \rceil}$, $t_{\lceil k/2 \rceil}$, $s_{\lfloor k/2 \rfloor}$, $t_{\lfloor k/2 \rfloor}$ should be re-labeled, otherwise many vertices will share the same label even though we consider them all distinct. The parameters a_k and b_k of $Z(k)$ are set to

$$a_k = 1 + \alpha\left(\left\lfloor \frac{k}{2} \right\rfloor\right) - \alpha\left(\left\lceil \frac{k}{2} \right\rceil\right) \quad b_k = 1 - \alpha\left(\left\lfloor \frac{k}{2} \right\rfloor\right) + \alpha\left(\left\lceil \frac{k}{2} \right\rceil\right), \tag{27}$$

with $a_1 = b_1 = 1$ and α is defined in (21). Note that $a_k, b_k \geq 0$ because $0 \leq \alpha(\lceil k/2 \rceil) - \alpha(\lfloor k/2 \rfloor) \leq 1$.

It is not difficult to show, by induction, that

$$\alpha(k) = \frac{1}{2}\left[1 + \alpha\left(\left\lceil \frac{k}{2} \right\rceil\right) + \alpha\left(\left\lfloor \frac{k}{2} \right\rfloor\right) \right]. \tag{28}$$

Let $LP(k)$ and $OPT(k)$ denote, respectively, the relaxed and optimal revenue values on $Z(k)$. Let $\mathcal{U}(k)$ be the length of the shortest toll-free path from s_k to t_k in $Z(k)$. We claim that $\mathcal{U}(k) = 2\alpha(k)$. Indeed, we have $\mathcal{U}(1) = 2\alpha(1) = 2$ and, assuming that $\mathcal{U}(k') = 2\alpha(k')$ for all $k' < k$, it follows from (28) that:

$$\mathcal{U}(k) = \min\left\{ \mathcal{U}\left(\left\lfloor \frac{k}{2} \right\rfloor\right) + b_k, \mathcal{U}\left(\left\lceil \frac{k}{2} \right\rceil\right) + a_k, \mathcal{U}\left(\left\lceil \frac{k}{2} \right\rceil\right) + \mathcal{U}\left(\left\lfloor \frac{k}{2} \right\rfloor\right) \right\} = 2\alpha(k). \tag{29}$$

Clearly, $OPT(1) = LP(1) = 2$ and $LP(1)/OPT(1) = \alpha(1) = 1$. For $k > 1$, the shortest path in $Z(k)$ with tolls set to 0 has length 0. This implies that $LP(k) = \mathcal{U}(k) = 2\alpha(k)$. To conclude, we need to show (by induction) that $OPT(k) = 2$. We consider two cases. If the optimal path on $Z(k)$ goes through the arc joining $t_{\lfloor k/2 \rfloor}$ and $s_{\lceil k/2 \rceil}$, then $OPT(k) \leq 2$ because $a_k + b_k = 2$. Otherwise, the optimal path belongs entirely to $Z(\lfloor k/2 \rfloor)$ or $Z(\lceil k/2 \rceil)$ in which case $OPT(k) = 2$ by the induction hypothesis. [3]

---

[3]It is straightforward to check that the same argument works in the negative tolls case.
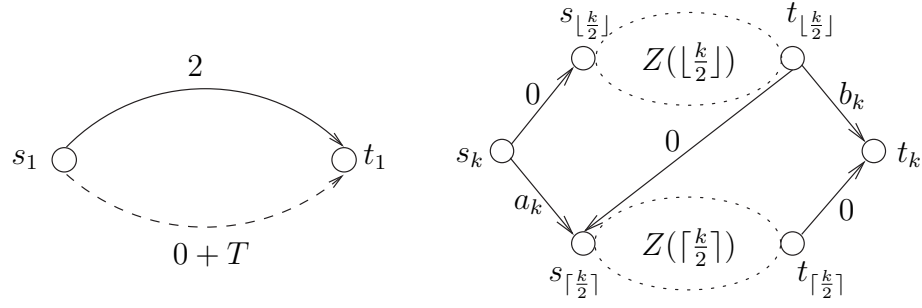
Figure 8: Networks $Z(1)$, left, and $Z(k)$, right. Toll arcs are dashed.

$\square$

We have just proved the optimality of our approximation factor with respect to the upper bound. We can prove more than that: the same family of instances, under a slight modification, can be used to show that our analysis of EXPLOREDESCENDANTS is tight. For this purpose, we require instances where APP is much smaller than OPT. This is not the case in the examples of Figure 8 where, indeed, $APP = OPT = 2$. However, our aim is attained if we add to $Z(k)$ a toll arc of fixed cost 1 from s^k to t^k. Then $OPT = LP - 1 = 2\alpha(k) - 1$ (the optimal path being the new toll arc) yet APP is still 2 since the algorithm starts with a path of length zero and misses the optimal path of length 1.

4.3 Complexity 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

29

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

$$\mathbb{T}(m_T^P) = \mathbb{T}(m_T^{P_1}) + \mathbb{T}(m_T^{P_2}) + O((m_T^P)^3) \tag{30}$$

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   Concluding remarks

Our algorithm can also be applied to the multi-commodity 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 multi-commodity 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

30

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. Indeed, we believe that the instances of Figure 8 can be generalized to the multi-commodity case.

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 $\frac{1}{2}\log m_T + 1$ guarantee be improved? Such result would obviously require a tighter upper bound than the one used in this paper.

Acknowledgements

References

[1] Audet, C., Hansen, P., Jaumard, B and Savard, G., Links between Linear Bilevel and Mixed 0-1 Programming Problems, Journal of Optimization Theory and Applications 93 (1997), 273–300.

[2] Ausiello, G. et al., 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 Transactions on 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, New York, 1979.

[5] Hearn, D. and Ramana, M.V., "Solving congestion toll pricing models", in: Equilibrium and Advanced Transportation Modelling, Marcotte and Nguyen (Editors), Kluwer, 1998,pp. 109–124.

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

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

[8] Korilis, Y.A., Lazar, A.A. and Orda, A., Achieving Network Optima Using Stackelberg Routing Strategies, IEEE/ACM Transactions on Networking 1 (1997), 161–173.

[9] Larsson, T. and Patriksson, M., "Side constrained traffic equilibrium models – Traffic management through link tolls", in: Equilibrium and Advanced Transportation Modelling, Marcotte and Nguyen (Editors), Kluwer, 1998, pp. 125–151.

[10] Labbé, M., Marcotte, P. and Savard, G., A bilevel model of taxation and its application to optimal highway pricing, Management Science 44 (1998), 1595–1607.

32

[11] Labbé, M. Marcotte, P. and Savard, G., "On a class of bilevel programs", in: Nonlinear Optimization and Related Topics. Di Pillo and Giannessi (Editors), Kluwer Academic Publishers, 1999, pp. 183–206.

[12] Luo, Z.-Q., Pang, J.-S. and Ralph, D., Mathematical Programming with Equilibrium Constraints, Cambridge University Press, UK, 1996.

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

[14] Marcotte,P., Savard, G. and Semet, F., A Bilevel ProgrammingApproach to the Travelling Salesman Problem. Report G-2002-73, GERAD (Montréal), 2002.

[15] Pigou, A.C., The Economics of Welfare, Macmillan and Co., London, 1920.

[16] Roughgarden, T., Stackelberg Scheduling Strategies, Proceedings of the 33rd ACM Symposium on Theory of Computing, Crete, 2001, pp. 104–113.

[17] 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.

[18] Vazirani, V.V., Approximation Algorithms, Springer, Berlin, 2001.

[19] Verhoef, E.T., P. Nijkamp and P. Rietveld, Second-best congestion pricing: the case of an untolled alternative, Journal of Urban Economics 40 (1996), 279–302.

[20] Vicente, L., Savard, G. and Júdice, J., Descent Approaches for Quadratic Bilevel Programming, Journal of Optimization Theory and Applications 81 (1994), 379–399.