A Heuristic Solution of the Toll Optimal Problem With Congestion Affected Costs *

Vyacheslav Kalashnikov[1,2,3], José G. Florez-Muñiz[4], Nataliya Kalashnykova[3,4], Vladik Kreinovich[5]

[1]*Tecnológico de Monterrey (ITESM), Campus Monterrey, Monterrey, NL, Mexico, 64849; kalash@itesm.mx*

[2]*Central Economics and Mathematics Institute (CEMI), Moscow, Russian Federation, 117418; slavkamx@mail.ru*

[3]*Sumy State University, Sumy, Ukraine, 40007; slavkmax@gmail.com*

[4]*Universidad Autónoma de Nuevo León (UANL), San Nicolás de los Garza, NL, Mexico; jose_guadalupe64@hotmail.com*

[5]*University of Texas at El Paso (UTEP), El Paso, TX 79968, USA; vladik@utep.edu*

***Abstract*— An important problem concerning the toll roads is the setting of appropriate costs for driving along paid arcs of a transportation network. Our paper treats this problem as a bilevel programming model. At the upper level, decisions are made by a public regulator/private company that administers the toll roads endeavoring to elevate their benefits. At the lower level, several transportation companies/individual users appease the existing demand for transportation of goods or passengers while selecting the routes that would minimize their total travel costs. In contrast to the previous models, here the lower level problem assumes quadratic costs implied by the possible traffic congestion. Aiming to find a solution to the bilevel programming problem, a plain method based on sensitivity analysis for quadratic programs is brought forward. In order to "jump" (if necessary) from a local maximum of the upper level objective function to a vicinity of another, the "filled function" move is applied. The proposed algorithms are genuine and work efficiently enough when employed to solve small- and medium-sized test numerical problems.**

***Keywords*— Bilevel problems; congestion-affected costs; sensitivity analysis; "filled functions" method**

I. INTRODUCTION

Along the first periods of the industrial revolution, the producing plants and factories used to be located not too far from the markets, since the goods movement was rather costly, taking a lot of time, and lacking security. With the development of the modern transportation systems the producers are able to compete in front of the distant consumers as well, which promotes economies of scale by increasing sales volume.

The up-to-date complexity of production and global approach, distribution and supply chains have become widespread and greatly encompassing, which, in turn, has made logistics costs grow enormously. As the IMF (International Monetary Fund) asserts, logistics costs contribute as much as 12% of the gross national product, so ranging from 5% to 31% of the sales at the plant/factory caliber.

The latter development taken into account, many companies and states have appended gross relevancy to the enlargement and modernization of the infrastructure to reach a better participation in the global economy. There exist bodies dealing with the improvement of communications and transference infrastructure, inventing new technologies in order to enhance the amends, excellence, and durability of the infrastructure. In Mexico, operations with new (private) highways are usually granted to non-government enterprises, state governments, or financial institutions (banks, holdings, etc.), who assign the movement toll to pick up payments from the motorists.

During the last ten years, it has been noticed that there is less heavy traffic under the concession model applied to govern these tolled highways. One of the efficient ways to elevate the use of toll highways is the tight control of the tolls (pass rates). Therefore, a natural question arises about the appropriate criteria to evaluate these rates.

Hence, this is the problem of assigning optimal tolls to the arcs of a multi-commodity transportation network. The toll optimization problem (TOP) is commonly formulated as a bilevel mathematical programming model where the upper level is operated under a control by a company (or a public regulator) raising profits from the tolls assigned to (certain) arcs of the network, while the lower level comprises an array of drivers (transportation companies) riding the cheapest paths with respect to a generalized travel cost. The problem can be read as how to find a kind of equilibrium among the toll values that produce high revenues and still are attractive for the drivers.

The considered problem question has been examined by many a prominent researcher. To demonstrate that we provide a comprehensive literature review related to the TOP. First, Magnanti and Wong [1] presented a complete theoretical basis for the drivers and restrictions of network design based on integer programming accompanied by several models and algorithms. This approach rendered a kind of unification of network design models, as well as a general blueprint to develop network design algorithms. They noticed that

researchers had been motivated to develop a variety of solution techniques such as linear approximation methods and the search of vertices adjacent to the lowest cost flow problem treated as a network design problem (NDP).

Marcotte [2] asserts that the NDP principally works with the optimal balance either of the transportation, investment, or maintenance costs of the networks subject to congestion. The drivers mostly behave according to Wardrop's first principle of traffic equilibrium. Marcotte [2] also assumes that the NDP could be modeled as a multilevel mathematical programming problem.

Dempe and Starostina [3] examined TOP problem by designing "fuzzy" algorithm for the TOP. Next, Lohse and Dempe [4] based their studies on the analysis of optimization problem in some sense reverse to the TOP. In addition, Didi-Biha et al. [5] developed an algorithm based on the calculation of lower and upper bounds to determine the maximum gain from the tolls on a subset of arcs of a network transporting various commodities.

As is possible to assert, bilevel programming offers a handy carcass for modeling the optimal toll problems, since it permits one to make use of the drivers' behavior explicitly. In contrast to the works mentioned above, Labbé et al. [6] treated the TOP as a sequential game involving the owners of highway networks (the leaders) and the drivers (the followers) as the players, which suits exactly the scheme of a bilevel programming problem. Such a scheme is also studied and developed by Brotcorne [7] for the problem of fixing tariffs on cargo transportation. In the latter case, the leader comprises a group of competing companies, and their revenues are formed by the gross profits from the rates, while the follower is a carrier who tries to drop travel expenditures, given the toll values assigned by the leader(s).

One of the most primitive instances was examined in Kalashnikov et al. [8], where a TOP delineated for a congestion-free, multi-commodity transportation network was investigated. In the latter framework, a motorway administrator (the leader) appoints tolls on a subset of arcs of the network, while the users (the followers) select the shortest paths (in generalized time units) connecting their origin and destination nodes. The aim of the leader in this setting is to maximize the toll revenue. Hence, it is contrary to its benefit to arrange unwisely high tolls, because this would discourage the users from using the tolled sub-network. The problem could be formulated as a combinatorial program comprising NP-hard tasks, such as the Traveling Salesman Problem (*see*, Labbé et al. [9], for a reduction method). Exploiting the well-known NP-hardness proofs provided by previous authors, Roch et al. [10] established interesting results about the existing methods' computational complexity and approximation values.

Recently, Brotcorne et al. [11] examined this problem under a bit different assumptions; namely, they permitted the network to be subsided; this is, they allowed the toll values to be unconstrained. The authors concocted a method that constructing paths and then forming columns in order to find the optimal tariff values for the current path (the lower bound). After that, they amended the profit upper bound and in the end, realized a diversification stage. They also tested their numerical procedure on various examples of the problem in question to wind up that the bidding method performed quite well for networks with a limited number of toll arcs. The authors continued their work over the same problem later [12] having brought up a tabu search algorithm, which helped them to infer that their heuristics did obtain better results than other combinatorial methods. Dempe and Zemkoho [13] also explored the TOP and introduced its restatement founded on the optimal-value-function technique. This restatement is better than that making use of the Karush-Kuhn-Tucker (KKT) optimality conditions since the former accumulates the information about the congestion in the network. The authors [13] deduced the optimality conditions appropriate for this restatement and studied some related theoretical properties of the latter.

The objective of the present paper is to devise an algorithm founded upon the allowable ranges to stay optimal (ARSO) deduced with the aid of sensitivity analysis applied to the lower level quadratic problem (*cf.*, [14]–[16]). This effective instrument helps analyze allowable variations of the coefficients of the objective function that do not spoil the optimal solution. Also, it empowers one to examine the changes in the optimal solution when the parameters accept values outside the ARSO. This work has been enkindled by the previous attempts described in [10].

Besides our making use of the allowable ranges, the developed approach also exercises the notion of "filled function" (*see* [17]–[19]). The latter is daubed on when a local maximum has been encountered. In that case, the "filled function" procedure permits one either to run into another local maximum, which improves the previous ones, or to conclude that (probably) the best feasible optimal solution has been discovered. The stopping point is selected based upon certain tolerance criterion.

The cogency and soundness of the proposed method are confirmed by the results of numerical experiments with test examples used to compare the developed approach against the other well-known algorithms. The procedure's robustness is eventually illustrated by the above-mentioned numerical experiments as well.

The paper has the following structure: Section II provides the statement of the model together with the involved parameters. Section III deals with the algorithm's description, while in Section IV, the results of numerical experiments with several toll optimization test problems are presented. Section V comprises conclusions and the targets for future research. The acknowledgments and the list of references finish the paper.

II. TOLL OPTIMIZATION BILEVEL PROGRAM

In this section, we present a new formulation of the Toll Optimization Problem (TOP) that takes into account bounded capacities of the arcs of the transportation network. TOP can be analyzed as a leader-follower game that takes place on a multi-commodity network $G=(K,N,A)$ defined by a set of

origin-destination couples K, a set of nodes N, and a set of arcs A. The latter is partitioned into the subset A_1 of toll arcs and the complementary subset A_2 of toll-free arcs; here $|A_1| = M_1$ and $|A_2| = M_2$, thus yielding $M = M_1 + M_2$. We endow each arc $a \in A$ with a fixed travel delay c_a and a congestion parameter d_a. Also, there is an upper limit capacity q_a associated with each arc $a \in A$ in the network. Toll arcs $a \in A_1$ also involve a toll component t_a to be determined, which, for the sake of consistency, is also expressed in time units. This vector t_a of toll values is restricted by the vector $t^{max} = \{t_a^{max} : a \in A_1\}$. The numbers n^k denote the demand for transportation of a commodity $k \in K$ between the origin node $o(k)$ and the destination node $d(k)$; the total quantity of commodities is betokened as $|K| = r$. The demand for each commodity $k \in K$ in every node i is referred to by the values b_i^k (combined in the demand vector b^k) defined as follows:

$$b_i^k = \begin{cases} -n^k, \text{ if } \quad i = o(k), \\ n^k, \text{ if } \quad i = d(k), \\ 0, \quad \text{otherwise.} \end{cases} \tag{1}$$

Letting $\{x_a^k\}_{a \in A}$ denote the set of commodity flows, and i^+ (respectively, i^-) the set of arcs having i as their head node (respectively, tail node), TOP can be formulated as a bilevel program:

TOP:

$$F(x,t) = \sum_{k \in K} \sum_{a \in A_1} t_a x_a^k \to \max_{t,x}, \tag{2}$$

subject to

$$t_a \le t_a^{max}, \qquad \forall a \in A_1, \tag{3}$$

$$t_a \ge 0, \qquad \forall a \in A_1, \tag{4}$$

$$\forall k \in K \begin{cases} x^k \in Arg \min_{\bar{x}} \left[\sum_{a \in A_1} (c_a + t_a) \bar{x}_a^k + \sum_{a \in A_2} c_a \bar{x}_a^k + \frac{1}{2} \sum_{a \in A} d_a (\bar{x}_a^k)^2 \right], \\ \text{subject to} \\ -\sum_{a \in i^-} \bar{x}_a^k + \sum_{a \in i^+} \bar{x}_a^k = b_i^k, \quad \forall i \in N, \\ \bar{x}_a^k \ge 0, \qquad \forall a \in A, \\ \sum_{\ell \in K} \bar{x}_a^\ell \le q_a, \qquad \forall a \in A. \end{cases} \tag{5}$$

In the above formulation, the leader controls both the toll and flow variables. However, the lower level problem's 'Argmin' constraint reflects the followers' objective of minimizing the aggregate "transportation costs" (in the form of "time delay units" times the respective flow levels) subject to the current toll values and the supply-demand bounds. We would also like to point out that the TOP considered in [5] does not include the capacity constraints.

III. The Proposed Heuristics

In this algorithm, we blend the main structure of the method described in [8] with a following new idea. First, we determine the vector of "fastest increase" for the upper level objective function restricted to the toll variables. The "formal gradient" of this objective F function defined in (2) can be characterized by the present flows assigned to the toll arcs:

$$\frac{\partial F}{\partial t_a}(x,t) = \sum_{k \in K} \left(x_a^k + t_a \frac{\partial x_a^k}{\partial t_a} \right), \quad a \in A_1. \tag{6}$$

According to [20] – [21], the lower level Nash equilibrium problem (5) can be equivalently replaced with the quadratic programming problem with the objective function equal to the sum of all followers' objective functions:

$$f(x,t) = \sum_{k \in K} \left\{ \sum_{a \in A_1} t_a x_a^k + \sum_{a \in A} \left[c_a x_a^k + \frac{1}{2} d_a (x_a^k)^2 \right] \right\} \to \min_x, \tag{7}$$

s.t.

$$-\sum_{a \in i^-} x_a^k + \sum_{a \in i^+} x_a^k = b_i^k, \quad \forall i \in N, \forall k \in K; \tag{8}$$

$$\sum_{k \in K} x_a^k \le q_a, \quad \forall a \in A; \tag{9}$$

$$x_a^k \ge 0, \quad \forall k \in K, \forall a \in A. \tag{10}$$

The main difference of the lower level program (7) – (10) from the previous models mentioned in the Introduction is in the following. Here, apart from the explicit capacity constraint (9), the traffic congestion affects the transportation cost, too. The latter is exposed in the linear (not necessarily constant) marginal cost $\bar{c}_a(x) = c_a + 1/2\, d_a x_a^k$ for each good $k \in K$ transported along any arc $a \in A$, which clearly conduces to the quadratic cost terms appearing in (5) and (7). Therefore, the lower level program is quadratic but not linear as it was assumed in all previous papers [1] – [13] and [17] – [21]. However, making use of the well-known sensitivity analysis techniques available for quadratic programming problems, we develop the heuristic method described below in application to the TOP bilevel program (2) – (5), (7) – (10).

Step 1. Set $m = 0$ and $t_a^{(m)} = 0$, $a \in A_1$. Solve the lower level quadratic program (7) – (10), e.g., with the conjugate gradient method, thus finding an optimal response

$x^{(0)} = x\left(t^{(0)}\right)$. Whenever the optimal solution of the lower level problem is multiple, we accept the "optimistic" version, that is, such of them that maximizes the upper level objective function *F*. Go to Step 2.

Step 2. Evaluate the upper level objective function

$$F\left(x\left(t^{(m)}\right), t^{(m)}\right) = \sum_{k\in K}\sum_{a\in A_1} t_a^{(m)} x_a^k\left(t^{(m)}\right). \quad (11)$$

If $m \geq 1$ then match the upper level objective function value (11) with the same for the previous value of *m*, and when $F\left(t^{(m)}, x\left(t^{(m)}\right)\right) > F\left(t^{(m-1)}, x\left(t^{(m-1)}\right)\right)$ then go to Step 3. Otherwise, go to Step 5. If this jump from Step 1to Step 5 happens, say, 10 times in a row, go to Step 6.

Step 3. Having found the allowable ranges to stay optimal (ARSO) provided by the sensitivity analysis applied to the quadratic programming problem (7)–(10), select the maximum increase and decrease parameters $\Delta_a^{k,+}$ and $\Delta_a^{k,-}$, respectively, for the toll-arc basic variables x_a^k, $a \in A_1$. Define the subset of indices

$$A_1^+ = \left\{ a \in A_1 \left| \sum_{k\in K} x_a^k\left(t^{(m)}\right) > 0 \right.\right\}. \quad (12)$$

According to (6), the positive values shown in (12) are the component of the "formal gradient" vector of the leader's objective function. If $A_1^+ = \varnothing$ go to Step 5, otherwise go to Step 4.

Step 4. The toll variation procedure can be accomplished in four different modes. The first (more precautious) one makes an increase of the present toll level by the maximum allowable increment slots $\Delta_a^{k,+}$, $a \in A_1^+$, but not exceeding the corresponding component of the "formal gradient":

$$t_a^{(m+1)} = \begin{cases} min\left\{ t_a^{max}, t_a^{(m)} + \max_{k\in K} min\left\{ \sum_{s\in K} x_a^s\left(t^{(m)}\right), \Delta_a^{k,+} \right\} \right\}, & \text{if } a \in A_1^+; \\ max\left\{0, t_a^{(m)} - \min_{k\in K} \Delta_a^{k,-}\right\}, & \text{otherwise.} \end{cases}$$

The second mode of computing the toll increment is determined by the combination of the allowable increase values:

$$t_a^{(m+1)} = \begin{cases} min\left\{ t_a^{max}, t_a^{(m)} + \sum_{k\in K} \beta_k \, min\left\{ \sum_{s\in K} x_a^s\left(t^{(m)}\right), \Delta_a^{k,+} \right\} \right\}, & \text{if } a \in A_1^+; \\ max\left\{0, t_a^{(m)} - \min_{k\in K} \Delta_a^{k,-}\right\}, & \text{otherwise.} \end{cases}$$

Here, the nonnegative coefficients $\beta_k \geq 0, k \in K$, and such that $\sum_{k\in K} \beta_k = 1$ can be selected by the well-known 100 percent rule of sensitivity analysis. Next, if $t_a^{(m+1)} \neq t_a^{(m)}$ for at least one $a \in A_1^+$, then update $m := m+1$ and close the loop by returning to Step 1 to minimize the lower level aggregate objective function with the updated toll values. Otherwise, i.e., if no toll value has been changed, go to Step 5.

Remark. Modes 3 and 4 deal with the positive toll values assigned to the arcs with zero flows at the current solution. Then, on the contrary to modes 1 and 2, the present tolls are *decreased* by the values of $\Delta_a^{k,-}$ (one by one, like in mode 1, or in combination, similar to that of mode 2). This decrease may lead to the increase of the flow of transportation along those arcs (which are now empty).

Step 5. The present set of toll values $\left\{t_a^{(m)}\right\}_{a\in A_1}$ clearly provides a local maximum of the leader's objective function. In order to "jump" to some other local maximum solution, apply the "filled function" technique (*cf.*, [17] – [19]). Then come back to Step 1 and minimize the lower level aggregate objective function under the updated toll values.

Step 6. If after a number (say, 7 to 10) of times that Step 5 has been repeated, it seems to be impossible to increase the leader's objective function's value, hence stop the procedure, and report the current vectors $t = \left\{t_a^{(m)}\right\}_{a\in A_1}$ and $x = x\left(t^{(m)}\right)$ as an approximate (global) optimum solution.

The algorithm's flow chart can be depicted as follows.

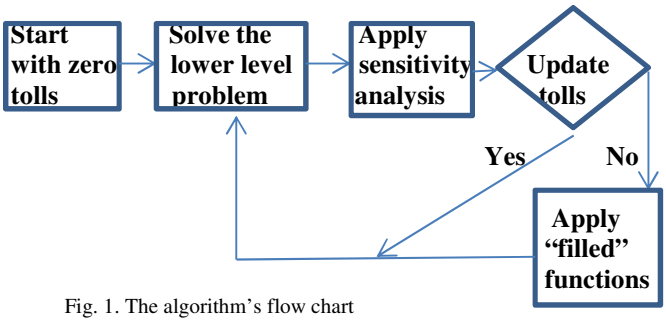

Fig. 1. The algorithm's flow chart

Figure 1 shows that first, the initial zero values assigned to the tolls. After solving the quadratic programming problem of the follower finding the flow in the arcs and yielding a value for the leader's objective function, apply sensitivity analysis taking into account only toll-arc variables. Having listed the possible increases and decreases in the coefficients of the objective function and based on the formal "gradient" vector of the upper level objective function *F* we update active toll vector $\left\{t_a^{(m)}\right\}_{a\in A_1}$. When positive increments of *t* cannot be obtained, perform the "filled function" method. Once there is a new toll vector, go to Step 1 and close the loop. The algorithm stops when the "filled function" method fails to provide a better value for the leader's objective function after several attempts in a row, which can mean that an approximate global optimum has been reached. The multi-commodity flow corresponding to the final toll values provides the optimal solution for the followers, too.

A. Application of the adapted "filled function" techniques

The above-described heuristic algorithm based upon sensitivity analysis (SA) includes sometimes an application of

the "filled function" technique first proposed in Renpu [17], adapted by to the maximization procedures (*see*, [20]). The original "filled function" method works under the assumption that a local minimum of a differentiable function $u = u(t)$ defined over a polyhedral set $T \subset R^n$, has been found. The aim is to find another (better than the current) local minimum of the function in question, or determine that this is an (approximate) global minimum of the function within the domain *T*. Since the pioneering work by Renpu [17] and the recent papers [18] and [19] defined a "filled function" for a *minimization* problem, we need to adapt several versions of the "filled function" definitions and properties to deal with a *maximization* problem. For simplicity purpose, we assume that any local maximum point of the objective function provides a positive value of the latter. Of course, the procedure is easily extended to the case where the value of the objective function is negative, too.

Filled functions can be defined in many ways. One of such possible modes is the following already described by the authors in [20].

Definition 1. Let $\overline{t_0} \in T$ satisfy $\overline{t_0} \neq t^*$ and $u(\overline{t_0}) \geq \frac{9}{5}u(t^*)$. A continuously differentiable function $Q_{t^*} = Q_{t^*}(t)$ is said to be a *"filled function"* for the maximization problem $max_{t \in T}\, u(t)$ at a point t^* with $u(t^*) > 0$, if

- t^* is a strict local minimizer of $Q_{t^*} = Q_{t^*}(t)$ on *T*;
- any local maximizer $\overline{t}$ of $Q_{t^*} = Q_{t^*}(t)$ on *T* satisfies $u(\overline{t}) > \frac{8}{5}u(t^*)$, or $\overline{t}$ is a vertex of *T*;
- any local maximizer $\hat{t}$ of the optimization problem $max_{t \in T}\, u(t)$ with $u(\hat{t}) \geq \frac{9}{5}u(t^*)$ is a local maximizer of $Q_{t^*} = Q_{t^*}(t)$ on *T*;
- any $\tilde{t} \in T$ with $\nabla Q_{t^*}(\tilde{t}) = 0$ implies $u(\tilde{t}) > \frac{8}{5}u(t^*)$.

Now, to construct a typical "filled function" in the sense of Definition 1, define two auxiliary functions as follows.

For arbitrary t and $t^* \in T$, denote $b = u(t^*) > 0$, and $v = u(t)$, then define

$$g_b(v) = \begin{cases} 1, & \text{if } v \geq \frac{4}{5}b; \\ 5 - \frac{30}{b}v + \frac{225}{4b^2}v^2 - \frac{125}{4b^3}v^3, & \text{if } \frac{2}{5}b \leq v \leq \frac{4}{5}b; \\ 0, & \text{if } v \leq \frac{2}{5}b, \end{cases} \quad (13)$$

and

$$s_b(v) = \begin{cases} v - \frac{2}{5}, & \text{if } v \leq \frac{2}{5}b; \\ 5 - \frac{8}{5}b + \left(8 - \frac{30}{b}\right)v - \frac{25}{2b}\left(1 - \frac{9}{2b}\right)v^2 + \frac{25}{4b^2}\left(1 - \frac{5}{b}\right)v^3, & \text{if } \frac{2}{5}b \leq v \leq \frac{4}{5}b; \\ 1, & \text{if } \frac{4}{5}b \leq v \leq \frac{8}{5}b; \\ 1217 - \frac{2160}{b}v + \frac{1275}{b^2}v^2 - \frac{250}{b^3}v^3, & \text{if } \frac{8}{5}b \leq v \leq \frac{9}{5}b; \\ 2, & \text{if } v \geq \frac{9}{5}b. \end{cases} \quad (14)$$

Now given a point $t^* \in T$ such that $u(t^*) > 0$ we define the following "filled function":

$$Q_{\rho,t^*}(t) := -exp\left(-\|t - t^*\|^2\right) g_{\frac{2}{5}u(t^*)}(u(t)) - \rho s_{\frac{2}{5}u(t^*)}(u(t)), \quad (15)$$

where $\rho > 0$ is a parameter. This "filled function" will be used in our algorithms.

First, based on [17] we have the following theorem:

Theorem 1. Assume that the function $u = u(t)$ is continuously differentiable and there exists a polyhedron $T \subset R^n$, and a point $t_0 \in T$ such that $u(t) \leq \frac{4}{5}u(t_0)$ for any $t \in R^n \setminus int\, T$. Let $\overline{t}_0, t^* \in T$ and such that $\overline{t}_0 \neq t^*$ satisfy the inequality $u(t^*) - u(\overline{t}_0) \leq \frac{2}{5}u(t^*)$. Then

- there exists a value $\rho^1_{t^*} \geq 0$ such that when $\rho > \rho^1_{t^*}$ any local maximizer $\overline{t}$ of the problem $\max_{t \in T} Q_{\rho,t^*}(t)$ obtained via the search starting from $\overline{t}_0$ satisfies $\overline{t} \in int\, T$;
- there exists a value $\rho^2_{t^*} > 0$ such that when $0 < \rho < \rho^2_{t^*}$ then for any stationary point $\tilde{t} \in T$ with $\tilde{t} \neq t^*$ of the function $Q_{\rho,t^*}(t)$, the following estimate holds: $u(\tilde{t}) > \frac{8}{5}u(t^*)$. ■

B. Auxiliary steps related to the "filled function" procedure

Now we come back to Step 5 of the algorithm developed above in Section III. We realize the "filled function" procedure making use of (15) when no toll values are allowed to change based on sensitivity analysis. The description of these auxiliary steps denoted as *FFSteps* mainly follows the ideas from [21].

FFStep 1. Assume that the present toll values $\left\{t_a^{(m)}\right\}_{a \in A_1}$ are such that update formulas of Step 4 of the main algorithm

permit no change. The latter may mean that the toll vector $t_0^* = t^{(m)}$ provides for a local maximum for the upper level objective function defined $\tilde{F}\left(t^{(m)}\right) := F\left(x\left(t^{(m)}\right), t^{(m)}\right)$ by (2) over the polyhedron described by inequalities (3). Go to *FFStep* 2.

FFStep 2. If it is possible select some previous iterate toll $t^{(i)}, 0 \le i \le m-1$, such that $\tilde{F}\left(t^{(i)}\right) > 0.6\tilde{F}\left(t^{(m)}\right)$ then the vectors $t^* := t^{(m)}$ and $\overline{t}_0 := t^{(i)}$ satisfy the conditions of Theorem 1. Our task is to find a *better* local maximizer for problem (2) subject to (3) – (5). To this end, find a (local) maximum point of the following auxiliary "filled function" problem:

$$\max_{t \in T} Q_{\rho, t^*}(t), \tag{16}$$

where the function $Q_{\rho, t^*}(t)$ is defined by (15), and the parameters $\rho_{t^*}^1 \ge 0$ and $\rho_{t^*}^2 > 0$ are also updated by the procedure described in [21]. Go to *FFStep* 3.

FFStep 3. If $\rho_{t^*}^1 < \rho_{t^*}^2$ and $\rho > 0$ is in between: $\rho_{t^*}^1 < \rho < \rho_{t^*}^2$ then the local maximizer of (16) provides a new initial point $t_1 := t^{(m+1)}$ to problem (2) subject to (3) – (5) with $u(t_1) \equiv \tilde{F}\left(t^{(m+1)}\right) > \frac{8}{5}\tilde{F}\left(t^{(m)}\right) = \frac{8}{5}u\left(t^*\right)$. Return to the main algorithm. Otherwise, go to *FFStep* 4.

FFStep 4. If $\rho \ge \rho_{t^*}^1$ then any local maximizer $t_1 := t^{(m+1)}$ of problem (16) satisfies either $u(t_1) > \frac{8}{5}u\left(t^*\right)$, or $u(t_1) \le \frac{4}{5}u\left(t^*\right)$ with $\nabla u(t_1) \ne 0$. If $u(t_1) > \frac{8}{5}u\left(t^*\right)$ we obtain a better local maximizer; otherwise, if $u(t_1) \le \frac{8}{5}u\left(t^*\right)$ then we adjust the parameter ρ (e.g., by dividing it by half) and solve problem (16) again.

FFStep 5. When $\rho \le \rho_{t^*}^2$ then any local maximizer $t_1 := t^{(m+1)}$ of problem (16) satisfies either $u(t_1) > \frac{8}{5}u\left(t^*\right)$ or $t_1 := t^{(m+1)}$ is a vertex of *T*. In the first case, we can obtain a better local maximizer of $u(t)$ on T; in the second outcome, we adjust the parameter ρ (e.g., by doubling it) and solve (16) again.

IV. NUMERICAL EXPERIMENTS

With the aim to test the algorithm, numerical experiments on three various graphs with five different instances each were conducted. To compare the efficiency and computational time of the proposed algorithm the numerical experiments reported as conducted in [8], [20], with four different algorithms, were repeated.

To apply the sensitivity analysis algorithm and the method of the "filled function" (FF), a PC Hewlett-Packard was used. The characteristics of the computer equipment used for the development and implementation of the algorithm are Intel (R) Atom (TM) CPU N455 with a speed 2.00 GHz of 1.67 GB RAM memory. The coding algorithm was written in the Matlab mathematical software in its version MatLab R2010a. This software was used due to its linear programming tools found in the "Optimization Toolbox". One of the functions used was *quadprog* because in the case of unbounded capacity, i.e., $q_a = +\infty$ for all $a \in A$ in constraints (9), the lower level of the TOP can be separated without loss of generality into independent minimal cost flow problems, which are quadratic programming problems.

The main parameters of the problems are the ones that define the size of the network; i.e., the number of nodes $|N|$, of arcs $|A|$, of toll-arcs$|A_1|$, and of commodities $|K|$. Each free arc and toll-arc has been assigned a fixed time-delay value c_a generated pseudo-randomly. The problems involved in this report are of minimum, small, and medium size, with two or three commodities:

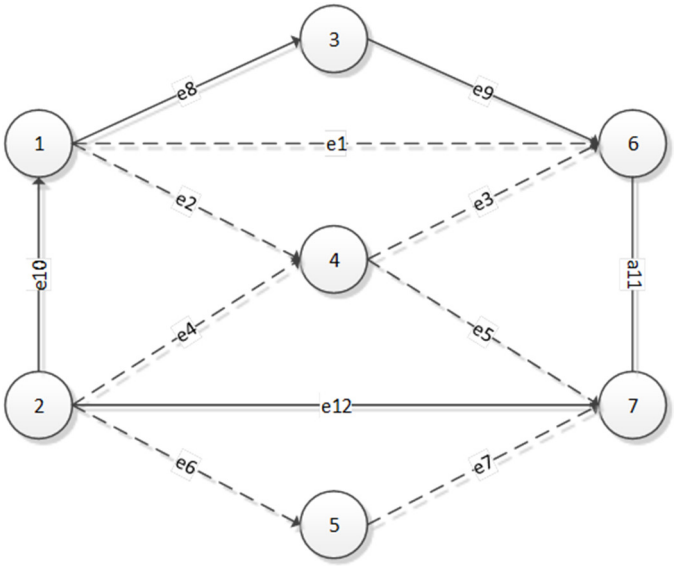

Fig. 2. Network 1 with 7 nodes, 12 arcs where 7 are toll arcs.

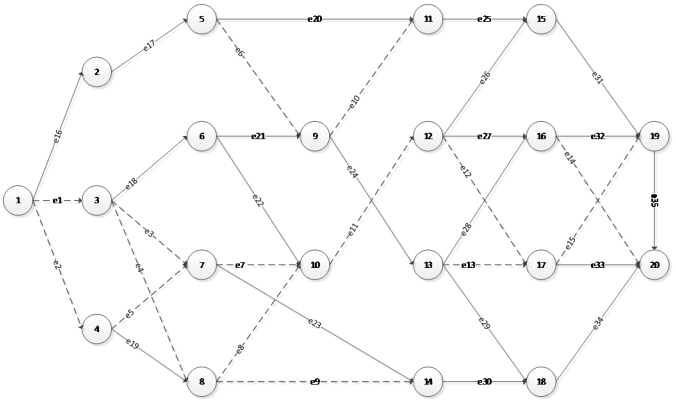

Fig. 3. Network 2 with 25 nodes, 40 arcs, where 20 are toll arcs.

The results for each example can be seen in the tables below. The first column (SA+FF) shows the leader's profit obtained by the algorithm when the increase in the tolls after sensitivity analysis is conducted by the first formula of Step 4. In the second column (SA +FF 100%), we put the data of the best solution obtained by our algorithms with the second toll update formula of Step 4 making use of the well-known 100% rule. The remaining four columns show the results obtained after having emulated the algorithms proposed in [8], which are Nelder-Mead (NM), Penalization (P), Quasi-Newton (QN), and Gradient (G) methods. The best result is marked in bold.

TABLE I. NUMBER OF ITERATIONS TO SOLVE TOP FOR NETWORK 1

N1	SA+FF	SA +FF 100%	NM	P	QN	G
1	273	290	377	389	**40**	370
2	322	309	469	384	**28**	497
3	248	277	375	387	**38**	529
4	185	124	264	218	**33**	436
5	270	299	305	269	**30**	308

TABLE II. NUMBER OF ITERATIONS TO SOLVE TOP FOR NETWORK 2

N2	SA+FF	SA +FF 100%	NM	P	QN	G
1	707	**577**	1312	987	591	945
2	734	636	1687	585	**596**	1012
3	822	594	1564	733	**574**	696
4	674	591	1486	649	**624**	993
5	666	599	1598	691	**509**	750

Tables I and II illustrate that the number of iterations the tested algorithms needed to reach the approximately optimal solutions to both test sample problems has the same order, with a single exception of the Nelder-Mead method. The latter is known to be quite slow, being a derivative-free algorithm, i.e., not using even the first derivatives of the upper level objective function.

One of the possible ways of measuring the algorithms efficiency is to compare, first, the number of iterations required for each algorithm to reach an approximate solution for a given tolerance value, and second, estimate the average computational cost (the number of iterations necessary on average) to decrease the error by one decimal order. This metric is calculated by the following formula:

$$Cost_{iter} := \frac{Number\ of\ \ iterations\ m}{log_{10}\,\varepsilon_0 - log_{10}\,\varepsilon_m},$$

where m denotes the number of iterations needed to reach the desired tolerance $\varepsilon_m > 0$ and ε_0 is the initial error computed as the difference between the initial leader's objective function value and the final one reached by the algorithm, that is, $\varepsilon_0 := \left|\tilde{F}\left(t^{(0)}\right) - \tilde{F}\left(t^{(m)}\right)\right|$. Tables I and II present the total number of iterations required for each algorithm, and Tables III and IV show the average cost of the number of iterations required to reduce the error by one decimal order.

TABLE III. AVERAGE NUMBER OF ITERATIONS TO REDUCE THE ERROR FOR TOP ON NETWORK 1

N1	SA+FF	SA +FF 100%	NM	P	QN	G
1	156	123	137	149	**6**	142
2	142	76	139	185	**50**	123
3	103	82	144	176	**19**	186
4	170	133	126	126	**41**	109
5	94	84	134	101	**39**	135

TABLE IV. AVERAGE NUMBER OF ITERATIONS TO REDUCE THE ERROR FOR TOP ON NETWORK 2

N2	SA+ FF	SA +FF 100%	NM	P	QN	G
1	419	407	1015	996	386	**339**
2	379	**374**	977	792	377	414
3	553	334	875	918	**287**	391
4	336	322	565	719	**261**	359
5	303	**315**	1044	793	460	396

Tables III and IV show that our sensitivity analysis-based algorithms are quite competitive against the other methods when the dimension of the test problem is rather high. Such robustness of the procedure may help when dealing with real-life problems, which are usually of higher dimensions.

In the next four tables, we also measured the number of values of the upper level objective function calculated during the performance of the algorithms, and the average computational cost (measured in the number of objective functions evaluations necessary to reduce the error by one decimal order). The evaluation formula used in Tables VII and VIII is:

$$Cost_{ObjFun} := \frac{\#_{ObjFun}}{log_{10}\,\varepsilon_0 - log_{10}\,\varepsilon_m},$$

where $\#_{ObjFun}$ is the number of the leader's objective function values calculated until the algorithm stops.

TABLE V. NUMBER OF OBJECTIVE FUNCTION EVALUATIONS TO SOLVE TOP ON NETWORK 1

N1	SA+FF	SA +FF 100%	NM	P	QN	G
1	2557	2169	4249	692	**327**	5506
2	2628	1814	4860	687	**431**	9347
3	1539	1508	4948	**690**	707	12713
4	2809	2517	3367	721	**598**	8993
5	1576	1388	3939	672	**584**	4142

TABLE VI. NUMBER OF OBJECTIVE FUNCTION EVALUATIONS TO SOLVE TOP ON NETWORK 2

N2	SA+FF	SA +FF 100%	NM	P	QN	G
1	5601	**4433**	24224	7257	5445	8311
2	5711	**4038**	21996	8037	7083	8752
3	6260	**4255**	10773	9779	6697	9937
4	5368	**4502**	9386	9534	5544	7227
5	5321	**4444**	9194	8736	6392	8781

Again, the proposed sensitivity analysis-based methods performed at a quite high level of efficiency compared to the best (quasi-Newton) algorithm but only for bigger problems (*cf.*, Table VI).

TABLE VII. AVERAGE NUMBER OF OBJECTIVE FUNCTION EVALUATIONS TO REDUCE THE ERROR FOR TOP ON NETWORK 1

N1	SA+FF	SA +FF 100%	NM	P	QN	G
1	508	316	619	441	**301**	713
2	404	**226**	648	376	450	896
3	251	**223**	721	368	622	821
4	438	496	467	**327**	923	791
5	215	**104**	574	494	925	809

TABLE VIII. AVERAGE NUMBER OF OBJECTIVE FUNCTION EVALUATIONS TO REDUCE THE ERROR FOR TOP ON NETWORK 2

N2	SA+FF	SA +FF 100%	NM	P	QN	G
1	1175	**944**	6969	2013	1723	15967
2	937	**615**	5949	1403	1787	19565
3	944	**674**	2820	1397	1685	16041
4	853	**656**	5228	2474	1641	21257
5	751	**664**	9441.	1151	1721	17110

According to the above Tables VII and VIII, with respect to the average cost in the number of values of the leader's objective function calculated to reduce the order of error by 1, our sensitivity analysis-based methods performed better at least the medium-size test problems, which is a promising feature.

The last measure we checked in order to compare the algorithms' performance is the computational time they needed to reach a good approximate solution. It is important to mention that we emulated the benchmark algorithms, so the required time is going to be valid because we have run all the experiments on the same computer. Tables IX and X present the time (in seconds) used for each instance and each network.

TABLE IX. RUNNING TIME (SEC) REQUIRED TO SOLVE TOP ON NETWORK 1

N1	SA +FF	SA +FF 100%	NM	P	QN	G
1	28.41	21.88	37.96	33.41	**3.28**	213.85
2	30.88	22.16	40.54	33.61	**6.79**	221.38
3	23.21	20.12	44.82	33.21	**9.95**	301.33
4	25.76	22.76	48.92	33.44	**7.38**	235.50
5	21.97	20.60	44.26	32.23	**5.68**	314.42

TABLE X. RUNNING TIME (SEC) REQUIRED TO SOLVE TOP ON NETWORK 2

N2	SA +FF	SA +FF 100%	NM	P	QN	G
1	429.67	**358.23**	639.90	387.69	576.61	747.85
2	558.98	467.50	588.63	416.33	**275.25**	717.73
3	580.22	**431.74**	742.40	678.38	652.56	685.57
4	323.65	268.22	520.63	345.25	**223.19**	841.82
5	722.86	**511.30**	671.22	665.36	668.99	617.34

The last two Tables IX and X again demonstrate that our algorithms cede the leading position only to the quasi-Newton method that has proved to be extremely fast when applied to the low-dimensional problems. However, for the higher-dimensional examples, the sensitivity-analysis-based procedure didn't lag behind, overwhelming all the other methods tested here.

V. CONCLUSIONS

The paper proposes and tests two versions of the heuristic algorithm to solve the Toll Optimization Problem (TOP) based upon sensitivity analysis for quadratic programming problems. The algorithm also makes use of the "filled" function technicalities in order to reach the global optimum when "jammed" at some local optimum.

Numerical experiments with a series of small and medium dimension test problems show the proposed algorithm's robustness and decent convergence characteristics.

In our future research, we are going to expand the above-described technique to the more complicated TOP problem with several objective functions.

REFERENCES

[1] T.L. Magnanti and R.T. Wong, "Network design and transportation planning: Models and algorithms," Transp. Sci., vol. 18, no. 1, pp. 1–55, 1984.

[2] P. Marcotte, "Network design problem with congestion effects: a case of bilevel programming," Math. Programming, vol. 34, no. 1, pp. 142–162, 1986.

[3] S. Dempe and T. Starostina, T., "Optimal toll charges: Fuzzy optimization approach," in Methods of Multicriteria Decision – Theory and Applications, F. Heyde, A. Lóhne, C. Tammer, Eds. Aachen: Shaker Verlag, 2009, pp. 29–45.

[4] S. Lohse and S. Dempe, "Best highway toll assigning models and an optimality test" (in German), Preprint, TU Bergakademie Freiberg, Nr. 2005-6, Fakultät für Mathematik und Informatik, Freiberg, 2005.

[5] M. Didi-Biha, P. Marcotte, and G. Savard, "Path-based formulation of a bilevel toll setting problem," in Optimization with Multi-Valued Mappings: Theory, Applications and Algorithms, S. Dempe and V.V. Kalashnikov, Eds. Boston: Springer Science, 2006, pp. 29–50.

[6] M. Labbé, P. Marcotte, and G. Savard, "On a class of bilevel programs," in Nonlinear Optimization and Related Topics, G. Di Pillo and F. Giannessi, Eds. Dordrecht: Kluwer Academic Publishers, 2000, pp. 183–206.

[7] L. Brotcorne, "Operational and strategic approaches to traffic routers' problems" (in French). Ph.D. dissertation, Université Libre de Bruxelles, 1998.

[8] V.V. Kalashnikov, N.I. Kalashnykova, and R.C. Herrera Maldonado, "Solving bilevel toll optimization problems by a direct algorithm using sensitivity analysis," in Proc. 2011 New Orleans Int. Acad. Conf., New Orleans, LA, March 21-23, 2011, pp. 1009 –1018.

[9] M. Labbé, P. Marcotte and G. Savard, "A bilevel model of taxation and its applications to optimal highway pricing," Manag. Sci., vol. 44, pp. 1608–1622, 1998.

[10] S. Roch, G. Savard and P. Marcotte, "Design and analysis of an algorithm for Stackelberg network pricing," Networks, vol. 46, no. 1, pp. 57–67, 2005.

[11] L. Brotcorne, F. Cirinei, P. Marcotte and G. Savard, "An exact algorithm for the network pricing problem," Discrete Optim., vol. 8, pp. 246–258, 2011.

[12] L. Brotcorne, F. Cirinei, P. Marcotte and G. Savard, "A tabu search algorithm for the network pricing problem," Comput. Oper. Res., vol. 39, pp. 2603–2611, 2012.

[13] S. Dempe and A.B. Zemkoho, "Bilevel road pricing; theoretical analysis and optimality conditions," Ann. Oper. Res., vol. 196, pp. 223–240, 2012.

[14] J.C.G. Boot, "On sensitivity analysis in convex quadratic programming problems," Oper. Res., vol. 11, pp. 771 –786, 1963.

[15] B. Jansen, Interior Point Techniques in Optimization. Complementarity, Sensitivity and Algorithms. Dordrecht, The Netherlands: Springer-Science+Business Media, B.V. 1997.

[16] A. G. Hadigheh, O. Romanko, and T. Terlaky, "Sensitivity analysis in convex quadratic optimization: Simultaneous perturbation of the objective and right-hand-side vectors," Algorithmic Oper. Res., vol. 2, pp. 94–111, 2007.

[17] G.E. Renpu, "A filled function method for finding a global minimizer of a function of several variables," Math. Programming, vol. 46, pp. 191–204, 1998.

[18] Z.Y. Wu, M. Mammadov, F.S. Bai and Y.J. Yang, "A filled function method for nonlinear equations," App. Math. Comput., vol. 189, pp.1196–1204, 2007.

[19] Z. Wan, L. Yuan and J. Chen, "A filled function method for nonlinear systems of equalities and inequalities," Comput. App. Math., vol. 31, no. 2, pp. 391–405, 2012.

[20] V.V. Kalashnikov, R.C. Herrera-Maldonado, J.-F. Camacho-Vallejo, and N.I. Kalashnykova "A heuristic algorithm solving bilevel toll optimization problems," Int. J. Logistics Manag., vol. 27, issue 1, pp. 31– 51, 2016.

[21] S. Dempe, V.V. Kalashnikov, G.A. Pérez-Valdés, and N.I. Kalashnykova, Bilevel Programming Problems. Theory, Algorithms, and Applications to Energy Networks. Heidelberg-New York-Dordrecht-London: Springer, 2015.