Numerical Studies on Reformulation Techniques for Continuous Network Design with Asymmetric User Equilibria

Xuegang (Jeff) Ban*
Assistant Professor
Department of Civil and Environmental Engineering
Rensselaer Polytechnic Institute (RPI)
110 Eighth Street, room JEC 4034
Troy, NY 12180-3590
Phone: (518) 276-8043 Fax: (518) 276-4833
* **Corresponding Author**

Michael C. Ferris
Professor
Computer Sciences Department
University of Wisconsin – Madison
4381 Computer Sciences & Statistics Building
1210 West Dayton Street
Madison, WI 53706
Tel: (608) 262-4281, Fax: (608) 262-9777
Email: ferris@cs.wisc.edu

Henry X. Liu
Assistant Professor
Department of Civil Engineering
University of Minnesota – Twin Cities
122 Civil Engineering Building
500 Pillsbury Drive S.E.
Minneapolis, MN 55455
Tel: (612) 625-6347, Fax: (612) 626-7750
Email: henryliu@umn.edu

Dr. Jeff Ban is an Assistant Professor of Civil and Environmental Engineering Department at Rensselaer Polytechnic Institute (RPI). He received his PhD from the University of Wisconsin at Madison in 2005. Dr. Ban's research interests are in the areas of transportation modeling and traffic simulation, traffic control and operations, network modeling, and Intelligent Transportation Systems (ITS). His recent research activities include modeling mobile sensors (like GPS cellular phones) as traffic probes, applications of microscopic traffic simulation to corridor-based transportation management, and robust congestion pricing. He presented his research at national and international conferences, including Transportation Research Board Annual Meetings, International Conference on Complementarity Problems (ICCP), and INFORMS. His articles appeared in *Transportation Research Part B*, *Mathematical and Computer Modeling*, *Networks and Spatial Economics*, *International Journal of Operations Research and Information Systems,* and *Transportation Research Record*.

Dr. Michael Ferris' Michael C. Ferris is Professor of Computer Sciences and Industrial and Systems Engineering at the University of Wisconsin, Madison, USA. He received his PhD from the University of Cambridge, England in 1989. Dr. Ferris' research is concerned with algorithmic and interface development for large scale problems in mathematical programming, including links to the GAMS and AMPL modeling languages, and general purpose software such as PATH, NLPEC and FATCOP. He has worked on several applications of both optimization and complementarity, including cancer treatment plan development, radiation therapy, video-on-demand data delivery, economic and traffic equilibria, structural and mechanical engineering. Ferris received the Beale-Orchard-Hays prize from the Mathematical Programming Society and is a past recipient of a NSF Presidential Young Investigator Award, and a Guggenheim Fellowship. He serves as co-editor of Mathematical Programming, and is on the editorial boards of SIAM Journal on Optimization, Transactions of Mathematical Software, and Optimization Methods and Software.

Dr. Henry Liu is currently an Assistant Professor of Civil Engineering at the University of Minnesota, Twin Cities. His research interests are in the areas of traffic control and operations, traffic flow theory, and transportation network modeling. His recent research activities include arterial traffic data collection and signal optimization, real-time traffic management for emergency evacuation, analytical dynamic traffic assignment, and application of microscopic traffic simulation models. He has published more than 30 refereed journal articles in these areas. Dr. Liu is a member of Transportation Network Modeling Committee of Transportation Research Board (TRB) and a member of Traffic Signal Operations Committee of Institute of Transportation Engineers (ITE). In 2008, Dr. Liu received the New Faculty Member Award presented by the Council of University of Transportation Research Centers (CUTC) and American Road and Transportation Builders association (ARTBA).

Numerical Studies on Reformulation Techniques for Continuous Network Design with Asymmetric User Equilibria

Xuegang (Jeff) Ban*, Rensselaer Polytechnic Institute, USA
Michael Ferris, University of Wisconsin-Madison, USA
Henry Liu, University of Minnesota, USA

ABSTRACT
In this paper, we aim to find the most effective reformulation techniques to solve the MPCC (mathematical program with complementarity constraints) model that we proposed recently for continuous network design problems under asymmetric user equilibria. The MPCC model is based on a link-node nonlinear complementarity formulation for asymmetric user equilibria. By applying various reformulation techniques for the lower level nonlinear complementarity, the original bilevel formulation can be converted to a single level nonlinear programming problem. We show that certain reformulations are more effective than others to solve the proposed MPCC model. Recommendations are thus provided on how to choose a reformulation of the continuous network design problem that can be solved effectively and/or efficiently.

Keywords: continuous network design; user equilibrium; asymmetric user equilibrium; mathematical programs with equilibrium constraints; mathematical programs with complementarity constraints; nonlinear complementarity problems

INTRODUCTION AND MOTIVATION

The continuous network design problem (CNDP) has been proposed and studied for the last several decades, aiming to determine the optimal capacity expansion for a set of selected links in a given transportation network (see Morlok (1973), Abdulaal and LeBlanc (1979), Tan (1979), Marcotte (1983), Suwansirikul et al. (1987), Friesz et al. (1992), Yang (1997), Josefsson and Patriksson (2008), to name just a few). CNDP can be formulated as a bilevel programming problem – the upper level is a nonlinear programming problem (NLP) to minimize certain system cost, and the lower level is a user equilibrium (UE) problem to account for drivers' route choice behavior. Such a bilevel formulation is also termed mathematical programs with equilibrium constraints (MPEC), which are generally non-convex and nonsmooth, and even fail to satisfy certain constraint qualifications (e.g. the Mangasarian Fromovitz Constraint Qualification, see Luo et al., 1996). Therefore, solving the bilevel CNDP problem is usually challenging, and despite the extensive research so far, no algorithm has been developed that can efficiently solve CNDPs with guaranteed convergence (Yang and Bell, 2001).

One of the widely used solution techniques for CNDP is the sensitivity-based algorithm (Tobin and Friesz, 1988), based on the observation that although CNDP is not smooth everywhere, it IS relatively smooth (Friesz et al., 1990), i.e. smooth *almost* everywhere. Therefore, the gradient of the objective function with respect to the upper level decision variable can be calculated using the sensitivity analysis technique. Such a method however may fail especially when the problem is not smooth at the optimal solution (Josefsson and Patriksson, 2008). More recent approaches for modeling CNDP focus on reformulating the problem using certain forms of smooth gap functions for the lower-level UE. Meng et al. (2001) formulated CNDP as a bilevel problem with the lower level UE an NLP. Using a particular gap function for the lower level problem, the bilevel model was converted to a single level (yet smooth) NLP and solved using the augmented Lagrangian method. Nevertheless, the model was based on the symmetry assumption on the lower-level UE problem. Here we note that a UE is *symmetric* if the travel time of any link depends on only the traffic flow of itself (also called *separable*) or the dependence of travel times on link flows are symmetric; otherwise, an *asymmetric* UE (AUE) will have to be considered if the dependence is not symmetric. For formal definitions of symmetric and asymmetric UEs, one can refer to Patriksson (1994, equation (2.44)). Symmetric UE is therefore a special case of the asymmetric UE (AUE), and can be formulated as an NLP (Beckmann et al., 1956). Beckmann-type NLP formulation however does not exist for AUE, which has to be formulated as a nonlinear complementarity problem (NCP) or variational inequality (VI). By defining certain gap functions for the lower level VI, Marcotte (1996) transferred the bilevel problem to a single level one and solved using a penalty method. Patriksson and Rockafellar (2002) presented a reformulation technique to convert a CNDP into a constrained and locally Lipschitz minimization problem which can be further solved using an algorithm proposed in the same paper. However, neither Marcotte (1996) nor Patriksson and Rockafellar (2002) tested their models using well-known CNDP examples in the transportation field.

The authors of this paper recently formulated CNDP as a bilevel problem with the lower level a link-node based NCP formulation to represent the user equilibrium (Ban et al., 2006a, 2006b). To solve the model, the product reformulation was applied to convert the bilevel model to a single level one. A decomposition scheme was also discussed to solve the proposed model (Ban et al., 2006b). As there are various techniques to convert a bilevel mode to a single level model (particularly for the NCP-based UE formulation, see Ferris, 2004), one interesting question is which reformulation method is the most effective in terms of solution quality (i.e. the objective value of the upper level) or computational efficiency (i.e. the CPU time used to produce the solution) or both. A solid answer to this question is expected to provide useful information to select the most appropriate reformulation technique for CNDP; it can also provide insights to solving similar bilevel models for applications beyond CNDP.

In this paper, we focus on numerical experiments to find the most effective reformulation techniques. We first summarize the properties of the link-node NCP UE model, and show that under mild conditions, the NCP model has at least one solution which is unique in terms of total link flows. Based on the NCP model, we formulate CNDP as a mathematical program with complementarity constraints (MPCC). As a special case of MPEC, MPCC has specific structures that can be exploited in the solution process. A variety of methods can be applied to convert an MPCC to a single level NLP that can be solved using existing solution techniques (Scheel and Scholte, 2000; Jiang and Ralph, 2000; Hu and Ralph, 2002; Anitescu, 2004; Fletcher and Leyffer, 2004). To solve the proposed MPCC-based CNDP model, we apply a relaxation method that relaxes the lower level complementarity condition using a relaxation parameter. To find out the most appropriate reformulation techniques for CNDP, we conduct numerical studies in this paper using the NLPEC (nonlinear programs with equilibrium constraints) solver as the solution tool (Ferris et al., 2005). The performance measures include both solution quality and efficiency. On some test CNDP problems, we examine various options available in NLPEC for reformulating the lower level NCP. We find that certain reformulation types and options are particularly effective for solving CNDP. The solutions found for two well-known CNDPs in the transportation literature are better than those obtained using traditional methods. A scheme is also developed in this paper to automatically determine the relaxation settings for a CNDP, together with investigations of the impact of link interactions on solution quality and efficiency. The remainder of this paper is organized as follows. Section 2 presents the MPCC model for CNDP. The relaxation solution algorithm, especially the NLPEC solver, is briefly discussed in Section 3. Section 4 provides numerical experiments conducted to determine the best reformulation techniques for solving MPCC based CNDP. Section 5 concludes our study.

MPCC MODEL FOR CNDP

Link-Node NCP Formulation for Asymmetric User Equilibria

User Equilibrium (UE) assignment aims to (simultaneously) determine the distribution of traffic flow from any origin to any destination in a given transportation network. As shown in Wardrop's first principle (Wardrop, 1952), a UE state will be reached if no driver can reduce his/her travel time by unilaterally changing the route. In other words, under UE, all drivers between an origin-destination pair will only use routes having minimum travel time. The route choice condition for AUE may be expressed in a link-node manner as follows:

$$\begin{cases} \pi_j^s + t_{ij}(\sum_{s \in S} v^s) - \pi_i^s > 0 \Rightarrow v_{ij}^s = 0, \forall (i,j) \in A, \forall s \in S, \\ \pi_j^s + t_{ij}(\sum_{s \in S} v^s) - \pi_i^s = 0 \Rightarrow v_{ij}^s \geq 0, \forall (i,j) \in A, \forall s \in S. \end{cases} \quad (1)$$

Here S and A are the set of destination nodes and all links, respectively. Let v_{ij}^s denote the disaggregated link flow on link (i,j) with respect to destination $s \in S$, and $v^s = (v_{ij}^s)_{(i,j)\in A} \in \Re^{|A|}$. Let π_i^s be the minimum travel time from node i to destination s, and by convention, we set $\pi_s^s = 0$. In addition, $t_{ij}(\sum_{s \in S} v^s)$ is the link travel time for link (i,j).

Note that since t_{ij} is a function of the aggregated link flow $v^A = \sum_{s\in S} v^s \in \Re^{|A|}$, (1) captures both symmetric UE and AUE. As shown in Ban et al. (2006), (1) is equivalent to the following nonlinear complementarity problems (NCP):

$$0 \le [\Lambda_s v^s - d^s] \perp \pi^s \ge 0, \forall s \in S, \qquad (2a)$$

$$0 \le [-\Lambda_s^T \pi^s + t(\sum_{s\in S} v^s)] \perp v^s \ge 0, \forall s \in S, \qquad (2b)$$

where $d^s = (d_i^s)_{i\in N, i\neq s} \in \Re^{|N|-1}$. Here $\perp$ represents the "perpendicular" operation, i.e. $x \perp y \Leftrightarrow x^T y = 0$. Further denote $\Lambda \in \Re^{|N|\times|A|}$ as the link-node incidence matrix, and $\Lambda_s \in \Re^{(|N|-1)\times|A|}$ represents Λ with the row corresponding to destination s removed which guarantees Λ_s is a full row rank matrix (Ahuja et al., 1993).

Model Properties

In this section, we present solution properties of the NCP model (2). Most of the results shown here are well-known in the literature (e.g. Ahuja, 1993; Tobin and Freisz, 1988). We state them here without proof. We first start with the assumptions on link travel time function.

Assumption 1 The link travel time t may have the following properties:

(a). t is a smooth function of the aggregated link flow $v^A = \sum_{s\in S} v^s$, i.e. $t = t(\sum_{s\in S} v^s)$.

(b). $t_{ij} = t_{ij}(\sum_{s\in S} v^s), \forall (i,j) \in A$ is strictly positive for any $v_{ij}^A = \sum_{s\in S} v_{ij}^s > 0$.

(c). t is strictly monotone in terms of v^A, i.e. $(v_1^A - v_2^A)^T [t(v_1^A) - t(v_2^A)] > 0$, $\forall v_1^A \neq v_2^A$.

We can easily observe that:

(a) Under Assumption 1(a), the partial derivatives of the link travel time function to the disaggregated link flow variables corresponding to different destinations are the same for a given total link flow vector v^A. That is,

$$\partial t / \partial v^r = \partial t / \partial v^s = Q, \forall r, s \in S. \qquad (3)$$

(b) Assumes Assumption 1(a) – 1(c) holds. Then for a given destination $s \in S$, if the disaggregated link flows with respect to all other destinations are fixed, then the link travel time is a strictly monotone function in terms of v^s, i.e. the matrix Q in (3) is positive semi-definite.

It is easy to say that the following theorem holds.

Theorem 1. If the following two conditions hold, NCP model (2) will have at least one solution:

(a) Assumption 1(a) and 1(b) hold, and

(b) There exists at least one path from node i to destination s for any pair (i,s) if the demand $d_i^s > 0$ (which is equivalent to say that the AUE problem is feasible).

We can see that (2) is a well-defined NCP that has at least one solution. It is defined on disaggregated link flow variables, while the link travel time can be treated as a function of the total link flow. Since the link travel time is a function of all $v^s, \forall s \in S$, model (2) can represent AUE for which the link interactions need to be considered. The Jacobian matrix of NCP (2) is:

$$M = \begin{array}{c} \begin{array}{cccccc} \pi^1 & \cdots & \pi^{|S|} & v^1 & \cdots & v^{|S|} \end{array} \\ \begin{bmatrix} 0 & \cdots & 0 & \Lambda_1 & 0 & 0 \\ \vdots & \ddots & \vdots & 0 & \ddots & 0 \\ 0 & \cdots & 0 & 0 & 0 & \Lambda_{|S|} \\ -\Lambda_1^T & 0 & 0 & \partial t/\partial v^1 & \cdots & \partial t/\partial v^{|S|} \\ 0 & \ddots & 0 & \vdots & \ddots & \vdots \\ 0 & 0 & -\Lambda_{|S|}^T & \partial t/\partial v^1 & \cdots & \partial t/\partial v^{|S|} \end{bmatrix} \end{array} \begin{array}{c} \pi^1 \\ \cdots \\ \pi^{|S|} \\ v^1 \\ \cdots \\ v^{|S|} \end{array}. \qquad (4)$$

This is equivalent to (due to equation (3)):

$$M = \begin{array}{c} \begin{array}{cccccc} \pi^1 & \cdots & \pi^{|S|} & v^1 & \cdots & v^{|S|} \end{array} \\ \begin{bmatrix} 0 & \cdots & 0 & \Lambda_1 & 0 & 0 \\ \vdots & \ddots & \vdots & 0 & \ddots & 0 \\ 0 & \cdots & 0 & 0 & 0 & \Lambda_{|S|} \\ -\Lambda_1^T & 0 & 0 & Q & \cdots & Q \\ 0 & \ddots & 0 & \vdots & \ddots & \vdots \\ 0 & 0 & -\Lambda_{|S|}^T & Q & \cdots & Q \end{bmatrix} \end{array} \begin{array}{c} \pi^1 \\ \cdots \\ \pi^{|S|} \\ v^1 \\ \cdots \\ v^{|S|} \end{array}, \qquad (5)$$

where Q is positive semi-definite. It can then be easily seen that the Jacobian matrix M in (5) is positive semi-definite since Q is so. Note that M is not positive definite which implies that multiple solutions may exist for model (2). However, as stated in Theorem 2, the aggregated link flow must be unique if Assumption 1 holds. The proof of Theorem 2 can be found in Tobin and Friesz (1988).

Theorem 2. Denote $v^A = \sum_{s \in S} v^s$ the vector of total link flows. Assume Assumption 1 holds. Then NCP (2) has a unique solution in terms of the **total link flow** vector v^A. □

If we denote vector $x = [(\pi^1)^T \quad \cdots \quad (\pi^{|S|})^T \quad (v^1)^T \quad \cdots \quad (v^{|S|})^T]^T \in \Re^{(|A|+|N|-1)\cdot|S|}$, and a function

$$F(x) = \begin{bmatrix} \Lambda_1 v^1 - d^1 & & & & & \\ & \ddots & & & & \\ & & \Lambda_{|S|} v^{|S|} - d^{|S|} & & & \\ & & & -\Lambda_1^T \pi^1 + t(\sum_{s \in S} v^s) & & \\ & & & & \ddots & \\ & & & & & -\Lambda_{|S|}^T \pi^{|S|} + t(\sum_{s \in S} v^s) \end{bmatrix} \in \Re^{(|A|+|N|-1)\cdot|S|}, \quad (6)$$

then (2) can be rewritten in a standard form as *finding a vector x which satisfies*:

$$0 \le x \perp F(x) \ge 0. \quad (7)$$

Equation (7) is the link-node NCP formulation that will be used later for modeling the bilevel CNDP. We can easily observe that (7) 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 AUE problem itself and the CNDP problem constructed on (7).

MPCC Formulation for CNDP

In this section, we present the MPCC model for CNDP. Additional notation is first listed.

y_{ij} = the capacity enhancement for link $(i, j) \in A$, the upper level decision variable; $y = (y_{ij})_{(i,j)\in A} \in \Re^{|A|}$

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

$g_{ij}(y_{ij})$ = the cost function of capacity enhancement for link $(i, j) \in A$; $g = (g_{ij})_{(i,j)\in A} \in \Re^{|A|}$

θ = the relative weight of total capacity enhancement cost and total travel cost in the system design objective function.

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

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

With this in place, we can formulate CNDP with AUE as the following MPCC:

$$\min_{y, v^1, \cdots v^s, v^{|S|}, \pi^1, \cdots \pi^s, \pi^{|S|}} \sum_{(i,j)\in A} [t_{ij}(\sum_{s\in S} v^s, y_{ij}) \cdot \sum_{s\in S} v_{ij}^s] + \theta \sum_{(i,j)\in A} g_{ij}(y_{ij}), \quad (8a)$$

subject to

$$l_{ij} \le y_{ij} \le u_{ij}, \ \forall (i, j) \in A, \quad (8b)$$

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

$$\begin{aligned} &0 \le [\pi_j^s + t_{ij}(\sum_{s\in S} v^s, y_{ij}) - \pi_i^s] \perp v_{ij}^s \ge 0, \forall (i, j) \in A, s \in S, \\ &0 \le [\sum_{(i,j)\in A} v_{ij}^s - \sum_{(k,i)\in A} v_{ki}^s - d_i^s] \perp \pi_i^s \ge 0, \forall i \in N, i \ne s, s \in S. \end{aligned} \quad (8c)$$

Using matrix notation, (8) can be rewritten as:

$$\min_{x \in \Re^{(|A|+|N|-1)\cdot|S|}, y \in \Re^{|A|}} f(x, y) \quad (9a)$$

subject to

$$l \le y \le u, \quad (9b)$$

$$0 \le x \perp F(x, y) \ge 0, \quad (9c)$$

where $f(x,y)$ is defined as

$$f(x, y) = t(\sum_{s\in S} v^s, y)]^T (\sum_{s\in S} v^s) + \theta e^T g(y). \quad (9d)$$

Obviously, the MPCC based CNDP model (9) is defined on the upper level decision variable y and the lower level variable x. Equation (9d) 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; (9b) is the bound constraints for the upper level decision variable; and (9c) is the lower level NCP formulation that x must satisfy. In this paper, we simply assume the weighing factor θ in (9d) is given. However, we recognize that determining a proper value of θ in practice may not be trivial. Extensive research has been conducted in the regard, aiming to determine the "value of time" (e.g. Lam and Small, 2001).

SOLUTION ALGORITHM – THE NLPEC SOLVER

As mentioned above, the advantage of MPCC over MPEC is that the lower level NCP can be easily converted into certain equivalent conditions (constraints). This way, the bilevel MPCC can be reformulated and solved as a single level NLP. Various techniques are available to perform the reformulation. In this paper, to facilitate our discussion, we adopt the NLPEC solver developed by Ferris et al. (2005). NLPEC can automatically convert an MPCC to an equivalent single level NLP which can be further solved using existing NLP solvers in GAMS (Brooke et al., 1998). A distinct feature of NLPEC is that it provides a collection of options, ranging from different NCP reformulation types and options, relaxation settings, to various NLP solvers available in GAMS. This is particularly useful to numerical studies to investigate the best settings for solving CNDP. As a specific example, we give the formulation of the single level NLP by reformulating the complementarity condition using the so-called *product* reformulation (Ferris, 2004).

$$\min_{x\in\Re^{(|A|+|N|-1)\cdot|S|},\, y\in\Re^{|A|}} f(x,y) \qquad (10a)$$

subject to

$$l \le y \le u \qquad (10b)$$

$$x_i \ge 0, i = 1,\cdots,(|A|+|N|-1)\cdot|S|, \qquad (10c)$$

$$F_i(x,y) \ge 0, i = 1,\cdots,(|A|+|N|-1)\cdot|S|, \qquad (10d)$$

$$x_i F_i(x,y) \le 0, i = 1,\cdots,(|A|+|N|-1)\cdot|S|. \qquad (10e)$$

Clearly, the single level NLP model (10) involves only continuously differentiable functions with respect to *x* and *y*. Hence, it is a smooth optimization problem. However, the MFCQ (Mangasarian Fromovitz Constraint Qualification) does not hold because of the complementarity slackness constraints (10e) (Luo et al., 1996). Therefore, solving the single level NLP directly is typically difficult, and NLPEC adopts a relaxation scheme to iteratively solve (10). Using the product reformulation (10) as an example, the scheme relaxes (10e) by the following condition.

$$x_i F_i(x,y) \le \mu, \forall i = 1,2,\cdots,(|A|+|N|-1)\cdot|S|, \qquad (11)$$

where $\mu > 0$ is the relaxation parameter. By progressively reducing the value of μ, the single level NLP can be finally solved by replacing (10e) using (11). This iterative algorithm can be illustrated as follows.

Step 1 Initialization. Choose an initial relaxation parameter $\mu^0 > 0$. Set iteration limit *M*, update factor $0 < \lambda < 1$, and $k = 0$.

Step 2 Major Iteration.

Step 2.1 Solve current relaxed single level NLP (10a) – (10d) and (11) using the solution from last iteration as the starting point. Use μ^k as the relaxation parameter in (11).

Step 2.2 Update and Move. If $k \le M$, set $\mu^{(k+1)} = \lambda\mu^k$, $k = k+1$ and go to Step 2.1; go to Step 3.

Step 3 Final Solve. Set the final relaxation parameter as μ^f which is a pre-defined value. If it is successful, we obtain an optimal solution for the CNDP problem; otherwise, an approximate solution is achieved from the last run of Step 2.2.

For a selected 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. Here we only illustrate the relaxation solution procedure using the product reformulation. For all other reformulation types and options, the solution process is quite similar. We refer to Ferris (2004) for more details. In the next section, we will discuss in detail the options that are most relevant for solving the MPCC model.

NUMERICAL EXPERIMENTS

There are a number of ways to convert and solve a bilevel MPCC-based CNDP due to plentiful possibilities to reformulate the lower level NCP, various available NLP solvers, and different relaxation settings. Since one objective of our study is to investigate these myriad conversion and solving options to find the best one(s) for solving CNDPs, we experiment on existing and randomly generated CNDPs for possible option combinations. Note that our analyses are based on option combinations (such as the product formulation with inequality constraint, no aggregate, and no slack variable). In general, we are not interested in the performance of each individual option (such as with/without slack variables) because of the interaction effects of different options. However, from the best option combinations identified in later sections, we found that certain options are superior to others in terms of solving CNDPs.

The experiments were designed in two stages. In the first stage, we conducted a preliminary test on the performances of all possible option combinations. We found that certain options are not effective to solve the proposed CNDP model, which were eliminated for further evaluations (Section 4.3). In the second stage, we performed more detailed experiments for the remaining option combinations. We found that such a two-stage method can reduce the computational efforts needed for the experimental study, while still maintain the integrity of the analysis.

We tested the options on 10 networks for both the symmetric and asymmetric cases, resulting in 20 test problems. Two of the ten networks are well-known in the CNDP literature and the other eight are randomly generated (see Section 4.2 for details). The performance measures we consider in this paper include solution quality and efficiency. Solution quality refers to the objective value obtained for each test problem, while solution efficiency is related to the CPU time used to solve the problem. Obviously, the most desirable option combinations are the ones that can generate the smallest objective value with the least CPU time.

Link Travel Time Function

We first provide link travel time functions used for both symmetric and asymmetric cases. For the symmetric case, we use separable travel time functions, implying the travel time on a particular link only depends on its own traffic flow. In the transportation area, the most popularly used functional form for the separable case is the BPR (Bureau of Public Roads) function:

$$t_{ij}(v_{ij}^A, y_{ij}) = A_{ij} + B_{ij}(v_{ij}^A/(K_{ij}+y_{ij}))^4, \quad (12)$$

where A_{ij}, B_{ij}, and K_{ij} are constants for a given link (i,j).

For the asymmetric case, however, no appropriate functional form has been suggested in the literature. In this study, we simply adopt the following expression:

$$t_{ij}(v^A, y_{ij}) = A_{ij} + B_{ij}[(\sum_{\substack{(i_1,j_1)\in A,\\ j=j_1 \text{ or } j=i_1}} \rho_{ij,i_1j_1} v_{i_1j_1}^A)/(K_{ij}+y_{ij})]^4. \quad (13)$$

Here $0 \le \rho_{ij,i_1j_1} \le 1$ denotes the "impact factor" of the flow on link (i_1, j_1) to the travel cost of link (i, j) for every link (i_1, j_1) that has node j as its starting or ending node. Apparently, we have $\rho_{ij,ij} = 1, \forall (i,j) \in A$. If $\rho_{ij,i_1j_1} = 0$, $\forall (i,j) \in A, (i_1, j_1) \in A, j = j_1$, or $j = i_1, i \ne i_1$ equation (13) will reduce to the standard BPR function in (12). Further, according to the definition of matrix Q in equation (5), one can see that

$$Q_{ij,i_1j_1} = \partial t_{ij}/\partial v_{i_1j_1}^A = \frac{4B_{ij}\rho_{ij,i_1j_1}}{(K_{ij}+y_{ij})^4}\left[\sum_{\substack{(i',j')\in A,\\ j=j' \text{ or } j=i'}} \rho_{ij,i'j'} v_{i'j'}^A\right]^3. \quad (14)$$

If we look at a particular row ij of matrix Q, its diagonal element $\frac{4B_{ij}}{(K_{ij}+y_{ij})^4}\left[\sum_{\substack{(i',j')\in A,\\ j=j' \text{ or } j=i'}} \rho_{ij,i'j'} v_{i'j'}^A\right]^3 \ge 0$ will be fixed.

Therefore, if $\sum_{\substack{(i',j')\in A,\\ j=j' \text{ or } j=i', i'\neq i}} \rho_{ij,i'j'} < 1$, matrix Q is diagonal dominant, implying that Q is positive semi-definite (so is the Jacobian matrix of the NCP model (2) as shown in equation (5)). This will likely happen if the levles of link interactions among different links are small; as the values of link interactions increase, Q will tend to be not positive semi-definite. Section 4.6 discusses in more detail how the link interactions impact the solution process of the CNDP model proposed in this paper using the SF network as an example.

We should point out that equation (13) is just an intuitive way to achieve (asymmetric) link interactions among adjacent links to illustrate our proposed model and algorithm. How to design a practically reasonable asymmetric link cost function for a given network is out of the scope of this paper. Note that we use $\rho_{ij,i_1j_1} = 0.15, \quad \forall (i,j) \in A, \forall (i_1, j_1) \in A,$ $j = j_1$ or $j = i_1, i \neq i_1$ up to Section 4.5.

Test Networks

Discussions of this paper are based on two well-tested networks in the CNDP literature (Suwansirikul et al., 1987; Friesz, 1992; Meng et al., 2001) and eight randomly generated grid networks. The first one, denoted as **SN** (short for "six-node"), is a simple network with 6 nodes, 16 links, and 2 origin-destination (OD) pairs. The configuration and data of the network (e.g., the values of the weighing factor θ) can be found in Suwansirikul et al. (1987). Note that three demand levels were originally proposed, and only the third level (i.e. the one with the highest demand) is tested in this paper. The cost function of capacity enhancement is given as $g_{ij}(y_{ij}) = y_{ij}, \forall (i,j) \in A$. In other words, the upper level objective function of the CNDP is

$$f(x,y) = \sum_{(i,j)\in A} [t_{ij}(\sum_{s\in S} v^s, y_{ij}) \cdot \sum_{s\in S} v^s_{ij} + \theta_{ij} y_{ij}]. \quad (15)$$

The second test network, denoted as **SF**, is the Sioux-Falls network (Adbulaal and LeBlance, 1979). It contains 24 nodes, 76 links, and 528 OD pairs. Ten links were chosen for capacity enhancement (Suwansirikul et al., 1987). The cost function for capacity enhancement of SF is $g_{ij}(y_{ij}) = 0.001 y_{ij}^2, \forall (i,j) \in A$. In other words, the upper level objective function of the CNDP is

$$f(x,y) = \sum_{(i,j)\in A} [t_{ij}(\sum_{s\in S} v^s, y_{ij}) \cdot \sum_{s\in S} v^s_{ij} + 0.001\theta_{ij} y_{ij}^2]. \quad (16)$$

The eight random networks are denoted as *n*/*m*/*l*/*k*, where *n* denotes the number of nodes, *m* the number of links, *l* the number of OD pairs, and *k* the number of links whose capacities are to be improved. Except these four parameters, most data of the random networks are generated purely randomly, including existing link capacities, OD demands, and which links to enhance, etc. The average volume/capacity ratio is set as 0.8 to represent a relatively congested traffic pattern. The reason of using random grid networks is that they are usually more difficult to solve compared with other types of networks. The cost function for capacity enhancement for all random networks is $g_{ij}(y_{ij}) = 0.01 y_{ij}^2, \forall (i,j) \in A$, implying the upper level objective function of CNDP is

$$f(x,y) = \sum_{(i,j)\in A} [t_{ij}(\sum_{s\in S} v^s, y_{ij}) \cdot \sum_{s\in S} v^s_{ij} + 0.01\theta_{ij} y_{ij}^2]. \quad (17)$$

In summary, we have in total 20 test problems, including both symmetric and asymmetric cases for each of the ten test networks.

Reformulation Options

Different reformulations for rewriting the bilevel MPCC to a single level NLP are given in this section, based on whether the initial MCP solve is performed, available NLP solvers in GAMS, various reformulation types and options, and different relaxation settings.

Initial MCP Solve

The initial MCP solve implies that before running the NLPEC solver, we will temporarily fix the value of the upper level decision variable (i.e., variable *y* in equation (12)). This will result in an MCP problem that can be solved in GAMS (by default the PATH solver is used). The advantage for such an initial MCP solve is that for a properly chosen value of *y* (e.g., close to the final solution of *y*), the initial solve may provide a good starting point for NLPEC; the latter may then converge to an optimal solution relatively easier and faster. To test the significance of the initial MCP solve, we run two options: with and without the initial solve. Three alternatives for fixed *y* values were tested: $y=l$, $y=u$, and $y=(l+u)/2$. Here *l* and *u* denote the lower and upper bounds of *y*, respectively. No significant difference was observed among these three alternatives. Therefore, we use $y=(l+u)/2$ for cases with the initial MCP solve.

NLP Solvers

We tested five NLP solvers that are currently available in GAMS, namely CONOPT, MINOS, SNOPT, KNITRO, and IPOPT. Based on our test experience, MINOS, KNITRO, and IPOPT are not as effective for solving CNDPs. Therefore, we only test CONOPT and SNOPT in this paper. A particular option file can be set for each solver to maximize its performance. In our study, we use the default options for each solver except that the convergence tolerance is set as 10^{-6}.

Reformulation Types

Several reformulation types are available in NLPEC, including the product (i.e., *mult*), NCP functions, and penalty reformulations (Ferris et al., 2005). We found that none of the NCP function reformulations are effective for solving an MPCC-based CNDP. In this study, we only test the product and penalty reformulations.

The product reformulation is a popularly adopted technique and was formally defined in (10). The reformulation in (10) has inequality constraints, no aggregation, and no slack variable, as will be further described in Section 4.3.4. Another type is the penalty reformulation. For our specific MPCC model (9), the penalty reformulation can be expressed as follows.

$$\min_{x\in\Re^{(|A|+|N|-1)\cdot|S|},\,y\in\Re^{|A|}} f(x,y)+\frac{1}{\mu}x^T F(x,y), \qquad (18a)$$

subject to

$$l \le y \le u, \qquad (18b)$$

$$x \ge 0, \qquad (18c)$$

$$F(x,y) \ge 0. \qquad (18d)$$

In the penalty reformulation (18), the theoretically problematic constraint (10e) is moved to the objective function, and $1/\mu$ is the (positive) penalty parameter. It can be shown under certain conditions that as $1/\mu \to +\infty$ (i.e., $\mu \to 0$), the solution of (18) also solves (10).

Reformulation Options

Besides the different reformulation types discussed in previous section, there are various options available to convert a bilevel problem to a single level one. These options are related to whether aggregation is performed, the types of constraints, and if slack variables are used for the defining function of the lower level NCP (see also Ferris et al., 2005).

We use the product reformulation to illustrate these options. First the **aggregate** option defines whether we express the complementarity conditions (10e) component-wise or as a single aggregated constraint. No aggregation (with equality constraints and no slack variable) results in the following reformulation.

$$x_i \geq 0, \forall i = 1,2,\cdots,n$$
$$F_i(x,y) \geq 0, \forall i = 1,2,\cdots,n \qquad (19a)$$
$$F_i(x,y)\cdot x_i = 0, \forall i = 1,2,\cdots,n.$$

The full aggregation option can be represented as (with equality constraints and no slack variable):

$$x \geq 0,$$
$$F(x,y) \geq 0, \qquad (19b)$$
$$F(x,y)^T x = 0.$$

Clearly, (19a) and (19b) are equivalent to each other in terms of ensuring the complementarity between x and $F(x,y)$ but result in different NLP's to solve which may prove advantageous to particular solvers.

The **constraint** option implies whether equalities or inequalities are used to represent the complementarity condition. Firstly (19b) illustrates the option of equality constraints with aggregation and no slack variable, whereas inequality constraints can be used as in (20) with aggregation and no slack variable:

$$x \geq 0,$$
$$F(x,y) \geq 0, \qquad (20)$$
$$F(x,y)^T x \leq 0.$$

Finally, the **slack** option represents whether a slack variable is used for the defining function of the lower level NCP. All options we discussed so far have no slack variable. For positive slack variables, we can use the following expression

$$\omega = F(x,y)$$
$$x \geq 0, \qquad (21)$$
$$\omega \geq 0,$$
$$\omega^T x = 0.$$

In (21), a slack variable ω is defined to represent the function $F(x,y)$ of the NCP. This problem has extra variables and constraints, but the form of the constraints is simpler, and typically reduces the number of times that F is evaluated by a solver. Note that we use equality constrains and full aggregation in (21).

For the specific CNDP model (9), there are eight combinations for the product reformulation by considering the two possibilities for each of the **constraint**, **aggregate** and **slack** options. For the penalty reformulation, there are only two options depending on whether slack variables are used or not. Therefore, in total we have ten different reformulations for a given CNDP problem.

Relaxation Settings

The relaxation settings turn out to be crucial for solving an MPCC model. The initial and final relaxation parameters (μ^0 and μ^f), number of NLP solves (M), and update factor (λ) need to be determined before each NLPEC solve. First, μ^f represents the required solution accuracy and thus should be input by users. In this paper, we set $\mu^f = 10^{-6}$. The initial μ^0 may be problem-specific and we test on two values $\mu^0 = 1, 10$. In Section 4.5, we will propose a scheme that can automatically determine μ^0. We further test on three possible values of the update factor: $\lambda = 0.1, 0.3, 0.6$. To ensure a smooth reduction of the relaxation parameter for the algorithm in Section 3, we set M as $\mu^0 \cdot \lambda^{M+2} = \mu^f$. That is,

$$M = \left[\log \frac{\mu^f}{\mu^0} \Big/ \log \lambda \right] - 2 \,. \qquad (22)$$

Here $[x]$ denotes the nearest integer that is greater than x.

The above analysis results in six (6) relaxation settings. We further test on a direct solve (i.e., no relaxation). Hence, seven groups of settings are tested, denoted as **R0** – **R6** respectively, as shown in Table 1. Note that **R0** represents a direct solve since μ^0 is set as 0.

Table 1 Relaxation Settings

	R0	R1	R2	R3	R4	R5	R6
initMu μ^0	**0**	1	1	1	10	10	10
finalMu μ^f	/	1.0e-6	1.0e-6	1.0e-6	1.0e-6	1.0e-6	1.0e-6
updateFac λ	/	0.1	0.3	0.6	0.1	0.3	0.6

To sum up, we have totally 2×2×10×7=280 options for each of the 20 test problems. Or equivalently, we are going to run 280×20 = 5600 NLPEC solves.

Result Analysis

We tested the relaxation algorithm with aforementioned option combinations on an Intel Pentium-4 PC with CPU 2.8 GHZ and 2GB memory. The best objective value and the median CPU time that produces the best solution are first listed for each problem in Table 2. For each problem (one row in Table 2), there may be multiple option combinations that can solve the problem to the best objective value. The median value of CPU times of all these option combinations is defined as the "Median CPU Time for Best Objective Value" for each problem, listed in the last column of Table 2. Since CNDP is normally conducted in the planning stage and computational efficiency is not as critical as the solution quality, the median CPU time should provide a reasonable estimate of the difficulty to solve each test problem. The table also shows that, on a given network, the asymmetric case is usually more difficult to solve than its symmetric counterpart since both the objective value and CPU time are larger for the former case.

Table 2 Best Objective Values and Median CPU Times

Network	Symmetry	Best Objective Value	Median CPU Time (Seconds) for Best Objective Value
SN	Symmetric	522.6439	0.342
SN	Asymmetric	662.1176	0.61
SF	Symmetric	80.5148	166.89
SF	Asymmetric	644.0102	1089.359
9/24/12/10	Symmetric	27563.8495	1.1485
9/24/12/10	Asymmetric	72398.8151	2.12
16/48/56/10	Symmetric	317339.5871	21.886
16/48/56/10	Asymmetric	1363608.5066	64.744
25/80/20/10	Symmetric	1761932.2960	29.9565
25/80/20/10	Asymmetric	4449762.7638	71.191
36/120/42/10	Symmetric	739422.6182	211.1675
36/120/42/10	Asymmetric	1832088.5308	300.297
6/14/6/10	Symmetric	6494.8869	0.3755
6/14/6/10	Asymmetric	18812.2567	0.604
12/34/30/10	Symmetric	109503.2433	5.4135
12/34/30/10	Asymmetric	565790.6267	11.8145
20/62/56/10	Symmetric	192486.3159	43.485
20/62/56/10	Asymmetric	824430.3857	84.475
30/98/56/10	Symmetric	241948.9115	188.735
30/98/56/10	Asymmetric	1372222.0190	515.894

To better illustrate and compare the performances of different option combinations based on objective values, we summarize, for each option combination, the total number of problems that can be solved with objective values no higher than a certain percentage over the best values. This number ranges from 0 to 20, with a larger number indicating a better performance. The percentages tested are from 0% – 10%. Table 3 below depicts the top 10 option combinations. Table 4 shows the number of problems each option can successfully solve according to different percentages. Clearly, the first option has the best performance, i.e., it can solve 19 out 20 problems to the best solutions and all problems to within 1% of the best solutions obtained.

Table 3 Top 10 Option Combinations Based on Objective Values

Option	Solver	initSolve	μ^0	μ^f	λ	refType	aggregate	constraint	slack
1	CONOPT	TRUE	10	.000001	.6	mult	none	inequality	positive

2	CONOPT	TRUE	10	.000001	.1	mult	none	inequality	positive
3	CONOPT	TRUE	10	.000001	.6	penalty	-	-	positive
4	CONOPT	TRUE	10	.000001	.1	penalty	-	-	positive
5	CONOPT	TRUE	10	.000001	.6	mult	none	inequality	none
6	CONOPT	TRUE	10	.000001	.3	mult	none	inequality	none
7	CONOPT	FALSE	10	.000001	.6	mult	none	inequality	positive
8	CONOPT	TRUE	10	.000001	.3	mult	none	inequality	positive
9	CONOPT	TRUE	1	.000001	.1	mult	none	inequality	positive
10	CONOPT	TRUE	1	.000001	.3	mult	none	inequality	positive

Table 4 Number of Successful Solves for the Top 10 Option Combinations Based on Objective Values

Option	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
1	19	20	20	20	20	20	20	20	20	20	20
2	18	19	19	19	19	19	20	20	20	20	20
3	18	19	19	19	19	19	19	20	20	20	20
4	18	19	19	19	19	19	19	20	20	20	20
5	18	19	19	19	19	19	19	19	19	19	19
6	18	19	19	19	19	19	19	19	19	19	19
7	18	18	18	18	18	18	18	18	18	18	18
8	17	18	19	19	19	19	19	19	19	19	19
9	17	18	18	18	18	18	18	19	19	19	19
10	17	18	18	18	18	18	18	19	19	19	19

To study the computational efficiency of different options, we list in Table 5 the top 10 option combinations that are the most computationally efficient. The criterion is the number of problems that can be successively solved with objective values no higher than 10% of the best values, provided the CPU times are no higher than 10% of the Median CPU times in Table 2.

Table 5 Top 10 Option Combinations Based on CPU Times

Option	Solver	initSolve	μ^0	μ^f	λ	refType	aggregate	constraint	slack
1	CONOPT	TRUE	10	.000001	.3	penalty	-	-	positive
2	CONOPT	TRUE	10	.000001	.1	penalty	-	-	positive
3	CONOPT	TRUE	1	.000001	.3	penalty	-	-	positive
4	CONOPT	TRUE	1	.000001	.1	penalty	-	-	positive
5	CONOPT	TRUE	1	.000001	.6	penalty	-	-	positive
6	CONOPT	TRUE	1	.000001	.1	mult	none	inequality	positive
7	CONOPT	TRUE	10	.000001	.1	mult	none	inequality	positive
8	CONOPT	TRUE	10	.000001	.6	penalty	-	-	positive
9	SNOPT	TRUE	10	.000001	.1	mult	full	inequality	positive
10	SNOPT	TRUE	10	.000001	.1	mult	none	inequality	none

From Table 3 and 5, we can observe that:

- The initial MCP solve is desirable for the later solution process, implying that it does provide a good starting point;
- For different problems, the optimal relaxation settings may vary, however, a relatively large update factor (>=0.3) is preferable for producing the best objective values as shown in Table 3. On the other hand, for computational efficiency, a smaller update factor (0.1) may be more desirable as shown in Table 5;
- The product reformulation is more able to produce better quality solutions, while the penalty reformulation is more (time) efficient for solving CNDPs;
- For the product reformulation, it is more effective to use inequality constraints and not to aggregate the complementarity constraints;
- Setting the slack option to *positive* is preferable for the product and penalty reformulations and from the

considerations of both solution quality and computational efficiency;

- CONOPT is more effective to solve MPCC based CNDPs.

Ralph and Wright (2004) proved that the product reformulation with no aggregation and inequality constraints requires weaker conditions for the relaxation algorithm to converge to a local solution, while the penalty reformulation needs a sufficiently large penalty parameter. The recommended reformulation types and options for CNDP in our study, especially items (c) – (e) above, amplify the previous theoretical study by Ralph and Wright (2004).

We also note from the above discussions that the recommendations for solution quality are sometimes not consistent with those for solution efficiency. Since solution quality is considered as the primary criterion in our study, we focus on those option combinations that are more desirable from solution quality point of view. In Section 4.5 and 4.6, therefore, we will use the product reformulation to test an automatic scheme for determining relaxation settings and to study the impact of link interactions, respectively.

The symmetric cases of SN and SF have been well studied in the transportation literature. In Ban et al. (2006a), we showed that for SN the product reformulation can generate a solution with an objective value significantly smaller than the previously known best one. Similarly for SF, the product reformulation can produce a solution with the objective value slightly smaller than the previously known best one. Details can be found in Ban et al. (2006).

Automatic Determination of Relaxation Settings

The relaxation settings are crucial for the relaxation algorithm. In previous testing, we simply used a group of fixed settings for comparison purposes. Ban et al. (2006a) provided some general findings on how to choose μ^0, μ^f, λ, and *M*. It seems that a good setting is very likely problem-specific, and it is thus desirable to devise certain scheme which can automatically determine the relaxation setting for solving a given MPCC based CNDP.

As aforementioned, $\lambda \geq 0.3$ is preferable in order to ensure the smoothness of the relaxation process. We thus choose λ=0.3 for the automatic scheme in this paper. On the other hand, μ^f represents the required solution accuracy and thus should be input by users. If we further know μ^0, *M* can be computed as in equation (22).

Hence, the problem now reduces to how to choose a proper initial relaxation parameter μ^0. This can be determined by the initial MCP solve. In this paper, we develop a simple and intuitive method. First, we assign initial values for *x*, and *y* in our MPCC model (9) as $\bar{x}$ and $\bar{y}$ respectively. We can then compute the initial violation of complementarity as: $\bar{\mu} = \sqrt{\bar{x}^T F(\bar{x}, \bar{y}) / n}$. Here *n* is the dimension of *x*. Next, to solve the initial MCP, we fix *y* as $\bar{y}$ and set the initial value of *x* as $\bar{x}$. Further set the convergence tolerance of MCP solver as $\delta_1 \bar{\mu}$, where $0 < \delta_1 << 1$. If the initial MCP can be successfully solved, the initial relaxation parameter μ^0 can be set as $\delta_1 \bar{\mu} / \delta_2$, where $0 < \delta_2 < 1$. In GAMS codes, we can then output values of the relaxation setting, together with other recommended options, to the option file of NLPEC.

The above idea is implemented using the 20 test problems in this paper. In particular, we experiment on the product reformulation with no aggregation, inequality constraints, and positive slack variables. We further set μ^f =10^{-6}, $\delta_1 = 0.01$, and $\delta_2 = 0.005$ (also λ =0.3). It turns out that the proposed automatic scheme can solve all the 20 problems to optimality. Table 6 summarizes the objective value and CPU time for each problem generated by the automatic scheme.

Clearly, the optimal solutions found are almost exactly the same as the corresponding best solutions (to the level of 10^{-4}). The CPU times, nevertheless, may be (and in some cases significantly) higher or lower than the median times for producing the best solution in Table 2. Since solution quality is our primary objective, this table shows that the automatic scheme works well in terms of solving the MPCC-based CNDPs.

Table 6 Results by the Automatic Scheme and H0 Heuristics

Network	Symmetry	Auto Objective Value	Auto CPU Time (seconds)	H0 Objective Value	H0 CPU Time (seconds)	Improvement of Auto (%) in Obj. Value
SN	Symmetric	522.6439	0.499	527.7916	0.0462	0.9849
SN	Asymmetric	662.1176	0.668	681.3049	0.0466	2.8979
SF	Symmetric	80.5148	665.289	80.86526	1.0616	0.4353
SF	Asymmetric	644.0102	2310.718	651.2517	8.0938	1.1244
9/24/12/10	Symmetric	27563.8495	2.032	27563.85	0.0462	0.0000
9/24/12/10	Asymmetric	72398.8151	3.702	72398.91	0.0466	0.0001
16/48/56/10	Symmetric	317408.984	30.062	317409	0.0929	0.0219
16/48/56/10	Asymmetric	1363610.11	71.261	1375931	0.2804	0.9037
25/80/20/10	Symmetric	1762110.051	19.939	1762110	0.1555	0.0101
25/80/20/10	Asymmetric	4449746.369	108.424	4449919	0.1869	0.0035
36/120/42/10	Symmetric	739489.0156	106.878	739544.8	0.2491	0.0165
36/120/42/10	Asymmetric	1834100.637	497.367	1834267	0.6245	0.1189
6/14/6/10	Symmetric	6494.8869	0.386	6494.887	0.0462	0.0000
6/14/6/10	Asymmetric	18812.2638	0.709	18813.07	0.0623	0.0043
12/34/30/10	Symmetric	109492.6838	4.39	109503.3	0.0772	0.0000
12/34/30/10	Asymmetric	565790.7532	23.17	566059.5	0.0936	0.0475
20/62/56/10	Symmetric	192507.17	22.307	192510.3	0.1089	0.0125
20/62/56/10	Asymmetric	824430.5889	99.191	825402.8	0.2969	0.1180
30/98/56/10	Symmetric	241929.7959	110.089	241950.3	0.2185	0.0006
30/98/56/10	Asymmetric	1372418.994	421.708	1372903	0.7496	0.0496

Marcotte (1986) developed six heuristics for solving the bilevel CNDP model. In particular, heuristics H0 in Marcotte (1986) determines the optimal capacity by solving a system optimal problem. We implemented such a method in this paper. The system optimal problem was solved by first converting it into a UE problem (see Sheffi, 1985). The UE problem can then be solved using a decomposition scheme based on individual origins (Ban et al., 2006c).

Table 6 also presents the comparison of performances of the automatic scheme vs. the H0 heuristics using the 20 test problems. We can see that the objective values obtained from the automatic scheme are always slightly superior to the H0 heuristic method; however, the running time of the H0 heuristic method is smaller because the H0 method only requires to solve a SO problem. This indicates a trade-off between the solution quality and efficiency. One should also note that in theory the worst case of the H0 method could produce an objective value that is at least one time larger than the optimal objective value (Marcotte, 1986).

Discussion of Link Interactions

Impact factors are introduced in this paper to represent the level of interactions among adjacent links when asymmetric user equilibrium is considered. In this section, we study how the values of the impact factors may influence the solution quality and efficiency of CNDP. The test is done on the SF network by setting $\rho_{ij,i_1 j_1} = \zeta$,

$\forall (i,j) \in A, (i_1, j_1) \in A, j = j_1$ or $j = i_1, i \neq i_1$
with ζ varying from 0 to 0.3 using 0.01 as the increment. The options used for solving these problems are the same as those in Section 4.5.

Figure 1 depicts the objective values and CPU times as ζ changes from 0.01 to 0.3. Note that we also display the objective value and CPU time for the symmetric case (with ζ =0). First, the optimal objective values increase monotonically as so does ζ. This is intuitive since larger ζ 's imply more link interactions and thus longer link travel times.

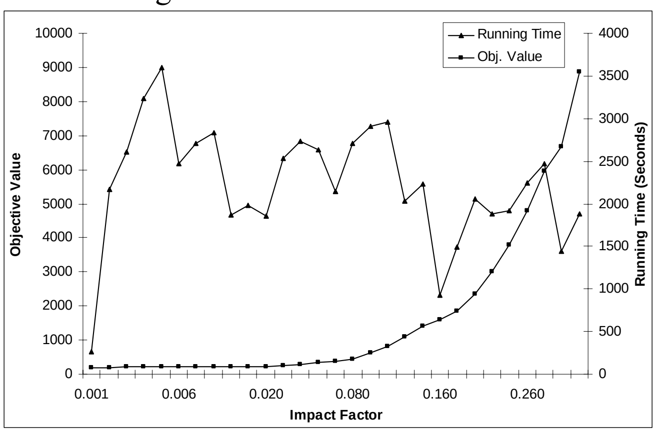

Figure 1 The Impact of Link Interactions – Objective Value & CPU Time

For CPU times, we observe that asymmetric cases (those with ζ >0) require significantly larger CPU times than the symmetric case (the one with ζ =0). This confirms that a CNDP with AUE is "harder" to solver than its symmetric counterpart. In addition, as ζ (i.e., the asymmetry) increases from 0.01 to 0.05, the CUP time increases dramatically. It then starts to fluctuate. In fact, except ζ =0.05 and 0.15, the CPU times for other cases are relatively stable. This means that the CPU time is not sensitive to the level of link interactions when $0 < \zeta \leq 0.3$. However, if we further increase ζ to be more than 0.3, the relaxation scheme tends to fail to solve the CNDP model. This also illustrates that as link interaction increases, the CNDP model becomes harder to solve.

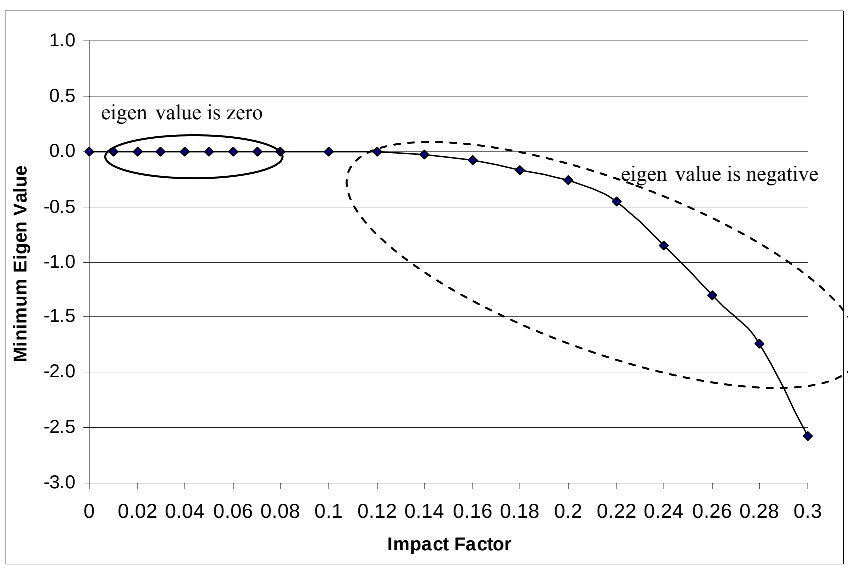

Figure 2 The Impact of Link Interactions – Minimum Eigen Value

Figure 2 shows the minimum eigen values of matrix Q at the optimal solutions for different impact factors. The figure shows that if ζ <=0.06, the minimum eigen value is zero; the minimum eigen value becomes negative if ζ >0.06. Thus Figure 2 illustrate that as the impact factors increase, matrix Q (at the optimal solution) becomes from positive semi-definite to non positive semi-define.

CONCLUSIONS

In this paper, we extensively evaluated the impacts of different reformulation techniques, NLP solvers, relaxation settings, and other options for solving CNDP. The tests were conducted on both well-known CNDP examples in the transportation literature and some randomly generated grid networks. It turned out that the product reformulation (with no aggregation, inequality constraints, and positive slack variables) works the best for CNDP in terms of solution quality, while the penalty reformulation (with positive slack variables) is more desirable for computational efficiency. We further developed an automatic scheme to determine the relaxation setting for a given CNDP, based on the initial MCP solve. It was shown using the 20 test problems that this scheme is effective. We also tested the impact of link interactions using the SF network and found that as the link interactions increase, so does the objective value while the CPU times are rather stable.

Note that above conclusions are made based on the 20 small size test problems only. The findings need to be verified on large size networks. There is however no well-established test problems in the transportation field for CNDP (the largest test example in the most recent study by Josefsson and Patriksson (2008) is the Sioux-Falls network). Therefore, our next step will be construction of large scale CNDP test problems. Meanwhile, due to the fact that the final single level model is constructed using disaggregated variables, its dimension could be large, especially for large-scale “many origins to many destinations” problems. However, the model has a very special structure which allows us to deploy certain decomposition schemes that have already been adopted for large-scale static traffic assignment problems (Ban, 2005). We have recently reported some preliminary results by applying the Gauss-Seidel decomposition scheme for solving the proposed MPCC model (Ban et al., 2006b). The study showed that the product reformulation with no aggregation and inequality constraints, as recommended in this paper, can be used in the decomposition scheme.

CNDP in the transportation area represents the most basic form of the so-called network design problem that has a variety of applications such as optimal toll pricing (Lawphongpanich and Hearn, 2004), origin-destination demand estimation (Yang et al., 1992), optimal traffic control strategies (Yang and Yager, 1994), etc. Although the practicality of CNDP itself has gradually faded as major capacity enhancement is less frequent in today’s maturing transportation systems, the advancement of theoretical understanding or algorithm development of CNDP is expected to provide insights for this and other network design problems and applications. In particular, the proposed MPCC model provides a general framework which may be used to study other types of network design problems, especially for those that need to consider AUE. We recently applied the reformulation types and options recommended in this paper to dynamic congestion pricing problems (Ban and Liu, 2009). Research in this direction will be further pursued by the authors.

ACKNOWLEDGEMENT

The authors would like to thank the four anonymous reviewers for their insightful comments that help improve an earlier version of the paper. The first author thanks Dr. David Boyce from Northwestern University for his helpful comments on an earlier version of the paper.

REFERENCES

Abdulaal, M., & LeBlanc, L.J. (1979). Continuous equilibrium network design models. *Transportation Research Part B*, 13, 19-32.

Ahuja, R.K., Magnani, T.L., & Orlin, J. (1993). *Network Flows*, Prentice Hall.

Anitescu, M. (2004). *Global convergence of an elastic mode approach for a class of mathematical programs with complementarity constraints*. Preprint ANL/MCS-P1143-0404, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL, USA.

Aashtiani, H.Z., & Magnanti, T.L. (1981). Equilibria on a congested transportation network. *SIAM,* 2(3), 213-226.

Ban, X. (2005). *Quasi-Variational Inequality Formulations and Solution Approaches for Dynamic User Equilibria,* Ph.D. Thesis, University of Wisconsin-Madison.

Ban, X., Liu, H., Ferris, M.C., & Ran, B. (2006a). A general MPCC model and its solution algorithm for continuous network design problem. *Mathematical and Computer Modeling,* 43(5-6), 493-505.

Ban, X., Liu, H., Lu, J., & Ferris, M.C. (2006b). A decomposition scheme for continuous network design problem with asymmetric user equilibrium. *Transportation Research Record,* 1964, 185-192.

Ban, X., Liu, H, & Ferris, M.C. (2006c). A link-node based complementarity model and its solution algorithm for asymmetric user equilibria. In *Proceedings of the 85th Transportation Research Board Annual Meeting (CD-ROM).*

Ban, X., & Liu, H. (2009). A link-node discrete-time dynamic second best toll pricing model with a relaxation solution algorithm. *Networks and Spatial Economics, in Press*.

Beckmann, M., McGuire, C. B., & Winsten, C. B. (1956). *Studies in the economics of transportation*. Yale University Press, New Haven, Connecticut; also published as Rand-RM-1488-PR, Rand Corporation, Santa Monica, CA, May 12, 1955.

Brooke, A., Kendrick, D., & Meeraus, A., et al. (1998). *GAMS, a user's guide*. GAMS Development Corporation.

Dafermos, S. C. (1968). *Traffic Assignment and Resource Allocation in Transportation Networks*. PhD thesis, Operations Research, Johns Hopkins University, Baltimore, Maryland.

Ferris, M.C., Dirkse, S.P., & Meeraus, A. (2005). Mathematical programs with equilibrium constraints: Automatic reformulation and solution via constrained optimization. In T.J. Kehoe, T.N. Srinivasan, and J.Whalley, editors, *Frontiers in Applied General Equilibrium Modeling*. Cambridge University Press, 67-93.

Ferris, M.C. (2004). *NLPEC Manual*. University of Wisconsin-Madison.

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

Friesz, T.L., Cho, H.-J., & Mehta, N.J., et al. (1992). A simulated annealing approach to the network design problem with variational inequality constraints. *Transportation Science* 26, 18-26.

Hu, X. & Ralph, D. (2002). A note on sensitivity of value functions of mathematical programs with complementarity constraints. *Mathematical Programming*, 93(2), 265-279.

Jiang, H. & Ralph, D. (2000). Smooth SQP methods for mathematical programs with nonlinear complementarity constraints. *SIAM Journal of Optimization*, 10(3), 779-808.

Josefsson, M., & Patrksson, M. (2008). Sensitivity analysis of separable traffic equilibria with application to bilevel optimization in network design. *Transportation Research Part B*, 41(1), 4-31.

Lam, T.C., & Small, K.A. (2001). The value of time and reliability: measurement from a value pricing experiment. *Transportation Research Part E*, 37, 231-251.

Lawphongpanich, S., & Hearn, D. (2004) An MPEC approach to second best toll pricing. *Mathematical Programming*, 33-55.

Luo, Z.-Q., Pang, J.-S., & Ralph, D. (1996). *Mathematical Programs with Equilibrium Constraints*. Cambridge University Press, Cambridge, UK.

Marcotte, P. (1983). Network optimization with continuous control parameters. *Transportation Science* 17, 181-197.

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

Marcotte, P. & Zhu, D.L. (1996). Exact and inexact penalty methods for the generalized bilevel programming problem. *Mathematical Programming* 74, 141-157.

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

Morlok, E., Schofer, J., Pierskalla, W., and et al. (1973). *Development and Application of a Highway Network Design Model*. Final Report: FHWA Contract Number DOT-PH-11, Northwestern University.

Murtagh, M., & Saunders, M. (1982). A projected Lagrangian algorithm and its implementation for sparse nonlinear constraints. *Mathematical Programming Study* 16, 84-117.

Patriksson, M. (1994). *The Traffic Assignment Problem, Models and Methods*. VSP.

Patriksson, M., & Rockafellar, R.T. (2002). A mathematical model and descent algorithm for bilevel traffic management. *Transportation Science* 36, 271-291.

Ralph, D, & Wright, S.J. (2004). Some properties of regularization and penalization schemes for MPECs. *Optimization Methods and Software* 19, 527-556.

Scheel, H., & Scholte, S. (2000). Mathematical programs with complementarity constraints: stationarity, optimality, and sensitivity. *Mathematics of Operations Research*, 22(1), 1-22.

Sheffi, Y. (1985) *Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods*. Prentice-Hall, Inc.

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

Tan, H.N., Gershwin, S.B., & Athana, M. (1979). *Hybrid optimization in urban traffic networks*. Report No. DOT-TSC-RSPA-79-7, Laboratory for Information and Decision System, MIT, Cambridge, MA.

Wardrop, J.G. (1952). Some theoretical aspects of road traffic research. In *Proceedings of the Institution of Civil Engineers*, Part II, 1, 325-378.

Yang, H., Sasaki, T., Iida, Y., & Asakura, Y. (1992). Estimation of origin-destination matrices from link traffic counts on congested networks. *Transportation Research, Part B,* 26, 1-18.

Yang, H., & Yagar, S. (1994). Traffic assignment and traffic control in general freeway-arterial corridor systems. *Transportation Research Part B,* 28, 463-486.

Yang, H. (1997). Sensitivity analysis for the elastic demand network equilibrium problem with applications. *Transportation Research Part B,* 31, 55-70.

Yang, H., & Bell, M.G.H. (2001). Transport bilevel programming problems: recent methodological advances. *Transportation Research Part B*, 35, 1-4.