Computers & Industrial Engineering 107 (2017) 120–127

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Estimating the minimal revenue tolls in large-scale roadway networks using the dynamic penalty function method

Mohammadali Shirazi [a,*], Hedayat Z. Aashtiani [b], Luca Quadrifoglio [a]

[a] Zachry Department of Civil Engineering, Texas A&M University, College Station, TX 77843, United States
[b] Department of Civil Engineering, Sharif University of Technology, P.O. Box 11365-9313, Tehran, Iran

ARTICLE INFO

Article history:
Received 17 September 2016
Received in revised form 2 March 2017
Accepted 2 March 2017
Available online 4 March 2017

Keywords:
Transportation network
Toll pricing
Minimum toll revenue
Large-scale networks
Dynamic penalty function

ABSTRACT

Congestion toll pricing is an inexpensive management way to mitigate the traffic congestion and reduce the delay in the network. One of the models that were proposed for toll pricing is the minimum toll revenue (MinRev) problem. The objective of this model is to find link-tolls that simultaneously cause users to efficiently use the network and to minimize the total toll revenues to be collected. Although it can be written as a linear model, when applied to road networks in practice, this model is difficult to be solved optimally in a reasonable time, due to its large size. This paper proposes a method to approximately estimate the minimal revenue tolls in large-scale roadway networks. The method was implemented for four real network ranged from medium to large, and two large random networks. Implementation of this method indicated that this technique can find an approximate toll vector that is within 0.5% of the optimal solution after just a few seconds. Furthermore, this method allows to perform sensitivity or trade-off analysis between the total collected tolls, the number of tolled links and the desired network improvement, which could suggest implementing more practically efficient solutions with substantially fewer tolled links and even quicker solution time at a negligible additional network cost.

1. Introduction

Nowadays, network congestion has become a crucial problem in many populous cities. Constructing new roads not only needs a significant investment, but also may induce more demand. Therefore, transportation planners are looking for inexpensive management ways to mitigate the traffic congestion. As such, toll pricing has become a subject of interest in many populous cities as a method to reduce the traffic congestion. The toll pricing problem is originated from the Wardrop equilibrium principles (Wardrop, 1952) in traffic assignment: user equilibrium (UE) and system optimum (SO). Based on the UE principle, network users try to minimize their own travel time without considering the impact of their decisions on the network. Therefore, their decisions are not necessarily the best for the network, and the total travel time in user equilibrium is more than the one in system optimum. Toll pricing policies manage to impact the travel cost function in order to change the equilibrium and improve the network condition.

The minimum toll revenue (MinRev) problem is one of the models that were proposed for toll pricing (Bergendorff, Hearn, & Ramana, 1997; Hearn & Ramana, 1998). The focus of this paper is on the application of the MinRev problem in large-scale roadway networks; however, this model can be applied to other large-scale networks such as internet and telecommunication as well (Bai, Hearn, & Lawphongpanich, 2004). The objective of this model is to find link-tolls that simultaneously cause users to efficiently use the network and to minimize the total toll revenues to be collected. MinRev tolls are desirable due to their greater stability, simplicity, and perceived equity as well as their lower out-of-pocket costs (Penchina, 2009). Since its introduction, several researchers have tried to write different linear formulations for this model, such as: variational inequality (VI) formulation (Hearn, Yildirim, Ramana, & Bai, 2001), node-link formulation (Bai et al., 2004) and path-link formulation (Shirazi & Aashtiani, 2015). However, regardless of what formulation is used, when applied to road networks in practice (i.e.: applied to large-scale roadway networks), this model is challenging to solve optimally due to its large size. Therefore, several studies have tried to find an efficient method to solve this problem in large-scale networks. Below we review a few of them.

Dial (1999) proposed a fast algorithm to find the minimal tolls in single-origin networks. Next, in another paper, Dial (2000) generalized his method to find an approximate solution in multiple-origin networks. However, this paper did not provide

* Corresponding author.
E-mail addresses: alishirazi@tamu.edu (M. Shirazi), ashtiani@sharif.edu (H.Z. Aashtiani), lquadrifoglio@civil.tamu.edu (L. Quadrifoglio).

http://dx.doi.org/10.1016/j.cie.2017.03.008
0360-8352/

any convergence analysis or large-scale implementations to evaluate the performance of the method in real networks. Later, Penchina (2004) illustrated that the single-origin algorithm that was proposed by Dial (1999) can be modified to be used in single-destination networks as well.

Hearn et al. (2001) used the variational inequality notation of the MinRev model and a cutting plane method to propose an algorithm to solve the MinRev problem optimally. This method divides the MinRev problem into a master problem and a subproblem. The master problem is a linear program (LP) that in each iteration, based on the subproblem results, some constraints are added to the model. The subproblem of the method includes solving an all-or-nothing assignment for each origin-destination (OD) pair of the network. Implementation of this method to large-scale networks can face numerical difficulties (Bai et al., 2004).

Bai et al. (2004) used the node-link notation of the MinRev model, and the Dantzig-Wolfe (DW) decomposition method to solve the problem in large-scale networks. Similar to the cutting plane algorithm, this algorithm also divides the MinRev problem into two problems, a master problem and a subproblem. However, in this algorithm, based on subproblem results, some variables are generated and added to the master problem. The subproblem of this algorithm includes solving a minimum-cost-network-flow problem for each origin of the network. It was reported that the DW decomposition method is an effective method to employ when it is applied to large-scale networks, when the MinRev model cannot be solved directly using commercial software, such as CPLEX, or too much time is required.

Shirazi and Aashtiani (2015) used the path-link notation of the MinRev model, and a path generation (PG) method to propose an algorithm for solving the problem optimally in real and large-scale networks. The PG method has an important advantage over the link-based methods. In fact, its convergence is significantly facilitated by using the path information derived from solving the SO problem, as initial paths for the PG method. Implementation of the PG method to several large-scale networks indicated that this algorithm has a better performance over the DW decomposition method and can find the optimal solution of the MinRev problem after a few iterations and an appropriate CPU time. Therefore, the PG algorithm is used in this paper as benchmark to evaluate the performance of our newly proposed method.

In the following sections, we initially define the MinRev model and provide a brief review of the PG method. Next, our proposed methodology to estimate the MinRev tolls is described in two sub-sections. First, a model that would guarantee the system optimum flows in the network is presented; then, the dynamic penalty function (DPF) method (Shahpar, Aashtiani, & Babazadeh, 2008) is used to solve this model iteratively and estimate a valid toll vector (Note: a toll vector that guarantees the SO flows in the network is referred to as a valid toll vector). This toll vector is then compared against the optimal MinRev solution for several real and random networks. Results show that this toll vector is a good estimation for the MinRev tolls. Lastly, the proposed method is used as a tool to perform sensitivity analysis to evaluate the effect of the number of tolled links in the estimated MinRev solution on the network improvement.

2. MinRev model and path-generation method

As noted in Section 1, so far, several notations have been used to formulate the MinRev problem, such as: VI formulation (Hearn et al., 2001), node-link formulation (Bai et al., 2004) and path-link (or path-based) formulation (Shirazi & Aashtiani, 2015). Given that our proposed method to estimate the minimal revenue tolls is also a path-based method, in this section we describe the path-based version (Shirazi & Aashtiani, 2015) of the original MinRev model proposed by Hearn and Ramana (1998).

It is assumed that the OD demand is fixed, network users have the same value of time, and every road in the network is tollable. Let the index 'r' and index 's', respectively, denote the origin and the destination of the OD pair (r, s). In addition, Let K_{rs} and A, respectively, denote the non-empty set of paths between the OD pair (r, s) and the set of all links in the network. The path-link complementarity formulation of the MinRev problem is given as follows (Shirazi & Aashtiani, 2015):

$$min\ z = \sum_{a\in A}\beta_a\bar{x}_a \tag{1a}$$

Subject to:

$$\left(\sum_{a\in A}\delta_{ak}^{rs}(t_a(\bar{x}_a)+\beta_a)-u_{rs}\right)f_k^{rs}=0 \quad \forall k\in K_{rs}, rs \tag{1b}$$

$$\sum_{a\in A}\delta_{ak}^{rs}(t_a(\bar{x}_a)+\beta_a)-u_{rs}\geqslant 0 \quad \forall k\in K_{rs}, rs \tag{1c}$$

$$\sum_{k\in K_{rs}}f_k^{rs}=q_{rs} \quad \forall rs \tag{1d}$$

$$\bar{x}_a=\sum_{rs}\sum_{k\in K_{rs}}\delta_{ak}^{rs}f_k^{rs} \quad \forall a \tag{1e}$$

$$f_k^{rs}\geqslant 0 \quad \forall k\in K_{rs}, rs \tag{1f}$$

$$\beta_a\geqslant 0 \quad \forall a \tag{1g}$$

where

q_{rs} = demand between the OD pair (r, s).
$\bar{x}_a$ = system optimum traffic flow on link 'a'.
$t_a(\bar{x}_a)$ = system optimum travel time on link 'a'.
β_a = toll[1] on link 'a'.
f_k^{rs} = traffic flow on path 'k' between the OD pair (r, s).
u_{rs} = least generalized cost[2] between the OD pair (r, s).
δ_{ak}^{rs} = indicator that is equal to 1 if link a lies on path $k \in K_{rs}$ and 0 otherwise.

Model (1) minimizes the total toll revenue, and at the same time ensures the system optimum flow in the network and satisfies the equilibrium conditions. With some manipulations in constraints (see Shirazi & Aashtiani, 2015), Model (1) can be written as the following linear model (MinRev-LP), Model (2), that is a path-based version of the original link-based model proposed by Hearn and Ramana (1998).

$$min\ z = \sum_{a\in A}\beta_a\bar{x}_a \tag{2a}$$

Subject to:

$$\sum_{a\in A}(t_a(\bar{x}_a)+\beta_a)\bar{x}_a=\sum_{rs}q_{rs}u_{rs} \tag{2b}$$

$$\sum_{a\in A}\delta_{ak}^{rs}(t_a(\bar{x}_a)+\beta_a)-u_{rs}\geqslant 0 \quad \forall k\in K_{rs}, rs \tag{2c}$$

$$\beta_a\geqslant 0 \quad \forall a \tag{2d}$$

In order to solve Model (2), directly, one is required to generate every possible path that is between each OD pair of the network

[1] Note: The toll component (β_a) is in time unit.
[2] Note: The generalized cost of a path involves the costs associated with the combination of the travel time and toll.

and include them in the model. However, since the number of paths exponentially increases by increasing the network dimensions, it becomes apparent that it is not practical to count and include all of them in the model, when the MinRev model is applied to large-scale networks. In order to overcome this difficulty, the PG algorithm (Shirazi & Aashtiani, 2015) solves the model iteratively. Beginning with the set of SO positive-flow paths, in each iteration, a restricted version of Model (2) is solved with a set of generated paths until constraint (2c) is satisfied for all paths in the network with an ε-accuracy.

3. Dynamic penalty function method

This section documents the proposed methodology. This section is divided into two parts. First, a model that would guarantee the system optimum flows in the network is presented. It is shown that when this model is reconstructed with Kuhn–Tucker conditions, the generalized cost function of network links would involve toll components that ensure the SO flows in the network. In the second subsection, the DPF method is used to solve this model and estimate the tolls (a valid toll vector). Numerical results show that with a very good precision, the estimated toll vector is an approximate solution for the MinRev problem.

3.1. A model to find a valid toll vector

As it was illustrated by Bergendorff et al. (1997), infinite toll vectors (or to be exact, valid toll vectors) can be found to satisfy constraints (1b)–(1g) of Model (1) and ensure the SO flows in the network. In this section, a model is presented to find one of these toll vectors.

If the SO flows ($\bar{x}_a$) were considered as an upper bound to link flows (x_a) in the Beckmann UE model (Beckmann, McGuire, & Winsten, 1956), the following model would be resulted:

$$\min \sum_{a\in A}\int_0^{x_a} t_a(w)dw \tag{3a}$$

Subject to:

$$\sum_{k\in K_{rs}} f_k^{rs} = q_{rs} \quad \forall rs \tag{3b}$$

$$x_a = \sum_{rs}\sum_{k\in K_{rs}} \delta_{ak}^{rs} f_k^{rs} \quad \forall a \tag{3c}$$

$$f_k^{rs} \geqslant 0 \quad \forall k \in K_{rs}, rs \tag{3d}$$

$$\frac{x_a}{\bar{x}_a} \leqslant 1 \quad \forall a \tag{3e}$$

Let t_a denote the travel time function. Assuming that t_a is positive, continuous and strictly increasing, Model (3) becomes a convex optimization model. Let u_{rs} and v_a denote the lagrangian multipliers of Eqs. (3b) and (3e), respectively. The optimal solution of Model (3) can be characterized by Kuhn–Tucker conditions as follows:

$$\left(\sum_{a\in A}\delta_{ak}^{rs}\left(t_a(x_a)+\frac{v_a}{\bar{x}_a}\right)-u_{rs}\right)f_k^{rs}=0 \quad \forall k\in K_{rs}, rs \tag{4a}$$

$$\sum_{a\in A}\delta_{ak}^{rs}\left(t_a(x_a)+\frac{v_a}{\bar{x}_a}\right)-u_{rs}\geqslant 0 \quad \forall k\in K_{rs}, rs \tag{4b}$$

$$\sum_{k\in K_{rs}} f_k^{rs} = q_{rs} \quad \forall rs \tag{4c}$$

$$x_a = \sum_{rs}\sum_{k\in K_{rs}} \delta_{ak}^{rs} f_k^{rs} \quad \forall a \tag{4d}$$

$$f_k^{rs} \geqslant 0 \quad \forall k \in K_{rs}, rs \tag{4e}$$

$$\left(1-\frac{x_a}{\bar{x}_a}\right)v_a = 0 \quad \forall a \tag{4f}$$

$$\frac{x_a}{\bar{x}_a} \leqslant 1 \quad \forall a \tag{4g}$$

$$v_a \geqslant 0 \quad \forall a \tag{4h}$$

Given the SO principle and its definition, it becomes apparent that at the optimal solution, constraint (4g) will stop at its upper bound ($x_a = \bar{x}_a$). In other words, Model (4) ensures the SO flows in the network. Besides, if we let $\beta_a = v_a/\bar{x}_a$ where $\beta_a \geqslant 0$, it would become apparent that Eqs. (4a)–(4e) are identical to Eqs. (1b)–(1f). Therefore, If Model (4) is solved and the toll vector (β) is found, this toll vector will be a valid toll vector to reach the SO flows in the network. In the next section, a method is proposed to solve Model (4).

3.2. Solving the model using the DPF method

The Dynamic Penalty Function (DPF) method has originally been introduced by Shahpar et al. (2008) to solve traffic assignment problems with side constraints. We use this methodology to solve Model (4) and estimate a valid toll vector.

Let $P_a(x_a, \lambda_a)$, with $\lambda_a \geqslant 0$, be a differentiable, monotone, and non-negative function. If this function has the following three properties, it can replace the lagrangian multiplier (v_a) (Larsson & Patriksson, 1999; Shahpar et al., 2008) in Model (4).

1. If $x_a/\bar{x}_a = 1$, $P_a(x_a, \lambda_a)$ equals λ_a.
2. If $x_a/\bar{x}_a > 1$, $P_a(x_a, \lambda_a)$ is sufficiently large.
3. If $x_a/\bar{x}_a < 1$, $P_a(x_a, \lambda_a)$ is sufficiently small.

The dynamic penalty function (DPF) (Shahpar et al., 2008) is a function that satisfies all these conditions. Eqs. (5) and (6) show the adopted DPF equation for Model (4) from the one originally used in Shahpar et al. (2008):

$$P_a(x_a, \lambda_a) = \lambda_a \psi\left(\frac{x_a}{\bar{x}_a}\right) \quad \forall a \tag{5}$$

where for each $\rho \in (0,1)$, $\psi(y)$ is defined as follows:

$$\psi(y) = \begin{cases} \frac{\rho}{2(1-y)} & y < 1-\rho \\ \frac{y-1}{2\rho}+1 & y \geqslant 1-\rho \end{cases} \tag{6}$$

The function ψ is differentiable, monotone, and non-negative (Shahpar et al., 2008). Besides, if $\rho \to 0$, this function will have all characteristics required to substitute the lagrangian multiplier v_a. For each link 'a', the toll function $\beta_a(x_a, \lambda_a)$ is defined as follows:

$$\beta_a(x_a, \lambda_a) = \frac{\lambda_a \psi\left(\frac{x_a}{\bar{x}_a}\right)}{\bar{x}_a} \quad \forall a \tag{7}$$

If $\beta_a(x_a, \lambda_a)$ is replaced in Model (4) and Eqs. (4f) and (4g) are considered implicitly, Model (8) will be derived as:

$$\left(\sum_{a\in A}\delta_{ak}^{rs}(t_a(x_a)+\beta_a(x_a,\lambda_a))-u_{rs}\right)f_k^{rs}=0 \quad \forall k\in K_{rs}, rs \tag{8a}$$

$$\sum_{a\in A}\delta_{ak}^{rs}(t_a(x_a)+\beta_a(x_a,\lambda_a))-u_{rs}\geqslant 0 \quad \forall k\in K_{rs}, rs \tag{8b}$$

$$\sum_{k\in K_{rs}} f_k^{rs} = q_{rs} \quad \forall rs \tag{8c}$$

$$x_a = \sum_{rs}\sum_{k\in K_{rs}} \delta_{ak}^{rs} f_k^{rs} \quad \forall a \tag{8d}$$

$$f_k^{rs} \geqslant 0 \quad \forall k \in K_{rs}, rs \tag{8e}$$

If λ is fixed, then, Model (8) would become a traffic assignment problem with generalized cost function, which can be solved by Aashtiani's complementarity algorithm (Aashtiani, 1979; Aashtiani & Magnanti, 1982; Shahpar et al., 2008). Let us define the flow on link 'a' as:

$$x_a(f) = \sum_{rs}\sum_{k\in K_{rs}} \delta_{ak}^{rs} f_k^{rs}$$

and the generalized cost function on path 'k' as:

$$c_k^{rs}(f,\lambda) = \sum_{a\in A} (t_a(x_a(f)) + \beta_a(x_a(f),\lambda_a))\delta_{ak}^{rs}$$

Therefore, the complementarity problem of Model (8) would be derived as follows:

$$(c_k^{rs}(f,\lambda) - u_{rs}) f_k^{rs} = 0 \quad \forall k \in K_{rs}, rs \tag{9a}$$

$$c_k^{rs}(f,\lambda) - u_{rs} \geqslant 0 \quad \forall k \in K_{rs}, rs \tag{9b}$$

$$f_k^{rs} \geqslant 0 \quad \forall k \in K_{rs}, rs \tag{9c}$$

$$\left(\sum_{k\in K_{rs}} f_k^{rs} - q_{rs}\right) u_{rs} = 0 \quad \forall rs \tag{9d}$$

$$\sum_{k\in K_{rs}} f_k^{rs} - q_{rs} \geqslant 0 \quad \forall rs \tag{9e}$$

$$u_{rs} \geqslant 0 \quad \forall rs \tag{9f}$$

Model (9) is a nonlinear complementarity problem [NCP(λ)] and is too large to be solved directly, when applied to large-scale networks. However, Aashtiani (1979) and Aashtiani and Magnanti (1982) proposed an iterative decomposition and linearization method to solve the problem iteratively.

In each iteration of the decomposition and linearization algorithm, NCP(λ) is decomposed by each OD pair into subproblems denoted by NCP$_{rs}$(λ). In these subproblems, all components of 'f' are fixed at the current solution except those of $f^{rs} = (f_k)_{k\in K_{rs}}$. Since the number of paths between each OD pair (r,s) is too large to solve the NCP$_{rs}$(λ) subproblems directly, Aashtiani and Magnanti (1982) introduced a method to keep only paths that have a positive flow. These set of paths are referred to as the set of working paths and are denoted by the variable K_{rs}^w for each given OD pair (r,s). The working paths are gradually updated in each iteration of the algorithm by eliminating those with zero flow and adding the new shortest path. Let $\bar{u}_{rs}$ be the cost of the shortest-path for the OD pair (r,s). The shortest path will be added to K_{rs}^w if the following condition is satisfied. Parameter ε_c specifies the required accuracy.

$$\frac{\min_{k\in K_{rs}^w} c_k^{rs}(f,\lambda) - \bar{u}_{rs}}{\bar{u}_{rs}} \geqslant \varepsilon_c \tag{10}$$

Although the number of working paths is small, still it is not efficient to use the general non-linear complementarity algorithms to solve the model. In order to overcome this difficulty, Aashtiani (1979) proposed an iterative linearization method. This iterative linearization method linearizes the model at the current flow ($\bar{f}$) to reduce the original model to a linear complementarity problem (LCP). The linearization algorithm continues until following stop condition is satisfied:

$$\frac{\max_{k\in K_{rs}^w f_k>0} c_k^{rs}(f,\lambda) - u_{rs}}{\max_{k\in K_{rs}^w f_k>0} c_k^{rs}(f,\lambda)} \leqslant \varepsilon_c \tag{11}$$

Until now, all discussions were under the assumption that λ is known and fixed; however, λ is unknown. In order to overcome this difficulty, NCP(λ) can be solved iteratively by updating λ in each iteration. Let λ_a^n denote the value of λ_a in iteration 'n' of the algorithm, in the next iteration, λ_a^{n+1} is updated as follows:

$$\lambda_a^{n+1} = \lambda_a^n \psi\left(\frac{x_a^n}{\bar{x}_a}\right) \quad \forall a \tag{12}$$

In order to solve the traffic assignment problems, the iterative decomposition and linearization algorithm continues until it reaches the equilibrium in the network. However, in order to solve Model (8), this algorithm must be continued until both equilibrium and implicit constraint (4g) are satisfied. Let ε_1 and ε_2 be two small positive numbers. Moreover, let Err^{eq} denote the relative-gap error. The algorithm stops when Eqs. (13) and (14) are satisfied simultaneously.

$$Err^{eq} < \varepsilon_1 \tag{13}$$

$$\max\left(\frac{x_a}{\bar{x}_a}\right) < (1+\varepsilon_2) \tag{14}$$

It is worth pointing out that having the SO flows as an upper bound to link flows at initial iterations of this iterative algorithm may cause a sharp change in DPF and adversely affect the algorithm convergence. Instead, in order to achieve a better convergence, the upper bound can be relaxed a little bit in initial iterations; then, in next iterations, when equilibrium is more established, upper bounds can get closer to SO flows. For this purpose, at the iteration 'n' of the algorithm, for each link 'a', C_a^n is calculated as shown in Eq. (15) and is used instead of $\bar{x}_a$ in DPF.

$$C_a^n = \bar{x}_a \times \left(1 + \frac{\gamma\times\rho}{n}\right) \quad \forall a \tag{15}$$

In Eq. (15), the value of $\gamma\times\rho$ represents the relaxation ratio at the first iteration of the algorithm. Note that such relaxation is also useful to avoid numerical difficulties when the value of $\bar{x}_a$ is equal to zero. Taking Eq. (15) into account, Eqs. (7) and (12) are modified as follows:

$$\beta_a(x_a,\lambda_a) = \frac{\lambda_a \psi\left(\frac{x_a}{C_a^n}\right)}{\bar{x}_a} \quad \forall a \tag{16}$$

$$\lambda_a^{n+1} = \lambda_a^n \psi\left(\frac{x_a^n}{C_a^n}\right) \quad \forall a \tag{17}$$

It is also worth pointing out that, similar to the PG method, in order to facilitate the convergence of the algorithm, it is recommended to initialize the set of the working paths with the SO positive-flow paths. The method described above to find a valid toll vector is referred to as the DPF method in this study. In the next section, this toll vector is compared with the MinRev optimal solution. It is shown that the total toll of this toll vector is very close to the MinRev solution. The detailed steps of the DPF method to find a valid toll vector are as follows:

Step 1 - Initialization

1.1. Select appropriate values for ρ, γ, ε_1, ε_2, ε_c and vector λ^1.
1.2. Let $n = 1$.

1.3. For each OD pair (r,s), set the SO positive-flow paths in K^w_{rs}. Then, let $x^1 = \bar{x}$, find a feasible flow for each path 'k' in K^w_{rs}, and set its value in f^{rs}_k.
1.4. Update the travel time vector (t^n), upper bound vector (C^n) in Eq. (15), and the toll vector (β^n) in Eq. (16).

Step 2 - linearization and decomposition (Aashtiani & Magnanti, 1982; Shahpar et al., 2008)

2.1. For each origin 'r':
2.1.1. Find the shortest path from origin 'r' to all destinations 's' based on $(t^n + \beta^n)$. Let k_{rs} be the shortest path and $\bar{u}_{rs}$ denote its cost.
2.1.2. For each destination 's':
2.1.2.1. If Eq. (10) is satisfied, set $K^w_{rs} = K^w_{rs} \cup \{k_{rs}\}$ and $f_{k_{rs}} = 0$.
2.1.2.2. Set $\bar{f}_k = f_k$ for $k \in K_{rs}$.
2.1.2.3. Linearize $NCP_{rs}(\lambda^n)$ at $\bar{f}$ and solve the linearized problem to obtain (f^{rs}, u_{rs}).
2.1.2.4. If Eq. (11) is not satisfied, go to step (2.1.2.2).

Step 3 - updating

3.1. Update the link flow vector (x^{n+1}), travel time vector (t^{n+1}), upper bound vector (C^{n+1}) in Eq. (15), and the toll vector (β^{n+1}) in Eq. (16).
3.2. Update λ^{n+1} in Eq. (17).
3.3. Find the relative-gap error.

Step 4 - stopping test

If Eqs. (13) and (14) are satisfied simultaneously, stop the algorithm and introduce β^{n+1} as a valid toll vector. Otherwise, Let $n = n + 1$ and go to step 2.

4. Numerical results

This section is divided into two sub-sections. First, the results from the application of the DPF method to real large-scale networks is discussed and compared to the optimal solution. Then, in the second sub-section, the DPF method is used as a tool to perform sensitivity analysis on the number of tolled links in the MinRev solution, and investigate the tradeoff between the total collected tolls, the number of tolled links and the desired network improvement.

4.1. Application of the dynamic penalty function method to large-scale networks

The Sioux Falls, Mashhad, Winnipeg and Chicago Sketch networks are used to evaluate the performance of the DPF method for real networks. The information of these networks is shown in Table 1. For each network, the total travel time in UE and SO problems were also calculated and shown in the table. The performance of the DPF method is compared with the performance of the PG method (Shirazi & Aashtiani, 2015) and the MinRev optimal solution. The PG method was used for comparison since not only it performs better than other methods such as DW decomposition when applied to large-scale networks (Shirazi & Aashtiani, 2015), but also it has a similar structure as the DPF method (i.e.: both are path-based methods).

Both algorithms (PG and DPF) were written in VISUAL C++. They both were implemented on a PC with 2.66 GHz CPU and 4 GB of RAM. For the PG method, the dual simplex routine of CPLEX (version 12) was used to solve the LP problems, and the Bellman method (Golden, 1976) to compute the shortest paths. The equilibrium accuracy error was set to 0.0001 ($\varepsilon = 0.0001$). For the DPF method, the Lemke algorithm (Lemke, 1965) was used to solve the LCP problems, and the Bellman method to compute the shortest paths. The parameters and initial values were set to the following values:

$\rho = 0.01$
$\gamma = 3.0$ (Upper bounds were relaxed by 3% at the first iteration)
$\lambda^1_a = \bar{x}_a t(\bar{x}_a)\ \forall a$
$\varepsilon_1 = 0.0001$
$\varepsilon_2 = 0.001$ (for Sioux Falls) and $\varepsilon_2 = 0.01$ (other networks)

Table 2 indicates the results of applying the PG and DPF methods to real networks. As shown in the table, even though the total toll is not minimized directly when using the DPF method, this method can have a very good estimation for the MinRev solution, after just a few CPU seconds. The difference between the value of the total toll in the DPF solution and the one in the optimal solution (PG solution) is less than 0.5% for the networks we analyzed.[3] In addition, the value of the total travel time in the DPF solution is just slightly more than the system optimum. Results show that as the network size increases, the DPF method becomes a more efficient option to employ. For instance, for the Chicago Sketch network, this method can find an approximate solution after just 33.81 CPU seconds, that is about 100 times faster than the PG method that finds the optimal solution. Furthermore, the DPF method is more memory efficient. Recall that the DPF method only keeps the paths with positive flows at a time, and discards those with no (or zero) flow; hence, it needs less memory to preserve the path information. As shown in the table, the number of paths at the final iteration of the DPF method is substantially less than the one in the PG method. Overall, the results show that the DPF method is much faster and more memory efficient method than the PG method, especially as the network size increases, and can be used as a heuristic method to estimate the MinRev tolls in large scale networks.

As discussed in the previous section, two stopping conditions should be satisfied to terminate the DPF method: (1) the relative gap error (Eq. (13)) and (2) reaching the SO flows (Eq. (14)). Figs. 1 and 2, respectively, show the relative gap error (Eq. (13)) and the number of links that did not satisfy the condition that is described in Eq. (14), at each iteration of the DPF method. Our investigation on these stopping conditions indicated that the later condition is much stronger than the former. Hence, the DPF method usually stops when the later condition is satisfied (i.e.: the relative-gap error is usually satisfied in earlier iterations of the method). For instance, for the Winnipeg network, the relative gap error is satisfied after 42 iterations, but the algorithm still continues with another 39 iterations to make sure that the flow on all links falls below the required accuracy. For having a more efficient implementation, one may decide to stop the algorithm at iteration 42, when the relative gap error is satisfied, and ignore the reaming four links that are still above the required ε_2-accuracy of SO flows. Even in this case, the estimated total toll revenue will be equal to 157,034 units of time that is still within the 1% of the optimal total toll, and a reliable option to be used instead of the optimal solution. Therefore, as the number of links in the network increases, it is more efficient to stop the algorithm when a reasonable number of links satisfy the condition described in Eq. (14), and ignore the condition for the remaining links. This 'reasonable' number of links can be a parameter discretionarily set by the analyst.

[3] Recall that constraint (4g) was slightly relaxed by ε_2-accuracy when the DPF method is used (see Eq. (14)). Hence, it is expected that the DPF method could find better solutions than the MinRev optimal.

Table 1
Network information.

Network	Nodes	Links	Zones	OD pairs	UE travel time	SO travel time
Sioux Falls	24	76	24	528	74.80	71.94
Mashhad	917	2526	163	7157	1,417,048	1,386,486
Winnipeg	1067	2836	147	4345	925,828	890,048
Chicago Sketch	933	2950	387	93135	18,935,450	18,518,578

Table 2
Comparing the DPF and PG methods for real networks.

Network	Sioux Falls		Mashhad		Winnipeg		Chicago Sketch	
Method	PG	DPF	PG	DPF	PG	DPF	PG	DPF
Number of iterations	2	45	10	49	14	81	7	128
Total toll	20.67	20.59	185617	185596	155652	155031	2794856	2784579
Difference compared to the MinRev optimal total toll	Optimal	0.39%	Optimal	0.01%	Optimal	0.40%	Optimal	0.37%
Total travel time	71.94	71.95	1386484	1386524	890048	890063	18518578	18518764
Number of final paths	727	723	16348	10400	17934	6552	176991	114971
CPU time (s)	0.14	0.03	12.12	2.59	38.45	7.12	3288.49	33.81

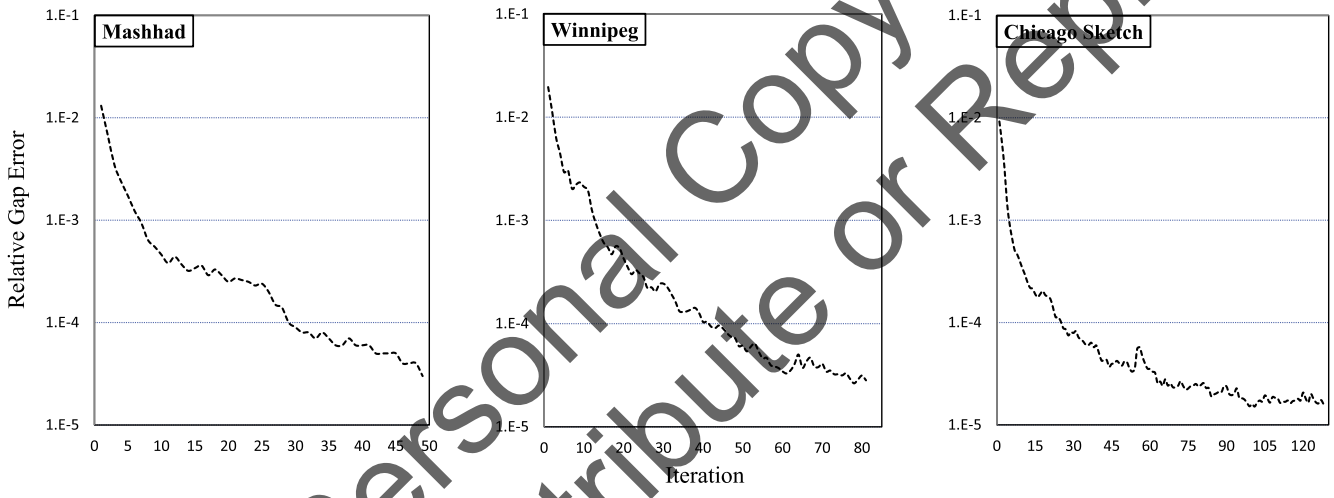

Fig. 1. Relative-gap error at each iteration of the DPF method.

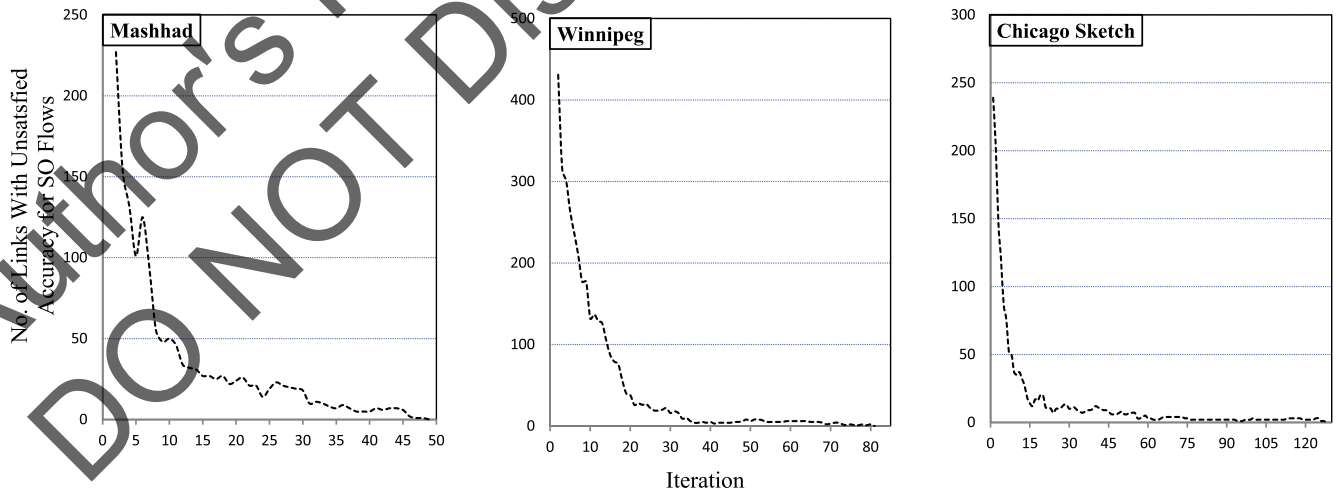

Fig. 2. The number of links that have a traffic flow that is above the ε_2-accuracy of the SO solution, at each iteration of the DPF method.

Two large-scale random networks (Nie, 2010) with a lot of links were used to (1) further investigate the observation described in the previous paragraph and (2) evaluate the efficiency of the method on networks with numerous links. The first one is a 40×40 square network and has 1600 nodes, 200 zones, 6240 links and 39,800 OD pairs. The second is a 50×50 square network and

has 2500 nodes, 312 zones, 9800 links and 97,032 OD pairs. In this implementation, the DPF method was set to stop when more than 99% of the links satisfy the condition described in Eq. (14). Table 3 shows the results of the implementations. For both networks, the DPF method estimates the solution with a very good precision, even though the second stopping condition was ignored for the 1% of the links; hence, it is reasonable to ignore Eq. (14) for a small portion of links (say 1%) when we have a network with numerous links. For the 40×40 network, the DPF method finds an approximate solution for the MinRev problem after 9.6 CPU seconds that is more than 50 times faster than the PG method. The difference between the estimated total toll and the optimal one is less than 0.5%. For the 50×50 network, the DPF method estimates an approximate toll vector after 53.4 s that is more than 100 times faster than the PG method. The difference between the estimated total toll and the optimal one is less than 0.1% for this network. Recall that the DPF method is more memory efficient too. This can be verified by looking at the number of final paths in Table 3.

4.2. Sensitivity analysis on the number of tolled links in MinRev solution

In Section 4.1, we showed that the DPF method is a fast and memory efficient method to estimate an approximate solution for the MinRev model. However, there is one more advantage in this method. Recall that constraint (3e) in Model (3) was considered for all links in the network. However, this may not be the most efficient way for the network, as not all tolled links impact the network the same. In order to have more efficient implementations, we can consider the SO flows on some selected links, run the DPF method again and find a toll vector. This would result in having fewer tolled-links with SO flows. In this case, even though the network will not reach the SO solution, it can still be improved to a good level between the UE and SO solutions. The main justification for this approach would be practical rather than theoretical and would suggest to focus on finding an approximate toll vector within a small percentage of the optimal solution much more easily and quickly.

In this section, we use this advantage to perform a sensitivity analysis to analyze the trade-off between the total collected tolls, the number of tolled links, and the desired network improvement. The analysis is performed by taking the following two steps. First, given the approximate (or optimal) MinRev tolls, the network links are ranked from the ones with the largest to the lowest toll. Then, the DPF method is run again. However, this time, the SO flow is considered only for those links with the highest toll values (for example, only for the top 20% of the links that have the highest tolls in MinRev solution).

The Mashhad and Winnipeg networks, respectively, have 516 and 615 tolled-links in MinRev optimal solution (PG solution). In approximate solution (DPF solution), all links will have tolls, although the value of the toll for most of them is small or negligible. Therefore, it is worth adopting the approach described above to find an efficient way to decrease the number of tolled links and associated costs. Figs. 3 and 4 show the sensitivity analysis on MinRev tolled links for the Mashhad and Winnipeg networks, respectively. As it is indicated in Fig. 3, the Mashhad network can be improved by 2.16% if a minimal total toll that is equal to 185,617 units of time is collected from 516 tolled links. However, it is still possible to reduce the number of tolled links to 250 (cut the number in half) and improve the network by 1.92%. In this case, the users will be charged for a total toll of 144,997 units of time. For the Winnipeg network, the analyst can collect a minimal total toll of 155,652 units of time from 615 tolled links to reach the SO solution and improve the network by 3.86%. However, Fig. 4 shows that one can decide to adopt a more efficient approach and improve the network by 3.56% (only 0.3% less than the optimal solution) by collecting 147,932 units of time from only 300 tolled links.

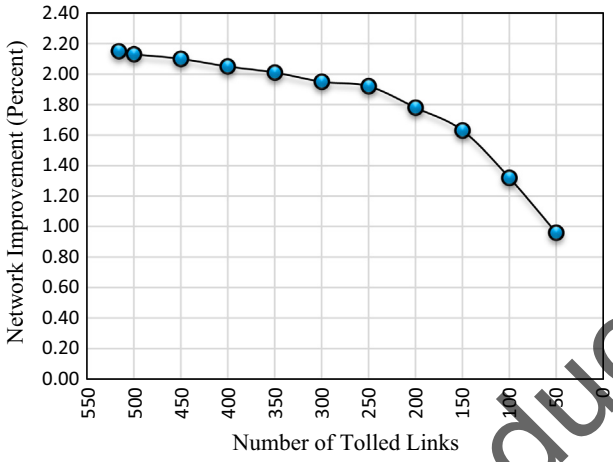

Fig. 3. Sensitivity analysis for the Mashhad network.

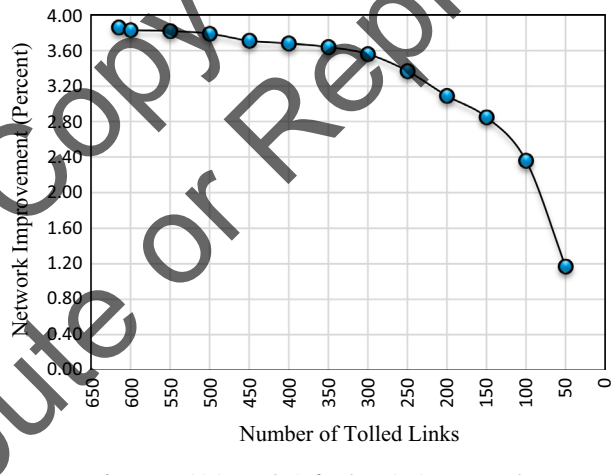

Fig. 4. Sensitivity analysis for the Winnipeg network.

Table 3
Comparing the results of the DPF and PG algorithms for random networks.

Network	Random 40×40		Random 50×50	
Method	PG	DPF	PG	DPF
Number of iterations	7	31	10	62
Total tolls	10817	10765	20462	20442
Difference compared to the MinRev optimal total toll	Optimal	0.48%	Optimal	0.10%
Total travel time	35263	35265	52342	52343
Number of final paths	105468	55269	295101	129220
CPU time (s)	486.4	9.6	6109.2	53.4

Using the analysis described in this section, planners can decide on different tolling strategies to implement these tolls. Recall that tolls are in time units. They can be implemented either by charging equivalent monetary values or by any means that result in changing the travel time on the roadway, such as alternating the signal timing or implementing variable speed limits. This analysis can also be useful in many instances where geography, terrain and other constraints might in fact limit the potential adoption of tolls on certain links.

5. Summary and conclusion

This paper proposed a method to estimate (or approximate) the minimal revenue tolls in large-scale roadway networks. First, a model was presented to find a toll vector that guarantees the system optimum in the network. Next, the dynamic penalty function algorithm was used to solve the model iteratively. The proposed method was applied to four real and two random networks. Numerical results show that this method can find an approximate solution that is within 0.5% of the optimal solution after just a few seconds. Hence, this method can be used as a heuristic method to estimate the MinRev tolls. Not only it is fast, but this method is also memory efficient. In addition, using this approximate method, one can consider the system optimum flows as an upper bound to selected number of links in the network and improve the network to a level between the UE and SO solutions. Hence, the analyst can make his decision based on a trade-off between the total collected tolls, the number of tolled links and the desired network improvement. The sensitivity analysis shows that while reaching the system optimum solution with minimal revenue tolls may require implementing tolls on many network links, one can adopt a solution still very close to the system optimum but with much fewer tolled links and smaller total toll. Further research can be done to compare the results of this analysis with the Minimum Toll Booth problem (the minimum tolled-links required to have the SO solution in the network) (Bai, Hearn, & Lawphongpanich, 2010; Hearn & Ramana, 1998).

References

Aashtiani, H. Z. (1979). *The multi-modal traffic assignment problem* (Ph.D. Dissertation). Cambridge, MA: MIT.

Aashtiani, H. Z., & Magnanti, T. L. (1982). A linearization and decomposition algorithm for computing urban traffic equilibria. In *Proc. 1982 IEEE large scale systems symposium* (pp. 8–19).

Bai, L., Hearn, D. W., & Lawphongpanich, S. (2004). Decomposition techniques for the minimum toll revenue problem. *Networks, 44*(2), 142–150.

Bai, L., Hearn, D. W., & Lawphongpanich, S. (2010). A heuristic method for the minimum toll booth problem. *Journal of Global Optimization, 48*(4), 533–548.

Beckmann, M., McGuire, C. B., & Winsten, C. B. (1956). *Studies in the economics of transportation*. New Haven, Connecticut: Yale University Press.

Bergendorff, P., Hearn, D. W., & Ramana, M. V. (1997). Congestion toll pricing of traffic networks. In P. M. Pardalos, D. W. Hearn, & W. W. Hager (Eds.), *Network optimization. Lecture notes in economics and mathematical systems* (Vol. 450, pp. 51–71). Springer-Verlag.

Dial, R. (1999). Minimal-revenue congestion pricing part I: A fast algorithm for the single-origin case. *Transportation Research Part B, 33*, 189–202.

Dial, R. (2000). Minimal-revenue congestion pricing part II: An efficient algorithm for the general case. *Transportation Research Part B, 34*, 645–665.

Golden, B. (1976). Shortest-path algorithms: A comparison. *Operations Research, 24*, 1164–1168.

Hearn, D. W., & Ramana, M. V. (1998). Solving congestion toll pricing models. In P. Marcotte & S. Nguyen (Eds.), *Equilibrium and advanced transportation modeling* (pp. 109–124). Kluwer Academic Publishers.

Hearn, D. W., Yildirim, M. B., Ramana, M. V., & Bai, L. (2001). Computational methods for congestion toll pricing models. *IEEE Intelligent Transportation Systems Proceedings*, 257–262.

Larsson, T., & Patriksson, M. (1999). Side constrained traffic equilibrium models-analysis, computation and applications. *Transportation Research Part B, 33*(4), 233–264.

Lemke, C. E. (1965). Bimatrix equilibrium points and mathematical programming. *Management Science, 11*, 681–689.

Nie, Y. (2010). A class of bush-based algorithms for the traffic assignment problem. *Transportation Research Part B, 44*, 73–89.

Penchina, M. (2004). Minimal-revenue congestion pricing: Some more good-news and bad-news. *Transportation Research Part B, 38*, 559–570.

Penchina, M. (2009). Minimal revenue tolls: Price stability for networks with BPR cost function. *The Open Transportation Journal, 3*, 87–92.

Shahpari, A. H., Aashtiani, H. Z., & Babazadeh, A. (2008). Dynamic penalty function method for the side constrained traffic assignment problem. *Applied Mathematics and Computation, 206*(1), 332–345.

Shirazi, M., & Aashtiani, H. Z. (2015). Solving the minimum toll revenue problem in real transportation networks. *Optimization Letters, 9*(6), 1187–1197.

Wardrop, J. G. (1952). Some theoretical aspects of road traffic research. *Proceedings of the Institution of Civil Engineers*, 325–378.