Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
Mathematical and Computer Modelling 43 (2006) 493–505
MATHEMATICAL AND COMPUTER MODELLING
www.elsevier.com/locate/mcm

# A general MPCC model and its solution algorithm for continuous network design problem

Jeff X. Ban[a,*], Henry X. Liu[b,1], Michael C. Ferris[c,2], Bin Ran[d,3]

[a] *California Center for Innovative Transportation (CCIT), University of California, 2105 Bancroft Way, Suite 300, Berkeley, CA 94720-3830, United States*
[b] *Department of Civil Engineering, University of Minnesota, Minneapolis, United States*
[c] *Computer Sciences Department, University of Wisconsin, Madison, United States*
[d] *Department of Civil and Environmental Engineering, University of Wisconsin, Madison, United States*

Received 18 April 2005; received in revised form 2 November 2005; accepted 7 November 2005

**Abstract**

This paper formulates the continuous network design problem as a mathematical program with complementarity constraints (MPCC), with the upper level a nonlinear programming problem and the lower level a nonlinear complementarity problem. Unlike in most previous studies, the proposed framework is more general, in which both symmetric and asymmetric user equilibria can be captured. By applying the complementarity slackness condition of the lower-level problem, the original bilevel formulation can be converted into a single-level and smooth nonlinear programming problem. In order to solve the problem, a relaxation scheme is applied by progressively restricting the complementarity condition, which has been proven to be a rigorous approach under certain conditions. The model and solution algorithm are tested for well-known network design problems and promising results are shown.


*Keywords:* Continuous network design problem (CNDP); Mathematical programs with equilibrium constraints (MPEC); Mathematical program with complementarity constraints (MPCC); Traffic assignment; User equilibrium

## 1. Introduction and motivation

To improve the performance of existing transportation networks and thus reduce traffic congestions, the continuous network design problem (CNDP) has been introduced and studied for almost three decades [1]. CNDP aims to determine the optimal capacity enhancement for a set of selected links in a given network by minimizing the total system cost as well as considering the route choice behavior of individual users. Due to multiple objectives for formulating CNDP, it is natural to model it as a bilevel programming problem with the upper level a nonlinear

* Corresponding author. Tel.: +1 510 642 5112; fax: +1 510 642 0910.
*E-mail addresses:* xban@calccit.org (J.X. Ban), henryliu@umn.edu (H.X. Liu), ferris@cs.wisc.edu (M.C. Ferris), bran@engr.wisc.edu (B. Ran).
[1] Tel.: +1 612 625 6347; fax: +1 612 626 7750.
[2] Tel.: +1 608 262 4281; fax: +1 608 262 9777.
[3] Tel.: +1 608 262 0052; fax: +1 608 262 5199.


doi:10.1016/j.mcm.2005.11.001

programming problem to minimize the system cost and the lower level a user equilibrium (UE) problem to account for drivers' route choice behavior. In the mathematical programming literature, the bilevel programming problem is also frequently referred to as a mathematical program with equilibrium constraints (MPEC), which has been extensively studied [2]. However, due to the non-convex and non-smooth characteristics of an MPEC, solving such a problem is normally difficult. In the transportation area, CNDP was first proposed by Morlok et al. [3] and subsequently studied by many researchers [4–8]. Most of the early works on CNDP were focusing on heuristic approaches for solving the bilevel model. For more detailed reviews on CNDP prior to 2001, we refer to discussions in [9,10].

More recent approaches for modeling CNDP focus on reformulating the problem using a certain form of smooth gap function for the lower-level UE problem. By exploring the special structure of CNDP, Meng et al. [10] converted the bilevel problem to a single-level yet smooth one through introducing a particular gap function for the lower-level UE problem which is formulated as a nonlinear programming problem (NLP). Although still a non-convex model, the resulting single-level problem can be solved using existing solution algorithms for the NLP. Nevertheless, Meng's model was based on the symmetry assumption on the lower-level problem, i.e., there is no interaction among flows on different links. A general UE problem cannot be formulated as an NLP; instead, a nonlinear complementarity problem (NCP) or variational inequality (VI) formulation needs to be adopted. Marcotte [11] investigated such a general bilevel model, i.e., an NLP for the upper level and a VI for the lower level. By defining certain gap functions, the bilevel problem was transferred to a single-level one and solved using the penalty method. More recently, Patriksson and Rockafellar [12] presented a new reformulation technique to convert an MPEC into a constrained and locally Lipschitz minimization problem which can be further solved using a decent algorithm proposed in the same paper. However, both Marcotte [11] and Patriksson and Rockafellar [12] did not further test their models using well-known CNDP examples in transportation field.

In this paper, by formulating the general UE (both symmetric and asymmetric) as a link-node based NCP, we develop a general CNDP model as a mathematical program with complementarity constraints (MPCC). As a special case of an MPEC, MPCC has been extensively studied recently [13–19], since a variety of methods can be applied to convert an MPCC to a single-level NLP and then solve the NLP using existing solution techniques. In this paper, we adopt a popularly applied method, called the relaxation approach, to solve the proposed MPCC model. This approach relaxes the strict complementarity condition by a relaxation parameter. Then this parameter is progressively reduced, with the resulting relaxed NLP solved using existing NLP solvers. Ralph and Wright [20] further proved that under certain conditions the relaxation scheme can guarantee to solve the original MPCC successfully. This makes the relaxation scheme a rigorous solution approach for CNDP. Numerical examples show that such a relaxation scheme is effective and efficient for solving CNDP, at least for tested small-scale problems.

This paper is organized as follows. An MPCC based CNDP model is presented in Section 2, based upon the link-node NCP formulation. In Section 3, the solution approach for the proposed MPCC model is discussed, including the conversion of the bilevel formulation to a single-level NLP and an iterative algorithm based on the relaxation scheme. Section 4 provides numerical examples showing the effectiveness of the proposed model and algorithm. Finally, concluding remarks and future study directions are given in Section 5.

## 2. An MPCC model for CNDP

### 2.1. A link-node NCP formulation for asymmetric user equilibrium

Following Wardrop's first Principle [21], various models have been proposed for the static UE problem [22]. In particular, we are focusing on the so-called asymmetric UE (AUE), a more general case of UE in which the link interactions are considered. In particular, Ban [23] has shown that a link-node based NCP formulation exists for the AUE problem, defined on the disaggregated link flows.

Assume a given transportation network can be represented as $G(N, A)$, where $N$ is the set of nodes and $A$ is the set of links. We use index $i$, $j$ to denote nodes in $N$ and $(i, j)$ or $ij$ to denote a link in $A$. Denote $R$ as the origin node set which is a subset of $N$ and generates origin–destination (OD) trips. Similarly, set $S$ is defined as the destination set which is also a subset of $N$ and absorbs OD trips. Further denote $\pi_i^s$ the minimum travel cost from node $i$ to destination $s$, $d_i^s$ the trip demand from node $i$ to destination $s$, $v_{ij}^s$ the (disaggregated) flow for link $(i, j)$ with respect to destination $s$, $x_{ij} = \sum_{s \in S} v_{ij}^s$ the total (aggregated) flow for link $(i, j)$, and $t_{ij}$ the link travel cost for link $(i, j)$.

Then the AUE can be mathematically formulated as [23]:

$$\begin{cases} 0 \le \left[\pi_j^s + t_{ij}\left(\sum_{s\in S} v_{ij}^s\right) - \pi_i^s\right] \perp v_{ij}^s \ge 0, & \forall (i,j)\in A, s\in S \\ 0 \le \left[\sum_{(i,j)\in A} v_{ij}^s - \sum_{(k,i)\in A} v_{ki}^s - d_i^s\right] \perp \pi_i^s \ge 0, & \forall i \in N, i\ne s, s\in S, \end{cases} \tag{1}$$

where the symbol "⊥" is the "perp" operator such that, $x \perp y \Leftrightarrow x^{\mathrm{T}} y = 0$. Denote vectors $\pi^s = (\pi_i^s)_{i\in N, i\ne s}$, $v^s = (v_{ij}^s)_{(i,j)\in A}$, $d^s = (d_i^s)_{i\in N, i\ne s}$ for any given destination node $s \in S$, and $t = (t_{ij})_{(i,j)\in A}$. Also notice that the standard node-link incidence matrix can be represented as $\Lambda$. Then Eq. (1) can be rewritten in a matrix form as:

$$\begin{cases} 0 \le \left[-\Lambda_s^{\mathrm{T}}\pi^s + t\left(\sum_{s\in S} v^s\right)\right] \perp v^s \ge 0 \\ 0 \le [\Lambda_s v^s - d^s] \perp \pi^s \ge 0, \end{cases} \quad \forall s \in S, \tag{2}$$

where $\Lambda_s$ denotes $\Lambda$ with the row corresponding to destination $s$ removed which guarantees that $\Lambda_s$ is full row rank. Eq. (2) is the link-node NCP formulation for the AUE which will be utilized later for modeling the bilevel CNDP. Further investigations on NCP (2) can reveal that it has amenable properties under mild assumptions. Firstly, the link travel time $t$ is a function of the aggregated link flow $x = \sum_{s\in S} v^s$ and we further impose the following assumptions on $t$.

**Assumption 1.** The link travel time $t$ has the following properties:

(a) $t$ is a smooth function of the aggregated link flow $x = \sum_{s\in S} v^s$, i.e., $t = t(\sum_{s\in S} v^s)$.
(b) $t$ is strictly monotone in terms of $x$, i.e., $(x_1 - x_2)^{\mathrm{T}}[t(x_1) - t(x_2)] > 0, \forall x_1 \ne x_2$.
(c) $t(x) > 0, \forall x \ge 0$.
(d) $t(x) < \infty$ if $x < \infty$.
(e) $t$ is a coercive function in terms of $v = (v^s)_{s\in S}$, i.e., $\lim_{v\to\infty} \frac{v^{\mathrm{T}} t(v)}{\|v\|} = \infty$.

Then we can prove that the NCP (2) has at least one solution and the total link flow is unique, as shown in Theorem 1. Note that the conditions in Assumption 1 are mild and easy to be satisfied. For example, the widely used BPR (Bureau of Public Roads) function for symmetric UE satisfies all these five conditions.

**Theorem 1.** *The following statements hold for the NCP model* (2) *under* Assumption 1.

(a) *The model has at least one solution if the problem is feasible.*
(b) *The model has a unique solution in terms of the aggregated link flow vector $x$.*

For detailed proofs of Theorem 1 and other related properties of NCP (2), we refer to [23]. We can also easily observe that (2) has a very special structure such that it can be naturally decomposed according to individual destinations. The only place in which interactions exist for variables related to different destinations is the link travel cost vector $t$ since $t$ is defined on the aggregated link flows. This special structure has important impacts on how to design a solution algorithm for both the UE problem itself and the CNDP problem constructed based upon (2). Ban et al. [24] developed a decomposition method for solving CNDP by exploring such a special structure in (2).

## *2.2. MPCC formulation for the CNDP*

In this section, we present the MPCC model for CNDP. Firstly, additional notations are listed as follows.

$y_{ij}$ = the capacity enhancement for link $(i, j) \in A$

$y$ = the vector of $y_{ij}$, $y \in R^{|A|}$

$t_{ij}(x, y_{ij})$ = the travel cost on link $(i, j) \in A$, defined as a function of the aggregated link flow $x$ and the capacity enhancement of $(i, j)$, i.e., $y_{ij}$

$g_{ij}(y_{ij})$ = the cost function of capacity enhancement for link $(i, j) \in A$

$g$ = the vector of $g_{ij}(y_{ij})$, $g \in R^{|A|}$

$\theta$ = the relative weight of total capacity enhancement cost and total travel cost in the system design objective function

$l_{ij}, u_{ij}$ = the lower bound and upper bound for the capacity enhancement for link $(i, j) \in A$

$l, u$ = the vector of $l_{ij}$ and $u_{ij}$, respectively, $l, u \in R^{|A|}$.

With these notations in place, the goal of CNDP is to minimize the total system travel cost and the cost for enhancement, while the driver's route choice behavior (the user equilibrium in this case) must be respected [1]. It is well-known that such a problem can be formulated as a bilevel formulation; and particularly in this paper we can formulate a general CNDP model which captures both UE and AUE as the following MPCC:

$$\min_{y,v^1,\ldots v^s,v^{|S|},\pi^1,\ldots\pi^s,\pi^{|S|}} \sum_{(i,j)\in A} \left[ t_{ij}\left(\sum_{s\in S} v^s, y_{ij}\right) \cdot \sum_{s\in S} v^s_{ij} \right] + \theta \sum_{(i,j)\in A} g_{ij}(y_{ij}), \tag{3a}$$

subject to

$$l_{ij} \le y_{ij} \le u_{ij}, \quad \forall (i, j) \in A, \tag{3b}$$

where $(v^s, \pi^s), \forall s \in S$ is the solution to the following NCP problem:

$$\begin{aligned} &0 \le \left[\pi^s_j + t_{ij}\left(\sum_{s\in S} v^s, y_{ij}\right) - \pi^s_i\right] \perp v^s_{ij} \ge 0, \quad \forall (i, j) \in A, s \in S, \\ &0 \le \left[\sum_{(i,j)\in A} v^s_{ij} - \sum_{(k,i)\in A} v^s_{ki} - d^s_i\right] \perp \pi^s_i \ge 0, \quad \forall i \in N, i \ne s, s \in S. \end{aligned} \tag{3c}$$

Obviously, the MPCC based CNDP model (3) is defined on $y_{ij}, \forall (i, j) \in A$ and $(v^s, \pi^s), \forall s \in S$. Eq. (3a) is the upper-level objective of the MPCC model which tries to minimize a weighted summation of the total system travel cost and the enhancement cost, (3b) is bound constraints for the upper-level decision variable $y_{ij}, \forall (i, j) \in A$, and (3c) is the lower-level AUE formulation that $(v^s, \pi^s), \forall s \in S$ must satisfy. Using matrix notations, (3) can be rewritten as:

$$\min_{y,v^1,\ldots v^s,v^{|S|},\pi^1,\ldots\pi^s,\pi^{|S|}} \left[ t\left(\sum_{s\in S} v^s, y\right)\right]^{\mathrm{T}} \left(\sum_{s\in S} v^s\right) + \theta e^{\mathrm{T}} g(y), \tag{4a}$$

subject to

$$l \le y \le u, \tag{4b}$$

where $e$ is the vector of all 1's and $\{(v^s, \pi^s), \forall s \in S\}$ is a solution to the following NCP model:

$$\begin{cases} 0 \le \left[-\Lambda^{\mathrm{T}}_s \pi^s + t\left(\sum_{s\in S} v^s, y\right)\right] \perp v^s \ge 0 \\ 0 \le [\Lambda_s v^s - d^s] \perp \pi^s \ge 0, \end{cases} \quad \forall s \in S. \tag{4c}$$

Similarly to [20], the MPCC model (4) can be tackled by converting it to a single-level equivalence and then solving the latter using a relaxation scheme. Under certain conditions, such a relaxation scheme can guarantee to generate an optimal solution of (4). The detailed discussions of this relaxation scheme will be provided in the next section. Notice here that (4) is a general CNDP model since the constraint (4c) can represent both symmetric and asymmetric user equilibria.

## 3. Solution algorithm

### 3.1. Single-level NLP formulation for the CNDP

Because the NCP formulation (4c) can be readily replaced by its equivalent complementarity slackness condition and additional nonnegativity constraints, the MPCC model of CNDP (4) can be straightforwardly converted into a single-level NLP as follows.

$$\min_{y,v^1,\ldots v^s,v^{|S|},\pi^1,\ldots\pi^s,\pi^{|S|}} \left[t\left(\sum_{s\in S} v^s, y\right)\right]^{\mathrm{T}} \left(\sum_{s\in S} v^s\right) + \theta e^{\mathrm{T}} g(y), \tag{5a}$$

subject to

$$l \le y \le u, \tag{5b}$$

$$-\Lambda_s^{\mathrm{T}}\pi^s + t\left(\sum_{s\in S} v^s, y\right) \ge 0, \quad \forall s \in S, \tag{5c}$$

$$\Lambda_s v^s - d^s \ge 0, \quad \forall s \in S, \tag{5d}$$

$$v^s \ge 0, \quad \forall s \in S, \tag{5e}$$

$$\pi^s \ge 0, \quad \forall s \in S, \tag{5f}$$

$$(\Lambda_s v^s - d^s)_i \pi_i^s = 0, \quad \forall s \in S, i \in N, i \ne s, \tag{5g}$$

$$\left[-\Lambda_s^{\mathrm{T}}\pi^s + t\left(\sum_{s\in S} v^s, y\right)\right]_{ij} v_{ij}^s = 0, \quad \forall s \in S, (i, j) \in A, \tag{5h}$$

where the lower-level NCP formulation in (4c) is replaced by its equivalent complementarity slackness conditions in (5c)–(5h). Evidently, under the assumption that both the link travel cost function $t$ and function $g$ are smooth, the single-level NLP model (5) involves only smooth functions with respect to $(y, v, v^1, \ldots v^s, v^{|S|}, \pi^1, \ldots \pi^s, \pi^{|S|})$. Hence, it is a smooth and nonlinear optimization problem. However, this NLP formulation lacks sound mathematical properties because of the complementarity slackness constraints (5g) and (5h). Actually, because of these two constraints, the single-level model is non-convex and most importantly, the Mangasarian Fromovitz Constraint Qualification (MFCQ) does not hold [2]. Hence, solving the single-level NLP directly is usually difficult and a relaxation algorithm will be adopted instead in the next section.

Note that the NLP equivalence (5) clearly involves the disaggregated variables explicitly and hence has a large dimension for large scale problems. However, as aforementioned, all the constraints (5c)–(5h) are defined according to individual destinations, except for the interaction of disaggregated link flows on the link travel cost function $t$. This feature makes it possible to employ certain decomposition techniques in order to solve the single-level NLP model efficiently [23]. It is worth noting that many other reformulation techniques are also available to convert the MPCC model (4) to a single-level NLP, as investigated in [27]. The one we adopted in (5) turns out to have the best performance.

### 3.2. A relaxation scheme

In order to solve the non-convex single-level model (5), a relaxation scheme is proposed in this section to iteratively tackle this NLP. The main idea of the algorithm is to introduce an auxiliary parameter $\mu^s > 0, \forall s \in S$, which can be used to define the relaxed complementarity slackness conditions for each destination $s \in S$, rather than the exact ones as shown in (5g) and (5h). In other words, in each iteration, (5g) and (5h) are replaced by the following two conditions, respectively,

$$(\Lambda_s v^s - d^s)_i \pi_i^s \le \mu^s, \quad \forall s \in S, i \in N, i \ne s, \tag{6a}$$

$$\left[-\Lambda_s^{\mathrm{T}}\pi^s + t\left(\sum_{s\in S} v^s, y\right)\right]_{ij} v_{ij}^s \le \mu^s, \quad \forall s \in S, (i, j) \in A. \tag{6b}$$

Then the relaxed NLP problem (5a)–(5f) and (6a) and (6b) can be solved repeatedly by constantly reducing the value of $\mu^s > 0, \forall s \in S$ by some predefined factor. It is clear that although this relaxed NLP is still non-convex, the MFCQ holds [2] which implies that existing NLP solution algorithms can be adopted to solve the relaxed NLP. The iterative algorithm can be illustrated as follows.

*Step 1* *Initialization*

Choose an initial auxiliary parameter $\mu^{s0} > 0$ for each destination $s \in S$. Set the iteration limit $M$, update factor $0 < \lambda < 1$, and $k = 0$.

*Step 2* *Major Iteration*

*Step 2.1* Solve the current relaxed single-level NLP (5a)–(5f) and (6a), (6b). Use $\mu^{sk}$ as the auxiliary parameter for each $s \in S$ in (6a) and (6b).

*Step 2.2* Update and Move. If $k \leq M$, set $\mu^{s(k+1)} = \lambda\mu^{sk}, \forall s \in S, k = k + 1$ and go to Step 2.1; otherwise, go to Step 3.

*Step 3* *Final Solve*

Solve the exact single-level NLP (5a)–(5h). If it is successful, we obtain an exact solution for the CNDP problem; otherwise, an approximate solution is achieved from the last run of Step 2.2.

For a given iteration limit $M$, in total $M + 2$ runs ($M + 1$ from Step 2 and the last one from the final solve in Step 3) will be performed by the algorithm. The algorithm requires three parameters as input, namely, the initial value of the auxiliary parameter $\mu^{s0}$ (if we set $\mu^{s0}$ the same for any $s \in S$), the update factor $\lambda$, and the number of solves $M$. For proper choices of these parameters which guarantee $\mu^{sM}, \forall s \in S$ is sufficiently small, we can obtain a point which is very close to the true solution of the single-level NLP (5) after Step 2. Consequently, in Step 3, we can successfully solve the exact single-level NLP for most of the cases. Note that in Step 2.1, solving each relaxed single-level NLP is roughly equivalent to solving a UE problem. Since we usually set $M$ to be 5–20, the relaxation method has the potential to significantly reduce the computational time because hundreds of UE problems normally need to be solved by other CNDP algorithms. However, since a direct solve is applied in this paper for the single-level NLP without exploring the structure of the problem, the actual running time of the proposed relaxation algorithm could be large. A decomposition method has just been reported by the authors [24]. Also notice that the relaxation algorithm presented here is straightforward and easily implemented. In particular, many existing solution techniques for the NLP can be used in Step 2.1 to tackle the single-level NLP.

## 4. Numerical examples

In this section, we test our proposed model and solution approach using two networks. The first one, as shown in Fig. 1, was first presented by Harker and Friesz [25] for studying CNDP, and the second one, depicted in Fig. 2, is for the City of Sioux Falls firstly used by LeBlanc et al. [26]. Both of these two networks have been extensively tested in the CNDP literature [4,6,10], and the network configurations and associated data can be found in [6]. In particular, the first network has two OD pairs, $1 \rightarrow 6$ and $6 \rightarrow 1$. For the second network, there are 528 OD pairs and only ten links (namely, link 16, 17, 19, 20, 25, 26, 29, 39, 48 and 74) are selected for capacity enhancement. These ten links were selected since they were among the most congested links in the network [7]. They were first chosen by Abdulaal and LeBlanc [7] and followed by other researchers for testing various CNDP models and solution algorithms [4,6,10].

In this paper, to reduce the effort of our implementation, we adopt the NLPEC (nonlinear program with equilibrium constraints) solver by Ferris et al. [27] to solve our proposed MPCC model. NLPEC can automatically convert a given MPCC model (e.g., our CNDP model (4)) to an equivalent NLP problem (e.g., the model (5)) and solve it using the relaxation algorithm (e.g., replacing (5g) and (5h) with (6a) and (6b)).

The models and solution algorithms by all the aforementioned authors dealt with only the symmetric case (i.e., the underlying user equilibrium is indeed an NLP). Since our model can handle both the symmetric and asymmetric cases, we will first compare the results of our method with existing models for the symmetric case and then we will test the proposed model for asymmetric cases.

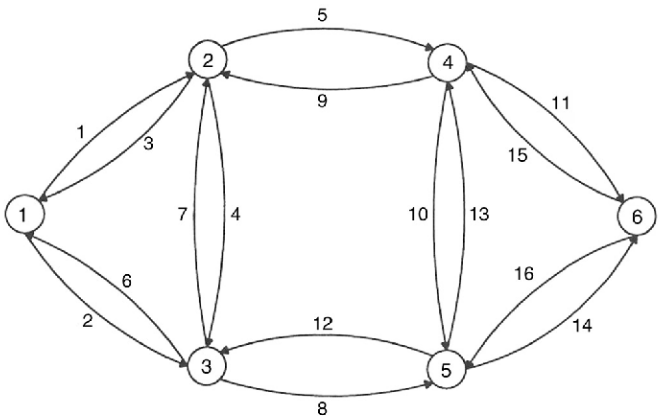


Fig. 1. Testing network #1.

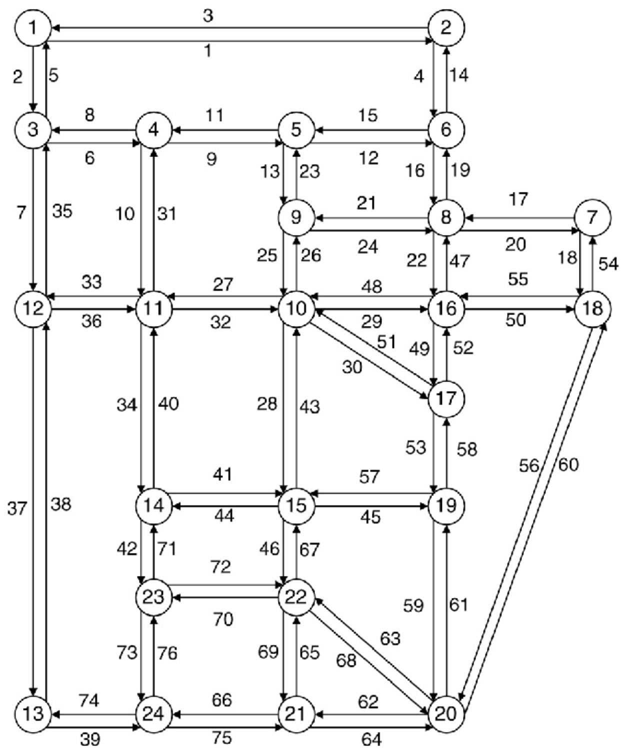


Fig. 2. Testing network #2 (Sioux Falls).

## *4.1. Symmetric cases*

Symmetric cases adopt the BPR function to compute link travel cost since it has been widely accepted and applied in many static traffic assignment applications. The BPR function can be expressed in Eq. (7).

$$t_{ij}(x_{ij}, y_{ij}) = A_{ij} + B_{ij}(x_{ij}/(K_{ij} + y_{ij}))^4, \tag{7}$$

where $A_{ij}$, $B_{ij}$, and $K_{ij}$ are parameters for link $(i, j)$ as listed in [6]; in particular, $K_{ij}$ is the capacity for link $(i, j)$.

For the convenience of comparison, we first list the abbreviations of the previous algorithms as well as the one in this paper in Table 1 (revised from Table 1 in [10]). Among all the previous models, SA is regarded as the one which could produce the global optimal solution [4].

Table 1
Abbreviations of algorithms

| Abbreviation | Title of the algorithm | Source |
|---|---|---|
| MINOS | Modular in-core nonlinear system | Suwansirikul et al. [6] |
| HJ | Hooke–Jeeves algorithm | Abdulaal and LeBlanc [7] |
| EDO | Equilibrium decomposed optimization | Suwansirikul et al. [6] |
| SA | Simulated annealing algorithm | Friesz et al. [4] |
| AL | Augmented Lagrangian algorithm | Meng et al. [10] |
| RELAX | Relaxation method | Presented in this paper |

Table 2
Comparison of results for scenario I of network 1[a]

| Variable (thousand vehicles) | MINOS | HJ | EDO | SA | AL | RELAX |
|---|---|---|---|---|---|---|
| $y_{2.1}$ (Link 3) | | 1.2 | 0.13 | | 0.0062 | |
| $y_{3.1}$ (Link 6) | 6.58 | 3 | 6.26 | 3.1639 | 5.2631 | 5.19458 |
| $y_{3.2}$ (Link 7) | | | | | 0.0032 | |
| $y_{4.6}$ (Link 11) | | | | | 0.0064 | |
| $y_{6.4}$ (Link 15) | 7.01 | 3 | 0.13 | | 0.7171 | |
| $y_{6.5}$ (Link 16) | 0.22 | 2.8 | 6.26 | 6.724 | 6.7561 | 7.596208 |
| Obj. value (thousand vehicle hours) | 211.25 | 215.08 | 201.84 | **198.1038** | 202.9913 | 199.6253 |
| Number of solved UE problems | – | 54 | 10 | 18,300 | 2700 | – |

[a] Note: Demand $1 \to 6 = 5$, Demand $6 \to 1 = 10$, total traffic demand is 15.0. $0 \leq y_{ij} \leq 10, \forall(i, j) \in A$ for EDO, SA, AL, and RELAX.

Table 3
Comparison of results for scenario II of network 1[a]

| Variable (thousand vehicles) | MINOS | HJ | EDO | SA | AL | RELAX |
|---|---|---|---|---|---|---|
| $y_{1.3}$ (Link 2) | 4.61 | 5.4 | 4.88 | | 4.6153 | 4.614426 |
| $y_{2.1}$ (Link 3) | 9.86 | 8.18 | 8.59 | 10.174 | 9.8804 | 9.910446 |
| $y_{3.1}$ (Link 6) | 7.71 | 8.1 | 7.48 | 5.7769 | 7.5995 | 7.373796 |
| $y_{3.2}$ (Link 7) | | | 0.26 | | 0.0016 | |
| $y_{3.5}$ (Link 8) | 0.59 | 0.9 | 0.85 | | 0.6001 | 0.592238 |
| $y_{4.2}$ (Link 9) | | | | | 0.001 | |
| $y_{5.3}$ (Link 12) | | | | | 0.113 | |
| $y_{5.6}$ (Link 14) | 1.32 | 3.9 | 1.54 | | 1.3184 | 1.315255 |
| $y_{6.4}$ (Link 15) | 19.14 | 8.1 | 0.26 | | 2.7265 | |
| $y_{6.5}$ (Link 16) | 0.85 | 8.4 | 12.52 | 17.2786 | 17.5774 | 20.000000 |
| Obj. value (thousand vehicle hours) | 557.14 | 557.22 | 540.74 | 528.497 | 532.71 | **522.6439** |
| Number of solved UE problems | – | 134 | 12 | 24,300 | 4000 | – |

[a] Note: Demand $1 \to 6 = 10$, Demand $6 \to 1 = 20$, total traffic demand is 30.0. $0 \leq y_{ij} \leq 20, \forall(i, j) \in A$ for EDO, SA, AL, and RELAX.

Two scenarios for the network in Fig. 1 are tested in this section, with the results shown in Tables 2 and 3. Scenario I is designed to test for a lower demand level, while scenario II aims to test on a higher demand level. Notice that the upper bound of capacity enhancement for each link in these two cases are also different. For the relaxation algorithm, we set the initial auxiliary parameter $\mu^{s0} = 10, \forall s \in S$, the iteration limit $M = 6$, and the update factor $\lambda = 0.1$. We can see from Table 2 that for scenario I, we achieve a solution whose objective value is very close to that obtained by the SA method, while for scenario 2 in Table 3, our method can find a solution whose objective value is significantly smaller than the one obtained by SA. Another advantage of the relaxation method is that we do not need to solve so many UE problems as are generally required by other methods, as shown in Tables 2 and 3, which could be expensive for large-scale networks.

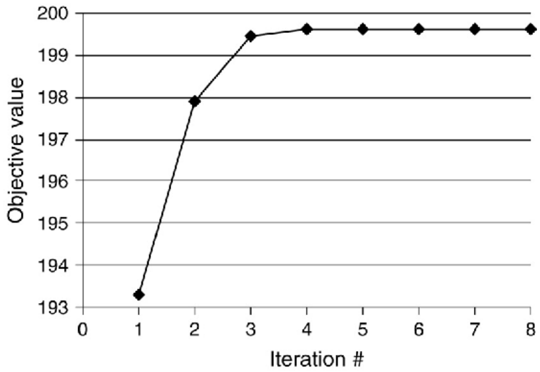


Fig. 3. Objective value vs. iteration # (Scenario I).

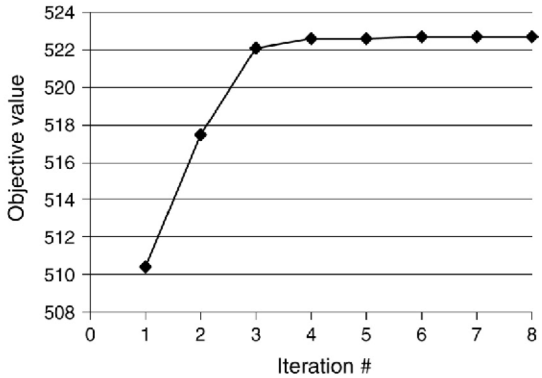


Fig. 4. Objective value vs. integration # (Scenario II).

In Figs. 3 and 4, we show the value of the objective function at each iteration for scenario 1 and 2, respectively. Note that initially the value of the auxiliary parameter is large (10 in our case), implying that the lower-level variables $(v^s, \pi^s), \forall s \in S$ are constrained by a larger set. Thus we can obtain a smaller objective value. However, as we restrict the auxiliary parameter values in later iterations closer and closer to zero (with a scaling factor 0.1), we essentially reduce the defining set of $(v^s, \pi^s), \forall s \in S$. Then, larger objective values would be gradually obtained. Moreover, for the final solve, we enforce the auxiliary parameter value to be zero which means that the lower-level user equilibrium condition holds exactly. It turns out that RELAX can perform successfully for the final solves for both scenarios; therefore, we eventually obtain (local) optimal solutions for them. It is worth noting that an objective value which is very close to the optimum can be obtained even when the value of the auxiliary parameter is not trivial. For example, at iteration #4 in Fig. 3, the objective value (199.6069) is very close to the final one (199.6253, at iteration #8) although the value of the auxiliary parameter is still pretty large (0.01). This implies that in order to obtain an approximate solution to the original bilevel model, it is reasonable to adopt a relaxation scheme to the lower-level user equilibrium condition. Similar results can also be observed for scenario II as shown in Fig. 4.

One scenario is tested for the Sioux Falls network. We set the initial auxiliary parameter $\mu^{s0} = 1, \forall s \in S$, the iteration limit $M = 10$, and the update factor $\lambda = 0.4$ for this particular example. The results are shown in Table 4. Similarly as for network 1, RELAX can find a solution whose objective value is slightly smaller than that by the SA method. Moreover, the value of the objective function at each iteration for network 2 also changes in the same manner as for network 1. Such observations may have further impacts on designing solution algorithms for CNDP, especially when an approximate solution is desired. In other words, if an approximate solution is required with objective value close to the true minimized one, we only need to solve the relaxed NLP several times (4 or 5 in the two examples). This will significantly reduce the computational time of the solution process.

Table 4
Comparisons of results for network 2[a]

| Variable (thousand vehicles) | MINOS | HJ | EDO | SA | AJ | RELAX |
|---|---|---|---|---|---|---|
| Initial value of $y_{ij}$ | 2 | 1 | 12.5 | 6.25 | 12.5 | 12.5 |
| $y_{6.8}$ (Link 16) | 4.8 | 3.8 | 4.59 | 5.38 | 5.5728 | 5.279134 |
| $y_{7.8}$ (Link 17) | 1.2 | 3.6 | 1.52 | 2.26 | 1.6443 | 1.957131 |
| $y_{8.6}$ (Link 19) | 4.8 | 3.8 | 5.45 | 5.5 | 5.6228 | 5.279134 |
| $y_{8.7}$ (Link 20) | 0.8 | 2.4 | 2.33 | 2.01 | 1.6443 | 1.957131 |
| $y_{9.10}$ (Link 25) | 2 | 2.8 | 1.27 | 2.64 | 3.1437 | 2.51829 |
| $y_{10.9}$ (Link 26) | 2.6 | 1.4 | 2.33 | 2.47 | 3.2837 | 2.51829 |
| $y_{10.16}$ (Link 29) | 4.8 | 3.2 | 0.41 | 4.54 | 7.6519 | 3.038993 |
| $y_{13.24}$ (Link 39) | 4.4 | 4 | 4.59 | 4.45 | 3.8035 | 4.991152 |
| $y_{16.10}$ (Link 48) | 4.8 | 4 | 2.71 | 4.21 | 7.382 | 3.038993 |
| $y_{24.13}$ (Link 74) | 4.4 | 4 | 2.71 | 4.67 | 3.6935 | 4.991152 |
| Obj. value (thousand vehicle hours) | 81.25 | 81.77 | 83.47 | 80.87 | 81.752 | **80.5157** |
| Number of solved UE problems | 58 | 108 | 12 | 3900 | 2700 | – |

[a] Note: $0 \le y_{ij} \le 25, \forall (i, j) \in A$ for EDO, SA, AL, and RELAX.

### 4.2. Asymmetric cases

Interestingly, testing examples for CNDP under AUE are sparse in the literature. This is probably due to the fact that unlike the BPR function for the symmetric case, there is no standard way to design the link cost function for AUE, although some preliminary research has been done previously [28]. Normally empirical analysis needs to be done before any reasonable functional form can be derived. In this section, however, in order to demonstrate that our proposed model and solution method can be applied to asymmetric cases, we revise the BPR cost function in (7) so that link interactions could occur among adjacent links. In particular, the cost function for link $(i, j) \in A$ is defined as follows:

$$t_{ij}(x, y_{ij}) = A_{ij} + B_{ij}\left[\frac{\sum_{(k,j)\in A} \rho_{ij,kj} x_{kj} + \sum_{(j,l)\in A} \rho_{ij,jl} x_{jl}}{\alpha \cdot K_{ij} + y_{ij}}\right]^4, \tag{8}$$

where $0 \le \rho_{ij,kj} \le 1$ (or $0 \le \rho_{ij,jl} \le 1$) denotes the "impact factor" of the flow on link $(k, j)$ (or link $(j, l)$) to the travel cost of link $(i, j)$. Apparently, we have $\rho_{ij,ij} = 1, \forall (i, j) \in A$. If $\rho_{ij,kj} = 0, \forall j \in N, (i, j) \in A, (k, j) \in A, i \ne k$ and $\rho_{ij,jl} = 0, \forall j \in N, (i, j) \in A, (j, l) \in A$, (8) will reduce to the standard BPR function for the symmetric case. Note that $\alpha \ge 1$ in (8) represents a factor to increase the original capacity of the selected links. In this paper, we set $\alpha = 1$ for network 1 and $\alpha = 1.5$ for the Sioux Falls network since the latter is already in a high congestion level even for the symmetric case. Furthermore, for illustrative purposes, we set $\rho_{ij,kj} = 0.15, \forall j \in N, (i, j) \in A, (k, j) \in A, i \ne k$ and $\rho_{ij,jl} = 0.1, \forall j \in N, (i, j) \in A, (j, l) \in A$. Note that, in the literature, there are several ways to construct the link travel time for AUE [28]. Eq. (8) is probably the most general expression which considers the impacts of all adjacent links to a given link. However, we should point out here that although Eq. (8) is only an intuitive way to achieve (asymmetric) link interactions among adjacent links. How to design a practically reasonable asymmetric link cost function for a given network is beyond the scope of this paper.

We first test the two scenarios of network 1. We use the same settings for RELAX as those in previous section. The final results are shown in Table 5. As we would expect, the objective values for both scenarios are increased due to the link interactions (note that $\alpha = 1$ for network 1). Correspondingly, the optimal link enhancement for each individual link is also changed. The final results for network 2 for the asymmetric case are shown in Table 6. Note that although we increased the link capacity of each selected link by 50%, the objective value for AUE has still increased substantially due to link interactions. Similarly, the optimal link enhancement is also changed.

Table 5
Results of network 1 under AUE

| Variable (thousand vehicles) | Scenario I | Scenario II |
|---|---|---|
| $y_{1.3}$ (Link 2) | | 6.477080 |
| $y_{2.1}$ (Link 3) | 0.849351 | 12.376940 |
| $y_{3.1}$ (Link 6) | 6.046094 | 12.313064 |
| $y_{3.5}$ (Link 8) | | 1.942252 |
| $y_{5,3}$ (Link 12) | | 1.255003 |
| $y_{5,6}$ (Link 12) | | 4.662409 |
| $y_{6.4}$ (Link 16) | | 20.000000 |
| $y_{6.5}$ (Link 16) | 10.00000 | 2.438337 |
| Obj. value (thousand vehicle hours) | **221.7194** | **648.8585** |

Table 6
Results of network 2 under AUE

| Variable (thousand vehicles) | |
|---|---|
| Initial Value of $y_{ij}$ | 12.5 |
| $y_{6.8}$ (Link 16) | 6.153192 |
| $y_{7.8}$ (Link 17) | 2.919601 |
| $y_{8.6}$ (Link 19) | 5.548870 |
| $y_{8.7}$ (Link 20) | 4.381182 |
| $y_{9.10}$ (Link 25) | 4.477047 |
| $y_{10.9}$ (Link 26) | 2.261691 |
| $y_{10.16}$ (Link 29) | 5.389955 |
| $y_{13.24}$ (Link 39) | 5.462957 |
| $y_{16.10}$ (Link 48) | 9.358611 |
| $y_{24.13}$ (Link 74) | 5.891133 |
| Obj. value (thousand vehicle hours) | **140.1709** |

### *4.3. Discussions on relaxation parameters*

The three relaxation parameters are crucial for the relaxation algorithm. Firstly, the number of iterations of the algorithm, $M$, cannot be determined arbitrarily; rather, it has to be decided according to the value of the initial auxiliary parameter $\mu^{s0}$ and the update factor $\lambda$. Generally, for large values of $\mu^{s0}$ and $\lambda$, $M$ must be set large enough to assure that the last solve of Step 2 in the relaxation algorithm can produce a solution close enough to the true solution of the CNDP. This can be roughly checked by whether the auxiliary parameter for the last solve in Step 2, i.e. $\mu^{sM}$, is small enough.

To obtain an initially feasible solution to the CNDP easily, a large $\mu^{s0}$ is usually preferred. However, as $\mu^{s0}$ increases, $M$ has to be increased also, which may slow down the convergence of the relaxation algorithm. Therefore, a proper value of $\mu^{s0}$ is problem specific and represents a certain tradeoff between speed and convergence accuracy of the algorithm.

The update factor $\lambda$, on the other hand, represents how aggressively the auxiliary parameters are reduced. Large values of $\lambda$ (which should be less than 1) mean the change of auxiliary parameters between two iterations is small. Hence, the entire solution process tends to be smooth and easier. However, large values of $\lambda$ also imply that the number of iterations needed ($M$) is large, which might require more computational time. To further study the impact of $\lambda$ on the performance of the relaxation algorithm, we test on the two scenarios on network 1 by ranging $\lambda$ from very small (0.001) to very large (0.6). Fig. 5 depicts the average number of iterations for solving the relaxed single-level NLP in Step 2.1 with respect to the change of $\lambda$. In the figure, IS (IIS) and IA (IIA) denote the symmetric and asymmetric case for scenario I (II), respectively. Note that the solution algorithm for an NLP is itself an iterative procedure. We can easily observe from Fig. 5 that as $\lambda$ increases, the number of iterations required for solving the single-level NLP decreases approximately monotonically. This clearly shows that as $\lambda$ increases the effort for solving each single-level NLP is reduced. Therefore, although $M$ increases as $\lambda$ does, the computational effort for solving a CNDP is roughly

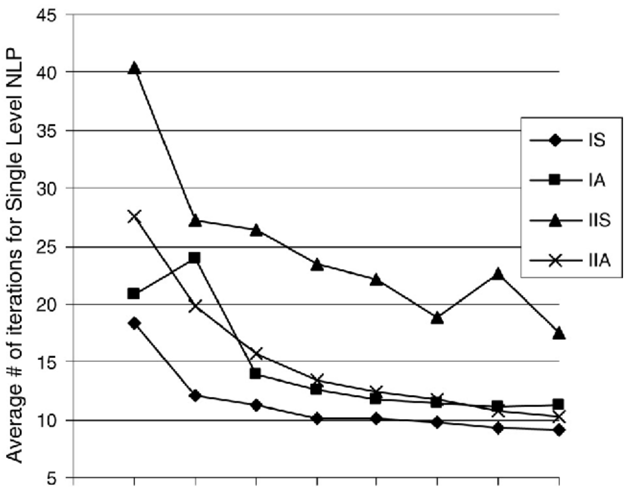


Fig. 5. Average # of iterations for a single-level NLP.

the same for different values of $\lambda$. Based on such an observation, $\lambda$ should be chosen large enough (normally $\geq 0.1$) to ensure the smoothness of the relaxation process.

## 5. Conclusions

In this paper, we formulated CNDP as an MPCC model with an NLP to represent the upper-level optimization problem and an NCP for the lower-level user equilibrium. The proposed formulation can handle both symmetric and asymmetric user equilibria, which extends previous CNDP studies. The MPCC model is converted to a single-level smooth NLP problem and solved by a relaxation scheme using existing NLP solution techniques. The model and solution approach were implemented and tested, and promising results were achieved for several well-known CNDP testing problems. In particular, we showed that in order to obtain an approximate solution, the relaxed problem needs only to be solved several times, which makes the solution approach computationally efficient.

For future study, we need to test the model and algorithm for large-scale problems. In order to do this, however, we notice that the lower-level AUE has to be defined using the disaggregated variables. This may bring in the dimensionality problem [10] for a large-scale CNDP if the single-level problem is to be solved directly. That is, the resulting single-level NLP might have a large number of defining variables, especially for multiple origin and multiple destination (many-to-many) problems [26]. However, the lower-level AUE has a very special structure such that it can be easily decomposed according to individual destinations. Therefore, certain decomposition approaches may be employed to efficiently solve the single-level NLP for large size problems. We have recently reported some preliminary results by applying the Gauss–Seidel decomposition scheme for solving the proposed MPCC model [24]. Further research on this topic is currently under investigation by the authors and more results will be reported in subsequent papers.

## Acknowledgements

The authors would like to thank two anonymous reviewers for their helpful comments.

## References

[1] L.J. Leblanc, D. Boyce, A bilevel programming algorithm for exact solution of the network design problem with user-optimal flows, Transportation Research 20B (1986) 259–265.
[2] Z.Q. Luo, J.S. Pang, D. Ralph, Mathematical Programs with Equilibrium Constraints, Cambridge University Press, Cambridge, UK, 1996.
[3] E. Morlok, J. Schofer, W. Pierskalla et al., Development and Application of a Highway Network Design Model, Final Report: FHWA Contract Number DOT-PH-11, Northwestern University, 1973.
[4] T.L. Friesz, H.J. Cho, N.J. Mehta et al., A simulated annealing approach to the network design problem with variational inequality constraints, Transportation Science 26 (1992) 18–26.

[5] P. Marcotte, Network optimization with continuous control parameters, Transportation Science 17 (1983) 181–197.

[6] C. Suwansirikul, T.L. Friesz, R.L. Tobin, Equilibrium decomposed optimization: a heuristic for the continuous equilibrium network design problem, Transportation Science 21 (1987) 254–263.

[7] M. Abdulaal, L.J. LeBlanc, Continuous equilibrium network design models, Transportation Research 13B (1979) 19–32.

[8] H. Yang, Sensitivity analysis for the elastic demand network equilibrium problem with applications, Transportation Research 31B (1997) 55–70.

[9] H. Yang, M.G.H. Bell, Transport bilevel programming problems: recent methodological advances, Transportation Research 35B (2001) 1–4.

[10] Q. Meng, H. Yang, M.G.H. Bell, An equivalent continuously differentiable model and a locally convergent algorithm for the continuous network design problem, Transportation Research 35B (2001) 83–105.

[11] P. Marcotte, D.L. Zhu, Exact and inexact penalty methods for the generalized bilevel programming problem, Mathematical Programming 74 (1996) 141–157.

[12] M. Patriksson, R.T Rockafellar, A mathematical model and descent algorithm for bilevel traffic management, Transportation Science 36 (2002) 271–291.

[13] M. Anitescu, On Solving Mathematical Programs with Complementarity Constraints as Nonlinear Programs, MCS Division, Argonne National Laboratory, Argonne, IL, 2000. Reprint ANL/MCS-P864-1200.

[14] M. Anitescu, Global Convergence of an Elastic Mode Approach for a Class of Mathematical Programs with Complementarity Constraints, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL, USA, 2004. Preprint ANL/MCS-P1143-0404.

[15] R. Fletcher, S. Leyffer, Solving mathematical program with complementarity constraints as nonlinear programs, Optimization Methods and Software 19 (1) (2004) 15–40.

[16] X. Hu, D. Ralph, A note on sensitivity of value functions of mathematical programs with complementarity constraints, Mathematical Programming 93 (2) (2002) 265–279.

[17] H. Jiang, D. Ralph, Smooth SQP methods for mathematical programs with nonlinear complementarity constraints, SIAM Journal of Optimization 10 (3) (2000) 779–808.

[18] H. Jiang, D. Ralph, Extension of quasi-Newton methods to mathematical programs with complementarity constraints, Computational Optimization and Applications 25 (1–3) (2003) 123–150.

[19] H. Scheel, S. Scholte, Mathematical programs with complementarity constraints: stationarity, optimality, and sensitivity, Mathematics of Operations Research 22 (1) (2000) 1–22.

[20] D. Ralph, S.J. Wright, Some properties of regularization and penalization schemes for MPECs, Optimization Methods and Software 19 (2004) 527–556.

[21] J.G. Wardrop, Some theoretical aspects of road traffic research, Proceedings of the Institution of Civil Engineers, Part II 1 (1952) 325–378.

[22] M. Patriksson, The Traffic Assignment Problem, Models and Methods, VSP, 1994.

[23] J.X. Ban, Quasi-variational inequality formulations and solution approaches for dynamic user equilibria, Ph.D. Thesis, University of Wisconsin–Madison, 2005.

[24] J.X. Ban, H.X. Liu, J.G. Lu, M.C. Ferris, A decomposition scheme for continuous network design problem with asymmetric user equilibrium. Accepted for presentation and pending for publication at Transportation Research Board Meeting.

[25] T.L. Harker, T.L. Friesz, Bounding the solution of the continuous equilibrium network design problem, in: Proceedings of the Ninth International Symposium on Transportation and Traffic Theory, VNU Science Press, Delft, Netherlands, 1984.

[26] L.J. Leblanc, E.K. Morlok, W.P. Pierskalla, An efficient approach to solving the road network equilibrium traffic assignment problem, Transportation Research 9 (1975) 309–318.

[27] M.C. Ferris, S.P. Dirkse, A. Meeraus, Mathematical programs with equilibrium constraints: automatic reformulation and solution via constrained optimization, Numerical Analysis Group Research Report NA-02/11, Oxford University Computing Laboratory, Oxford University, USA, 2002.

[28] A. Nagurney, Comparative test of multimodal traffic equilibrium methods, Transportation Research 18B (1984) 469–485.