CONF-950193--2

OAK RIDGE
NATIONAL
LABORATORY

MAXBAND Version 3.1: Heuristic and Optimal Approach for Setting the Left Turn Phase Sequences in Signalized Networks

Rekha S. Pillai[‡]
Ajay K. Rathi
Steve Cohen

MANAGED BY
MARTIN MARIETTA ENERGY SYSTEMS, INC.
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

‡ Postdoctoral Research Program, Oak Ridge Institute of Science and Education (ORISE), Oak Ridge, TN 37831

DISCLAIMER

Abstract

The main objective of synchronized signal timing is to keep traffic moving along arterial in platoons throughout the signal system by proper setting of left turn phase sequence at signals along the arterials/networks. The synchronization of traffic signals located along the urban/suburban arterials in metropolitan areas is perhaps one of the most cost-effective method for improving traffic flow along these streets. The popular technique for solving this problem formulates it as a mixed integer linear program and used Land and Powell branch and bound search to arrive at the optimal solution. The computation time tends to be excessive for realistic multiarterial network problems due to the exhaustive nature of the branch and bound search technique. Furthermore, the Land and Powell branch and bound code is known to be numerically unstable, which results in suboptimal solutions for network problems with a range on the cycle time variable. This paper presents the development of a fast and numerically stable heuristic, developed using MINOS linear programming solver. The new heuristic can generate optimal/near-optimal solutions in a fraction of the time needed to compute the optimal solution by Land and Powell code. The solution technique is based on restricted search using branch and bound technique. The efficiency of the heuristic approach is demonstrated by numerical results for a set of test problems.

A Restricted Branch and Bound Approach for Setting the Left Turn Phase Sequences in Signalized Networks

Rekha S. Pillai, Ph. D.
Center for Transportation Analysis
Oak Ridge National Laboratory
Oak Ridge, TN 37831

Ajay K. Rathi, Ph. D., P.E.
Group Leader, IVHS Research
Center for Transportation Analysis
Oak Ridge National Laboratory
Oak Ridge, TN 37831

Steve Cohen
Federal Highway Administration
Turner Fairbanks Highway Research Center
McLean, VA 22101

Submitted for presentation at the 74th Annual Meeting of Transportation Research Board
January, 1995

July 1994

1.0 Introduction

Efficient transportation is very important to the nation's economic health. Nearly all economic activity uses transportation directly or indirectly. The economic productivity of a nation is boosted by improving the efficiency of transportation systems. The synchronization of traffic signals, located along the urban/suburban arterials in metropolitan areas, is perhaps one of the most cost effective method for improving traffic flow along these sections of the urban street networks. The main objective of synchronized signal timing is to keep traffic moving along an arterial in platoons throughout the signal system by proper synchronization of green signals along the arterials/networks.

Over time, traffic engineering research has resulted in a number of techniques for setting traffic signals along arterials and networks. These models can be classified into two major categories: on-line models and off-line models. The on-line (also referred to as traffic adaptive) models compute signal settings in real-time and are used for controlling traffic dynamically. OPAC (Gartner [1983]) is an example of this type of model.

Off-line signal optimization models were developed in the late 1960's and early 1970's, and are used for computing signal settings for recurrent traffic flow conditions. The existing models for off-line determination of signal settings on single/multiarterial networks fall into one of two major categories. One set of models are based on the criteria of minimizing system delays and stops, while the other maximizes the progression bandwidth along the arterials. Delay minimization models lead to signal settings that minimize the number of stops and delays experienced by vehicles at intersections. Bandwidth maximization models lead to signal settings that maximize the proportion of traffic flowing unimpeded through the signals. TRANSYT (Robertson [1968], Wallace *et al.* [1988]) and SIGOP III (Lieberman *et al.* [1983]) are models that determine signal settings that minimize delay. These models combine macroscopic simulation and nonlinear optimization based gradient searches to determine the optimal signal settings. MAXBAND is a model that maximize bandwidth for multiarterials (Messer *et al.* [1987], Chang *et al.* [1988]).

The underlying optimization model in MAXBAND is a Mixed Integer Linear Programming (MILP) model, originally formulated by Little [1966]. This formulation was extended to triangular networks by Little, Kelson, and Gartner [1981]. Gartner *et al.* [1991] report the extension of the arterial MILP formulation to include multi-band capability. Chaudhary *et al.* [1993] report the development of bandwidth optimization formulation that include circular phasing of signals and the new model is called PASSER IV.

Cohen [1983], Cohen and Liu [1986], and Liu [1988] used the good features of both delay minimization model and bandwidth maximization model to obtain a combined model. These model attempt to find a signal settings that minimizes delays and while maximizing progression bandwidth.

TRANSYT is perhaps the most widely used traffic model in the practice of traffic engineering, in spite of its limitations. TRANSYT minimizes delay-based disutility functions from which green bands cannot always be found, and the model has no capability to perform left turn phase sequence optimization. MAXBAND model maximizes green bands and optimizes left turn phase sequences. Studies have shown (see Rogness [1981], Cohen *et al.* [1983]) that left turn phase sequence optimization can substantially improve performance of signal timing plans. But, experience with MAXBAND has shown that hours of computer time may be required to optimize a medium-sized network problem even on a mainframe computer. The computational inefficiencies make the current version of MAXBAND impractical for use by the traffic engineering community.

This paper describes the development of a new heuristic for the bandwidth maximization MILP using restricted branch and bound approach. The new heuristic is fast, numerically stable, and capable of generating an optimal/near-optimal solution for the MILP problem. The speed up in execution time will make bandwidth optimization usage attractive for real-time applications and for iterative use of bandwidth maximization with delay-minimization problems or simulation procedures.

The remainder of this paper is organized as follows. In the next section, the MILP and the heuristic are discussed. Section 3 discusses the LP solver and computer implementation. Section 4 reports the results for a number of network and arterial test problems. Finally, in section 5 the conclusions and directions for future work are discussed.

2.0 MILP Based Bandwidth Maximization

The current version of MAXBAND (known as MAXBAND-86 or MAXBAND Version 2.1) provided optimal solutions for arterial and network problems. This version of the program uses the integer programming code from Land and Powell [1973]. The code uses branch and bound logic in determining the optimal values of the integer variables. It requires an excessive amount of computer time to solve even a small instance of a network problem. The code was also known to be numerically unstable for problems where the optimal cycle time was to be selected from a range of cycle times. The numerical instability resulted in the runs ending prematurely with either a suboptimal solution or no solution at all. Some modifications were made to MAXBAND-86 to stabilize the numerical computations (see Solanki *et al.* [1993]). In this effort, the research staff at Oak Ridge National Laboratory (ORNL) developed a new heuristic solution technique that is faster, numerically stable and capable of generating optimal/near-optimal solutions for the bandwidth maximization MILP. The new heuristic is based on restricted branch and bound logic.

The MILP formulation for multiarterial networks consist of blocks of constraints dealing with individual arterials and some additional constraints that impose restrictions on loops of multiple arterials. The derivation of the constraints and details of the formulation are provided in Messer *et al.* [1987]. The difficulty in solving realistic problems arises due to the large number of integer variables in the formulation. The heuristic is a search procedure for suitable values of the integer variables. The set of integer variables in the MILP formulation can be divided into three sets:

Intra-loop variables (m_{ij}) : are a set of general integer variables. This variable denotes the number of cycles required to go from signal i to signal $i+1$ and back, on arterial j. The m_{ij}'s should assume integer values due to the fact that the progression bandwidth in a specified direction for arterial j should pass through the green interval of signal cycles at signal i and $i+1$. Little [1966] provides the analytical justification for the integral nature of this set of variables.

Inter-loop variables (n_L) : are a set of general integer variables. This variable denotes the number of cycles required for traversing arterials in the loop. The inter-loop variables are the reflection of network closure constriants which are required in a closed network consisting of intersecting arterials and running on a common cycle length. These variables state that the sum of the offsets around any closed loop in the network must be an integral multiple of the common cycle length. Messer *et al.* [1987] provide the analytical justification for the integer nature of this set of variables.

Left-turn-phase sequence variables (δ_{ij}) : are a set of binary variables. These variables are used to define the left turn phase sequence pattern on intersection i of arterial j.

The only known heuristics for the MILP are the *two-step* and *three-step* heuristic by Chaudhary *et al.* [1991,1993]. The first step of the two-step heuristic relaxes the δ_{ij}'s to continuous variables and searches for optimal m_{ij}'s and n_L's. The six best solutions obtained during the search are saved. For each of these six solutions, the integer values of the m_{ij}'s and n_L's are fixed in the second step, which searches for optimal integer values of the δ_{ij}'s. Similarly, the three-step heuristic solves the integer values of the n_L's, m_{ij}'s and δ_{ij}'s in three steps, where the integer values obtained in one step are fixed in the next step. As expected, the two-step heuristic produces better solutions but consumes significantly more time compared with the three-step heuristic. In both heuristic methods, at each step an exhaustive branch and bound search is required to obtain optimal integer values. It was observed that, for some problem instances, the time required by the multi-step method could be more than the time required for the simultaneous optimization of all integer variables.

The key observation of a good heuristic design is to identify suitable problems that can be solved quickly and repetitively to generate improving solutions over iterations. The heuristic developed for the bandwidth maximization MILP is a *restricted branch and bound algorithm.* The branch and bound search is restricted to portions of solution space which is likely to contain good solutions. Figure 1 gives an overview of the new heuristic. There are two key elements that characterize the algorithm described here:

1. a greedy heuristic to generate a good lower bound to be used at the root node of the branch and bound tree (*Greedy Heuristic I*), and
2. a *tree search* approach that combines branching and bounding techniques.

Efficient implementation of these key elements allow us to solve large problem instances of the MILP in reasonable time and memory allocations. Let P be the original MILP problem to be maximized. Let $V(P)$ be the optimal objective function value of P. Let P' be the LP relaxation of P, obtained by relaxing the integer variables m_{ij}'s, n_L's, and δ_{ij}'s. Then, $V(P')$ is the optimal objective value of P'. It is obvious that $V(P) \leq V(P')$. If the optimal value of the solutions vector corresponding to the variables m_{ij}'s, n_L's, and δ_{ij}'s are integer in P', then the solution is optimal to the original problem P. The greedy heuristic, developed to generate a lower bound that can be used in the tree search procedure, shall be discussed first. This paper then continues to discuss the restricted branch and bound algorithm.

Greedy heuristic I is based on the concept of local search in the space of integer variables. Heuristic algorithms based on local searches have been found to be very effective in a large variety of integer programming problems. The key to this algorithm is to restrict the integer variable ranges to those values which are likely to yield good solutions. The local search is performed by fixing the values of the integer variables. The objective of the restricted problem is evaluated by solving the resulting restricted linear program.

Algorithm I : Greedy Heuristic I

Input : P', the set of integer variables

Step 1 : Initialize the current incumbent, $Z^* = -\infty$.

Step 2 : Perform steps 3 through 9 two times. Go to step 10.

Step 3 : Order the set of integer variables as follows $I = \{n_1 , \ldots, n_L, m_{11} , \ldots, m_{KN} \}$

Step 4 : Solve LP problem P'.

Step 5 : If the set I is empty then go to step 8; otherwise pick the next variable from the ordered set I (say variable x_{ij}) and delete it from set I.

Step 6 : Set the upper and lower bounds of the variable x_{ij} as follows: $b^{ij}_{low}=b^{ij}_{up}=Int(x_{ij}+0.5)$, i.e. set the upper and lower bounds of the integer variable to be the integer value nearest the LP solution.

Step 7 : Solve the restricted LP. If the current LP is infeasible then reset the variable last set to the other end of the LP optimal solution (obtained in step 6) and re-solve. Go to step 5.

Step 8 : The algorithm reaches this step once all the integer variables have been set to the LP solution upper or lower bound. If the final LP is feasible then the solution is a valid lower bound for the problem P. If the objective is greater than the current incumbent, save the current solution as the incumbent. Reset the bounds of all the integer variables.

Step 9 : Reverse the order of the integer variables and put it in set I, i.e. this time the variable m_{KN} is the first variable and variable n_1 is the last variable. Go through steps 4 through 8.

Step 10: Fix the m_{ij}'s and n_L's at their best values and use branch and bound code to integerize the δ_{ij}'s.

The solution obtained at the end of step 10 of greedy heuristic I serves as a lower bound (best incumbent) during the branch and bound procedure. Such a bound restricts the growth of the tree and hence helps in faster resolution of the optimal solution. The restricted tree search algorithm (also called restricted branch and bound) developed for bandwidth optimization can then be described as follows: In the tree-search procedure, the range over which the integer variables, m_{ij}'s and n_L's, can vary, are restricted. Integer variable m_{ij}'s are allowed only two values and integer variable n_L's are allowed three values. The three values of n_L's are selected such that the incumbent value is the middle value. The two values of m_{ij}'s are selected such that the incumbent value is the upper bound of this variable. For ease of exposition let an integer variable be denoted x_{ij}. Let the set F_l be the set of the integer variables fixed at the lower bound during the branch and bound procedure i.e. $F_l = \{x_{ij} / b_{low} \leq x_{ij} \leq b_{low}\}$. Let F_m be the set of integer variables fixed at the middle value i.e. $F_m = \{x_{ij} / b_{low}+1 \leq x_{ij} \leq b_{low}+1\}$, and F_r be the set of integer variables fixed at the upper bound i.e. $F_r = \{x_{ij} / b_{up} \leq x_{ij} \leq b_{up}\}$. Then, let S be a family of ordered triple of node sets $<F_l, F_m, F_r>$, and let Z^*, referred to as an *incumbent,* be the incidence vector of some integer feasible solution.

To describe the restricted branch and bound algorithm the following terminologies are used. Let a *tree-node,* associated with the ordered set $<F_l, F_m, F_r>$, be the problem $P(F_l,F_m,F_r)$. This is a problem of finding a signal timing plan whose solution vector satisfies the inequalities (2.1a), (2.1b), and (2.1c) given below:

$b_{low} \leq x_{ij} \leq b_{low}$ for all integer variables in the set F_l (2.1a)

$b_{low}+1 \leq x_{ij} \leq b_{low}+1$ for all integer variables in the set F_m (2.1b)

$b_{up} \leq x_{ij} \leq b_{up}$ for all integer variables in the set F_r (2.1c)

Then, $P'(F_l,F_m,F_r)$ is the *linear relaxation* of $P(F_l,F_m,F_r)$ obtained by relaxing the integer variable not in the set F_l, F_m, and F_r. The tree-nodes are recorded by the ordered triple corresponding to it. A tree-node is considered fathomed if one or more of the following conditions are satisfied:

i. the optimal LP objective i.e. $V(P'(F_l,F_m,F_r))$, at this node is less than the current incumbent,

ii. the depth of this tree-node is equal to the maximum depth (*mdepth*) specified,

iii. the LP, $P'(F_l,F_m,F_r)$, is infeasible, or

iv. the optimal LP results in an integer feasible solution.

If the optimal solution of the current LP relaxation is fractional and the current depth (number of integer variables fixed) is less than maximum depth, the algorithm selects a *branching variable* x_{ij} and *branches*, thus providing up to three new tree-nodes ($<F_l \cup \{x_{ij}\}, F_m, F_r>$, $<F_l, F_m \cup \{x_{ij}\}, F_r>$, $<F_l, F_m, F_r \cup \{x_{ij}\}>$). The *root-node* of the search-tree is the tree-node $<\emptyset,\emptyset,\emptyset>$. During the algorithm the tree-nodes of the search-tree that are in S are called *active tree-nodes.* The restricted branch and bound algorithm can then be described as follows:

Algorithm II : Restricted Branch and Bound

Input : *Z^*, the LP problem P', the set of integer variables I.*

Step 1 : **(Initialization)** Set $S = \{ <\emptyset,\emptyset,\emptyset> \}$. Limit the ranges of the m_{ij}'s and n_L's. Select the maximum depth *(mdepth)* of the tree to be half of the number of integer variables in the problem. Number the integer variables such that the first consecutive number, (starting with number 1), are given to the m_{ij}'s, the next consecutive numbers are given to n_L's and finally number the δ_{ij}'s.

Step 2 : **(Select a tree-node for evaluation).** If $S = \emptyset$ then stop - the current incumbent is a local optima. Otherwise choose an ordered set $<F_l, F_m, F_r>$ from S and set $S = S \setminus <F_l, F_m, F_r>$.

Step 3 : **(Greedy heuristic II).** Fix the integer variables that are not yet fixed yet, (i.e. the set of integer variables $\{x_{ij} / x_{ij} \in I \setminus (F_l \cup F_m \cup F_r)\}$), to an integer value nearest to the LP solution, i.e. set $b^{ij}_{low} = b^{ij}_{up} = Int(x_{ij}+0.5)$. Solve the new LP. If the optimal objective is greater than Z^*, save the solution and reset the variables fixed in this step.

Step 4 : **(Evaluation of tree-node).** Solve the linear program, $P'(F_l,F_m,F_r)$, with the additional restriction. Let Z' be its optimal solution. If $Z' \leq Z^*$, go to step 2.

Step 5 : (Check for new incumbent) If Z' is integer feasible, and the optimal objective value is greater than Z^*, then set Z^* to Z'. Go to step 2.

Step 6 : (Create new set of tree-nodes) If the depth of the tree is greater than the maximum depth specified for the problem instance then go to step 2. Otherwise, select a fractional integer variable x_{ij} to branch on. Such a variable will be in $I \setminus (F_l \cup F_m \cup F_r)$. Set $S = S \cup < F_l \cup \{x_{ij}\}, F_m, F_r > \cup < F_l, F_m \cup \{x_{ij}\}, F_r > \cup < F_l, F_m, F_r \cup \{x_{ij}\} >$ and go to step 2.

Once an ordered triple is removed from S it is never again generated in Step 6, so the algorithm terminates in a finite number of steps. When the algorithm stops, Z^* is a local optima. The performance of Algorithm II depends significantly on certain implementation details. In particular, the following issues are key to the algorithm's performance:

(a) Whether or not early tree-nodes can be fathomed depends on the starting Z^*. If this value is close to the optimum, the search-tree will consist of few tree-nodes. Therefore it is necessary to generate good feasible solutions early in the procedure. This objective is achieved by the Greedy Heuristic I and Greedy Heuristic II.

(b) Steps 3 and 4 must be executed many times before a good solution is obtained. A large portion of the final execution time of the algorithm is devoted to solving the LPs. Therefore, it is important to use the LP-optimizer as efficiently as possible. The LP-optimizer of MINOS is very fast and numerically very stable since it uses the state-of-art techniques of numerical analysis and linear programming for updating basis and matrix inversions.

(c) The efficiency of the algorithm with respect to run time and memory usage depends on two things: the way the ordered triple is chosen in Step 2, and the way the tree-nodes are created. The tree-nodes were processed in a depth-first fashion (LIFO). The integer variables are ordered as follows: (m_{ij}'s, n_L's, and δ_{ij}'s). Through experimentation, it was found that this particular order led to incumbents that are close to optimal early on in the search tree. The depth of the search-tree was also restricted to half the number of integer variables, m_{ij}'s and n_L's. This restricted the number of tree-nodes generated and, hence, restricted the growth of the search tree. Further restriction on the range of integer variables also limited the number of tree-nodes generated. As will be seen in the numerical results both types of restriction helped in faster resolution of the optimal solution. The experimentation with δ variables revealed that these naturally turn out to be integer or can be rendered integer by a minimal amount of branching in the branch and bound search. Thus the δ values are searched using the exact optimization technique.

3.0 The LP Solver

For implementing of Algorithms I and II, the linear programming routines from MINOS 5.4 [1993] were used. MINOS is a FORTRAN-based computer system designed to solve large-scale linear and nonlinear optimization problems. It has a collection of high-performance mathematical subroutines which can be called from application programs. From this package we used only the subroutines required for solving linear programming problems. The main reasons for choosing this LP solver were the cost and the availability of source code. The availability of source code allowed customization and therefore helped to speed up the executable.

MINOS performs scaling of rows, right hand side vectors, and columns by choosing appropriate scale factors to make its rows and columns roughly the same length, in some appropriate norm during the solution process; whereas, in Land and Powell the scaling of a problem instance had to be performed by the user externally. In MINOS, the constraints and variables are scaled by an iterative procedure that attempts to make the matrix coefficients as close as possible to 1. This improves the solution performance. Data (both input and output) is stored within a work array that is partitioned by a set of pointers to starting locations of individual arrays needed by the procedure, each with an appropriate number of bytes that depends on whether the array is integer, single, or double precision floating-point. This makes implementation largely independent of data structures and it is then relatively easy to unplug one set of data structure and substitute another. MINOS uses primal simplex algorithm to find the optimal solution of a given LP. In the primal simplex algorithm basis inversion is done repeatedly. The basis inverse can be done either explicitly or as a product of a sequence of pivot matrices. The major drawback of basis inversion is that roundoff errors accumulate as the algorithm moves from step to step. LU-decomposition technique is used compute the basis inverse (see Murty [1983] for more on LU decomposition). During LU factorization the near zero pivot elements lead to uncontrollable growth in the elements and fill-in of L (lower triangular matrix) and U (upper triangular matrix). This in turn results in large numerical errors and large computational times for the primal simplex algorithm used for solving and LP. The solution is to choose pivot elements suitably so as to prevent such element growth and fill-in growth. MINOS implementation is based on the *Markowitz pivoting strategy* that balances considerations of stability and sparsity. The basis updating strategy used by MINOS is the *Bartels-Golub basis updating* strategy in which updating is carried out with a pivot strategy that balances considerations of stability and sparsity. The basis inverse is maintained implicitly in product form. MINOS has also implemented various selection strategies for actually making the choice of entering and exiting variables. These strategies lead to faster resolution of the optimal solution, degeneracy resolution and also leads to numerical stability.

4.0 Numerical Results

A number of network and single arterial problems were solved using the new model. The test data sets were obtained from FHWA. Tables I through IV report the solution quality and the computation times for these problems. Tables I and II show the results of the arterial test problems. Tables III and IV correspond to the network problem runs. The columns of Tables III and IV can be described as follows. Column 1 specifies the names of the data set as identified by the FHWA. The name is given here for reference purposes only. Column 2 contains the problem sizes showing the number of arterials and total number of intersections. Column 3 contains the optimal objective value. Column 4 contains the objective function value at the end of the LP based heuristic (greedy heuristic I); the numbers in parentheses show how close it is to the optimal value. Column 5 contains the time in seconds for greedy heuristic I. Column 6 contains the objective at the end of the restricted branch and bound procedure; the number in parentheses shows how close this value is to the optimal objective value. The numbers in column 7 show the time taken in seconds for the entire algorithm. The computation times are reported for an 80486/66 MHz personal computer. As is observed from the tables III and IV, the heuristic performs very well in generating optimal/near-optimal solutions in a short amount of time. The utility of the heuristic increases as the size of the problem grows and an exact search requires excessive computation time. The arterial problems are solved optimally since all of them can be solved in less than a minute.

5.0 Conclusions and Future Work

In this paper, we describe a fast heuristic for the bandwidth optimization MILP. The restricted branch and bound heuristic is able to obtain optimal and near-optimal solutions for a range of test problems in only a fraction of the time needed by MAXBAND. The reductions in computation times for difficult network problems is substantial, which allows for the use of repetitive solutions of the bandwidth maximization problem in conjunction with the delay minimization problem or simulation procedure. The arterial problems are solved optimally since they only require a few seconds to solve. Additionally since MINOS 5.4 uses state-of-the art techniques in numerical analysis and linear programming for inverting matrices and updating basis this code is much faster and numerically stable compared to the Land and Powell code.

Further work can be done to enhance the MILP formulation to include circular phasing and multiband capability. Work can also be done in building a combined model based on bandwidth maximization, delay minimization, and simulation.

References

[1] E.C.P. Chang, S.L. Cohen, C.C. Liu, N.A. Chaudhary, and C.J. Messer. **MAXBAND-86: Program for Optimizing Left Turn Phase Sequence in Multiarterial Closed Networks.** *Transportation Research Record* 1181:61-67. 1988.

[2] N.A. Chaudhary, A. Pinnoi, and C.J. Messer. **Proposed Enhancements to MAXBAND-86 Program.** *Transportation Research Record*, 98-104. 1991.

[3] N.A. Chaudhary and C.J. Messer. **PASSER-IV: A Program for Optimizing Signal Timing in Grid Networks.** *Paper presented at the 72nd Annual Meeting of the Transportation Research Board, Washington D.C.* 1993.

[4] H. Chen. **An Adaptive Control Strategy for Signalized Intersections.** *Proceedings of the North American Conference on Microcomputers in Transportation, Boston MA*. 1987a.

[5] H. Chen. **Simulation Study of OPAC: A Demand Responsive Strategy for Traffic Signal Control.** *Proceedings for the Tenth International Symposium on Transportation and Traffic Theory, Cambridge MA*. 1987b.

[6] S.L. Cohen. **Concurrent use of MAXBAND and TRANSYT Signal Timing Programs for Arterial Signal Optimization.** *Transportation Research Record*, 906:81-84. 1983.

[7] S.L. Cohen and C.C. Liu. **The Bandwidth-Constrained TRANSYT Signal Optimization Program.** *Transportation Research Record*, 1057:1-7. 1986.

[8] R. Fourer. **Solving Staircase Linear Programs by the Simplex Method.** *Math. Prog. 23, 274-313.* 1982.

[9] N.H. Gartner, M.H. Kaltenbach, and M.M. Miyamato. **Demand Responsive Decentralized Urban Traffic Control, Part II: Network Extensions.** *Technical Report DOT/RSPA/DPB, Department of Transportation, Washington D.C.* 1983.

[10] N.H. Gartner, S.F. Assmann, F.L. Lasaga, and D. L. Hout. **A Multi-Band Approach To Arterial Traffic Signal Optimization.** *Transportation Research Board,* Vol. 25B, No. 1, pp. 55-74. 1991.

[11] A.H. Land and S. Powell. **Fortran Codes for Mathematical Programming: Linear, Quadratic and Discrete.** *John Wiley & Sons, London.* 1973.

[12] E.B. Lieberman. **An Advanced Approach to Meeting Saturated Flow Requirements.** *Paper Presented at the 72nd Annual meeting of the Transportation Research Board, Washington D.C.* 1993.

[13] E.B. Lieberman, J. Lai, and R.E. Ellington. **SIGOP III User's Manual: Technical Report,** *Federal Highway Administration, Washington D.C.* 1983.

[14] J.D.C. Little. **The Synchronization of Traffic Signals by Mixed-Integer Linear Programming.** *Operations Research*, 14:568-594. 1966.

[15] J.D.C. Little, M.D. Kelson, and N.H. Gartner. **MAXBAND: A Program for Setting Signals on Arterials and Triangular Networks.** *Transportation Research Record,* 795. 1981.

[16] C.C. Liu. **Bandwidth Constrained Delay Optimization for Signal Systems.** *ITE Journal*, 58:21-26. 1988.

[17] C.J. Messer, G.L. Hogg, N.A. Chaudhary, and E.C.P. Chang. **Optimization of Left Turn Phase Sequence in Signalized Networks using MAXBAND 86, Vol.1 Summary Report.** *Technical Report FHWA/RD-87/109*, FHWA. 1987.

[18] K. Murty. **Linear Programming.** *John Wiley and Sons, Inc.* New York, 1983

[19] G.L. Nemhauser, L.A. Wolsey, **Integer and Combinatorial Optimization.** *A Wiley-Interscience Publication*, John Wiley and Sons, New York, 1988.

[20] J.K. Reid, **Fortran Subroutines for Handling Sparse Linear Programming Bases.** *Technical report R8269, Atomic Energy Research Establishment,* Harwell, England, 1976.

[21] J.K. Reid, **A Sparsity-Exploiting Variant of the Bartels-Golub Decomposition for Linear Programming Bases.** *Mathematical Programming,* 24, 55-69, 1982.

[22] D.I. Robertson, **TRANSYT: Traffic Network Study Tool,** *Fourth International Symposium on the Theory of Traffic Flow, Karlsruhe, Germany*. 1968.

[23] R.O.Rogness. **Evaluation of a Heuristic Programming Approach to Arterial Street Signal Timing Operation.** *Ph.D. Dissertation, Texas A&M University*. 1981.

[24] R.S. Solanki, A.K. Rathi, and S.L. Cohen. **Numerical Stabilization of MAXBAND 86 on PC**. *Paper presented at the 72nd Annual Conference of the Transportation Research Board, Washington D.C.* 1993.

[25] MINOS 5.4, **Stanford Business Software**, *Stanford University, Stanford, CA*. 1993.

[26] C.E. Wallace, K.G. Courage, D.P. Reaves, G.W. Shoene, G.W. Euler, and A. Wilbur. **TRANSYT-7F User's Manual (Release 6)**. *Technical Report, Prepared for FHWA by the Transportation Research Center, University of Florida, Gainesville, FL*. 1988.

Table I : Arterial problem without left-turn phase sequence variables

PROBLEM NAME	SIZE	Opt. Obj.	Time (Sec.)	Land & Powell Time (sec)**
Hawth	(1,13)	0.2305	12.14	106
N33rd	(1,9)	0.1292	7.47	19
Nicholas	(1,12)	0.5454	11.53	56
Univ	(1,10)	0.2166	4.89	19
Fredrica	(1,12)	0.6726	15.32	79
Mstreet	(1,8)	0.6993	2.52	4

** These timings are for 486/33 MHz personal computer.

Table II: Arterial problem with left-turn phase sequence variables

PROBLEM NAME	SIZE	Opt. Obj.	Time (Sec.)	Land & Powell Time (sec)**
Hawth.ph	(1,13)	0.5796	15.38	889
N33rd.ph	(1,9)	0.4735	16.53	44
Nicholas.ph	(1,12)	0.6254	16.92	257
Univ.ph	(1,10)	0.5000	7.36	23
Fredrica.ph	(1,12)	0.7106	33.83	209
Felipe.ph	(1,12)	0.6000	3.74	21

** These timings are for 486/33 MHz personal computer.

Table III : Network problem without left-turn phase sequence variables

PROBLEM NAME	SIZE	Opt. Obj.	Heu.Obj (%Opt)	Time (Sec.)	Heu. B&B Obj (%Opt)	Time (Sec.)	Land & Powell Time (sec)**
Daytona	(7,12)	1.6368	1.5381(94.0%)	46.58	1.5403(94.1%)	235.52	244592
Annarbor	(8,14)	3.3235	2.8010(84.3%)	57.40	3.3235(100%)	198.18	8912
Houston	(8,13)	2.5079	1.5816(63.1%)	58.50	2.3723(94.6%)	113.26	11765
Memphis	(8,17)	3.0479	2.5768(84.5%)	50.10	3.0044(98.6%)	582.59	28306
Ogden	(8,13)	2.6192	2.2981(87.7%)	62.40	2.5902(98.9%)	548.49	33365
Baycity	(8,16)	2.9658	2.5040(84.4%)	75.63	2.7094(91.4%)	348.18	26854
Owosso	(8,16)	4.0057	3.2217(80.4%)	45.21	3.9119(97.7%)	111.99	3804
Lax	(8,15)	3.4278	3.2157(93.8%)	65.63	3.4278(100%)	394.86	15745
Wlntcrk	(6,13)	2.0925	1.5478(74.0%)	28.62	2.0925(100%)	148.30	3525
Sanramon	(6,17)	1.8848	1.5416(81.8%)	70.03	1.8848(100%)	348.45	32812
Annarbo1	(9,20)	2.5599	1.6168(63.2%)	125.18	2.3363(91.3%)	1782.11	na
Wlntcrk1	(9,22)	2.6737	2.4303(90.9%)	75.80	2.4303(90.9%)	658.12	na
Annarbo2	(9,26)	3.0861	2.5016(81.1%)	161.81	2.7579(89.4%)	1443.50	na
Wlntcrk2	(10,27)	2.5763*	1.8660(72.4%)	133.36	2.4834(96.4%)	985.20	na

* Best known objective, not necessarily optimal.
** Time on 486-33 MHz microprocessor.

Table IV : Network problem with left-turn phase sequence variables†

PROBLEM NAME	SIZE	Opt. Obj.	Heu.Obj (%Opt)	Time (Sec.)	Heu. B&B Obj (%Opt)	Time (Sec.)
Daytona.ph	(7,12)	2.9445	2.2949(77.9%)	43.12	2.9445(100%)	474.72
Annarbor.ph	(8,14)	3.7652	3.4915(92.7%)	34.71	3.7095(98.5%)	314.56
Houston.ph	(8,13)	2.8171	2.0992(74.5%)	48.83	2.8171(100%)	106.78
Memphis.ph	(8,17)	3.4682	3.2214(92.9%)	31.14	3.4474(99.4%)	319.89
Ogden.ph	(8,13)	3.1220	2.9627(94.9%)	48.94	3.1029(99.4%)	654.11
Baycity.ph	(8,16)	3.8354	2.6903(70.1%)	54.98	3.7395(97.5%)	221.95
Owosso.ph	(8,16)	4.2978	3.5324(82.2%)	27.14	4.2321(98.5%)	79.48
Lax.ph	(8,15)	4.0004	2.9501(73.8%)	117.76	3.8862(97.2%)	939.01
Wlntcrk.ph	(6,13)	2.8304	2.3120(81.7%)	26.09	2.7231(96.2%)	103.37
Sanramon.ph	(6,17)	2.8012	2.3150(82.6%)	31.41	2.8012(100%)	181.26
Annarbo1.ph	(9,20)	2.9380	1.8481(62.9%)	118.09	2.7929(94.7%)	7389.95
Wlntcrk1.ph	(9,22)	3.3667	2.5443(75.6%)	62.29	3.2071(95.3%)	1494.02
Annarbo2.ph	(9,26)	3.6215*	1.9657(54.3%)	223.38	3.4078(94.1%)	6545.64
Wlntcrk2.ph	(10,27)	3.6644*	3.3740(92.1%)	125.62	3.6798(100.4)	6869.75

† Land & Powell code was unable to solve these problems in 5 days on a 486-33MHz personal computer.
* Best known objective, not necessarily optimal.