Computing Wardropian Equilibria in a Complementarity Framework

Michael C. Ferris* Alexander Meeraus† Thomas F. Rutherford‡

May 1998

Abstract

This note considers alternative methods for computing Wadropian (traffic network) equilibria using a multicommodity formulation in nonlinear program and complementarity formats. These methods compute exact equilibria, they are efficient and they can be implemented with standard modeling software.

Key Words. Transportation, multicommodity flows, complementarity problems

*Computer Sciences Department, University of Wisconsin, Madison, Wisconsin 53706, email: ferris@cs.wisc.edu .
The work of this author was based on research supported by the Department of Energy grant DE-FG03-94ER61915, the Air Force Office of Scientific Research grant F49620-94-1-0036 and the National Science Foundation grant CCR-9157632.

†GAMS Corporation, 1217 Potomac Street, N.W., Washington, D.C. 20007, email: alex@gams.com .

‡Department of Economics, The University of Colorado, Boulder, Colorado 80309, email: rutherford@colorado.edu.
The work of this author was based on research supported by the Department of Energy grant DE-FG03-94ER61915.

1 Introduction

The calculation of individually-rational equilibria for traffic networks has a long history. The fundamental assumption of this model is that drivers are fully informed: a driver going from point A to point B follows the fastest available route, taking the decisions of other drivers as given.

In the past, a range of solution methods have been proposed for the Wardropian model. Some of the earlier techniques include multipath assignment (Irwin et al. 1961) and variational inequality methods (Bertsekas and Gafni 1982). More recently, a tatonnement adjustment process has been suggested (Friesz, Bernstein, Mehta, Tobin and Ganjalizadeh 1994). All of these approaches suffer from the need for implicit or explicit enumeration of all paths between origin-destination pairs in the network. Other methods based on nonlinear complementarity (Aashanti and Magnanti, 1981) and optimization (Beckmann et al., 1956) have also been suggested. This paper illustrates how the complementarity and optimization approaches to the static traffic equilibrium problem can be implemented using standard modeling software. Furthermore, we demonstrate how a multicommodity formulation can provide a compact and efficient representation of the model, permitting direct solution with "off-the-shelf" algorithms. Surprisingly, we find that in many cases the larger complementarity model provides a more efficient formulation than either the primal or dual nonlinear program.

2 Multicommodity Formulation

We consider a network that is given by a set of nodes $\mathcal{N}$ and a set of arcs $\mathcal{A}$. We assume that the time cost of traveling along a given arc is a nonlinear (increasing) function of the total flow along that arc. There are two subsets of $\mathcal{N}$ that represent the set of origin nodes $\mathcal{O}$ and destination nodes $\mathcal{D}$ respectively. Associated with each origin-destination pair is a demand that represents the required flow from the origin node to the destination node.

2.1 A mixed complementarity model

We consider a multicommodity flow formulation of the problem in order to avoid explicit or implicit ennumeration of the network paths. We associate a "commodity" with each destination node (or alternatively with each origin node). We have two sets of variables in this formulation, x and t. The variables $x = (x^1, x^2, \ldots, x^k)$ represent the flows of the commodities $1, 2, \ldots, K = |\mathcal{D}|$ over the network with $x^k_{(i,j)}$ denoting the flow of commodity k on arc (i, j). The variables $t = (t^1, t^2, \ldots, t^k)$ are composed of components t^k_i that represent the minimum time to deliver commodity k from node i. Designating the demand at node i for commodity k as d^k_i, the total cost of shipments on arc ij by $c_{(i,j)}()$, and using $\perp$ to represent complementary slackness, the equilibrium conditions may be written as a nonlinear complementarity problem. We have two classes of constraints:

$$\sum_{j:(i,j)\in\mathcal{A}} x^k_{(i,j)} - \sum_{j:(j,i)\in\mathcal{A}} x^k_{(j,i)} = d^k_i \qquad \forall i \in \mathcal{N}, k \in \mathcal{D}, \tag{1}$$

that represents conservation of flow of commodity k at node i and

$$c_{(i,j)}\left(\sum_{l\in\mathcal{D}} x^l_{(i,j)}\right) + t^k_j \geq t^k_i \perp x^k_{(i,j)} \geq 0 \qquad \forall (i,j) \in \mathcal{A}, k \in \mathcal{D}, \tag{2}$$

that ensures that if there is positive flow of commodity k along arc (i, j), then the corresponding time to deliver that commodity is minimized, see (Wardrop 1952). Note that in terms of the standard node-arc incidence matrix A of the network, the constraints (1) can be rewritten as

$$Ax^k = d^k \qquad \forall k \in \mathcal{D}.$$

2.2 Nonlinear programming formulations

In most examples, the cost functions c_a are integrable as C_a for each $a \in \mathcal{A}$, in which case, the equilibrium conditions (1) and (2) are the first order optimality conditions of the nonlinear program

$$\begin{array}{ll} \min_x & \sum_{a\in\mathcal{A}} C_a(\sum_{k\in\mathcal{D}} x^k_a) \\ \text{subject to} & Ax^k = d^k, x^k \geq 0 \qquad \forall k \in \mathcal{D}. \end{array} \tag{3}$$

This formulation of the problem has received considerable attention in the literature, see for example (Steenbrink 1974). Note that it is a convex multicommodity network flow problem (LeBlanc, Morlok and Pierskalla 1975). Recent work has used parallel machines to exploit the network structure further.

For ease of exposition, let us define the flow vector f as

$$f_a = \sum_{k \in \mathcal{D}} x_a^k \qquad \forall a \in \mathcal{A},$$

or more simply by

$$f = \sum_{k \in \mathcal{D}} x^k.$$

Then a reformulation of (3) is given by

$$\begin{array}{ll} \min_{x,f} & \sum_{a \in \mathcal{A}} C_a(f_a) \\ \\ \text{subject to} & \end{array}$$

$$\begin{array}{rcl} \begin{bmatrix} A & & & \\ & A & & \\ & & \ddots & \\ & & & A \end{bmatrix} x & = & \begin{bmatrix} d^1 \\ d^2 \\ \vdots \\ d^k \end{bmatrix} \\ f - \begin{bmatrix} I & I & \cdots & I \end{bmatrix} x & = & 0 \\ x & \geq & 0 \end{array}$$

Note that in many practical instances, the variables f are upper bounded since the arcs have finite capacities.

The first order conditions of this problem are given by

$$\begin{array}{rcll} c_a(f_a) - v_a & = & 0 & \forall a \in \mathcal{A} \\ -A^T t^k + v & \geq & 0 \perp x^k \geq 0 & \forall k \in \mathcal{D} \\ Ax^k & = & d^k & \forall k \in \mathcal{D} \\ f & = & \sum_{k \in \mathcal{D}} x^k & \end{array} \tag{4}$$

By substituting out the first and fourth of these relationships, we recover (1) and (2). However, in many cases, it is further possible to use (4) to recover a dual nonlinear program. If the first relationship of (4) can be inverted to give

$$f_a = h_a(v_a) \tag{5}$$

and the $h_a(\cdot)$ are integrable (as $H_a(\cdot)$), this dual is the following problem

$$\begin{array}{lll} \max_{t,v} & \sum_{k\in\mathcal{D}}(d^k)^T t^k - \sum_{a\in\mathcal{A}} H_a(v_a) & \\ \text{subject to} & -A^T t^k + v \geq 0 & \forall k \in \mathcal{D}. \end{array} \tag{6}$$

3 Implementation and Testing

The modeling language GAMS (Brooke, Kendrick and Meeraus 1988) and its associated solvers allow all of the above formulations to be easily modeled and solved. For the purposes of this paper, we confine our attention to the Sioux Falls network that appeared in (Friesz et al. 1994). In this example, the network has 76 arcs, 24 nodes and d has 528 nonzero entries. The resulting equilibrium problem (4) has dimension 2376, using the first relationship of (4) to eliminate the variables v. The cost functions c_a are given by the following formula:

$$c_a(f_a) = A_a + B_a \left[\frac{f_a}{K_a}\right]^4 \qquad \forall a \in \mathcal{A},$$

for particular data A_a, B_a and K_a.

Note that c_a is integrable so that the primal nonlinear program (3) is well-defined. Furthermore, the functions h_a of (5) are given by

$$h_a(v_a) = K_a \left(\frac{v_a - A_a}{B_a}\right)^{1/4}$$

which are also integrable, making the dual problem well-defined.

We carried out two sets of experiments, the first having fixed demands d_i^k, and the second using elastic demands.

3.1 Fixed demand

The following table gives the solution times in seconds on a 150MHz Pentium processor for the fixed demand case. The equilibrium models are solved using PATH (Dirkse and Ferris 1995), an implementation of a pathsearch algorithm with nonmonotone stabilization originally due to Ralph (1994). The nonlinear programs are all solved using MINOS (Murtagh and Saunders 1983), a reduced gradient quasi-Newton method.

MCP	Primal NLP	Dual NLP
32.7	12.0	80.1

Table 1: Solution times

In these solutions, the complementarity and nonlinear optimization models agreed to five decimals, and the duality gap between the primal NLP and the dual NLP was zero. Other solvers such as SMOOTH (Chen and Mangasarian 1996) and CONOPT (Drud 1985) also solve the problems without any difficulties, although they take somewhat more time. Note that the primal nonlinear program is a convex nonlinear multicommodity network flow problem for which specialized algorithms will perform even better than MINOS.

3.2 Elastic demand

All of the above is for the case of fixed demands, that is d_i^k are constant. This would appear to be an unreasonable assumption in practical modeling, since in many situations the demand would be a function of t. The remainder of this paper looks at the case of elastic demands. For concrete illustration of these extensions we use the following demand function:

$$d_i^k(t_i^k) = \bar{d}_i^k \frac{\exp(-\rho t_i^k)}{\exp(-\rho t_i^k) + \exp(-\rho \bar{t}_i^k)},$$

where $\rho = 0.1$ and $\bar{d}_i^k$ are given constants and $\bar{t}_i^k$ represent the time on alternative transportation, also assumed constant.

In this case, the equilibrium problem (4) is unchanged except that the values of d^k are replaced by the above functions of t^k, resulting in the following equilibrium conditions:

$$\begin{array}{rcll} c_a(f_a) - v_a & = & 0 & \forall a \in \mathcal{A} \\ -A^T t^k + v & \geq & 0 \perp x^k \geq 0 & \forall k \in \mathcal{D} \\ A x^k & = & d^k & \forall k \in \mathcal{D} \\ f & = & \sum_{k\in\mathcal{D}} x^k & \\ d_i^k & = & \bar{d}_i^k \frac{\exp(-\rho t_i^k)}{\exp(-\rho t_i^k)+\exp(-\rho \bar{t}_i^k)} & \forall i \in \mathcal{N}, k \in \mathcal{D} \end{array} \tag{7}$$

Even in the Sioux Falls example, it remains possible to construct primal and dual problems that have such optimality conditions. The first step is to construct the inverse demand function using the last equation of (7), namely

$$t_i^k(d_i^k) = \bar{t}_i^k + 1/\rho \ln\left(\frac{\bar{d}_i^k}{d_i^k} - 1\right),$$

for $0 < d_i^k < \bar{d}_i^k$. We note that this function is integrable, and that its integral is

$$T_i^k(d_i^k) = \bar{t}_i^k d_i^k + 1/\rho\left(\bar{d}_i^k - d_i^k \ln(d_i^k) - (\bar{d}_i^k - d_i^k)\ln(\bar{d}_i^k - d_i^k)\right).$$

It follows that the nonlinear program

$$\begin{array}{ll} \min_{x,f,d} & \sum_{a\in\mathcal{A}} C_a(f_a) - \sum_{i\in\mathcal{N},k\in\mathcal{D}} T_i^k(d_i^k) \\ \text{subject to} & \begin{array}{l} A x^k - d^k = 0, x^k \geq 0 \qquad \forall k \in \mathcal{D} \\ f - \sum_{k\in\mathcal{D}} x^k = 0 \end{array} \end{array} \tag{8}$$

has first order optimality conditions

$$
\begin{array}{rcll}
c_a(f_a) - v_a & = & 0 & \forall a \in \mathcal{A} \\
-A^T t^k + v & \geq & 0 \perp x^k \geq 0 & \forall k \in \mathcal{D} \\
Ax^k - d^k & = & 0 & \forall k \in \mathcal{D} \\
f - \sum_{k \in \mathcal{D}} x^k & = & 0 & \\
t_i^k & = & \bar{t}_i^k + 1/\rho \ln \left(\frac{\bar{d}_i^k}{d_i^k} - 1 \right) & \forall i \in \mathcal{N}, k \in \mathcal{D}
\end{array}
$$

which are equivalent to (7). Note that the nonlinear program (8) is no longer a simple multicommodity network flow problem since d are now variables of the problem, not fixed demands. It is also clear that the final equation of (7) can be integrated as

$$D_i^k(t_i^k) = \bar{d}_i^k \left(t_i^k - (1/\rho) \ln \left(1 + \exp(\rho(t_i^k - \bar{t}_i^k)) \right) \right),$$

resulting in the following dual problem:

$$
\begin{array}{ll}
\max_{t,v} & \sum_{i \in \mathcal{N}, k \in \mathcal{D}} D_i^k(t_i^k) - \sum_{a \in \mathcal{A}} H_a(v_a) \\
\text{subject to} & -A^T t^k + v \geq 0 \qquad \forall k \in \mathcal{D}.
\end{array}
$$

Equilibrium models are not always representable as nonlinear programs. There are instances where the inverse demand function cannot be expressed in closed form, and even when it is, the resulting function may not be integrable. In the present application this is unimportant, since it appears that we are able to solve the equilibrium conditions (7) as a complementarity problem more easily than as a nonlinear program.

The numerical results we report below are obtained as follows. We use the values obtained as equilibrium values in the fixed demand case as the time for alternative transportation $\bar{t}_i^k$. The constants $\bar{d}_i^k$ are then taken as twice the fixed demand. With these values, we know that an equilibrium is achievable, setting $t_i^k = \bar{t}_i^k$ with all other variables at their fixed demand solutions. This was done in order to ensure that our algorithms were calculating the correct solutions. In the primal nonlinear programming formulation, to guarantee that the inverse demand functions could be evaluated, we imposed lower bounds of 0.001 on d_i^k and upper bounds of $0.999 * \bar{d}_i^k$. In the dual and complementarity formulation such bounds were unnecessary. The algorithms were started at the same starting points and all found the correct solution.

MCP	Primal NLP	Dual NLP
51.2	73.2	126.2

Table 2: Solution times

The GAMS files that we used to obtain the MCP results are included as appendix A. The GAMS files for the other problems are very similar.

4 Conclusion

In this paper we have demonstrated that the use of a standard mixed complementarity problem modeling language and solver is very effective for solving problems arising in traffic equilibria analysis. We have taken an alternative "multi-commodity" based approach for modeling the Wardropian equilibrium conditions and have been able to compute an equilibrium solution quickly, accurately and without modifying any parameters of the solution algorithm.

We believe that this approach should be considered whenever problems of this form are solved for two reasons. Firstly, the standard tools of the modeling language enable the data of the problem to be effectively communicated to a solver, and allow the modeler to easily test various solution strategies and modeling formats. Secondly, standard codes that are built on a Newton paradigm and exploit the automatic differentiation facilities of the modeling language allow a solution to be calculated very quickly and accurately. While it may be true that specialized algorithms for particular instances of these problems may outperform the standard codes in some cases, the additional flexibility and ease of development of the approach we outline here will without doubt effect a solution in shorter time.

References

Aashtiani, H. Z., and T. L. Magnanti (1981), "Equilibria on a Congested Transportation Network", *SIAM J. Algebraic Discrete Math."*, **2**, 213–226.

Beckmann, M., C. McGuire and C. Winsten (1956), *Studies in the Economics of Transportation*, Yale University Press, New Haven, CT.

Bertsekas, D. P. and E. M. Gafni (1982), "Projection methods for variational inequalities with application to the traffic assignment problem", *Mathematical Programming Study* **17**, 139–159.

Brooke, A., D. Kendrick and A. Meeraus (1988), *GAMS: A User's Guide*, The Scientific Press, South San Francisco, CA.

Chen, C. and O. L. Mangasarian (1996), "A class of smoothing functions for nonlinear and mixed complementarity problems", *Computational Optimization and Applications* **5**, 97–138.

Dirkse, S. P. and M. C. Ferris (1995), "The PATH solver: A non-monotone stabilization scheme for mixed complementarity problems", *Optimization Methods and Software* **5**, 123–156.

Drud, A. (1985), "CONOPT: A GRG code for large sparse dynamic nonlinear optimization problems", *Mathematical Programming* **31**, 153–191.

Friesz, T. L., D. Bernstein, N. J. Mehta, R. L. Tobin and S. Ganjalizadeh (1994), "Day-to-day dynamic network disequilibria and idealized traveller information systems", *Operations Research* **42**, 1120–1136.

Irwin, N. A., N. Dodd and H. G. Von Cube (1961), "Capacity Restraint in Assignment Program", *Highway Research Board* **297**, 109–127.

LeBlanc, L. J., E. K. Morlok and W. P. Pierskalla (1975), "An efficient approach to solving the road network equilibrium traffic assignment problem", *Transportation Research* **9**, 309–318.

Murtagh, B. A. and M. A. Saunders (1983), MINOS 5.0 user's guide, Technical Report SOL 83.20, Stanford University, Stanford, California.

Ralph, D. (1994), "Global convergence of damped Newton's method for nonsmooth equations, via the path search", *Mathematics of Operations Research* **19**, 352–389.

Steenbrink, P. A. (1974), *Optimization of Transport Networks*, John Wiley and Sons, London, England.

Wardrop, J. G. (1952), "Some theoretical aspects of road traffic research", *Proceeding of the Institute of Civil Engineers, Part II*, pp. 325–378.

A GAMS file listings

A.1 sioux-falls.dat

```
SET PARAM /A, B, K/

TABLE ARC_COST(I,J,PARAM)  Arc cost data
            A           B        K
1.2         6           90      25.9002
2.6         5           75      4.9582
3.12        4           60      23.4035
4.11        6           90      4.9088
5.9         5           75      10
7.8         3           45      7.8418
8.9        10           150     5.0502
9.10        3           45      13.9158
10.15       6           90      13.5120
10.17       8           120     4.9935
11.14       4           60      4.8765
13.24       4           60      5.0913
14.23       4           60      4.9248
15.22       4           60      10.3150
16.18       3           45      19.6799
18.20       4           60      23.4035
20.21       6           90       5.0599
21.22       2           30      5.2299
22.23       4           60      5.0000
1.3         4           60      23.4035
3.4         4           60      17.1105
4.5         2           30      17.7828
5.6         4           60      4.9480
6.8         2           30      4.8986
7.18        2           30      23.4035
8.16        5           75      5.0458
10.11       5           75      10.00
10.16       5           75      5.1335
11.12       6           90      4.9088
12.13       3           45      25.9002
14.15       5           75      5.1275
15.19       4           60      15.6508
16.17       2           30      5.2299
17.19       2           30      4.8240
19.20       4           60      5.0026
20.22       5           75      5.0757
21.24       3           45      4.8854
23.24       2           30      5.0785;

ARC_COST(I,J,PARAM)$ARC_COST(J,I,PARAM) = ARC_COST(J,I,PARAM);
```

```
ARC_COST(I,J,"A") = ARC_COST(I,J,"A");
ARC_COST(I,J,"B") = ARC_COST(I,J,"B") / 100;

COEF_A(I,J) = ARC_COST(I,J,"A");
COEF_B(I,J) = ARC_COST(I,J,"B");
COEF_K(I,J) = ARC_COST(I,J,"K");

TABLE DO(I,J)  Trip matrix
  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
1 1  1  5  2  3  5  8  5 13  5  2  5  3  5  5  4  1  3  3  1  4  3  1
2 0  1  2  1  4  2  4  2  6  2  1  3  1  1  4  2  0  1  1  0  1  0  0
3    0  2  1  3  1  2  1  3  3  2  1  1  1  2  1  0  0  0  0  1  1  0
4       0  5  4  4  7  7 12 14  6  6  5  5  8  5  1  2  3  2  4  5  2
5          0  2  2  5  8 10  5  2  2  1  2  5  2  0  1  1  1  2  1  0
6             0  4  8  4  8  4  2  2  1  2  9  5  1  2  3  1  2  1  1
7                0 10  6 19  5  7  4  2  5 14 10  2  4  5  2  5  2  1
8                   0  8 16  8  6  6  4  6 22 14  3  7  9  4  5  3  2
9                      0 28 14  6  6  6  9 14  9  2  4  6  3  7  5  2
10                        0 40 20 19 21 40 44 39  7 18 25 12 26 18  8
11                           0 14 10 16 14 14 10  1  4  6  4 11 13  6
12                              0 13  7  7  7  6  2  3  4  3  7  7  5
13                                 0  6  7  6  5  1  3  6  6 13  8  8
14                                    0 13  7  7  1  3  5  4 12 11  4
15                                       0 12 15  2  8 11  8 26 10  4
16                                          0 28  5 13 16  6 12  5  3
17                                             0  6 17 17  6 17  6  3
18                                                0  3  4  1  3  1  0
19                                                   0 12  4 12  3  1
20                                                      0 12 24  7  4
21                                                         0 18  7  5
22                                                            0 21 11
23                                                               0  7;

DO(I,J)$DO(J,I) = DO(J,I);
DO(I,J) = DO(I,J) * 0.11;

*       Identify arcs using flow cost parameter:
A(I,J) = YES$COEF_A(I,J);

*       Identify destination nodes using the trip table:
DEST(J) = YES$SUM(I, DO(I,J));
```

A.2 primal.gms

```
$TITLE TRAFFIC EQUILIBRIUM -- FRIESZ ET AL.

SCALAR RHO/0.3/;
```

```
SET     N NODES /1*24/;

ALIAS (I,N), (J,N), (K,N), (L,N);

SET     DEST(J) IDENTIFICATION OF DESTINATION NODES,
        A(N,N)  ARCS;

$INCLUDE sioux-falls.dat

PARAMETER TALT(I,J), DBAR(I,J); TALT(I,J) = 0;  DBAR(I,J)=2*D0(I,J);

SET ACTIVE(I,J);
ACTIVE(I,J) = YES$DEST(J);
ACTIVE(I,I) = NO;

VARIABLES       Y(I,J,K)        FLOW TO K ALONG ARC I-J,
D(I,J) TRIP DEMAND,
                X(I,J)          AGGREGATE FLOW ON ARC I-J
                OBJ             OBJECTIVE FOR NLP FORMULATION;

EQUATION        BALANCE(I,J)            MATERIAL BALANCE
                XDEF(I,J)               AGGREGATE FLOW DEFINITION
                OBJDEF                  DEFINES OBJECTIVE FOR NLP FORMULATION;

*       The stuff coming into a node equals the stuff going out:

BALANCE(I,J)$ACTIVE(I,J)..
  SUM(K$Y.UP(I,K,J), Y(I,K,J)) =E= SUM(K$Y.UP(K,I,J), Y(K,I,J)) + D(I,J);

*       Flow on a given arc constitutes flows to all destinations K:

XDEF(A)$X.UP(A)..
        X(A) =E= SUM(K$Y.UP(A,K), Y(A,K));

OBJDEF..
 OBJ =E= SUM(A, COEF_A(A)*X(A) + COEF_B(A)*POWER(X(A)/COEF_K(A),5)*COEF_K(A)/5)
- SUM((I,J)$D0(I,J), D(I,J) * TALT(I,J) +
(1/RHO) * (DBAR(I,J)
- D(I,J)*LOG(D(I,J))
- (DBAR(I,J)-D(I,J)) * LOG(DBAR(I,J)-D(I,J))));

MODEL TRAFFICNLP /OBJDEF, BALANCE, XDEF/;

*       Initial levels for arc flows are needed so that we can
*       properly evaluate the nonlinear functions:
```

```
X.L(A) = COEF_K(A);

*       Lower bounds are zero for flows, positive for times:

Y.LO(A,K) = 0;

*       Fixing values causes corresponding equilibrium conditions
*       to be dropped:

Y.FX(I,J,I) = 0;
Y.FX(I,I,J) = 0;
Y.FX(I,J,K)$(NOT A(I,J)) = 0;
Y.FX(I,J,K)$(NOT DEST(K)) = 0;
X.FX(I,J)$(NOT A(I,J)) = 0;

D.FX(I,J) = DO(I,J);

OPTION NLP = MINOS5;
TRAFFICNLP.ITERLIM = 3000;
TRAFFICNLP.OPTFILE=1;
SOLVE TRAFFICNLP USING NLP MINIMIZING OBJ;

D.LO(I,J)$DO(I,J) = 0.001;
D.UP(I,J)$DO(I,J) = 0.999*DBAR(I,J);
D.L(I,J)$DO(I,J) = D.LO(I,J);
X.L(A) = COEF_K(A);
Y.L(A,K) = 0;

TALT(I,J) = BALANCE.M(I,J);
BALANCE.M(I,J) = 0;
XDEF.M(A) = 0;

SOLVE TRAFFICNLP USING NLP MINIMIZING OBJ;
```

A.3 dual.gms

```
$TITLE TRAFFIC EQUILIBRIUM -- FRIESZ ET AL.

SCALAR RHO/0.3/;

SET     N NODES /1*24/;

ALIAS (I,N), (J,N), (K,N);

SET     DEST(J) IDENTIFICATION OF DESTINATION NODES,
        A(N,N)  ARCS;
```

```
$INCLUDE sioux-falls.dat

VARIABLES       T(I,J)          TIME TO GET FROM NODE I TO NODE J,
V(I,J) TIME TO TRAVERSE ARC I-J
                OBJ             OBJECTIVE FOR NLP FORMULATION;

EQUATION DUAL(I,J,K) COST MINIMIZATION (DUAL FORMULATION)
OBJDUAL OBJECTIVE FOR DUAL FORM
ELASOBJ OBJECTIVE IN ELASTIC DUAL;

PARAMETER T0(I,J) BENCHMARK TRAVEL TIME;

SET DUALCON(I,J,K);
DUALCON(I,J,K) = YES$(A(I,J)*DEST(K));
DUALCON(I,J,I) = NO;

DUAL(I,J,K)$DUALCON(I,J,K).. T(I,K) =L= V(I,J)+T(J,K);

V.LO(A) = COEF_A(A);
V.FX(I,J)$(NOT A(I,J)) = 0;
V.L(A) = V.LO(A);

T.LO(A) = 0.0;
T.FX(I,I) = 0;

OPTION ITERLIM = 8000;
OPTION DOMLIM = 100000;

OBJDUAL.. OBJ =E= SUM((I,K), D0(I,K) * T(I,K))
- SUM(A, (4/5) * (COEF_K(A)/COEF_B(A)**(1/4)) *
(V(A)-COEF_A(A))**1.25 );

MODEL TRAFFIC_D /OBJDUAL,DUAL/;
SOLVE TRAFFIC_D USING NLP MAXIMIZING OBJ;

T0(I,K) = T.L(I,K);
V.L(A) = V.LO(A);

T.L(A) = 0.0;
DUAL.M(I,J,K) = 0.0;

ELASOBJ.. OBJ =E= SUM((I,K)$D0(I,K),
2*D0(I,K) * ( T(I,K)
- (1/RHO) * LOG(1.0 + EXP(RHO*(T(I,K) - T0(I,K)))))
- SUM(A, (4/5) * (COEF_K(A)/COEF_B(A)**(1/4)) *
(V(A)-COEF_A(A))**1.25 );
```

```
MODEL TRAFFIC_E /ELASOBJ,DUAL/;
SOLVE TRAFFIC_E USING NLP MAXIMIZING OBJ;
```

A.4 mcp.gms

```
$TITLE TRAFFIC EQUILIBRIUM -- FRIESZ ET AL.
* see "Computing Wardropian Equilibria in a Complementarity Framework"
* M.C. Ferris, A. Meeraus and T. F. Rutherford
* Mathematical Programming Technical Report 95-03, February 1995.
* ftp://ftp.cs.wisc.edu:math-prog/tech-reports/95-03.ps.Z

SCALAR RHO /0.3/;

SET     N NODES /1*24/;

ALIAS (I,N), (J,N), (K,N), (L,N);

SET     DEST(J) IDENTIFICATION OF DESTINATION NODES,
ACTIVE(I,J,K) IDENTIFIES THE SET OF ACTIVE ARCS,
        A(N,N) ARCS;

$INCLUDE sioux-falls.dat

ACTIVE(A,K) = YES$DEST(K);
ACTIVE(I,J,I) = NO;
ACTIVE(I,I,J) = NO;

PARAMETER ETALT(I,J), DBAR(I,J); ETALT(I,J) = 0;  DBAR(I,J) = 2 * DO(I,J);

VARIABLES       T(I,J)          TIME TO GET FROM NODE I TO NODE J,
                X(I,J,K)        FLOW TO K ALONG ARC I-J,
                F(I,J)          AGGREGATE FLOW ON ARC I-J,
D(I,J) DEMAND FROM NODE I TO NODE J;

EQUATION        RATIONAL(I,J,K)         COST MINIMIZATION
                BALANCE(I,J)            MATERIAL BALANCE
DEMAND(I,J) ELASTIC DEMAND
                FDEF(I,J)               AGGREGATE FLOW DEFINITION;

*       The following constraint imposes individual rationality.

*       The time to reach node K from node I is no greater than
*       the time required to travel from node I to node J and then
*       from node J to node K.

RATIONAL(I,J,K)$ACTIVE(I,J,K)..
COEF_A(I,J) + COEF_B(I,J) * POWER(F(I,J)/COEF_K(I,J),4) + T(J,K)
```

```
=G= T(I,K);

*       The flow into a node equals demand plus flow out:

BALANCE(I,J)$T.UP(I,J)..
SUM(K$ACTIVE(I,K,J),X(I,K,J)) =G=
SUM(K$ACTIVE(K,I,J),X(K,I,J)) + D(I,J);

* Elastic demand:

DEMAND(I,J)$(D.UP(I,J) NE D.LO(I,J))..
    (D(I,J) - DBAR(I,J)) * EXP(-RHO*T(I,J)) + D(I,J) * ETALT(I,J) =E= 0.0;

*       Flow on a given arc constitutes flows to all destinations K:

FDEF(A)..
        F(A) =E= SUM(K$ACTIVE(A,K), X(A,K));

*       Here is the MCP model:

MODEL TRAFFIC /RATIONAL.X, BALANCE.T, FDEF.F, DEMAND.D/;

*       Initial levels for arc flows are needed so that we can
*       properly evaluate the nonlinear functions:

F.L(A) = COEF_K(A);
X.L(A,K) = 0.0;
T.L(I,J)   = COEF_A(I,J)$A(I,J) + SMIN(K$A(I,K),COEF_A(I,K))$(NOT A(I,J));

*       Lower bounds are zero for flows, positive for times:

X.LO(A,K) = 0.0;
T.LO(I,J)   = 0.0;

*       Fixing values causes corresponding equilibrium conditions
*       to be dropped:

T.FX(I,I) = 0;

F.FX(I,J)$(NOT A(I,J)) = 0;

* INITIALLY FIX DEMAND:

D.FX(I,J) = D0(I,J);

TRAFFIC.ITERLIM = 8000;
SOLVE TRAFFIC USING MCP;
```

```
* CONFIGURE THE ELASTIC-DEMAND MODEL:

ETALT(I,J)$D0(I,J) = EXP(-RHO*T.L(I,J));
D.UP(I,J)$D0(I,J) = +INF;
D.LO(I,J)$D0(I,J) = -INF;

D.L(I,J)$D0(I,J) = 0.0;
F.L(A) = COEF_K(A);
X.L(A,K) = 0.0;
T.L(I,J)   = COEF_A(I,J)$A(I,J) + SMIN(K$A(I,K),COEF_A(I,K))$(NOT A(I,J));

* CONFIRM THE SOLUTION:

SOLVE TRAFFIC USING MCP;
```