Computational Optimization and Applications (2018) 71:641–671
https://doi.org/10.1007/s10589-018-0032-0

Pricing and revenue maximization over a multicommodity transportation network: the nonlinear demand case

Aimé Kamgaing Kuiteing[1] · **Patrice Marcotte**[2] · **Gilles Savard**[1]

Received: 15 August 2016 / Published online: 31 August 2018

Abstract
This paper is concerned with the design of efficient exact and heuristic algorithms for addressing a bilevel network pricing problem where demand is a nonlinear function of travel cost. The exact method is based on the piecewise linear approximation of the demand function, yielding mixed integer programming formulations, while heuristic procedures are developed within a bilevel trust region framework.

Keywords Pricing · Bilevel programming · Elastic demand · Networks · Non-convex programming · Mixed integer programming

1 Introduction

In its simplest form (see Labbé et al. [10]), the network pricing problem consists in setting revenue maximizing tolls on a congestion-free, multi-commodity transportation network, under the assumption that flows are assigned to cheapest paths with respect to a generalized cost. Kuiteing et al. [8] considered an extension where demand decreases linearly with travel cost, and the goal of the present work is to address the case of nonlinear price-demand relationships, which is much more challenging from the computational point of view.

The key elements of the paper are: a formulation of the bilevel model; the development of a quasi-exact method based on approximation, discretization and linearization techniques; the design of two heuristics based on the trust region paradigm (see Conn et al. [4]); numerical experiments and sensitivity analysis with respect to demand elasticity.

✉ Patrice Marcotte
marcotte@iro.umontreal.ca

[1] Department of Mathematics and Industrial Engineering, École Polytechnique de Montréal, Montreal, Canada

[2] Department of Computer Science and Operations Research, Université de Montréal, Montreal, Canada

2 Model formulation

Consider a network defined over the underlying graph $G = (\mathcal{N}, \mathcal{A})$, with node set $\mathcal{N}$ and arc set $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. Each arc of $\mathcal{A}$ is endowed with a unit weight c_a, while a toll t_a, to be determined by the leader, is associated with arcs in the subset $\mathcal{A}_1 \subseteq \mathcal{A}$. In this context, the generic term 'weight' may designate an out-of-pocket cost, a delay, or any attribute of disutility. We assume that weights and tolls are expressed in the same units, yielding the generalized cost $c_a + t_a$. Let $\mathcal{K}$ denote the set of origin-destination (OD) pairs or commodities. For each $k \in \mathcal{K}$ demand from the origin $o(k)$ to its destination $d(k)$ is provided by a function $n_k(u_k)$ of the minimal travel cost u_k from $o(k)$ to $d(k)$.

For each $k \in \mathcal{K}$, let h_p^k denote the proportion of demand assigned to path $p \in \mathcal{P}_k$, where $\mathcal{P}_k$ is the set of paths from $o(k)$ to $d(k)$. Since only cheapest paths with respect to t can carry positive flow (proportions), we have that, for all k in $\mathcal{K}$,

$$h^k \in \arg\min_{h'^k} \sum_{p\in\mathcal{P}_k} \Big(\sum_{a\in p} c_a + \sum_{a\in p\cap A_1} t_a\Big) h_p'^k$$
$$\text{s.t.} \sum_{p\in\mathcal{P}_k} h_p'^k = 1,$$

and the cost of a cheapest path is denoted u_k, where

$$u_k = \sum_{p\in\mathcal{P}_k} \Big(\sum_{a\in p} c_a + \sum_{a\in p\cap A_1} t_a\Big) h_p^k.$$

The validity of the latter expression rests on the fact that paths with positive flow proportions must bear the same cheapest cost, and the flow proportions must sum up to one. This yields the bilevel mathematical program

$$\text{P}: \quad \max_{t\geq 0, h, u} \sum_{k\in\mathcal{K}} \sum_{p\in\mathcal{P}_k} \sum_{a\in p\cap\mathcal{A}_1} t_a n_k(u_k) h_p^k$$
$$\text{s.t.} \quad \forall\, k \in \mathcal{K} \begin{cases} u_k = \displaystyle\sum_{p\in\mathcal{P}_k} \Big(\sum_{a\in p} c_a + \sum_{a\in p\bigcap\mathcal{A}_1} t_a\Big) h_p^k \\ h^k \in \arg\min_{h'^k} \displaystyle\sum_{p\in\mathcal{P}_k} \Big(\sum_{a\in p} c_a + \sum_{a\in p\cap\mathcal{A}_1} t_a\Big) h_p'^k \\ \quad \text{s.t.} \displaystyle\sum_{p\in\mathcal{P}_k} h_p'^k = 1. \end{cases}$$

Two remarks are in order:

- The equation that defines u_k is an upper level constraint that must be enforced by the leader. Its role is to ensure compatibility between the least costs and the corresponding cheapest path flows. This constraint does not bind the followers, i.e., the road users.
- In this 'optimistic' framework, the leader is allowed to select, among cheapest paths, the ones that yield maximum revenue. It follows that there exists an optimal

solution where, for each origin-destination pair k, only one variable h_k^p is positive and its value must equal 1. Without loss of generality, we henceforth assume that flow proportions are binary-valued.

A single-level nonlinear reformulation of the bilevel model is readily obtained by following the steps suggested in Didi-Biha et al. [6]: introduce variables T^k equal to the revenue per flow unit (equivalently the toll on cheapest path) and express the lower-level optimality conditions as linear constraints involving big-M constants (see [8] for their estimates). This yields the formulation

$$\overline{\mathrm{P}}: \quad \max_{t,h,T,u} \sum_{k\in\mathcal{K}} n_k(u_k)T^k$$

$$\text{s.t.} \sum_{a\in p} c_a + \sum_{a\in p\cap\mathcal{A}_1} t_a - M_p^k\left(1-h_p^k\right) \le u_k \quad \forall\, p\in\mathcal{P}_k, \forall\, k\in\mathcal{K} \tag{1}$$

$$u_k \le \sum_{a\in p} c_a + \sum_{a\in p\bigcap\mathcal{A}_1} t_a \qquad \forall\, p\in\mathcal{P}_k, \forall\, k\in\mathcal{K} \tag{2}$$

$$u_k = T^k + \sum_{p\in\mathcal{P}_k} h_p^k \sum_{a\in p} c_a \qquad \forall\, k\in\mathcal{K} \tag{3}$$

$$\sum_{p\in\mathcal{P}_k} h_p^k = 1. \qquad \forall\, k\in\mathcal{K} \tag{4}$$

$$h_p^k \in \{0,1\} \qquad \forall p\in\mathcal{P}_k, \forall\, k\in\mathcal{K} \tag{5}$$

where constraints (1) and (2) ensure that flow is assigned to cheapest paths, (3) establishes consistency between T^k and u_k, and constraints (4) and (5) ensure that commodity flows are not split between paths. Based on Eq. (3), one can replace T^k by $u_k - \sum_{p\in\mathcal{P}_k} h_p^k \sum_{a\in p} c_a$. This yields the nonlinear binary program

$$\mathrm{P1}: \quad \max_{t,h,u} \sum_{k\in\mathcal{K}} n_k(u_k)\Big(u_k - \sum_{p\in\mathcal{P}_k} h_p^k \sum_{a\in p} c_a\Big)$$

$$\text{s.t. (1)} - \text{(2) and (4)} - \text{(5)},$$

which will be used from now on.

In this paper, travel demand either has a constant elasticity or decays exponentially, i.e., $n_k(u_k) = \alpha_k u_k^{-\beta_k}$ or $n_k(u_k) = \alpha_k \exp(-\beta_k u_k)$, where $\alpha_k \ge 0$ and $\beta_k > 1$. Both are convex and widely used in economics. (See Moore [12].) In both cases, the resulting revenue function $n_k(u_k)u_k$ is either convex and decreasing, or pseudo-concave (increasing then decreasing) as depicted in Fig. 1.

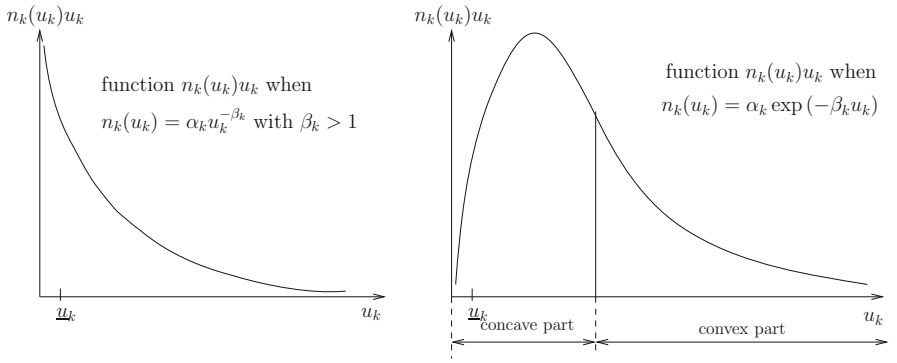

Fig. 1 Graph of function $n_k(u_k)u_k$; $\underline{u}_k$ is a lower bound on the cost of any path

3 An exact method

In this section, we describe an exact method inspired by the classical cutting plane principle introduced by Kelley [9], and applied to the hypograph (respectively epigraph) of the first term (respectively second term) in the objective. More precisely, similar to Audet et al. [2] in the context of non-convex quadratic programming, we replace these functions by piecewise linear over-estimations (respectively under-estimations), and derive an MIP that provides an upper bound on the objective of P1.

3.1 Approximation of P1

The toll revenue in the objective of P1 can be expressed as the difference between the total generalized cost and the total travel time, i.e.,

$$\sum_{k\in\mathcal{K}} n_k(u_k)u_k - \sum_{k\in\mathcal{K}} n_k(u_k)\Big(\sum_{p\in\mathcal{P}_k} h_p^k \sum_{a\in p} c_a\Big) \tag{6}$$

Let us first focus on the second term of (6). Since both forms of the demand function $n_k(u_k)$ are convex, the latter can be underestimated by its first-order Taylor development

$$n_k(u_k^l) + \frac{dn_k(u_k^l)}{du_k}(u_k - u_k^l).$$

Setting $b_k^l = \frac{dn_k(u_k^l)}{du_k}$ and $a_k^l = n_k(u_k^l) - u_k^l b_k^l$ and upon introduction of the variable y_k, we append to the model the constraints

$$y_k \geq a_k^l + b_k^l u_k. \tag{7}$$

At optimality, (u_k, y_k) must lie on a face of the polyhedron defined by constraints (7). The process is initiated with lower and upper bounds $\underline{u}_k$ and $\overline{u}_k$ determined a priori

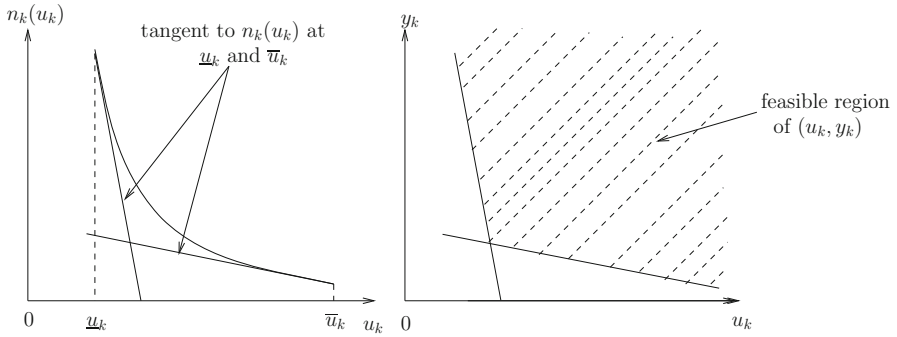

Fig. 2 Under-approximation of $n_k(u_k)$

(see Fig. 2). For instance, $\underline{u}_k$ and $\bar{u}_k$ can be set to the cost of a cheapest path for OD pair k when, respectively, $t_a = 0$ (no tolls) and $t_a = \infty$ (no toll arcs are allowed) for all $a \in \mathcal{A}_1$.

Upon substituting $n_k(u_k)$ by y_k, the expression $n_k(u_k) \sum_{p\in\mathcal{P}_k} h_p^k \sum_{a\in p} c_a$ in the objective function becomes $y_k \sum_{p\in\mathcal{P}_k} h_p^k \sum_{a\in p} c_a = \sum_{p\in\mathcal{P}_k} y_k h_p^k \sum_{a\in p} c_a$. The linearization of the bilinear term $y_k h_p^k$ is obtained by introducing a variable y_k^p set to $y_k h_p^k$ and a big-M constant M_k to force the equality $y_p^k = y_k h_p^k$:

$$0 \le y_k^p \le M_k h_p^k \qquad \forall p \in \mathcal{P}_k \tag{8}$$

$$y_k - \sum_{p\in\mathcal{P}_k} y_k^p = 0. \tag{9}$$

The parameter M_k can be set to $n_k(\underline{u}_k)$. In [8], it has been shown that this linearization outperforms the one where (9) is substituted to $-M_k(1-h_p^k) \le y_k - y_k^p \le M_k(1-h_p^k)$.

Let us now consider the piecewise linear approximation of the pseudo-concave term $n_k(u_k)u_k$, which is assumed concave over the interval $[\underline{u}_k, u_k^{I_k}]$, and convex over $[u_k^{I_k}, \overline{u}_k]$. Note that we can limit our attention to this case, since it subsumes the case when the function is convex.

Let w_k denote an over-estimate of the generalized cost. For its concave portion, the over-estimate is obtained using the tangential or first-order approximation (see Fig. 3):

$$w_k \le a_k^i + b_k^i u_k + M_k^i s_k \qquad \forall\, i = 1, \ldots, I_k \tag{10}$$

$$u_k \le u_k^{I_k}(1 - s_k) + \overline{u}_k s_k$$

$$s_k \in \{0, 1\} \tag{11}$$

where $M_k^i = \max_{\xi_k \le u_k \le \overline{u}_k} \{n_k(u_k)u_k\} + |\min\{0, a_k^i + b_k^i \overline{u}_k\}|$, a_k^i, b_k^i are the coefficients of the tangent to $n_k(u_k)u_k$, and s_k a binary variable equal to 0 if u_k lies in the concave part of $n_k(u_k)u_k$ and 1 otherwise.

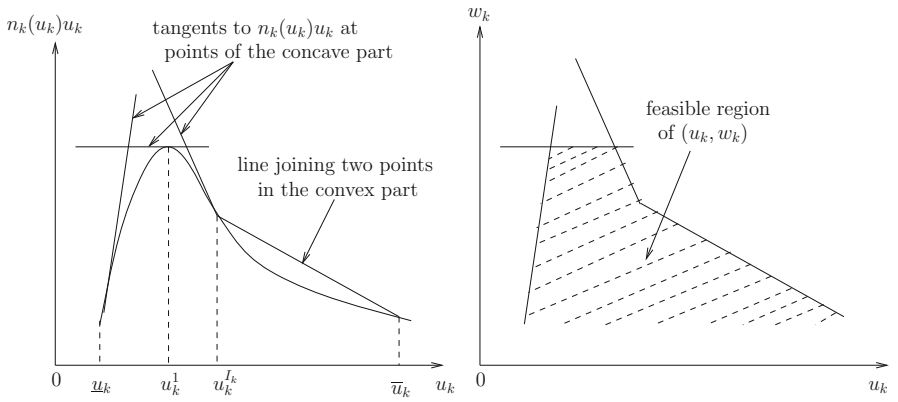

Fig. 3 Over-approximation of $n_k(u_k)u_k$

At optimality, the pair (u_k, w_k) lies on a face of the polyhedron determined by the tangents to $n_k(u_k)u_k$.

The over-approximation of the convex part of $n_k(u_k)u_k$ is obtained by appending to P1 a set of piecewise linear constraints. The interval $[u_k^{I_k}, \overline{u}_k]$ is split into J_k sub-intervals $[u_k^j, u_k^{j+1}]$ for $j = 1, \ldots, J_k - 1$. With each sub-interval $[u_k^j, u_k^{j+1}]$ is associated a binary variable z_k^j that indicates whether u_k belongs to it ($z_k^j = 1$) or not ($z_k^j = 0$). The linear constraint that defines the over-approximation of $n_k(u_k)u_k$ over the sub-interval $[u_k^j, u_k^{j+1}]$ corresponds to the segment joining the two points $\left(u_k^j, n_k(u_k^j)u_k^j\right)$ and $\left(u_k^{j+1}, n_k(u_k^{j+1})u_k^{j+1}\right)$. Mathematically, these constraints are expressed in the following way:

$$w_k \leq a_k^j + b_k^j u_k + M_k^j(1 - z_k^j) + N_k(1 - s_k) \qquad \forall\, j = 1, \ldots, J_k - 1 \qquad (12)$$

$$u_k \geq u_k^j z_k^j \qquad \forall\, j = 1, \ldots, J_k - 1 \qquad (13)$$

$$u_k \leq u_k^{j+1} z_k^j + \overline{u}_k(1 - z_k^j) \qquad \forall\, j = 1, \ldots, J_k - 1 \qquad (14)$$

$$z_k^j \in \{0, 1\} \qquad \forall\, j = 1, \ldots, J_k - 1 \qquad (15)$$

$$\sum_{j=1}^{J_k - 1} z_k^j = s_k \qquad (16)$$

where

$$M_k^j = \max_{\xi_k \leq u_k \leq \overline{u}_k} \{n_k(u_k)u_k\} + |\min\{0, a_k^i + b_k^i \overline{u}_k\}|$$

$$N_k = \max_{\underline{u}_k \leq u_k \leq \xi_k} \{n_k(u_k)u_k\} - \max_{\xi_k \leq u_k \leq \overline{u}_k} \{n_k(u_k)u_k\}$$

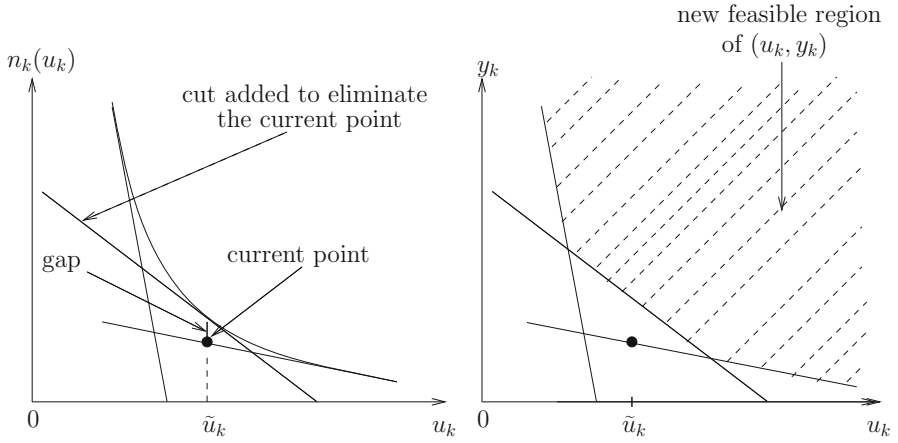

Fig. 4 Refining the under-approximation of $n_k(u_k)$

and a_k^j, b_k^j are parameters associated with the straight line going through $\left(u_k^j, n_k(u_k^j)u_k^j\right)$ and $\left(u_k^{j+1}, n_k(u_k^{j+1})u_k^{j+1}\right)$. At optimality, (u_k, w_k) lies on a face of the polyhedron bordered by piecewise linear functions defined over each sub-interval as depicted in Fig. 3.

The approximation of $n_k(u_k)$ and $n_k(u_k)u_k$ yields the mathematical program

$$\widetilde{\text{P1}}: \quad \max_{t,h,T,u,w,y,z,s} \sum_{k\in\mathcal{K}} \left(w_k - \sum_{p\in\mathcal{P}_k} y_k^p \sum_{a\in p} c_a \right)$$
$$\text{s.t. } (1)-(5) \text{ and } (7)-(16).$$

which provides an upper bound on the leader's revenue, as well as a lower bound obtained by solving the original lower level problem with respect to its toll solution.

3.2 Refining the functional approximations

When a refinement of the objective function is required, for instance if the gap between its actual value and the approximation is too wide at the current iterate $(\tilde{t}, \tilde{h}, \tilde{T}, \tilde{u}, \tilde{w}, \tilde{y}, \tilde{z}, \tilde{s})$, we proceed as follows.

For the under-estimate, if the difference between $\tilde{y}_k$ and $n_k(\tilde{u}_k)$ is large, we improve the approximation by appending the following tangential cut (see Fig. 4) to $\widetilde{\text{P1}}$:

$$y_k \geq a_k + b_k u_k \tag{17}$$

Next, consider the over-approximation of $n_k(u_k)u_k$. If $\widetilde{u}_k$ lies in the concave part of $n_k(u_k)u_k$, the refinement is obtained by appending to $\widetilde{\text{P1}}$ the tangential cut (see Fig. 5):

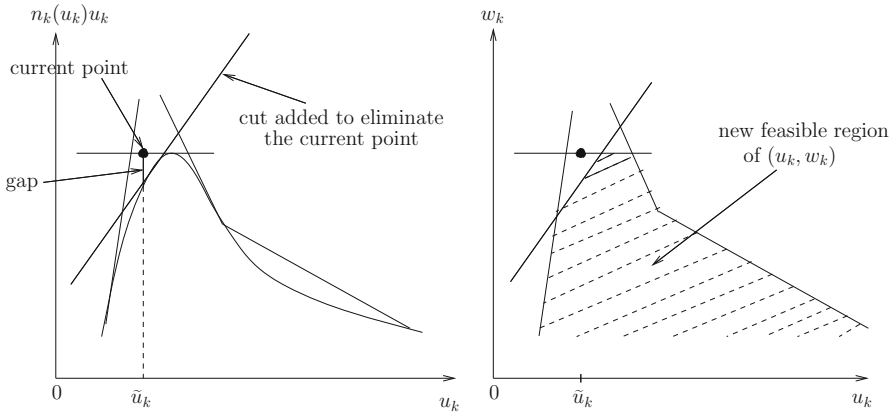

Fig. 5 Refining the over-approximation of $n_k(u_k)u_k$ over the concave part

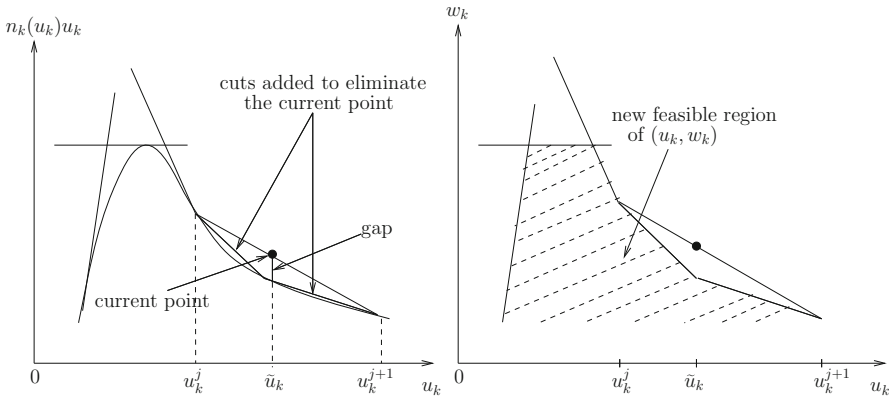

Fig. 6 Refining the over-approximation of $n_k(u_k)u_k$ over the convex part

$$w_k \leq a_k + b_k u_k + M_k s_k \tag{18}$$

where s_k is the binary variable that takes value 1 whenever $\tilde{u}_k$ lies in the concave part, and 0 otherwise.

If $\tilde{u}_k$ belongs to the convex part of $n_k(u_k)u_k$, with $\widetilde{u}_k \in [u_k^j, u_k^{j+1}]$, we divide the interval $[u_k^j, u_k^{j+1}]$ into two sub-intervals $[u_k^j, \widetilde{u}_k]$ and $[\widetilde{u}_k, u_k^{j+1}]$ and replace the cut associated with $[u_k^j, u_k^{j+1}]$ by the piecewise linear cuts illustrated in Fig. 6.

For the sake of consistency, the following constraints are also introduced in $\widetilde{\text{P1}}$:

$$w_k \leq a_k^1 + b_k^1 u_k + M_k^1(1 - z_k^1) + N_k(1 - s_k) \tag{19}$$

$$u_k \geq u_k^j z_k^1 \tag{20}$$

$$u_k \leq \tilde{u}_k z_k^1 + M_k^1(1 - z_k^1) \tag{21}$$

$$w_k \leq a_k^2 + b_k^2 u_k + M_k^2(1 - z_k^2) + N_k(1 - s_k) \tag{22}$$

$$u_k \geq \tilde{u}_k z_k^2 \tag{23}$$

$$u_k \leq u_k^{j+1} z_k^2 + M_k^2(1 - z_k^2) \tag{24}$$

$$z_k^1, z_k^2 \in \{0, 1\}. \tag{25}$$

3.3 Lower bound and inverse optimization

Let $(\tilde{t}, \tilde{h}, \tilde{T}, \tilde{u}, \tilde{w}, \tilde{y}, \tilde{z}, \tilde{s})$ be a current solution to $\widetilde{\text{P1}}$. Based on $\widetilde{t}$, we can solve the lower level problem and derive a lower bound on the leader's revenue. However, $\widetilde{t}$ does not necessarily yield the maximum revenue with respect to the flow vector $\widetilde{h}$. Assuming that $\widetilde{h}$ is known a priori, one can determine the compatible tolls that should be imposed in order to maximize the leader's revenue. This corresponds to solving an inverse optimization problem.

More precisely, let $\tilde{p}_k$ be the path used by commodity k, i.e., $\widetilde{h}^k_{\tilde{p}_k} = 1$ and $\widetilde{h}^k_p = 0$ for all $p \in \mathcal{P}_k \backslash \{\tilde{p}_k\}$. We write

$$u_k = T^k + \sum_{a \in \tilde{p}_k} c_a$$

$$n_k(u_k) = n_k \left(T^k + \sum_{a \in \tilde{p}_k} c_a \right) T^k$$

and consider the problem, for fixed vector flow $\widetilde{h}$, defined as

$$\text{IOP}(\widetilde{h}): \quad \max_{t,T,u} \sum_{k \in \mathcal{K}} n_k \left(\sum_{a \in \tilde{p}_k} c_a + T^k \right) T^k$$

$$\text{s.t.} \quad T^k = \sum_{a \in \widetilde{p}_k \cap \mathcal{A}_1} t_a \qquad \forall k \in \mathcal{K} \tag{26}$$

$$u_k \leq \sum_{a \in q} c_a + \sum_{a \in q \cap \mathcal{A}_1} t_a \qquad \forall q \in \mathcal{P}_k : \tilde{h}^k_q = 0 \quad \forall k \in \mathcal{K} \tag{27}$$

$$u_k = T^k + \sum_{a \in \widetilde{p}_k} c_a \qquad \forall k \in \mathcal{K}. \tag{28}$$

In the above objective, each term

$$n_k \left(\sum_{a \in \tilde{p}_k} c_a + T^k \right) T^k$$

inside the summation can be expressed as

$$n_k \left(\sum_{a \in \tilde{p}_k} c_a + T^k \right) \left(\sum_{a \in \tilde{p}_k} c_a + T^k \right) - n_k \left(\sum_{a \in \tilde{p}_k} c_a + T^k \right) \sum_{a \in \tilde{p}_k} c_a,$$

whose first term is pseudo-concave and the second is convex. By using the technique described in Sect. 3.1, an over-approximation of IOP($\widetilde{h}$) can be formulated as an MIP.

Once IOP($\widetilde{h}$) is solved, assume that the optimal value of IOP($\widetilde{h}$) is less than the incumbent lower bound, i.e., the optimal solution of the original problem does not match the solution associated with the vector flow $\widetilde{h}$. In order to forbid $\widetilde{h}$, the following 'flow cut' is appended:

$$\sum_{k \in \mathcal{K}} h^k_{\tilde{p}_k} \leq |\mathcal{K}| - 1. \tag{29}$$

3.4 Upper bound

Let p^k_0 (respectively p^k_∞) denote the cheapest path from $o(k)$ to $d(k)$ obtained by setting tolls to 0 (respectively $+\infty$) and let γ^k_0 (respectively γ^k_∞) denote the corresponding travel cost, i.e.,

$$\gamma^k_0 = \sum_{a \in p^k_0} c_a$$
$$\gamma^k_\infty = \sum_{a \in p^k_\infty} c_a.$$

If there is a path from $o(k)$ to $d(k)$ in $\mathcal{G}(\mathcal{N}, \mathcal{A}_2)$, γ^k_∞ equals the cost of the cheapest path on the subnetwork. Otherwise, all paths include at least one arc with infinite toll, and must therefore bear an infinite cost.

Proposition 1 *Let*

$$\overline{T}^k = \begin{cases} \min \left\{ \gamma^k_\infty - \gamma^k_0, \ \dfrac{1}{\beta_k - 1} \displaystyle\sum_{a \in p^k_0} c_a \right\} & \text{if } n_k(u_k) = \alpha_k u_k^{-\beta_k} \\ \min \left\{ \gamma^k_\infty - \gamma^k_0, \ \frac{1}{\beta_k} \right\} & \text{if } n_k(u_k) = \alpha_k \exp(-\beta_k u_k). \end{cases}$$

An upper bound of the leader's revenue is given by:

$$UB = \sum_{k \in \mathcal{K}} n_k \left(\sum_{a \in p^k_0} c_a + \overline{T}^k \right) \overline{T}^k$$

Proof The leader's objective function for commodity k is $n_k(u_k)T^k$. Since path p_0^k has the smallest travel fixed cost for commodity k and n_k is strictly decreasing, then

$$n_k(u_k)T^k \;\leq\; n_k\left(\sum_{a\in p_0^k} c_a + T^k\right)T^k.$$

The function $n_k(\sum_{a\in p_0^k} c_a + T^k)T^k$ is pseudo-concave (increasing then decreasing) and reaches its optimal value at π^k, the zero of its derivative, which is equal to $(1/(\beta_k - 1))\sum_{a\in p_0^k} c_a$ (respectively $1/\beta_k$) when $n_k(u_k) = \alpha_k(u_k)^{-\beta_k}$ (respectively $n_k(u_k) = \alpha_k \exp(-\beta_k u_k)$). Since $\gamma_\infty^k - \gamma_0^k$ is an upper bound on one unit of the leader's revenue T^k raised from commodity k, it follows that the maximum value of T^k is $\min\{\gamma_\infty^k - \gamma_0^k, \pi^k\}$. □

As a corollary to the previous theorem, it follows that a valid upper bound on the leader's objective is provided by the smaller value between UB and the objective value of $\widetilde{\text{P1}}$ constitutes a valid upper bound.

3.5 Bounds on u_k

As mentioned by Al-Khayyal [1], the difficulty of obtaining tight bounds on u_k, with the aim of obtaining good approximations of the functions $n_k(u_k)$ and $n_k(u_k)u_k$, is related to the number of commodities.

Let $[\underline{u}_k, \overline{u}_k]$ be the range of admissible generalized cost values u_k, and let N_{ij}^k be an upper bound on the toll on arc (i, j) with respect to commodity k. Let c_i (respectively c_j) denote the cheapest path from $o(k)$ to i (respectively from j to $d(k)$). In the fixed demand case, the upper bound

$$\overline{N}_{ij}^k = \gamma_\infty^k - (c_i^- + c_{ij} + c_j^+)$$

holds under the assumption that there exists a toll-free path from $o(k)$ to $d(k)$. If no such path exists, $\overline{N}_{ij}^k = \infty$ (see Dewez et al. [5]). However, when demand is elastic, an alternative bound can be derived, which holds even in the absence of toll-free paths.

Lemma 1 *For $t = 0$, let c_{ki}^- and c_{jk}^+ be the cost of the cheapest path, respectively, from $o(k)$ to i and from j to $d(k)$, and define*

$$\Omega_{ij}^k = \begin{cases} \dfrac{c_{ki}^- + c_a + c_{jk}^+}{\beta_k - 1} & \text{if } n_k(u_k) = \alpha_k u_k^{-\beta_k} \\ \dfrac{1}{\beta_k} & \text{if } n_k(u_k) = \alpha_k \exp(-\beta_k u_k). \end{cases}$$

If toll arc $a = (i, j)$ belongs to the path used by commodity k, then an upper bound on the corresponding toll is given by:

$$N_{ij}^k = \min\{\overline{N}_{ij}^k, \Omega_{ij}^k\}$$

Proof If the path used by commodity k goes through arc $a = (i, j)$, then $u_k = c_{ki}^- + c_a + c_{jk}^+ + T^k$. This implies that the leader's revenue raised by commodity k satisfies $n_k(u_k)T^k = n_k\left(c_{ki}^- + c_a + c_{jk}^+ + T^k\right)T^k$. Since the function on the right-hand side is pseudo-concave and reaches its maximum at $T^k = \Omega_{ij}^k$, the result follows. □

Proposition 2 *Let $\mathcal{A}_1^k$ be the set of toll arcs that can be used by commodity k, and $\mathcal{K}^a$ the set of commodities that can use toll arc $a = (i, j)$. The following lower and upper bounds on the total cost of any path used by commodity k hold:*

$$\underline{u}_k = \sum_{a \in p_0^k} c_a$$

$$\overline{u}_k = \min\left\{\gamma_\infty^k - \gamma_0^k + \sum_{a \in p_0^k} c_a, \max_{a \in \mathcal{A}_1^k}\left\{c_{ki}^- + c_a + c_{jk}^+ + \max_{k' \in \mathcal{K}^a} N_{ij}^{k'}\right\}\right\}.$$

Proof Since path p_0^k has the smallest fixed travel cost among the paths of commodity k, we have that $u_k \geq \sum_{a \in p_0^k} c_a$. Now, it follows from Lemma 1 that a toll arc $a = (i, j)$ will not carry any flow whenever its toll exceeds $\max_{k' \in \mathcal{K}^a} N_{ij}^{k'}$, which implies that $u_k \leq \max_{a \in \mathcal{A}_1^k}\{c_{ki}^- + c_a + c_{jk}^+ + \max_{k' \in \mathcal{K}^a} N_{ij}^{k'}\}$. Since $\gamma_\infty^k - \gamma_0^k$ is an upper bound on one unit of the leader's revenue, T^k, raised from commodity k, we have that $u_k \leq \gamma_\infty^k - \gamma_0^k + \sum_{a \in p_0^k} c_a$. Combining both upper bounds yields the desired result. □

3.6 The exact method and a variant thereof

The exact algorithm can be divided into two parts. The first consists in over-approximating the objective function after having computed bounds on variables u_k and T^k. The second step iteratively solves the approximated problem and refines the over-approximation.

Algorithm 1 Exact method

Require: n-point approximation of nonlinear functions; tolerance factor ε.

Compute bounds on variables u_k and T^k.
$k \leftarrow 0$, LB $\leftarrow -\infty$ and compute UB such as described in Sect. 3.4.
Let (t^*, h^*) be the best solution found.
StopCriterion $\leftarrow$ **false**
repeat
  $k \leftarrow k + 1$
  $(\widetilde{t}, \widetilde{h}, \widetilde{T}, \tilde{u}, \widetilde{w}, \tilde{y}, \tilde{z}, \tilde{s}) \leftarrow$ solution of $\widetilde{\text{P1}}$.
  $\widetilde{Z} \leftarrow$ objective value of $\widetilde{\text{P1}}$.
  UB $\leftarrow \max\{\text{UB}, \widetilde{Z}\}$
  $\text{Z}_{IOP} \leftarrow$ optimal value of IOP($\widetilde{h}$).
  $\bar{t} \leftarrow$ optimal solution of IOP($\widetilde{h}$).
  if LB $<$ Z_{IOP} **then**
    LB $\leftarrow \text{Z}_{\text{IOP}}$
    $(t^*, h^*) \leftarrow (\bar{t}, \widetilde{h})$
  else if LB $>$ Z_{IOP} **then**
    Append the 'flow' cut (29)
  end if
  StopCriterion $\leftarrow$ **true**
  for each commodity $k \in \mathcal{K}$ **do**
    if $\widetilde{y}_k < n_k(\widetilde{u}_k) - \varepsilon$ **then**
      StopCriterion $\leftarrow$ **false**
      Append the linear constraint (17) to $\widetilde{\text{P1}}$.
    end if
    if $\widetilde{w}_k - \varepsilon > n_k(\widetilde{u}_k)\widetilde{u}_k$ **then**
      StopCriterion $\leftarrow$ **false**
      if $\widetilde{u}_k$ lies in the concave part of $n_k(u_k)u_k$ **then**
        Append the linear constraint (18) to $\widetilde{\text{P1}}$.
      else
        Append the piecewise linear constraints (19)–(25) to $\widetilde{\text{P1}}$.
      end if
    end if
  end for
until StopCriterion **or** LB + ε $<$ UB

We now consider a variant of the exact method involving a more tractable objective. To this aim, we introduce the slack variable τ^k:

$$\tau^k = u_k - \sum_{a \in p_0^k} c_a. \tag{30}$$

By substituting u_k for (30) in the objective function, we obtain

$$\begin{aligned}
& n_k(u_k)\, u_k - n_k(u_k) \sum_{p \in \mathcal{P}_k} \sum_{a \in p} c_a h_p^k \\
&= n_k(u_k) \left(\tau^k + \sum_{a \in p_0^k} c_a \right) - n_k(u_k) \sum_{p \in \mathcal{P}_k} h_p^k \sum_{a \in p} c_a \\
&= n_k(u_k)\, \tau^k - n_k(u_k) \left(\sum_{p \in P_k} h_p^k \sum_{a \in p} c_a - \sum_{a \in p_0^k} c_a \right) \\
&= n_k \left(\tau^k + \sum_{a \in p_0^k} c_a \right) \tau^k + n_k(u_k) \left(\sum_{p \in \mathcal{P}_k} h_p^k \sum_{a \in p} c_a - \sum_{p \in \mathcal{P}_k} h_p^k \sum_{a \in p_0^k} c_a \right) \\
&= n_k \left(\tau^k + \sum_{a \in p_0^k} c_a \right) \tau^k - n_k(u_k) \sum_{p \in \mathcal{P}_k} h_p^k \left(\sum_{a \in p} c_a - \sum_{a \in p_0^k} c_a \right).
\end{aligned} \tag{31}$$

This yields the mathematical program obtained by replacing the objective of $\overline{\mathrm{P}}$ by (31) and appending constraints (30).

$$\begin{aligned}
\text{P2:} \quad & \max_{t,h,T,u} \sum_{k \in \mathcal{K}} n_k \left(\tau^k + \sum_{a \in p_0^k} c_a \right) \tau^k - n_k(u_k) \sum_{p \in \mathcal{P}_k} h_p^k \left(\sum_{a \in p} c_a - \sum_{a \in p_0^k} c_a \right) \\
& \text{s.t.} \qquad u_k = \tau^k + \sum_{a \in p_0^k} c_a \qquad \forall k \in \mathcal{K} \\
& \qquad (1) - (2) \text{ and } (4) - (5).
\end{aligned}$$

To solve P2, we adopt the strategy used for solving P1, and over-approximate $n_k(\tau^k + \sum_{a \in p_0^k} c_a)\tau^k$ in place of $n_k(u_k)u_k$. Note that the bounds on variables T^k are valid for the variable τ^k.

We observe that P2 is likely to outperform P1 in terms of speed. Indeed, the convex part of the function $n_k(u_k)u_k$ is always larger than the one of $n_k(\tau^k + \sum_{a \in p_0^k} c_a)\tau^k$, thus the over-approximation of $n_k(u_k)u_k$ involves a larger number of binary variables than the over-approximation of $n_k(\tau^k + \sum_{a \in p_0^k} c_a)\tau^k$. This leads to a lesser quality of the upper bound of P1 compared to P2.

4 Two heuristic algorithms

In this section, we present two heuristics based on the trust region paradigm, following the framework proposed by Colson et al. [3] to solve general non-linear bilevel programs. At each iteration, the trust region 'model' corresponds to a bilevel program where all functions are linearized, with the exception of the lower-level objective, which is replaced by a second-order approximation. Upon replacing the quadratic lower-level program by its optimality conditions, the resulting quadratic MIP can be solved using a commercial software.

Two strategies have been considered. In the first, linearization of the demand function yields the network pricing problem studied in [8]. In the second, one replaces the lower-level by its optimality conditions in order to derive a single-level program P1 whose major difficulty rests in the nonlinearity of the objective function.

4.1 The first heuristic model

Let $(\widetilde{t}, \widetilde{h}, \widetilde{T}, \widetilde{u})$ denote the incumbent solution of P. By linearizing the demand function $n_k(u_k)$ at $\widetilde{u}_k$ for each commodity, we have that $n_k(u_k) \approx a_k - b_k u_k$, where a_k and b_k are the coefficients of the tangent line to $n_k(u_k)$ at $\widetilde{u}_k$. For each commodity k, we substitute $n_k(u_k)$ for $a_k - b_k u_k$ in P and append the constraint $\widetilde{t} - \Delta t \leq t \leq \widetilde{t} + \Delta t$, where Δt is the radius of the trust region:

$$
\begin{aligned}
\text{TR1}: \quad & \max_{t \geq 0,\, h, u} \sum_{k \in \mathcal{K}} \sum_{p \in \mathcal{P}_k} (a_k - b_k u_k) \sum_{a \in p} t_a h_p^k \\
& \text{s.t.} \quad u_k = \sum_{p \in \mathcal{P}_k} \left(\sum_{a \in p} c_a + \sum_{a \in p \bigcap \mathcal{A}_1} t_a \right) h_p^k \\
& \qquad \forall k \in \mathcal{K} \begin{cases} h^k \in \arg\min_{h'^k} \sum_{p \in \mathcal{P}_k} \left(\sum_{a \in p} c_a + \sum_{a \in p \cap \mathcal{A}_1} t_a \right) h'^k_p \\ \qquad \text{s.t.} \sum_{p \in \mathcal{P}_k} h'^k_p = 1 \\ \qquad\qquad h'^k_p \in \{0, 1\}. \end{cases} \\
& \widetilde{t}_a - \Delta t_a \leq t_a \leq \widetilde{t}_a + \Delta t_a \qquad \forall a \in \mathcal{A}_1.
\end{aligned}
$$

This problem, which corresponds to the linear demand case, is efficiently addressed by the algorithm proposed in [8].

4.2 The second heuristic model

Let $(\widetilde{t}, \widetilde{h}, \widetilde{T}, \tilde{u})$ be the incumbent solution of $\overline{\mathrm{P}}$ and let n_k be the variable associated with the function $n_k(u_k)$. The linearization of the revenue function of commodity k at $(\widetilde{T}, \widetilde{u})$ is:

$$
\begin{aligned}
n_k(u_k) T^k &\approx n_k(\widetilde{u}_k) \widetilde{T}^k + n_k(\widetilde{u}_k)(T^k - \widetilde{T}^k) + \widetilde{T}^k (n_k - n_k(\widetilde{u}_k)) \\
&= n_k(\tilde{u}_k) T^k + \widetilde{T}^k n_k - n_k(\widetilde{u}_k) \widetilde{T}^k.
\end{aligned}
$$

For each commodity k, we substitute $n_k(u_k) T^k$ for $n_k(\widetilde{u}_k) T^k + \widetilde{T}^k n_k - n_k(\widetilde{u}_k) \widetilde{T}^k$ in $\overline{P}$ and append the constraint $\widetilde{t} - \Delta t \leq t \leq \widetilde{t} + \Delta t$. Since the variable n_k does not appear

in $\overline{P}$, we impose the constraint $n_k \leq a_k - b_k u_k$ where a_k, b_k are the coefficients of tangents to $n_k(u_k)$ at the point $\widetilde{u}_k$. This latter constraint prevents n_k from assuming an infinite value and enforces its proximity to the actual demand. This leads to the MIP

$$\begin{aligned} \text{TR2:} \quad & \max_{t,h,T,u} \sum_{k\in\mathcal{K}} (n_k(\widetilde{u}_k)T^k + \widetilde{T}^k n_k - n_k(\widetilde{u}_k)\widetilde{T}_k) \\ & \text{s.t.} \quad (1)-(2) \text{ and } (4)-(5) \\ & \qquad \widetilde{t}_a - \Delta t_a \leq t_a \leq \widetilde{t}_a + \Delta t_a \qquad \forall a \in \mathcal{A}_1 \\ & \qquad n_k \leq a_k - b_k u_k \qquad \forall k \in \mathcal{K}. \end{aligned}$$

The single-level reformulation of TR1 involves $\sum_{k\in\mathcal{K}} |\mathcal{P}_k| - |\mathcal{K}|$ continuous variables and $\sum_{k\in\mathcal{K}} |\mathcal{P}_k|$ more constraints than TR2.

5 Numerical results

Numerical tests have been performed on randomly generated grid networks, a topology that has been shown to be challenging for network pricing problems. Grid Networks were designed as indicated in [8]. It has been observed in practice that exact solutions to TR1 and TR2 are not necessary to obtain good toll solutions.

The main parameters of the test problems are $|\mathcal{N}|$ (number of nodes), $|\mathcal{A}|$ (number of arcs), $\%t$ (the percentage of toll arcs), and $|\mathcal{K}|$ (number of commodities). We consider demand functions as described in Sect. 2, i.e, they either have a constant elasticity or decay exponentially. The parameters of the demand functions were chosen to make most commodities attractive. When the demand functions have a constant elasticity, we set

$$\alpha_k = \left\lceil \frac{\xi_k (1.5\overline{C}_k)^{1.5}}{10^5} \right\rceil,$$

where $\overline{C}_k$ is the average fixed cost of paths associated with OD pair k, and ξ_k a random number drawn from the uniform distribution over the interval

$$\left[\exp\left(8\left\{\frac{1}{2} + \frac{3}{2}\frac{\gamma_k^\infty}{\overline{C}_k}\right\}\right)\right], \left[\exp\left(8.5\left\{\frac{1}{2} + \frac{3}{2}\frac{\gamma_k^\infty}{\overline{C}_k}\right\}\right)\right].$$

For demand functions with exponential decay,

$$\alpha_k = \left\lceil \frac{\xi_k \exp(1.5\overline{C}_k)}{10^5} \right\rceil,$$

with the aim of generating significant demand for all commodities.

Algorithms were coded in C++ and solved by the commercial MIP solver Cplex [7] under its default settings. The procedure 'IloPiecewiseLinear' of the 'ILOG CPLEX Callable Library' allowed the introduction of the piecewise linear cuts. The algorithms

Algorithm 2 Heuristic algorithm

Require: Trust region parameters Δ_0, $\Delta_{\min}$, $\eta_1 \leq \eta_2 < 1$, $< \delta_1 < 1 < \delta_2$, $k_{\max}$, ε.

Let $(t^{(0)}, h^{(0)}, T^{(0)}, u^{(0)})$ be an initial feasible solution.

$k \leftarrow 0$

StopCriterion $\leftarrow$ **false**.

repeat

Solve sub-problems (TR1 or TR2) for the respective solution $(\overline{t}, \overline{h}, \overline{T}, \overline{u})$.

$t^* \leftarrow \text{IOP}(\overline{h})$

Let R (respectively R_{m}) be the objective function of P (respectively TR1 or TR2).

$$\text{Let} \quad \phi_k \leftarrow \frac{R(t^{(k)}, h^{(k)}, T^{(k)}, u^{(k)}) - R(t^*, \overline{h}, T^*, u^*)}{R_{\text{m}}(t^{(k)}, h^{(k)}, T^{(k)}, u^{(k)}) - R_{\text{m}}(\overline{t}, \overline{h}, \overline{T}, \overline{u})}$$

if $\phi_k < \eta_1$ **then**

$(t^{(k+1)}, h^{(k+1)}, T^{(k+1)}, u^{(k+1)}) \leftarrow (t^{(k)}, h^{(k)}, T^{(k)}, u^{(k)})$

$\Delta_{k+1} \leftarrow \delta_1 \Delta_k$

else

$(t^{(k+1)}, h^{(k+1)}, T^{(k+1)}, u^{(k+1)}) \leftarrow (t^*, \overline{h}, T^*, u^*)$

if $\phi_k \geq \eta_2$ **then**

$\Delta_{k+1} \leftarrow \delta_2 \Delta_k$.

end if

end if

if $k > k_{\max}$ **or** $\|t^{(k+1)} - t^{(k)}\| \leq \varepsilon$ **or** $\phi_k \approx 1$ **or** $\Delta_{k+1} < \Delta_{\min}$ **or** $|R_{\text{m}}(t^{(k)}, h^{(k)}, T^{(k)}, u^{(k)}) - R_{\text{m}}(\overline{t}, \overline{h}, \overline{T}, \overline{u})| \leq \varepsilon$ **then**

StopCriterion $\leftarrow$ **true**

else

$k \leftarrow k + 1$

end if

until StopCriterion

were run on an Intel Core 2 Duo (2.13GHz) processor PC, under the GNU/Linux openSUSE operating system.

The major issue concerning the exact algorithm is the solution $\widetilde{\text{P1}}$, which takes 99% of the CPU time. In order to take advantage of the information provided by the Branch-and Bound (BB) procedure, inverse optimization was implemented with respect to the flow vectors, at every 1000th iteration. Whenever the inverse optimization yielded an objective value inferior to the incumbent lower bound, a flow cut (29) was appended to the model. Otherwise, the incumbent lower bound was updated. At each iteration

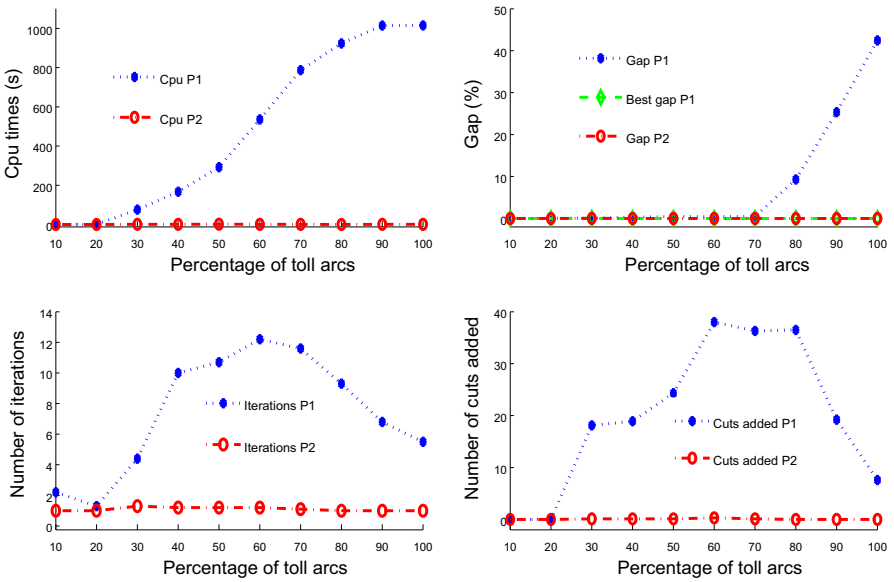

Fig. 7 Comparison between P1 and P2 on instances of 5 commodities. Constant elasticity set to 2. Average taken over 10 random instances. 'Best gap P1' is the absolute difference between the best solution achieved by P1 and the upper bound provided by P2

of the exact algorithm, an initial solution to $\widetilde{\text{P1}}$ provides a lower bound, used to prune nodes in the BB tree. At each iteration, initial flows are assigned to the cheapest paths according to the travel costs obtained at the previous iteration.

Figures 7 and 8 illustrate the respective performance of formulations P1 and P2, when demand elasticity is constant. The time limit is set to 1000 s, the stopping criterion to $\epsilon = .001$, and the number of discretization points to 20.

We observe that the best solution found by both methods are almost identical, but that the upper bound associated with P1 decreases very slowly compared to the upper bound provided by P2. Indeed, the over-approximation of $n_k(u_k)u_k$ in the objective of P1 requires several additional cuts (extra binary variables) before reaching an acceptable solution, in contrast with $n_k(\sum_{p\in p_0^k} c_a + \tau^k)\tau^k$ in the objective of $\widetilde{\text{P2}}$. Moreover, whenever elasticity increases, the difficulty of solving P1 (respectively P2) increases (respectively decreases). This is mainly due to the convex part of $n_k(u_k)u_k$ becoming wider and the curvature of $n_k(u_k)u_k$ higher as the elasticity increases, while the convex part of $n_k(\sum_{p\in p_0^k} c_a + \tau^k)\tau^k$ shrinks. Since these observations hold for all demand functions, we will only consider formulation P2 from now on.

From a computational point of view, a trade-off must be achieved between computational tractability and accuracy. We observe that it is worth increasing the number of discretization points, since the combinatorics of the problem comes more from the number of paths involved in the model than the number of linear pieces in the approximations of the functions. Of course, beyond a certain threshold, additional pieces are simply useless. Following experiments with 5, 10, 20 or 30 discretization points, it

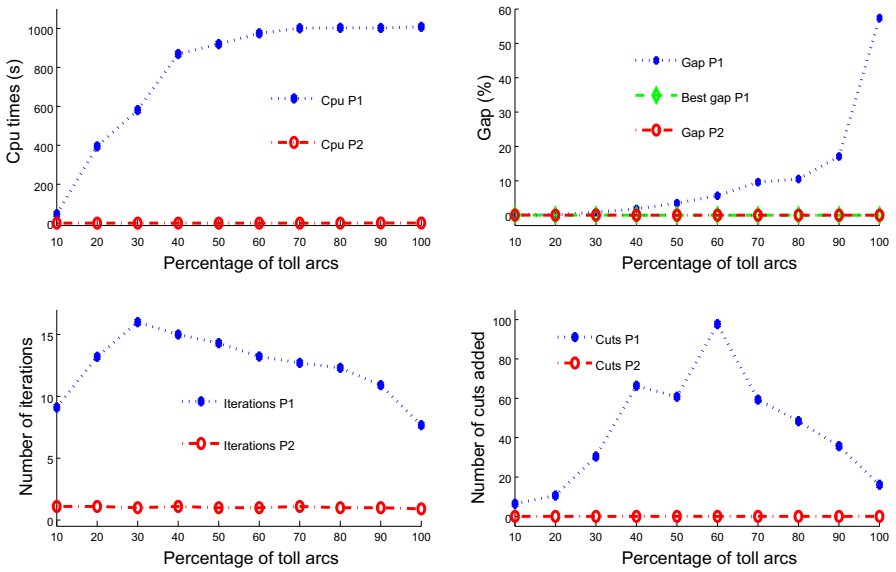

Fig. 8 Comparison between P1 and P2 on instances of 5 commodities. Constant elasticity set to 6. Average taken over 10 random instances

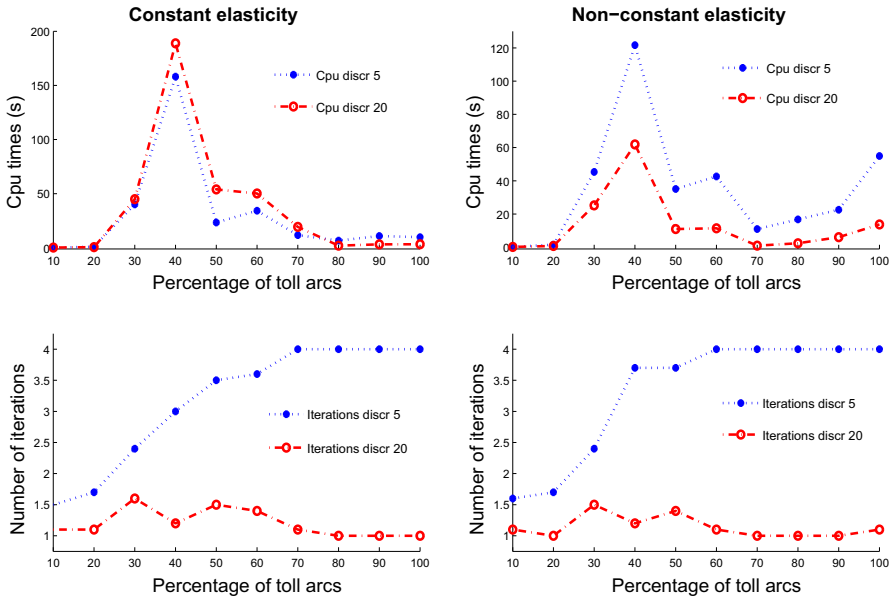

Fig. 9 Comparison between 5 and 20 discretization points. Average taken over 10 generated problems for 10 commodities. Constant elasticity set to 2

was observed that the number 20 represented a suitable compromise. For instance, see Fig. 9 for a comparison between 5 versus 20 discretization points.

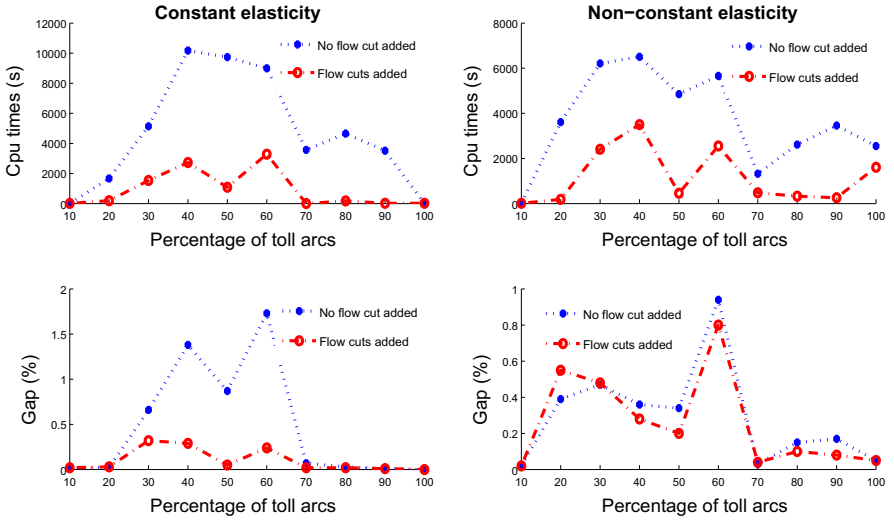

Fig. 10 Impact of flow cuts in formulation P2. Averages taken over 10 instances of 30 commodities

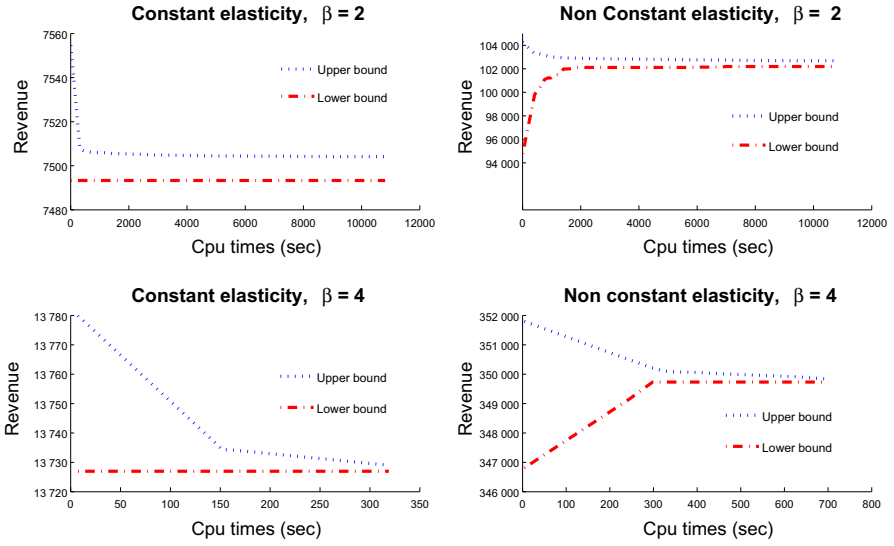

Fig. 11 Evolution of lower and upper bounds on instances of 40 commodities with 50% tolls arcs

Figure 10 highlights the performance of the exact method when flow cuts are appended to the original formulation. These allow to avoid paths with estimated high revenue, but actual revenue less than the incumbent lower bound. This resulted in an upper bound that decreases very fast, as well as a lower bound that increases in a stepwise manner (this behaviour can be observed on Fig. 11). We notice the good performance of the exact method when flow cuts are introduced during the process.

Table 1 TR2: elasticity set to 2

$\|\mathcal{K}\|$	%t	Constant elasticity						Non constant elasticity					
		cpu	cpu*	gap	#it	#fcut	#tlimit	cpu	cpu*	gap	#it	#fcut	#tlimit
10	10	0.20	0.15	0.01	1.1	0	0	0.22	0.15	0.01	1.1	0	0
	20	0.63	0.54	0.01	1.1	0	0	0.77	0.77	0.01	1.0	0	0
	30	44.94	9.06	0.02	1.6	0	0	25.19	5.77	0.01	1.5	0	0
	40	188.90	5.96	0.02	1.2	0	0	61.90	12.83	0.03	1.2	2	0
	50	53.91	11.39	0.01	1.5	2	0	10.97	7.24	0.02	1.4	0	0
	60	567.16	2.11	0.01	1.4	4	0	11.41	3.37	0.01	1.1	0	0
	70	19.32	0.98	0.01	1.1	0	0	0.98	0.95	0.01	1.0	0	0
	80	2.03	1.32	0.00	1.0	0	0	2.35	1.52	0.01	1.0	0	0
	90	3.16	2.67	0.00	1.0	0	0	6.04	2.96	0.01	1.0	0	0
	100	3.29	3.29	0.00	1.0	0	0	13.72	5.45	0.01	1.1	0	0
20	10	1.40	0.80	0.01	1.3	0	0	1.22	1.19	0.01	1.0	0	0
	20	21.45	14.89	0.02	1.1	0	0	44.38	33.84	0.02	1.1	2	0
	30	2311.64	1101.55	0.08	1.5	2	2	1119.24	541.50	0.03	1.5	0	0
	40	1828.67	355.88	0.07	1.8	13	1	1511.77	417.27	0.14	1.7	8	1
	50	1583.57	476.46	0.10	1.8	11	1	3301.31	1183.60	0.16	1.5	0	2
	60	3262.75	1237.35	0.09	1.8	11	2	2415.76	1182.18	0.14	1.6	0	2
	70	303.54	3.19	0.01	1.2	2	0	21.69	10.44	0.02	1.1	0	0
	80	1128.66	9.86	0.01	1.0	0	1	30.60	8.25	0.01	1.0	0	0
	90	8.13	8.13	0.00	1.0	0	0	25.93	8.94	0.01	1.0	0	0
	100	10.17	10.17	0.00	1.0	0	0	88.81	56.67	0.01	1.0	0	0

Table 1 continued

$\|\mathcal{K}\|$	%t	Constant elasticity						Non constant elasticity					
		cpu	cpu*	gap	#it	#fcut	#tlimit	cpu	cpu*	gap	#it	#fcut	#tlimit
30	10	11.27	8.02	0.02	1.3	0	0	12.53	11.34	0.02	1.0	0	0
	20	997.62	187.79	0.03	1.6	5	0	3681.90	189.40	0.55	1.4	7	1
	30	4315.61	1536.24	0.32	1.9	9	3	6810.82	2407.60	0.48	1.5	1	6
	40	4988.10	2730.24	0.29	2.3	27	4	6402.97	3500.91	0.28	2.1	7	4
	50	3305.77	1080.89	0.05	2.0	14	1	4795.23	458.77	0.20	1.7	5	3
	60	5580.26	3285.48	0.24	2.1	15	3	5619.77	2556.50	0.80	1.6	3	4
	70	1279.74	6.67	0.02	1.2	1	0	1759.76	479.80	0.04	1.5	0	0
	80	1414.95	179.71	0.02	1.1	0	1	2527.07	329.85	0.10	1.3	0	2
	90	17.29	17.22	0.01	1.0	0	0	3535.56	256.97	0.08	1.0	0	3
	100	20.79	20.79	0.00	1.0	0	0	3590.73	1610.81	0.05	1.2	0	2
40	10	35.04	20.03	0.02	1.3	0	0	36.18	7.35	0.02	1.0	0	0
	20	5521.78	2261.11	0.67	1.1	1	5	5906.16	1982.92	2.55	1.2	0	5
	30	6312.49	2738.21	0.65	1.5	4	5	8539.11	3785.09	1.25	1.4	1	7
	40	7658.57	2470.77	0.21	2.2	13	5	9529.25	4587.06	0.52	1.9	26	8
	50	5858.34	2490.00	0.27	1.5	9	3	6503.23	2332.74	0.20	1.8	7	4
	60	8880.35	5496.53	0.46	1.5	11	7	9824.63	4096.49	0.83	1.6	8	9
	70	6153.26	317.13	0.07	1.1	2	5	8365.21	2729.91	0.50	1.4	1	7
	80	2551.03	1064.43	0.04	1.1	2	1	6851.81	2970.97	0.50	1.1	2	6
	90	30.40	29.80	0.02	1.0	0	0	7313.39	3665.34	0.45	1.1	0	6
	100	33.13	33.13	0.00	1.0	0	0	6516.07	2502.75	0.56	1.2	0	6

Table 2 TR2: elasticity set to 6

$\|\mathcal{K}\|$	%t	Constant elasticity						Non constant elasticity					
		cpu	cpu*	gap	#it	#fcut	#tlimit	cpu	cpu*	gap	#it	#fcut	#tlimit
10	10	0.51	0.13	0.02	3.0	0	0	0.23	0.16	0.01	1.2	0	0
	20	0.38	0.15	0.02	1.3	0	0	0.57	0.34	0.02	1.4	0	0
	30	0.50	0.25	0.01	1.1	0	0	1.29	0.59	0.03	1.3	0	0
	40	0.43	0.30	0.01	1.0	0	0	0.91	0.40	0.02	1.0	0	0
	50	1.33	0.54	0.01	1.1	0	0	1.71	1.04	0.01	1.1	0	0
	60	2.24	0.73	0.01	1.0	0	0	3.33	0.76	0.01	1.0	0	0
	70	0.89	0.87	0.00	1.0	0	0	1.14	0.97	0.01	1.1	0	0
	80	1.27	1.26	0.00	1.0	0	0	1.85	1.51	0.01	1.1	0	0
	90	2.78	2.48	0.00	1.0	0	0	3.32	2.89	0.01	1.1	0	0
	100	2.73	2.73	0.00	1.0	0	0	8.59	3.68	0.01	1.2	0	0
20	10	1.04	0.65	0.02	1.3	0	0	2.71	2.33	0.01	1.4	0	0
	20	2.83	0.74	0.01	1.6	0	0	12.68	3.20	0.01	1.8	1	0
	30	3.83	2.33	0.02	1.0	0	0	13.56	4.95	0.03	1.3	0	0
	40	3.60	0.84	0.02	1.1	0	0	144.28	118.88	0.02	1.3	0	0
	50	10.31	2.23	0.01	1.0	0	0	29.92	12.16	0.02	1.4	0	0
	60	11.32	8.79	0.01	1.1	0	0	1096.98	662.25	0.16	1.3	0	1
	70	3.56	3.04	0.00	1.0	0	0	4.55	3.18	0.01	1.1	0	0
	80	8.92	4.25	0.00	1.0	0	0	9.05	4.64	0.01	1.1	0	0
	90	21.74	7.61	0.00	1.0	0	0	21.43	8.66	0.01	1.1	0	0
	100	8.29	8.29	0.00	1.0	0	0	41.67	11.91	0.01	1.2	0	0

Table 2 continued

$\|\mathcal{K}\|$	%t	Constant elasticity						Non constant elasticity					
		cpu	cpu*	gap	#it	#fcut	#tlimit	cpu	cpu*	gap	#it	#fcut	#tlimit
30	10	2.12	1.42	0.01	1.5	0	0	6.41	2.52	0.01	1.8	0	0
	20	5.61	2.02	0.02	1.3	0	0	42.69	25.26	0.01	1.9	0	0
	30	19.66	6.98	0.02	1.1	0	0	108.67	68.38	0.03	1.7	0	0
	40	20.16	1.58	0.03	1.1	0	0	385.54	360.17	0.03	1.3	0	0
	50	36.29	8.33	0.02	1.0	0	0	173.62	51.74	0.03	1.5	2	0
	60	32.48	18.69	0.01	1.1	0	0	1243.42	934.16	0.11	1.6	1	1
	70	8.02	6.33	0.01	1.1	0	0	74.42	16.35	0.01	1.3	0	0
	80	50.63	9.04	0.00	1.1	0	0	292.43	55.60	0.02	1.4	0	0
	90	57.15	16.13	0.00	1.0	0	0	443.05	64.31	0.01	1.3	0	0
	100	16.72	16.72	0.00	1.0	0	0	1684.33	324.48	0.03	1.4	0	1
40	10	14.25	2.00	0.01	1.5	0	0	14.95	10.22	0.02	1.4	0	0
	20	11.01	7.39	0.02	1.3	0	0	806.49	77.82	0.03	1.8	1	0
	30	45.89	33.67	0.02	1.3	0	0	158.44	61.16	0.02	1.7	0	0
	40	50.75	25.63	0.02	1.3	0	0	269.14	204.35	0.03	1.5	2	0
	50	72.13	27.53	0.01	1.3	0	0	754.11	313.30	0.03	1.8	0	0
	60	68.14	43.15	0.01	1.1	0	0	4464.12	2128.44	0.19	1.7	9	3
	70	34.64	10.46	0.01	1.1	0	0	4092.07	253.04	0.24	1.4	9	3
	80	14.79	13.85	0.01	1.0	0	0	4549.36	1622.67	0.24	1.6	0	3
	90	28.18	27.02	0.01	1.0	0	0	4222.72	1211.36	0.22	1.2	0	3
	100	25.81	25.81	0.00	1.0	0	0	5819.43	862.35	0.43	1.2	0	4

Table 3 Parameters for the exact method and column headers for tables

Parameters	
Stopping criterion (ε)	1e−2
Time limit	10,800 s
Percentage of toll arcs	Varies from 10 to 100%
Column headers	
cpu	Running time (s)
cpu*	Instant when the method achieves the best solution
gap	Gap between the upper and lower bounds
gap*	Gap between the best upper and lower bound
#it	Number of iterations
#fcuts	Number of flow cuts added
#tlimit	Number of instances that have reached the time limit

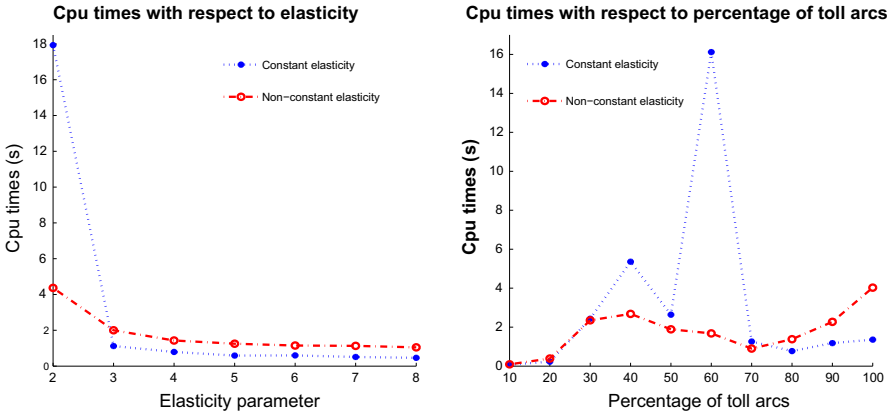

Fig. 12 Cpu times with respect to elasticity (left) and percentage of toll arcs (right). Average taken over 10 generated problems for 5–10 commodities

To assess the potential of the exact method on P2, we applied it to instances ranging from 10 to 40 commodities. The results of our computational experiments are presented in Tables 1 and 2 , each table summarizing an average taken over 10 randomly generated problems. The values of parameters and column headers for the tables are displayed in Table 3.

Formulation P2 allows to solve small to medium scale problems (up to 10,000 paths) at quasi optimality in $<$ 1,h of CPU time. On the larger instances, the running time increases very fast due to the number of binary variables that grows exponentially with network size. It is interesting to note (see Fig. 12) that running time decreases as elasticity increases. Intuitively, large elasticities imply a fast decrease of demand, which makes the path with smallest fixed travel cost all the more attractive for the leader. This reasoning is supported by the results displayed in Fig. 13, where the gap

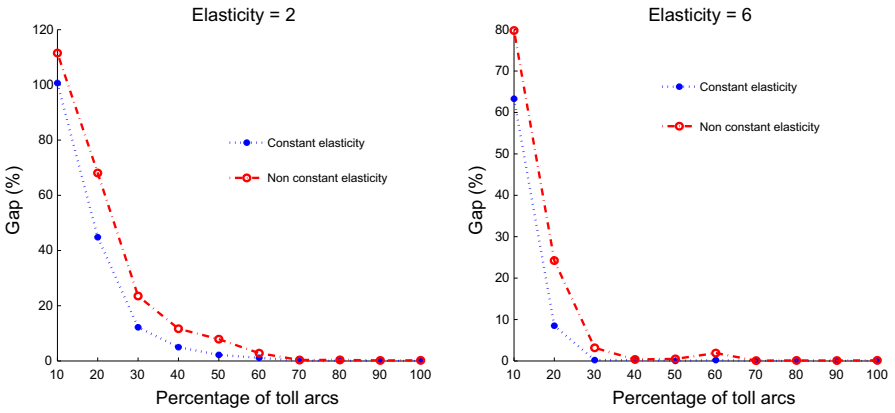

Fig. 13 Gap between upper bound provided by exact method and solution found when selected paths are the shortest ones with respect to the fixed travel cost. Average taken over 10 generated problems for 10, 20, 30 and 40 commodities

between the upper bound provided by $\widetilde{\text{P2}}$ and an inverse optimization performed on flows assigned to cheapest paths (with respect to fixed costs) decreases as elasticities increase. On the other hand, no clear conclusion can be drawn concerning the sensitivity with respect to the percentage of toll arcs. Notwithstanding, the solution obtained by selecting the paths with smallest fixed travel cost provides a good lower bound, although exact methods fail to provide a small gap, due to the bad quality of the upper bound. Note that, as a general rule, it is easier to solve instances where all arcs can be tolled.

We finally assessed the efficiency of the two heuristics, setting the parameters to those recommended in [3], with the exception of the number of iterations $k_{\max}$ and the maximum number of unsuccessful iterations mxnuns. These values are reported in Table 4 where nbpoint (respectively ptuns) denotes the number of starting points (respectively the number of consecutive points where the best solution is not improved).

While the models arising in the trust region algorithm are simpler than the original bilevel program, they are yet strongly NP-hard. For this reason, we decided to set the time limit to 1000 s and to allow more time for diversification. Alternatively, one can avoid generating already encountered flows through the introduction of the flow cut (29). In both cases, inverse optimization can be used to optimize revenues with respect to current feasible flow patterns. Table 5 illustrates the efficiency of the heuristic algorithms when elasticity is set to 2, comparing with P2 for various problem sizes.

We observe that the quality of solutions provided by the exact and heuristic method are similar when the number of commodities is small. This can also be observed on Figs. 14, 15 and 16. Heuristic algorithms always find quickly the best solution compared to the exact method (see column cpu*). Table 6 summarizes the ratio of running time over the three methods when they find the best solution.

Whenever the number of commodities exceeds 30, the ratio P1/TR2 increases and reaches 2.27, while no firm conclusion can be drawn concerning the ratio P2/TR1.

Table 4 Parameters for the heuristic algorithms

Trust region parameters	
Δ_0	10
η_1	0.01
η_2	0.9
γ_1	0.6
γ_2	1.4
nbpoint	11
Stopping criteria	
$k_{\max}$	30
$\Delta_{\min}$	$1e-6$
ε	$1e-6$
mxnuns	10
ptuns	5
Time limit	10,800 s

As can be observed in column #fcuts, the number of flow cuts is highest with TR2, followed by TR1 and the exact method, which might explain why TR2 finds good solutions quickly. Due to the difficulty of solving sub-problems involving a large number of commodities, #fcuts decreases as the number of commodities increases. Figure 17 shows that #fcuts is concave (increasing then decreasing) with respect to the percentage of toll arcs, and reaches its maximum (respectively minimum) value between 40 and 50% (respectively when all arcs are tolled). In summary, the heuristics provide a good compromise between computational time and solution quality whenever the number of commodities exceeds 30 and elasticities are small. When elasticities are high, the exact method is to be preferred.

6 Conclusion

In this paper, we have considered efficient formulations and algorithms for addressing a network pricing problem involving nonlinear demand functions. We proposed for its solution an asymptotically exact method, as well as powerful heuristic procedures based on a bilevel approximation of the original problem, cast within the framework of trust region methods. The numerical tests illustrate the good behavior of our methods and the robustness of the associated code. From a qualitative point of view, it was observed that the instances involving high elasticities were easier to solve, due to a decrease in the combinatorial complexity of the problem.

Although the introduction of variable and nonlinear demand is an important step forward, yet it does not account for situations where commodity flows are not assigned to the same path. For instance, dispersion of traffic along the paths of the network could result from congestion, or from the presence of rigid link capacities. In both cases, the structure of the lower-level problem is significantly modified, and the resulting

Table 5 Comparison between exact method and heuristics; $\beta_k = 2, 20, 30, 40$ and 50 commodities

Elasticity	#OD	P2				TR1				TR2			
		cpu	cpu*	gap	#fcuts	cpu	cpu*	gap	#fcuts	cpu	cpu*	gap	#fcuts
Constant	20	1046.01	321.83	0.04	4.02	1393.90	375.06	0.13	83.66	2603.81	188.24	0.14	143.07
	30	2204.21	907.38	0.10	7.38	3858.93	783.79	0.17	70.44	5071.33	622.98	0.15	133.29
	40	4364.72	1717.18	0.25	4.58	5177.20	1037.02	0.37	70.41	5906.78	1005.99	0.32	124.69
	50	5590.72	2955.09	0.58	4.77	6321.27	2070.72	0.70	67.19	6593.47	1301.85	0.65	79.75
	60	6743.04	3235.25	0.72	5.87	6690.63	1993.77	0.97	48.03	6979.14	1795.67	0.81	80.29
Non-constant	20	856.07	344.39	0.06	1.19	2488.39	404.83	0.23	101.81	3638.63	321.13	0.22	118.58
	30	3873.63	1180.19	0.26	2.70	5507.07	947.52	0.43	84.51	5224.95	954.65	0.39	92.83
	40	6938.50	2866.06	0.74	4.92	6087.03	1389.41	1.11	59.94	6246.27	1713.72	1.00	88.54
	50	8234.22	3529.06	1.31	3.66	6986.93	2270.89	1.87	51.44	6409.13	1675.65	1.89	62.63
	60	9149.47	4681.94	1.56	3.92	7131.06	2562.60	2.34	52.30	6762.47	2133.39	2.21	63.15

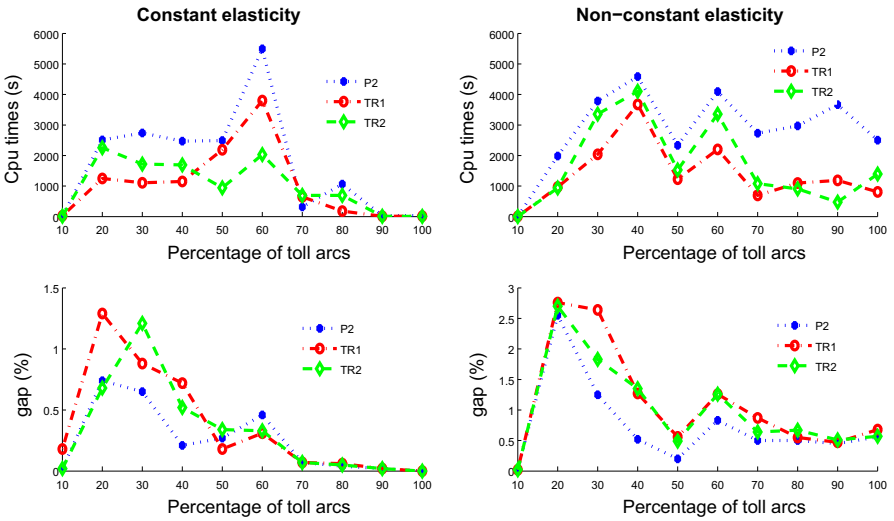

Fig. 14 Comparison between exact method and heuristics. Average taken over 10 generated problems for 40 commodities. Elasticity set to 2

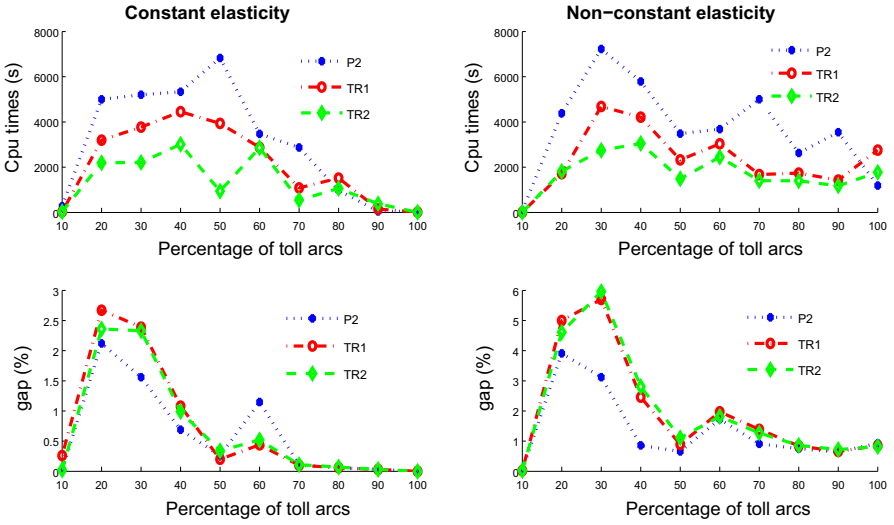

Fig. 15 Comparison between exact method and heuristics. Average taken over 10 generated problems for 50 commodities. Elasticity set to 2

model calls for a different algorithmic approach. Another key assumption of our model is that the value-of-time parameter is uniform throughout the user population, i.e., given the choice between two paths of equal costs, users always select the one with highest toll. This drawback could be eliminated by introducing a range of value-of-time parameters across users. If this range is continuous, the lower level problem becomes

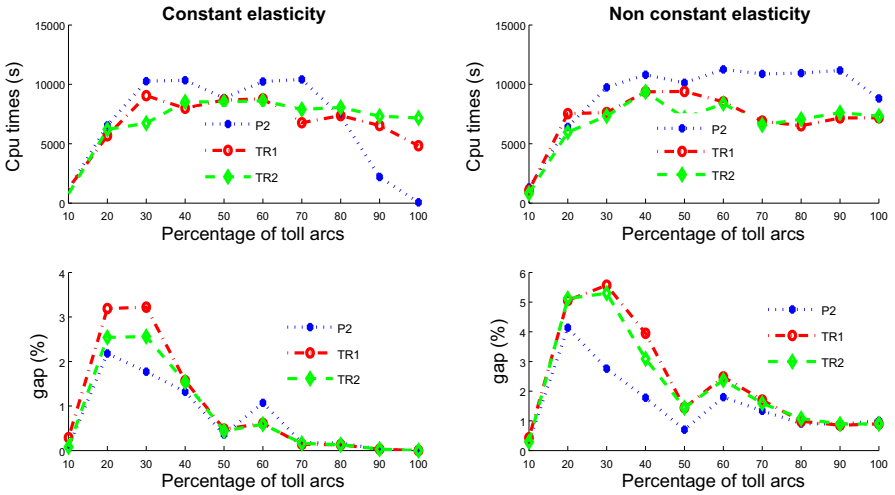

Fig. 16 TR2: Comparison between exact method and heuristics. Average taken over 10 generated problems for 60 commodities. Elasticity set to 2

Table 6 Running time ratio between exact method and heuristic algorithms; $\beta_k = 2$

#OD	Constant elasticity			Non-constant elasticity		
	P2/TR1	P2/TR2	TR1/TR2	P2/TR1	P2/TR2	TR1/TR2
20	0.86	1.71	1.99	0.85	1.07	1.26
30	1.16	1.46	1.26	1.25	1.24	0.99
40	1.66	1.71	1.03	2.06	1.67	0.81
50	1.43	2.27	1.59	1.55	2.11	1.36
60	1.62	1.80	1.11	1.83	2.19	1.20

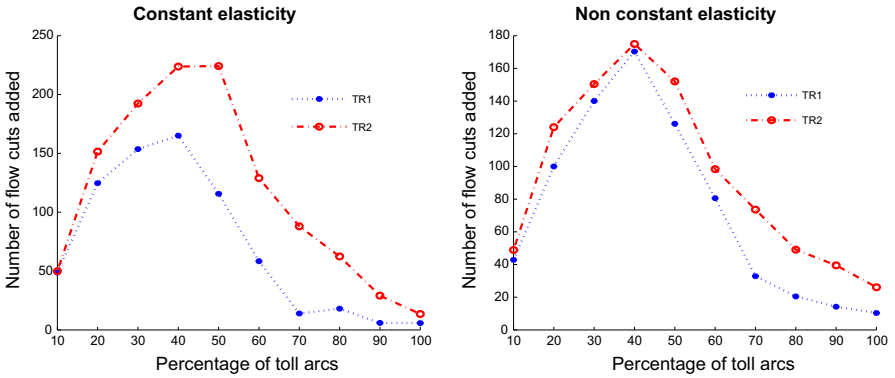

Fig. 17 Number of flow cuts (29) added during execution of heuristics with respect to the percentage of toll arcs. Average taken over 10 generated problems for 10, 20, 30, 40 and 50 commodities

an infinite-dimensional linear problem. As indicated by Marcotte and Zhu [11], an efficient solution to this lower level problem can be obtained by solving a parametric cheapest path problem. These extensions of the model are left to future studies.

References

1. Al-Khayyal, F.A.: Jointly constrained bilinear programs and related problems: an overview. Comput. Math. Appl. **19**, 53–62 (1990)
2. Audet, C., Hansen, P., Jaumard, B., Savard, G.: A branch and cut algorithm for nonconvex quadratically constrained quadratic programming. Math. Program. **87**, 131–152 (1999)
3. Colson, B., Marcotte, P., Savard, G.: A trust-region method for nonlinear bilevel programming: algorithm and computational experience. Comput. Optim. Appl. **30**, 211–227 (2005)
4. Conn, A.R., Gould, N.I.M., Toint, PhL: Trust-Region Methods. SIAM, Philadephia (2000)
5. Dewez, S., Labbé, M., Marcotte, P., Savard, G.: New formulations and valid inequalities for a bilevel pricing problem. Oper. Res. Lett. **36**, 141–149 (2008)
6. Didi-Biha, M., Marcotte, P., Savard, G.: Path-based formulations of a bilevel toll setting problem. In: Dempe, S., Kalashnikov, V. (eds.) Optimization with Multivalued Mappings Theory: Theory, Applications and Algorithms, pp. 29–50. Springer, Boston (2006)
7. ILOG CPLEX v10.1: Using CPLEX Callable Library and CPLEX Mixed Integer Programming (2006)
8. Kuiteing, A.K., Marcotte, P., Savard, G.: A network pricing problem under linear demand. Transp. Sci. **51**, 791–806 (2016)
9. Kelley Jr., J.E.: The cutting-plane method for solving convex programs. J. Soc. Ind. Appl. Math. **8**, 703–712 (1960)
10. Labbé, M., Marcotte, P., Savard, G.: A bilevel model of taxation and its application to optimal highway pricing. Manag. Sci. **44**, 1595–1807 (1998)
11. Marcotte, P., Zhu, D.L.: Equilibria with infinitely many differentiated classes of customers. In: Ferris, M.C., Pang, J.S. (eds.) Complementary and Variational Problems. State of Art. SIAM, Philadephia (1997)