TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No. FHWA/TX-87/21+356-2	2. Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle

NETWORK ASSIGNMENT METHODS FOR THE ANALYSIS
OF TRUCK-RELATED HIGHWAY IMPROVEMENTS

5. Report Date
November 1985

6. Performing Organization Code

7. Author(s)

Kyriacos Mouskos, Hani S. Mahmassani,
and C. Michael Walton

8. Performing Organization Report No.
Research Report 356-2

9. Performing Organization Name and Address

Center for Transportation Research
The University of Texas at Austin
Austin, Texas 78712-1075

10. Work Unit No.

11. Contract or Grant No.
Research Study 3-18-83-356

12. Sponsoring Agency Name and Address

Texas State Department of Highways and Public
Transportation; Transportation Planning Division
P. O. Box 5051
Austin, Texas 78763

13. Type of Report and Period Covered
Interim

14. Sponsoring Agency Code

15. Supplementary Notes

Study conducted in cooperation with the U. S. Department of Transportation, Federal Highway Administration
Research Study Title: "A Study of Truck Lane Needs"

16. Abstract

This report examines the application of the diagonalization algorithm to solve a two-class network equilibrium problem with asymmetric link interactions. The two classes that the traffic stream was divided into are passenger cars and trucks. Both traffic assignment rules, the User Equilibrium and System Optimum have been tested on three different networks. The third test network is a representation of the Texas highway network, thus providing a realistic case application.

An important feature developed and implemented in this study is a special structure of the network, where every link was coded in a way to account for exclusive lanes of either category of vehicles as well as common lanes for all traffic. This structure provides a tool to evaluate the performance of a network under different types of improvements involving the separation of the different categories of vehicles in the traffic stream.

The main aspects of the algorithm's performance examined in this study are its convergence characteristics as well as the effectiveness of some streamlining strategies aimed at improving its computational performance. Although convergence is not guaranteed, it was actually achieved in all the tests conducted, confirming the algorithm's appropriateness for this type of application. Furthermore, experience gained from the tests has identified powerful and relatively simple shortcuts for implementing the algorithm. Further research needed for implementation purposes is also discussed.

17. Key Words

highway improvements, truck-related, analysis, diagonalization algorithm, two-class network, passenger cars, trucks

18. Distribution Statement

No restrictions. This document is available to the public through the National Technical Information Service, Springfield, Virginia 22161.

19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 214	22. Price

NETWORK ASSIGNMENT METHODS FOR THE
ANALYSIS OF TRUCK-RELATED HIGHWAY IMPROVEMENTS

by

Kyriacos Mouskos
Hani S. Mahmassani
C. Michael Walton

Research Report Number 356-2

A Study of Truck Lane Needs
Research Project 3-18-83-356

conducted for

The Texas State Department of Highways and Public Transportation

by the

CENTER FOR TRANSPORTATION RESEARCH

THE UNIVERSITY OF TEXAS AT AUSTIN

November 1985

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the Federal Highway Administration. This report does not constitute a standard, specification, or regulation.

FOREWORD

This report documents the network traffic assignment procedures which constitute an essential component of the network modelling methodology developed for the study of truck lane needs in the Texas highway network. A general overview of the overall methodological approach, as well as a description of the model's capabilities and input requirements can be found in a companion report on the findings of study CTR 3-18-83-356. The present technical report is intended to fully document the research performed specifically in the development, refinement and testing of the traffic assignment procedures. The principal features of the assignment techniques presented here are: 1) the explicit consideration of two distinct classes of vehicles in the traffic stream; trucks and cars, 2) the modelling of interactions of these classes on highway links, in terms of the resulting effect on link travel times, and 3) the ability to represent and test the various link improvement options associated with the provision of special truck lanes, including restricted access of existing or new lanes to either vehicular class.

In addition to the theoretical and methodological aspects of these procedures, this report documents the computational experience conducted to develop guidelines for efficient implementation and use of a particular assignment algorithm, known as the diagonalization algorithm. The operational capability and usefulness of the model is demonstrated through application to the Texas highway network. The development of this network along with other more limited test networks is also documented.

In summary, a powerful tool has been developed to study the impact of implementing selected truck lanes on the highway system. It can benefit from further research in developing some of its inputs, particularly the link performance functions. While developed and adapted to the specific requirements of the truck lane needs study, this tool has broader applicability and can be used by the Texas SDHPT to analyze a variety of physical and operational improvements and measures aimed at coping with increasing truck traffic on the state's highway systems.

ABSTRACT

This report examines the application of the diagonalization algorithm to solve a two-class network equilibrium problem with asymmetric link interactions. The two classes that the traffic stream was divided into are passenger cars and trucks. Both traffic assignment rules, the User Equilibrium and System Optimum have been tested on three different networks. The third test network is a representation of the Texas highway network, thus providing a realistic case application.

An important feature developed and implemented in this study is a special structure of the network, where every link was coded in a way to account for exclusive lanes of either category of vehicles as well as common lanes for all traffic. This structure provides a tool to evaluate the performance of a network under different types of improvements involving the separation of the different categories of vehicles in the traffic stream. In particular, it can be used to evaluate the impacts of selected lane additions and exclusive lane designations aimed at coping with excessive truck traffic in certain parts of the network.

The main aspects of the algorithm's performance examined in this study are its convergence characteristics as well as the effectiveness of some streamlining strategies aimed at improving its computational performance. Although convergence is not guaranteed, it was actually achieved in all the tests conducted, confirming the algorithm's appropriateness for this type of application. Furthermore, experience gained from the tests has identified powerful and relatively simple shortcuts for implementing the algorithm. These shortcuts involve performing only a few "internal" iterations at each step of the algorithm instead of reaching an exact solution to a particular intermediate minimization problem. The results suggest the use of less than four internal iterations, with the use of two such iterations exhibiting the highest frequency of best performance in the tests conducted, followed by one and three internal iterations, respectively. Further research needed for implementation purposes is also discussed.

EXECUTIVE SUMMARY

This study is part of an integrated network modelling methodology which was developed to provide SDHPT engineers and planners with a tool to support the analysis, planning and design of highway link improvements aimed at coping with increasing truck sizes and flows in the network. An essential component of this methodology is the traffic assignment procedure, which allows the examination of the network-wide impacts of proposed link improvements. This report describes a traffic assignment approach, which is capable of producing the distribution of flows of different vehicle classes on the various links of the highway network. The traffic assignment approach used in this study takes into account the interaction of the passenger cars and trucks in the traffic stream. It can also readily be extended to account for a finer categorization of vehicles into more distinct classes.

The traffic assignment approach relies on the application of the diagonalization algorithm, which is used to distribute flows according to both traffic assignment rules (the User Equilibrium and System Optimum rules, respectively). This algorithm is capable of solving problems involving interaction between different classes of users operating on a given network. The algorithmic formulations for both the User Equilibrium and System Optimum are presented in this study as modified to account for two classes of users. Additionally, limited previous experience reported by other researchers on the diagonalization algorithm is briefly discussed.

The performance of the algorithm was tested on three different networks, including a coarsely aggregated representation of the Texas highway network, developed chiefly for testing purposes, in order to provide insight into the expected results for the larger more detailed version of the Texas network. In these networks, a special structure was devised to represent and test various improvement options with the provision and operation of special truck lanes, including restricted access of existing or new lanes to either cars or trucks.

The basic input required for the diagonalization algorithm, as in most traffic assignment methods, are the origin destination matrices for both classes of users, and the link characteristics required for the performance

functions. Unfortunately there exist no general-purpose calibrated link performance functions which take into account the interaction between passenger cars and trucks. In order to implement the algorithm the standard BPR functions developed for single class user were modified to take into account the interaction of both classes of users. The required parameters were then identified for all the links of the networks.

Implementation of the algorithm was achieved through the development of two computer programs, one for each of the User Equilibrium and System Optimum assignment rules respectively. The basic properties examined in each test include the convergence characteristics as well as possible shortcuts in implementing the algorithm so as to improve its computational performance.

An important conclusion of the tests conducted is that convergence was achieved for all tests. Since such convergence is not guaranteed a priori for this algorithm, the results of this study validate its applicability for the determination of truck lane needs and analysis of proposed related improvements in the Texas network. This conclusion was strengthened by the good performance of the algorithm for the full-scale detailed Texas test network. The second conclusion from the test results is that effective computational shortcuts can be adopted through streamlining strategies which achieve faster convergence of the algorithm. This in turn enhances the algorithm's applicability and usefulness for the analysis and design of truck related improvements to the highway network.

Given the encouraging positive results of these tests, it is recommended that further detailed development be conducted towards implementation of the algorithm. In particular, it is recommended that calibration of link performance functions based on actual observations of traffic behavior be conducted, in addition to the systematic development of the O-D matrices for the different classes of users. Furthermore, the representation of the appropriate highway network could be refined to better reflect local detail and address specific questions and improvements.

IMPLEMENTATION STATEMENT

The methodology developed in this study can assist the SDHPT in dealing with the questions of special lanes or facilities for truck traffic. Its applicability is however not limited to the analysis of exclusive truck lanes. It can handle a variety of highway link improvement options, involving capacity expansion jointly with operating strategies. The latter can include any combination of lane access restrictions to either cars or trucks, of existing as well as new lanes. As such, the network modelling methodology provides a flexible framework and tool to address a wide variety of measures aimed at relieving the problems associated with increasing flows of larger and heavier trucks in the highway system.

Naturally, some updating and fine-tuning of the network modelling methodology and its inputs to the specific needs of the implementing agency in any given problem situation is necessary. However, the requisite adaptability for such tasks is built into the structure of the methodology.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. INTRODUCTION

The purpose of this study is to examine the application of the diagonalization algorithm to solve a two-class network equilibrium problem with asymmetric link interactions resulting from shared use of the physical highway links by the two user classes. The convergence characteristics of this algorithm are studied under both the user equilibrium and the system optimum rules of traffic assignment. The two classes of vehicles that the traffic stream is divided into are the passenger cars and trucks. The distribution of the flows of these two groups on the network's links can provide valuable information to decision makers in the evaluation of changes and improvements to the transportation infrastructure and its operation.

1.1 Motivation

Over the past thirty years, there has been a considerable increase in the fleet of passenger cars and trucks, with an increasingly complex mix of vehicles in the traffic stream. Different types of vehicles are entering the road system, with different physical and performance characteristics. Recent trends toward less stringent regulations have allowed larger and heavier trucks in the highway system, jeopardizing geometric and capacity considerations in some parts of the system, and resulting in increased pavement deterioration rates. Furthermore the interaction of vehicles with different sizes and performance characteristics, such as large combination trucks on one hand and subcompact passenger cars on the other, may have resulted in more hazardous driving conditions, with increased potential severity of collisions.

The above concerns have led the appropriate agencies to consider the construction of exclusive facilities for different classes of users, as well as operational measures involving the restriction of access to existing selected lanes by certain vehicle types. The present work was conducted in conjunction with the development of a network modelling methodology for the identification and selection of good candidate highway links for the addition of special truck lanes. An essential element in a methodology to assess the impact of various selection criteria and proposed improvements is the prediction of the flows of both cars and trucks on the various links of the highway network. Flow prediction provides essential input to the analysis of the impacts on the highway users, carriers and shippers and on the operating agency, as well as to the determination of the costs and benefits of various link improvements.

1.2 General Background

The prediction of flows in transportation networks is an elaborate problem. Transportation science has provided many models for this purpose, employing both deterministic and stochastic approaches. A recent state-of-the-art review of these methods can be found in the text by Sheffi (1984). However to the extent that these models attempt to predict the outcome of human decisions, a certain amount of error is likely to be present in the results. It is difficult to collect the kind of data needed to determine all the factors that are taken into account by individuals in their route choice decision. Stochastic network assignment approaches attempt to account for this uncertainty through the specification of a random element in the route choice model. As such they are more general than their deterministic counterparts. However, they are more difficult to model and to solve, especially in the presence of link interactions, in which case existing algorithms for stochastic equilibrium assignment can be rather slow and inefficient and particularly costly in computational requirements. Furthermore, their potentially greater accuracy relative to deterministic approaches has not been verified. Therefore, since link interactions are of the essence, a deterministic network equilibrium approach based on the diagonalization algorithm is pursued in this study.

The principal variable that is used to determine the flows in the diagonalization algorithm, as well as in all traffic assignment procedures, is the travel time between an origin and a destination, taking into account congestion effects. Unlike most approaches currently found in practice, this algorithm takes into account the interaction between different classes of users sharing the transportation facilities through their respective effect on the travel time experienced by each category of vehicles. In addition, this interaction can be asymmetric, meaning that the marginal contribution of a vehicle belonging to a given category to the other class' travel time is different from the marginal contribution of a vehicle in the latter category to the former's travel time. This is expected to be the case in this study, where the two categories consist of passenger cars and trucks respectively.

It should be noted that the diagonalization algorithm has only recently received attention as a promising approach to solve for network equilibrium in the presence of asymmetric link interactions. In its complete version it is rather demanding computationally; however, some shortcuts have been suggested to improve this aspect, as discussed in the next chapter. However, these approaches remain to be tested, as current numerical experience seems to be limited to small unrealistic "toy" networks. A major objective of this study is to actually test these approaches and develop some computational experience in realistic networks, resulting in recommendations in view of its use as an operational tool to analyze truck-related improvements in a highway network.

As noted earlier, both User Equilibrium (UE) and System Optimum (SO) assignment rules are tested in this study. User Equilibrium assumes that each user behaves so as to minimize his/her own travel time (cost). The characterization of the User Equilibrium state is that no traveler can improve his travel time by unilaterally changing routes between any given origin and destination pair. These conditions do not generally imply that total travel time in the system is minimized. On the other hand, the System Optimum formulation minimizes the total travel time of all users in the network. The UE formulation is generally accepted as being more reasonable than the SO one, primarily because of its greater realism in depicting individual route choice behavior, whereby each user attempts to minimize his/her own travel time. In contrast, the SO assumptions do not seem as intuitively plausible, since it seems difficult to imagine that a tripmaker will always act, in the absence of special inducements or constraints, in such a way as to minimize the total travel time in the network, even if it means voluntarily using a longer route for one's particular trip. The SO formulation is however quite important for another reason, namely its role in network design models, which form the basis of the network modelling methodology for the selection of candidate links for truck related improvements, thus providing the motivation for its inclusion in this study.

1.3 Overview

This chapter has defined the problem addressed in this study and discussed its primary motivation in the general context of studies to analyze and design link improvements to deal with changing truck traffic in a highway network, as well as in the more specific context of network traffic assignment procedures. A more detailed description of the mathematical formulations of both the User Equilibrium and System Optimum approaches are presented in Chapter two, along with the associated assumptions. In addition the logic and structure of the diagonalization algorithm are presented in that chapter, and the results pertaining to the application of this algorithm to the two formulations are derived.

In Chapter three, the networks developed for this study are described, along with implementation details regarding the representation and coding of the truck-related improvements of interest. The convergence patterns associated with each network, based on the numerical testing conducted in this study are also presented in Chapter three. The principal results are summarized in the concluding fourth chapter, and application guidelines as well as recommendations for further research are presented.

The computer programs used to solve both the User Equilibrium and System Optimum formulations are presented in Appendix A, while the input data are presented in Appendix B. A listing of the computer programs is given in Appendix C, accompanied by a sample output of the programs.

CHAPTER 2 THE USER EQUILIBRIUM AND SYSTEM OPTIMUM FORMULATIONS, AND THE DIAGONALIZATION ALGORITHM

This chapter presents the assumptions, mathematical formulations and solution algorithms for the network traffic assignment problem, for both the user equilibrium and the system optimum decision rules, in the presence of multiple user classes with asymmetric interactions between the different classes. After discussing the two assignment rules, the diagonalization algorithm is presented for the solution of the network user equilibrium problem. Following the derivation of the mathematical formulation for the system optimum problem, the application of the diagonalization algorithm to this problem is discussed. The above mentioned user-equilibrium and system optimum decision rules are commonly attributed to Wardrop (1952). The system-optimum rule distributes the flows so as to minimize the total travel time experienced by users of the network under consideration. The user-equilibrium decision rule is a more realistic one, from a behavioral standpoint, since link flows at equilibrium satisfy the condition that no user from any given origin to any particular destination can improve his travel time (or cost) by unilaterally changing routes. The notion of equilibrium arises from the dependence of the link travel times on the link flows. The travel time (cost), in turn, is usually the criterion used to determine the flow pattern in a transportation network, thereby requiring the simultaneous solution of link flows and travel times in the network.

In the case where multiple user classes are present, the travel cost incurred by a particular user on a highway link depends on the characteristics of the link and the interaction between the volumes of all different classes of users utilizing that link. The user-equilibrium principle provides an abstraction and simplification of the complex real-world traffic assignment process. As typically implemented, it presumes that all users in a particular category are identical in their behavior, that they have full information about the network under consideration and that they consistently make correct decisions regarding route choice.

In order for the above assignment principles to yield operationally useful tools for planning and policy decisions, they have to be formulated mathematically in a manner that admits a computationally feasible solution procedure for large-scale networks. The user-equilibrium problem for a single user class was first formulated as a mathematical program by Beckman et. al. (1956). Practical exact solution algorithms started developing in the late 1960's and early 1970's. However, for the case of asymmetric interaction between the different classes of users, there is no presently known equivalent mathematical programming formulation for the user equilibrium. Nevertheless, several direct algorithms have been found to be successful in converging to the user equilibrium solution.

The system-optimum problem is easier to formulate due to the fact that there is an evident global function to minimize, which is the total travel time. In the remainder of this chapter, section 2.1 presents the diagonalization algorithm for the user equilibrium problem with asymmetric interaction between different classes of users, while section 2.2 presents the mathematical formulation of the system optimum network assignment problem under the same assumptions about the interaction among multiple user classes.

2.1 A Direct Algorithm For Solving The User-Equilibrium When There Is Asymmetric Interaction Between Different Classes of Users

As noted previously, no equivalent minimization program exists to solve for the equilibrium flows on the links in the case of asymmetric interaction between the different classes of users on a transportation network. In this section the diagonalization algorithm is briefly presented; a more detailed discussion can be found elsewhere, see Sheffi(1984).

In mathematical terms, the asymmetric interaction between the different classes can be expressed as follows:

$$\frac{\partial t_{ai}(x)}{\partial X_{aj}} \neq \frac{\partial t_{aj}(x)}{\partial X_{ai}} \qquad \forall\, ai \neq aj \qquad (2.1.1.)$$

where $t_{ai}(x)$ denotes the travel cost function of class i on link a which is dependent as the flow vector x of the different classes which use link a. Also X_{ai}, X_{aj} denote the flows of class i and class j on link a respectively. Relationship (2.1.1) can be stated as follows: The marginal contribution of the flow of class j, on the travel cost of class i on link a, is different from the marginal contribution of the flow of class i on the travel cost of class j on link a. These relationships are summarized in a general form in the Jacobian of the cost functions with respect to all flow classes; the Jacobian is the matrix of first order partial derivatives of these functions with respect to the flow of each class of users. The case of interest to this study is that where the Jacobian matrix is asymmetric. The Jacobian is denoted by $\Delta_x t$ and has the following form:

$$\nabla_x t = \begin{bmatrix} \frac{\partial t_{11}(x)}{\partial X_{11}} & \frac{\partial t_{12}(x)}{\partial X_{11}} \cdots & \frac{\partial t_{1I}(x)}{\partial X_{11}} \cdots \\ \frac{\partial t_{11}(x)}{\partial X_{12}} & \frac{\partial t_{12}(x)}{\partial X_{12}} \cdots & \frac{\partial t_{1I}(x)}{\partial X_{12}} \cdots \\ \vdots & \vdots & \vdots \\ \frac{\partial t_{11}(x)}{\partial X_{1I}} & \frac{\partial t_{12}(x)}{\partial X_{1I}} \cdots & \frac{\partial t_{1I}(x)}{\partial X_{1I}} \cdots \\ \vdots & \vdots & \vdots \end{bmatrix}$$

where I is the total number of user classes using a particular link

The interaction between the different classes on a given highway link is represented through the use of identical networks, all copies of each other, for each different class. In this way, each physical highway link is decomposed into as many "conceptual" links as the number of different user classes. Each of these links has its own cost function, and the flow on any given link consists of one designated class only. Interaction among the various classes using a particular physical link thus translates into interaction among links in this network representation, which is the more commonly found form of the network equilibrium problem with asymmetric link interactions.

The diagonalization algorithm involves solving a series of tractable UE programs. At the n-th iteration it fixes the crosslink effects at their current levels and solves the following UE mathematical program:

$$\min \tilde{Z}^n(x) = \sum_a \sum_i \int_0^{X_{ai}} t_{ai}\,(X^n_{a1},\ldots,X^n_{a,i-1},\, w,\, X^n_{a,i+1}\ldots X^n_{aI})\; dw \qquad (2.1.2a)$$

subject to

$$\sum_k f^{rs}_{ki} = q_{rsi} \qquad \forall k,r,s,i \qquad (2.1.2b)$$

$$f^{rs}_{ki} \geq 0 \qquad \forall k,r,s,i \qquad (2.1.2c)$$

where a denotes link a

i denotes class i

f_{ki}^{rs} denotes path k for traveler of class i from origin r to destination s

q_{rs} denotes the total flow of class i from origin r to destination s.

As mentioned before, the different classes are represented with "conceptual" links. Thus the final network has as many links as the physical network multiplied by the number of classes. The mathematical formulation of problem (2.1.2) can be expressed in the form of "conceptual" links as follows:

$$\min \tilde{Z}^n(x) = \sum_{\varepsilon} \int_0^{X_\varepsilon} t_\varepsilon(X_1^n, \ldots, X_{\varepsilon-1}^n, w, X_{\varepsilon+1}^n, \ldots, X_\varepsilon^n)\,dw \qquad (2.1.3a)$$

subject to

$$\sum_k f_k^{rs} = q_{rs} \qquad \forall k,r,s \qquad (2.1.3b)$$

$$f_k^{rs} \geq 0 \qquad \forall k,r,s \qquad (2.1.3c)$$

where ε denotes each link of the final network

X is the flow on link ε

f_k^{rs} denotes path k from origin r to destination s

and q_{rs} denotes the total flow from origin r to destination s.

This formulation is the same as the one presented in Sheffi (1984), where the interactions are also presented in terms of link flows.

For completeness of presentation purposes, the convex combinations algorithm, which solves the single class UE program, is first described.

STEP 0: Initialization. Perform all-or-nothing assignment based on $t_a = t_a(0)$, $\forall a$. This yields $\{X_a^1\}$. Set counter n: = 1

STEP 1: Update. Set $t_a^n = t_a(X_a^n)$, $\forall a$

STEP 2: Direction finding. Perform all-or-nothing assignment based on $\{t_a^n\}$. This yields a set of (auxiliary) flows $\{y_a^n\}$.

STEP 3: Line Search. Find α_n that solves

$$\min \sum_a \int_0^{X_a^n + \alpha_n (Y_a^n - X_a^n)} t_a(w)dw$$

subject to $0 \leq \alpha_n \leq 1$.

STEP 4: Set $X_a^{n+1} = X_a^n + \alpha_n (Y_a^n - X_a^n)$, $\forall a$

STEP 5: Convergence Test. If a convergence criterion is met stop (the current solution, $\{x_a^{n+1}\}$, is the set of equilibrium link flows); otherwise set n: = n+1 and GO TO STEP 1.

The above algorithm is most commonly known as Frank-Wolfe (1956). Its computational efficiency depends on the size of the network and the type of the travel cost functions. The step that requires more time to calculate is step three, where the shortest path is determined between an origin and a destination. Its popularity stems from the fact that it can handle very large networks. The same algorithm can be used to solve problem (2.1.3), where at the n-th iteration all cross link effects are fixed and the flow on one link depends only on its own flow. The Hessian of the program (2.1.2) is diagonal since all cross link effects are fixed; that is why the algorithm is called "diagonalization". The general steps of the diagonalization algorithm are given below.

STEP 0: Initialization. Find a feasible link flow vector x^n. Set n = 0.

STEP 1: Diagonalization. Solve subproblem (2.1.3). This yields a link flow vector x^{n+1}.

STEP 2: Convergence test. If $X^n \cong X^{n+1}$ STOP. If not set n = n + 1, and GO TO STEP 1.

Smith (1979) and Dafermos (1980) had shown that the equilibrium conditions can be formulated as a variational inequality, and uniqueness of the solution follows from a monotonicity assumption of the travel cost functions. Also Dafermos (1982) showed a formal proof of convergence of the diagonalization algorithm, requiring again that the cost interaction among the different classes be relatively weak. These conditions are, however, too strict. While they guarantee convergence, they are not necessary.

Researchers have reported success with this algorithm even when these conditions are violated, which is the case in this study. In addition, Sheffi (1984) presented a proof, following Abdulaal and LeBlanc (1979) that shows that if the algorithm converges, its solution is the equilibrium flow pattern, which is unique provided that the link-travel-time Jacobian is positive definite.

By noting that only the last iteration's flow pattern needs to be determined accurately, that problem [2.1.3] at each iteration is subject to the same set of constraints and that the solution of problem [2.1.3] is similar to the solution of a single user class, Sheffi (1984) suggested a "streamlined" version of the diagonalization algorithm, in an attempt to reduce the computational cost. It has to be noted that the convex combinations algorithm requires many iterations to reach convergence. Thus the solution of problem [2.1.3] is requiring a number of iterations to reach convergence per outer iteration of the diagonalization algorithm. The streamlined version applies only one iteration to problem [2.1.3], thus reducing it to a similar form as the convex-combination algorithm for a single user class. The streamlined algorithm is given below.

STEP 0: Initialization. Set $n = 0$. Find a feasible link-flow pattern vector x^n.

STEP 1: Travel time update. Set $t^n_{ai} = t_{ai}(X^n)$, $\forall\, a, i$

STEP 2: Direction finding. Assign O-D flows, $\{q_{rsi}\}$ to the network using the all-or-nothing based on $\{t^n_{ai}\}$. This yields a link flow pattern $\{y^n_{ai}\}$.

STEP 3: Move size determination. Find a scalar α_n, which solves the following program:

$$\min Z(\alpha_n) = \sum\sum_{ai} \int_0^{X^n_{ai} + \alpha_n (Y^n_{ai} - X^n_{ai})} t_{ai}(X^n_{a1}, \dots, X^n_{a,i-1}, w, X^n_{a,i+1}, \dots X^n_{an})\, dw$$

subject to $0 \leq \alpha_n \leq 1$

STEP 4: Update. Set $X^{n+1}_{ai} = X^n_{ai} + \alpha_n (y^n_{ai} - X^n_{ai})$, $\forall\, a, i$

STEP 5: Convergence test. If $X^{n+1}_{ai} \cong X^n_{ai}$ $\forall a, i$ STOP. The solution is x^{n+1}.

otherwise, set n - n+1 and go to STEP 1.

The streamlined version of the diagonalization algorithm was tested in this study. In addition, further tests were conducted, involving the solution of problem [2.1.3], using different numbers of maximum inner iterations, in order to examine the convergence pattern of these variants. The results are reported in Chapter 3. The following section presents the formulations of the system optimum program with multiclass user interaction.

2.2 The System-Optimum Formulation For Asymmetric Interactions Between the Different Users

The system-optimum formulation usually presumes the existence of some central agency, who knows a priori the O-D matrices of all different classes of users and assigns each traveler a definite path from its origin to its destination in a way that minimizes the total travel time in the network under consideration. Although this formulation overcomes the problem of user equilibrium, where no equivalent minimization program is found to exist for the case of asymmetric link interaction, its solution may not correspond to a stable condition. However, it can be used as a common measure of performance of a given network under different conditions. More importantly, it provides a lower bound to solutions of the UE program, which is particularly important for network design or link improvement selection problems. The equivalent minimization program is given below. The notation is the same as that used in the previous section.

$$\min \tilde{Z}(x) = \sum_{a}\sum_{i} X_{ai} t_{ai}(x) \qquad (2.2.1a)$$

subject to

$$\sum_{k} f_{ki}^{rs} = q_{rs} \qquad \forall\, r,s,i \qquad (2.2.1b)$$

$$f_{ki}^{rs} \geq 0 \qquad \forall\, k,r,s,i \qquad (2.2.1c)$$

This program is a minimization problem with linear equality and nonnegativity constraints. In order to find the necessary conditions for a minimum of the SO program the method of Lagrangian multipliers is used. These conditions are given by the first order conditions for a stationary point of the following Lagrangian program:

subject to the nonnegativity conditions

$$f_{ki}^{rs} \geq 0 \quad \forall\, k,r,s,i \qquad (2.2.2b)$$

The variable $\tilde{u}_{rsi}$ is the Lagrange multiplier associated with the flow conservation constraint of O-D pair r-s for class i. The first order conditions for a stationary point of the Lagrangian program are the following:

1st order conditions:

$$f_{ki}^{rs} \frac{\partial \tilde{L}(f,\tilde{u})}{\partial f_{ki}^{rs}} = 0 \quad \text{and} \quad \frac{\partial \tilde{L}(f,\tilde{u})}{\partial f_{ki}^{rs}} \geq 0 \quad \forall k,r,s,i \qquad (2.2.3a)$$

$$\frac{\partial \tilde{L}(f,\tilde{u})}{\partial \tilde{u}_{rsi}} = 0 \qquad \forall\, r,s,i \text{ (flow conservation)} \qquad (2.2.3b)$$

$$f_{ki}^{rs} \geq 0 \qquad \forall k,r,s,i \text{ (nonnegativity conditions)} \quad (2.2.3c)$$

writing equation [2.23a] explicitly.:

$$\frac{\partial \tilde{L}(f,\tilde{u})}{\partial f_{\ell i}^{mn}} = \frac{\partial}{\partial f_{\ell i}^{mn}} \tilde{Z}[x(f)] + \frac{\partial}{\partial f_{\ell i}^{mn}} \sum_{rsi}\sum \tilde{u}_{rsi}(q_{rsi} - \sum_{k} f_{ki}^{rs}) \quad \forall m,n,\ell,i \qquad (2.2.4)$$

The second term of the derivative yields the following:

$$\frac{\partial}{\partial f_{\ell i}^{mn}} \sum_{rsi}\sum \tilde{u}_{rsi} (q_{rs} - \sum_{k} f_{ki}^{rs}) = \tilde{u}_{mni} \qquad (2.2.5)$$

The first term yields the following:

$$\frac{\partial}{\partial f_{\ell i}^{mn}} \tilde{Z}[x(f)] = \sum_{bi}\sum \frac{\partial \tilde{Z}(x)}{\partial X_{bi}} \frac{\partial X_{bi}}{\partial f_{\ell}^{mn}} = \sum_{bi}\sum \frac{\partial \tilde{Z}(x)}{\partial X_{bi}} \delta_{bi,\ell i}^{mn}$$

$$= \sum_{bi}\sum \delta_{bi,\ell i}^{mn} \frac{\partial \sum_{ai}\sum X_{ai} t_{ai}(x)}{\partial X_{bi}}$$

$$= \sum_{bi}\sum \delta_{bi,\ell i}^{mn} [t_{bi}(x) + \sum_{ai}\sum X_{ai} \frac{dt_{ai}(x)}{dX_{ai}}], \quad \forall\, \ell,m,n,i \qquad (2.2.6)$$

Letting $\tilde{t}_{ai}(x) = t_{ai}(x) + \sum\limits_{ai}\sum X_{ai} \dfrac{dt_{ai}(x)}{dX_{ai}}$, $\forall a,i$ (2.2.7)

equation [2.2.6] can be written as

$$\frac{\partial \tilde{Z}\,[x(f)]}{\partial f^{mn}_{\ell i}} = \sum_{bi}\sum d^{mn}_{bi,\ell i}\tilde{t}_b = \tilde{C}^{mn}_{\ell i} \qquad (2.2.8)$$

$\tilde{t}_{ai}(X)$ can be interpreted as the marginal contribution of an additional traveler from each class on the a-th link to the total travel time of the user of class i on link a; whereas $\tilde{C}^{mn}_{\ell i}$ is the marginal total travel time induced by a user from class i on path ℓ connecting O-D pair m-n. The first order conditions of the SO program can now be written as

$f^{mn}_{\ell}(\tilde{C}^{mn}_{\ell i} - \tilde{u}_{mni}) = 0$ $\qquad \forall\ \ell,m,n,i$ (2.2.9a)

$\tilde{C}^{mn}_{\ell i} - \tilde{u}_{mni} \geq 0$ $\qquad \forall\ \ell,m,n,i$ (2.2.9b)

$\sum\limits_{\ell i}\sum f^{mn}_{\ell i} = q_{mni}$ $\qquad \forall\ m,n,i$ (2.2.9c)

$f^{mn}_{\ell i} \geq 0$ $\qquad \forall\ \ell,m,n,i$ (2.2.9d)

Equations [2.29a] and [2.29b] state that at optimality, the marginal total travel times on all used paths connecting a given O-D pair are equal. Any unused path has a marginal total travel time greater than or equal to the marginal total travel time of the used paths connecting an O-D pair. The marginal total travel time of all used paths between an O-D pair is given by the dual variable $\tilde{u}_{mni}$.

The sufficient condition needed to provide uniqueness of the SO program is for the Hessian of the objective function to be positive definite. The Hessian has the following form

$$\nabla^2\tilde{Z}(x) = \begin{bmatrix} \frac{\partial^2\tilde{Z}(x)}{\partial X_{a1}^2} & \frac{\partial^2\tilde{Z}(x)}{\partial X_{a1}\partial X_{a2}} \cdots & \frac{\partial^2\tilde{Z}(x)}{\partial X_{a1}\partial X_{aI}} & \cdots \\ & & \vdots & \\ \frac{\partial^2\tilde{Z}(x)}{\partial X_{a2}\partial X_{a1}} & \frac{\partial^2\tilde{Z}(x)}{\partial X_{a2}^2} \cdots & \frac{{}^2\tilde{Z}(x)}{\partial X_{a2}\partial X_{aI}} & \cdots \\ \vdots & \vdots & \vdots & \\ \frac{\partial^2\tilde{Z}(x)}{\partial X_{aI}\partial X_{a1}} & \cdots \quad \cdots & \frac{\partial^2\tilde{Z}(x)}{\partial X_{aI}^2} & \cdots \\ \vdots & & \vdots & \ddots \end{bmatrix}$$

where I is the total number of classes and X_{ai} denotes the flow of class i on link a. Note that the positive - definiteness of this Hessian cannot be established in the general case of asymmetric link interactions; therefore there is no a priori guarantee of uniqueness. However, since the principal motivation for solving this problem is to calculate the optimal value of the objective function, which serves as a lower bound in the discrete network design problem, non-uniqueness is not of major practical concern.

The system optimum flow pattern can be found using the diagonalization algorithm used in section 2.1 where the travel cost functions will be replaced by $\tilde{t}_{ai}(x)$ Eq. [2.2.7]. This method was used to compare the convergence patterns between the UE and SO multiclass user programs. As mentioned previously, only two classes of users are considered in this study; trucks and passenger cars. All results are reported in Chapter 3.

Although, in this study only the diagonalization (relaxation) method was used, it can be noted that other algorithms exist which solve these problems. The other major type of algorithms is referred to as the projection method. In a study conducted by Fisk and Nguyen (1982), the diagonalization method was found to be superior to the other algorithms. However, in a series of tests conducted by Nagurney (1983), the projection method was found to be superior to the relaxation method for some networks and inferior for some other networks. It was found that both the network structure and the type of the travel cost functions affect the efficiency of both methods, yet there are no general conclusions as to which method is more efficient. The diagonalization algorithm is easier to implement and interpret and is more widely used in the research community. Because of its streamlining possibilities, it was selected for this study, and tests were conducted to investigate the best streamlining strategy for the type of network and link interactions of interest.

The next chapter presents a description of the test networks used in this study and the numerical results for the convergence pattern of the diagonalization algorithm for both the SO and UE programs.

CHAPTER 3 NUMERICAL RESULTS ON THE DIAGONALIZATION ALGORITHM

This chapter presents the performance of the diagonalization algorithm, tested on different networks, where two classes of users were considered to operate, each class interacting with the other in an asymmetric way. The principal measure of performance used was the total number of internal iterations needed for the algorithm to reach the convergence criterion. The total number of internal iterations is the sum of the required number of internal iterations to solve the mathematical problem of STEP 1 of the algorithm described in section 2.1 in Chapter 2, per outer iteration, until convergence is reached. It should be noted that each internal iteration requires as many shortest path calculations as the number of O-D pairs. Three networks have been developed and a series of tests conducted on each one. Both the User Equilibrium and System Optimum traffic assignment rules were applied on each network. The first network developed is a hypothetical one whereas Networks 2 and 3 were developed from the Texas highway network. Network 2 was intended as a medium-sized abstraction of the Texas highway network to be used for methodological development and testing purposes. It was intended to capture the basic features of the state network with a minimum of unessential detail. On the other hand, Network 3 provides a more detailed representation of the Texas situation and can be used for actual planning studies.

A description of each network's characteristics is given hereafter in addition to the convergence patterns of the diagonalization algorithm. As mentioned before, the basic information to perform the traffic assignment is a graph representation of the network, the O-D matrix for each class of users and the performance functions of the links of the network. A description of the travel cost functions used in this study is presented in section 3.1, whereas sections 3.2, 3.3 and 3.4 describe networks 1,2, and 3 respectively, finally closing this chapter with section 3.4 which summarizes the results.

3.1 Travel Cost Functions

The travel cost functions are an integral part of the traffic assignment methodology. Unfortunately, there has been virtually no research on the functional forms and parameter estimates for travel cost functions which take into account the asymmetric interaction between the two classes of vehicles comprising the traffic stream in this study; the trucks and the passenger cars. However, some research has been conducted on the development of travel cost functions relating the travel time of a passenger car on a link to the flow (of passenger cars) on that link. Some of these include the travel cost functions developed by the U.S. Bureau of Public Roads (BPR) in 1964, Davidson (1966), Mosher (1963), Wardrop (1968) etc. In a review carried out by Branston (1976) many of the link performance functions were studied and it was concluded that it is difficult to identify the most suitable form of travel cost functions which can be used for any kind of network due to the lack of data. In this study, the BPR curves were chosen to be used in a modified form to take into account the interaction between trucks and passenger cars. The modification used is based on engineering considerations, not actual empirical observations, and thus might not represent accurately the actual interaction between the passenger cars and trucks. However, it is consistent with the accepted treatment of trucks in traffic engineering practice, and is believed to provide a good representation to serve as a tool to test the algorithm. The original formulation of the BPR curves is presented below, followed by the modified version.

As mentioned previously, the travel cost functions developed by the U.S. Bureau of Public Roads (BPR), relate the travel time of a vehicle on a link to the flow on that link, i.e. $t_a = f(X_a)$. These functions have the following form:

$$t_a = t_a^o \left[1 + \alpha \left(\frac{X_a}{C'_a} \right)^{\beta} \right]$$

where t_a is the travel time on link a

t_a^o is the free flow travel time on link a

α, β are parameters calibrated on the basis of the speed limit and the capacity of the link.

Table 3.1 – Volume / delay functions (Florian et. al. 1976)

Type	Speed limit mph	α	β	Free flow times/minutes per mile t_o
1	0-30	.731	3.660	15.0/4.00
2	0-30	.613	3.504	17.0/3.53
3	0-30	.877	4.461	20.0/3.00
4	0-30	.685	5.164	23.0/2.61
5	0-30	1.146	4.424	25.0/2.40
6	31-40	.619	3.654	30.0/2.00
7	31-40	.666	4.943	32.4/1.85
8	31-40	.622	5.141	32.4/1.85
9	31-40	1.030	5.523	35.3/1.70
10	41-50	.661	5.091	41.4/1.45
11	41-50	.542	5.789	41.4/1.45
12	41-50	1.009	6.586	41.4/1.45
13	+50	.878	4.929	55.0/1.09
14	+50	.770	5.344	55.0/1.09
15	+50	1.149	6.868	55.0/1.09

C'_a is the "practical capacity" of the link.

and X_a is the flow on link a

The BPR engineers suggested values of 1.15 and 4 for α and β respectively. Florian, et.al. (1979) calibrated the BPR curves using data collected at Winnipeg. Table 3.1 shows the estimated values of parameters a and b and the corresponding free flow travel time. These values were used also in the modified form in this study.

The modified version of the above travel cost functions has the following form:

$$t_{aA} = t^o_{aA} \left[1 + \alpha_A \left(\frac{X_{aA} + \varepsilon \, . \, X_{aT}}{C'_a} \right)^{\beta_A} \right]$$

$$t_{aT} = t^o_{aT} \left[1 + \alpha_T \left(\frac{X_{aA} + \varepsilon \, . \, X_{aT}}{C'_a} \right)^{\beta_T} \right]$$

Where t_{aA}, t_{aT} are the travel times of the passenger cars and trucks on link a respectively.

t^o_{aA}, t^o_{aT} are the free flow times of the passenger cars and trucks on link a respectively

α_A, β_A and α_T, β_T are parameters specific for the passenger cars and trucks respectively

C'_a is the capacity of link a in passenger car equivalents per unit time

X_{aA}, X_{aT} are the flows of the passenger cars and trucks on link a respectively

ε is a parameter transforming the trucks to passenger car equivalents.

Thus, the flow X_a from the original formulation is decomposed into X_{aA} and $\varepsilon .X_{aT}$, in an attempt to capture the interaction between the two classes of users. In order for the above model to become more useful, data should be collected to calibrate the parameters α and β and estimate them for each link of the network under consideration. However, the same value as the ones calibrated by Florian were used in this study. In order to distinguish between the passenger car parameter values and the trucks, values of a lower category curve were assigned to the trucks than the higher one selected for the passenger cars. Again, this is a methodological rather than an actual representation decision. The value of ε was selected to be 4 passenger car equivalents, taken as an average value from the Highway Capacity Manual (1965). Also here the value of ε for each link can be modified to better represent the characteristics of the link according to the new Highway Capacity Manual (1985). The following section presents a description of the networks tested, as well as the computational results of each test.

3.2 Performance of the Diagonalization Algorithm

The main goal of this study was to examine the convergence characteristics of the diagonalization algorithm. These are viewed from two perspectives. First, whether the algorithm converges and second, the convergence pattern of the algorithm. Of primary importance is the performance of this algorithm on large networks, where if applicable it can be very useful. In this study the algorithm was tested on a large network, Network 3, which was developed to represent the Texas highway system for the truck related improvement projects (see Mahmassani et al, 1985). Another goal of this study was to examine possible shortcuts of the diagonalization algorithm for faster convergence. The process followed in this study was to solve STEP 1 (see section 2.1) of the algorithm approximately. For each network, a series of runs, each with a maximum number of internal iterations ranging from 1 to 10 was used, except in Network 3 which was tested up to 5 internal iterations due to computational time considerations. It is to be noted that STEP 1 of the algorithm requires the solution of the mathematical program (2.1.1) to convergence. However, since only the last vector (the updated flows at convergence) of flows is of concern, then approximate solutions at the intermediate steps do not affect the final solution as discussed in Chapter two. Therefore, by examining first if convergence was achieved and second the performance of each of the ten maximum number of internal iterations used to solve STEP 1 approximately, it may be possible to identify guidelines regarding a possible "optimum" number which minimizes the total number of internal iterations needed to reach the equilibrium solution. Next, a description of the networks and a summary of the results are presented.

3.2.1 Network 1

Network 1 is depicted in Fig. 3.2.1.1, and the listing of the corresponding input data is given in Appendix B.1. The main features of Network 1 are the following:

Passenger car network coding:

Number of centroids: 11

Number of O-D pairs: 110

Number of centroid connectors: 11

Number of egress links: 11

Number of access links: 11

Number of one way highway links: 68

Origin nodes: from 1 to 11

Destination nodes: from 12 to 22

Highway nodes: from 23 to 39

As mentioned previously (section 2.1), the truck network is a replica of the passenger car network. The truck network node numbers follow sequentially those of the passenger car network nodes.

Truck network coding:

Origin nodes: from 40 to 50

Destination nodes: from 51 to 61

Highway nodes: from 62 to 78

For computational purposes, both networks are considered as one, thus depicting the following features:

Total number of O-D pairs: 220

Total number of nodes: 78

Total number of links: 202

As mentioned previously (Section 3.2), a series of tests to examine the convergence pattern was conducted by ranging the maximum number of internal iterations from 1 to 10 (inclusive). The results, for each maximum number of internal iterations, are presented in tables 3.2.1.1 to 3.2.1.10, respectively for the User Equilibrium assignment principle. Each table depicts the following information:

1. The number of internal iterations to reach either internal convergence or the maximum allowable number of internal iterations, whichever is lower, corresponding to each external (or outer) iteration of the

algorithm. This item is shown in column two, with the corresponding outer iteration number given in the first column of the table.

2. The sum of the internal iterations, up to the current outer iteration, given in column three.

3. The current level of the convergence measure is given in column four. In this study the following convergence measure was used:

$$\frac{1}{A} \sum_{a} \frac{|X_a^{n+1} - X_a^{n}|}{X_a^{n+1}} \leq k$$

where A is the total number of links (both the car and truck links), a denotes each link and k is a constant (.005 was used in all tests). This convergence measure is the summation of the absolute difference of the updated flows from the previous iteration's flows divided by the updated flow, divided by the total number of links.

In addition the CPU time needed for the algorithm to converge is given. the convergence measure versus the number of outer iterations is plotted in figures 3.2.1.2 to 3.2.1.11. A summary of the results for the User Equilibrium is given in Table 3.2.1.11. This table depicts the following information:

1. The total number of internal iterations required for convergence for each maximum allowable number of internal iterations is given in column three. The maximum allowable number of internal iterations is given in column one.

2. The corresponding required CPU time to reach convergence is given in column two.

Similar results for the System Optimum assignment principle were developed. Tables 3.2.1.12 to 3.2.1.21 present the results for each maximum allowable number of internal iterations ranging from 1 to 10 respectively. The convergence measure versus the number of outer iterations is plotted in Figures 3.2.1.12 to 3.2.1.21. Table 3.2.1.22 summarizes the results in a similar manner to that of the User Equilibrium.

3.2.2 Network 2

A graph representation of Network 2 is depicted in figure 3.2.2.1. This network is a highly aggregated version of Network 3 which was developed from

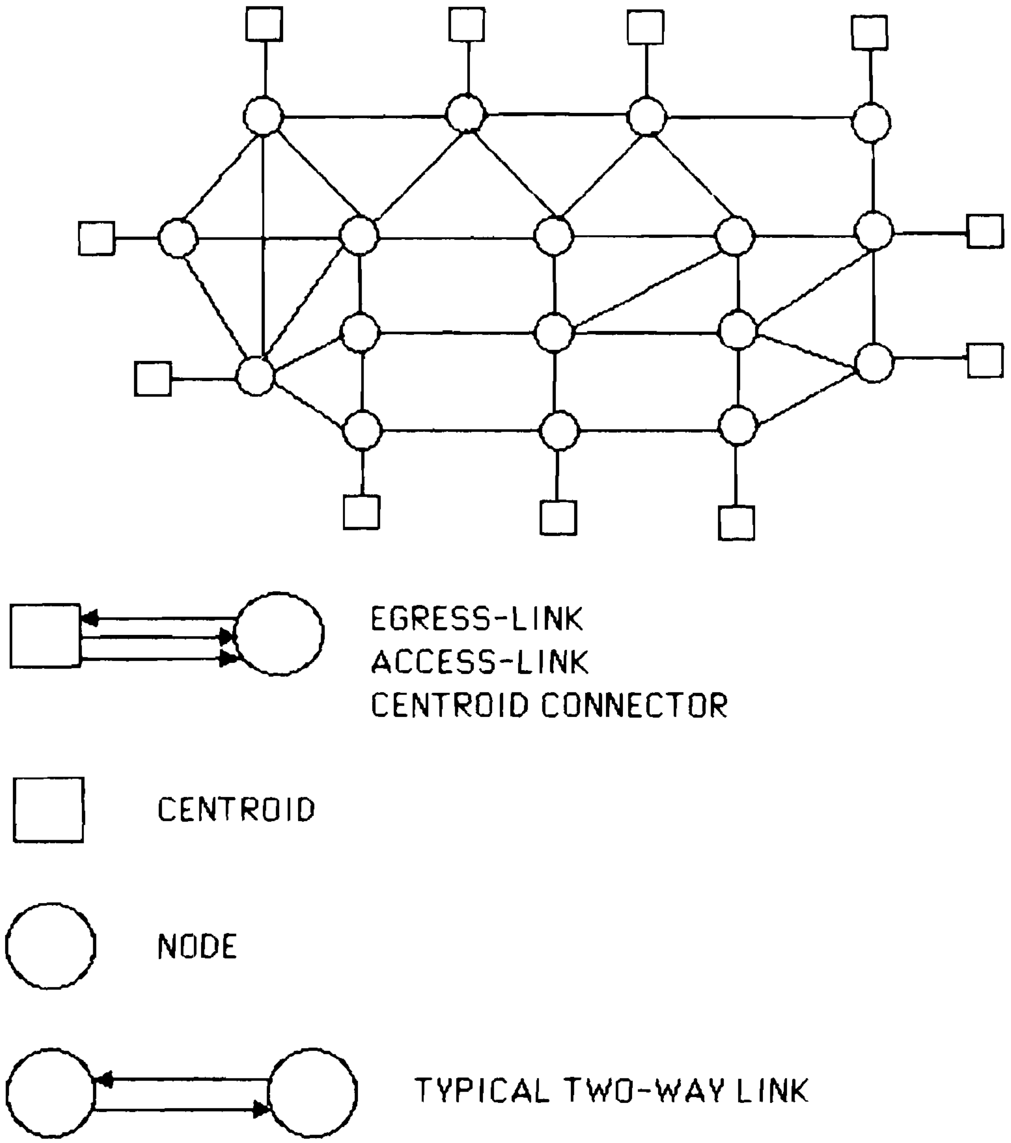

FIG. 3.2.1.1 NETWORK 1

Network #1 - User Equilibrium CPU = 9.994 seconds

Maximum # of Internal Iterations = 1

Table 3.2.1.1

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	1	1	.302
2	1	2	.512
3	1	3	.198
4	1	4	.125
5	1	5	.188
6	1	6	.110
7	1	7	.064
8	1	8	.089
30	1	30	.006
31	1	31	.004

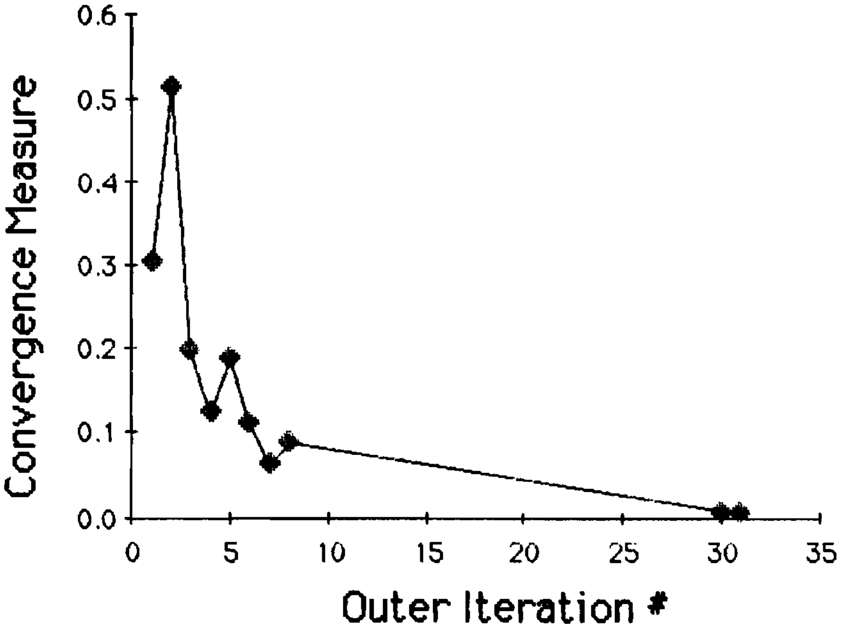

Fig.3.2.1.2

Network #1 - User Equilibrium CPU = 13.403 seconds

Maximum # of Internal Iterations = 2

Table 3.2.1.2

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	2	2	.386
2	2	4	.476
3	2	6	.613
4	2	8	.302
5	2	10	.176
6	2	12	.113
7	2	14	.080
8	2	16	.034
22	1	43	.003

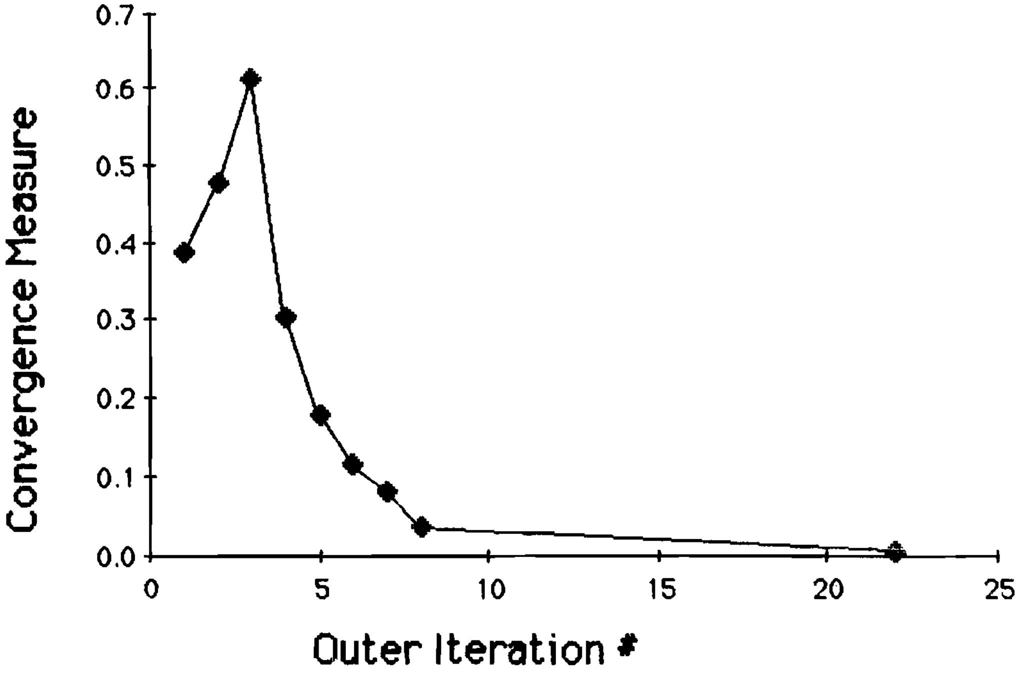

Fig. 3.2.1.3

Network *1 - User Equilibrium CPU = 15.944 seconds

Maximum # of Internal Iterations = 3

Table 3.2.1.3

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	3	3	.444
2	3	6	.840
3	3	9	.225
4	3	12	.260
5	3	15	.174
6	3	18	.126
7	3	21	.102
8	3	24	050
18	1	52	.004

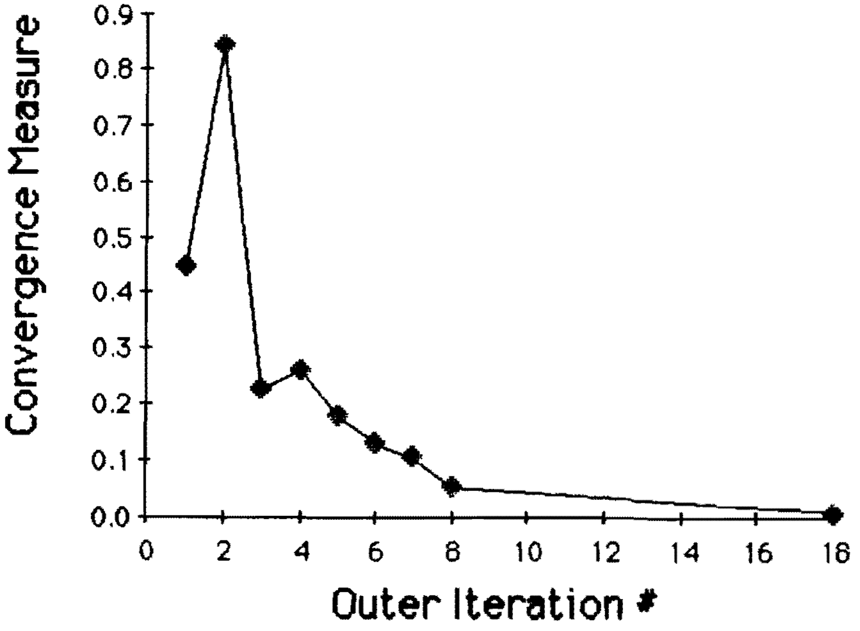

Fig. 3.2.1.4

Network #1 - User Equilibrium CPU =18.624 seconds

Maximum # of Internal Iterations = 4

Table 3.2.1.4

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	4	4	.584
2	4	8	.581
3	4	12	.259
4	4	16	.201
5	4	20	.153
6	4	24	.115
7	4	28	.091
8	4	32	.072
16	1	61	.004

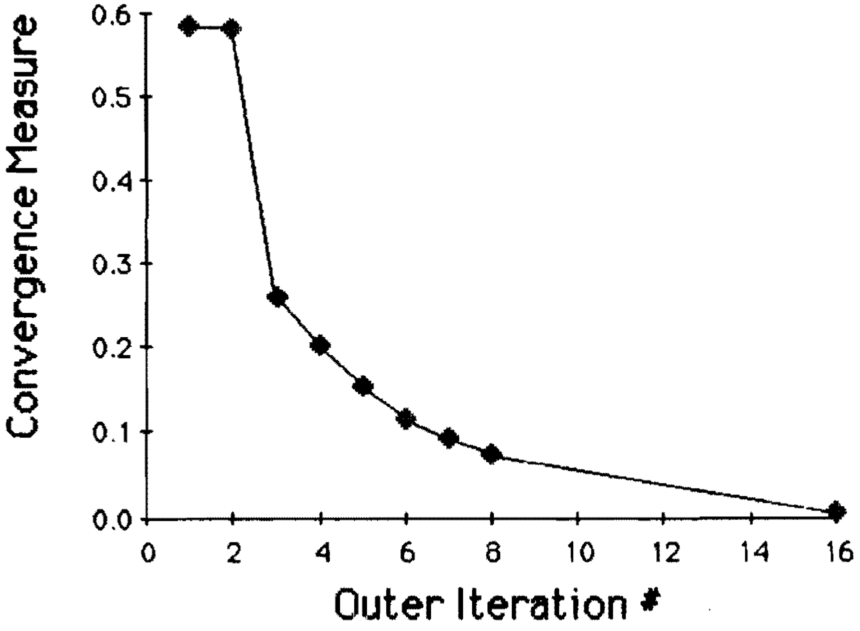

Fig. 3.2.1.5

Network #1 - User Equilibrium CPU =18.367 seconds

Maximum # of Internal Iterations = 5

Table 3.2.1.5

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	5	5	.466
2	5	10	.546
3	5	15	.256
4	5	20	.276
5	5	25	.152
6	5	30	.148
7	5	35	.068
8	5	40	.064
13	1	60	.005

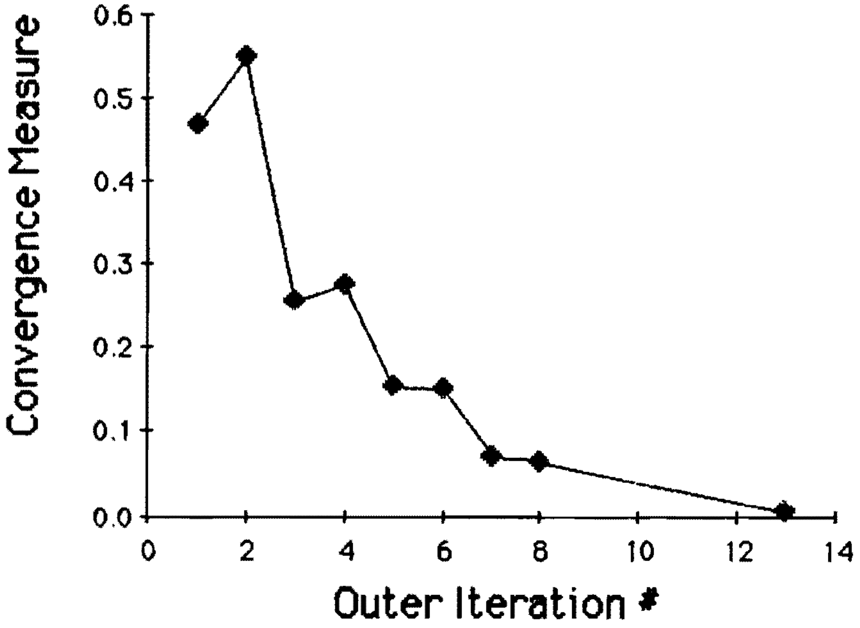

Fig. 3.2.1.6

Network #1 - User Equilibrium CPU =23.161 seconds

Maximum # of Internal Iterations = 6

Table 3.2.1.6

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	6	6	.491
2	6	12	.881
3	6	18	.249
4	6	24	.274
5	6	30	.153
6	6	36	.156
7	6	42	.083
8	6	48	.072
14	1	77	.004

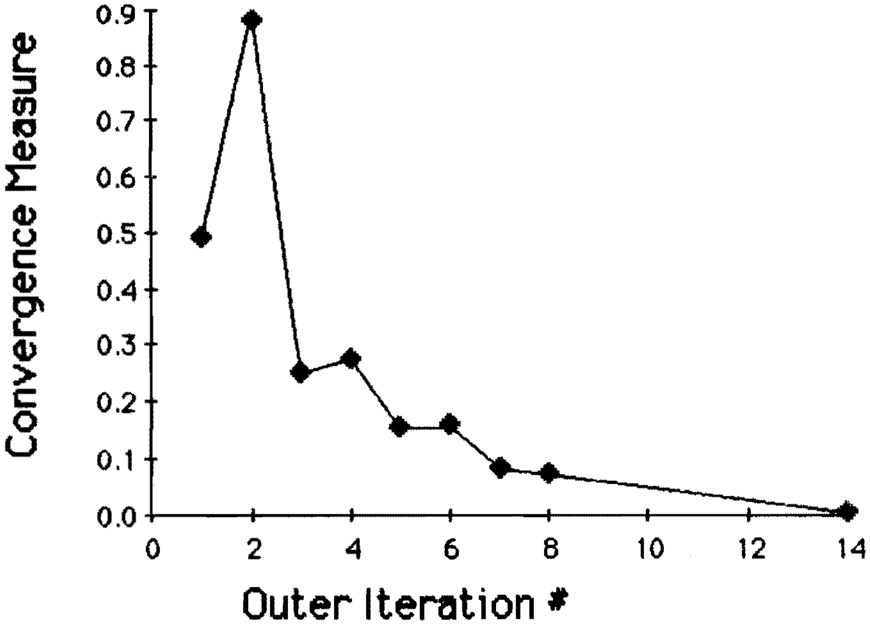

Fig. 3.2.1.7

Network #1 - User Equilibrium CPU =25.870 seconds

Maximum # of Internal Iterations = 7

Table 3.2.1.7

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	7	7	.523
2	7	14	.910
3	7	21	.291
4	7	28	.403
5	7	35	.213
6	7	42	.209
7	7	49	.114
8	7	56	.098
14	1	86	.004

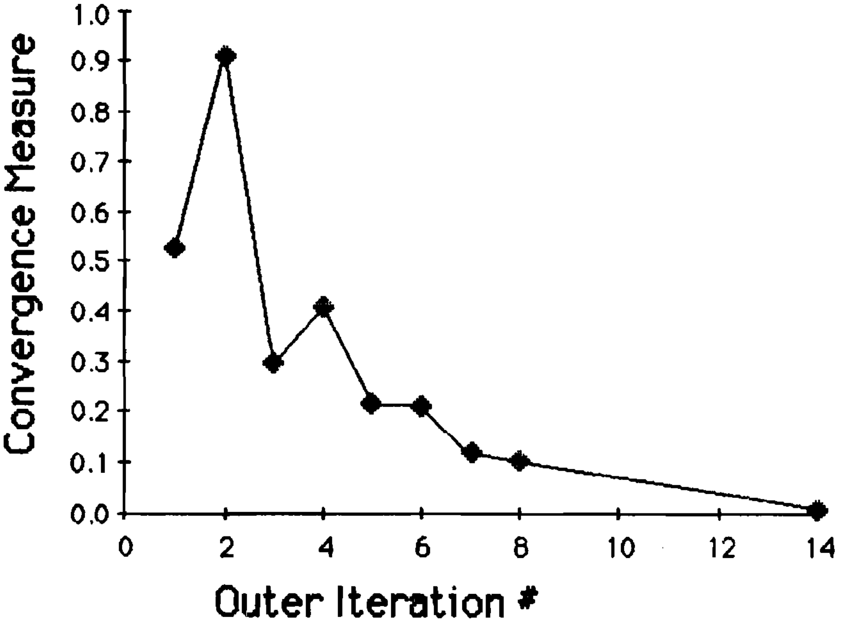

Fig. 3.2.1.8

Network #1 - User Equilibrium CPU =29.105 seconds

Maximum # of Internal Iterations = 8

Table 3.2.1.8

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	8	8	.492
2	8	16	.929
3	8	24	.524
4	8	32	.285
5	8	40	.123
6	8	48	.182
7	8	56	.090
8	8	64	.095
15	1	97	.003

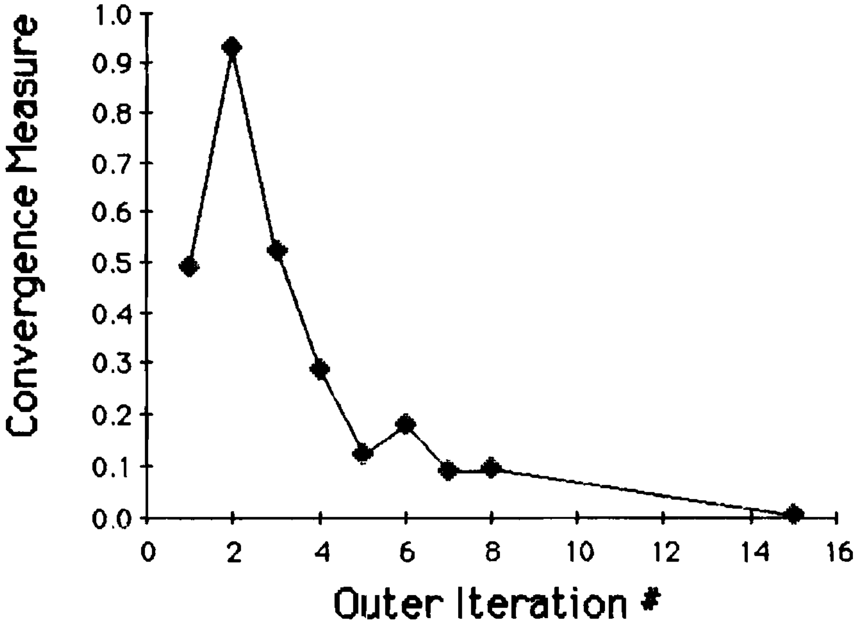

Fig. 3.2.1.9

Network #1 - User Equilibrium CPU =32.699 seconds

Maximum # of Internal Iterations =9

Table 3.2.1.9

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	9	9	.509
2	9	18	.113
3	9	27	.285
4	9	36	.321
5	9	45	.147
6	9	54	.180
7	9	63	.083
8	9	72	.088
18	1	109	.003

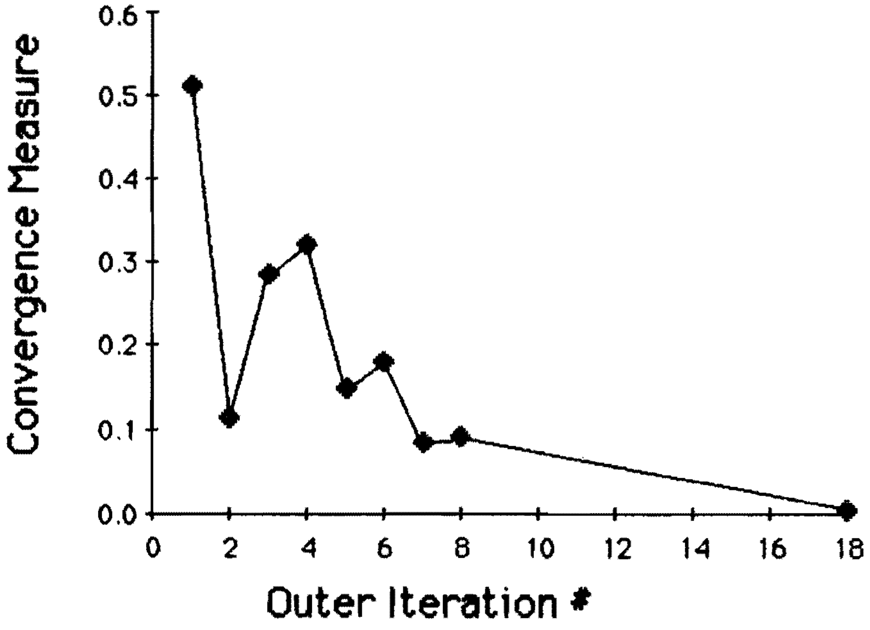

Fig. 3.2.1.10

Network #1 - User Equilibrium CPU =33.304 seconds

Maximum # of Internal Iterations = 10

Table 3.2.1.10

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	10	10	.492
2	10	20	.106
3	10	30	.360
4	10	40	.413
5	10	50	.166
6	10	60	.215
7	10	70	.103
8	10	80	.103
15	1	111	.004

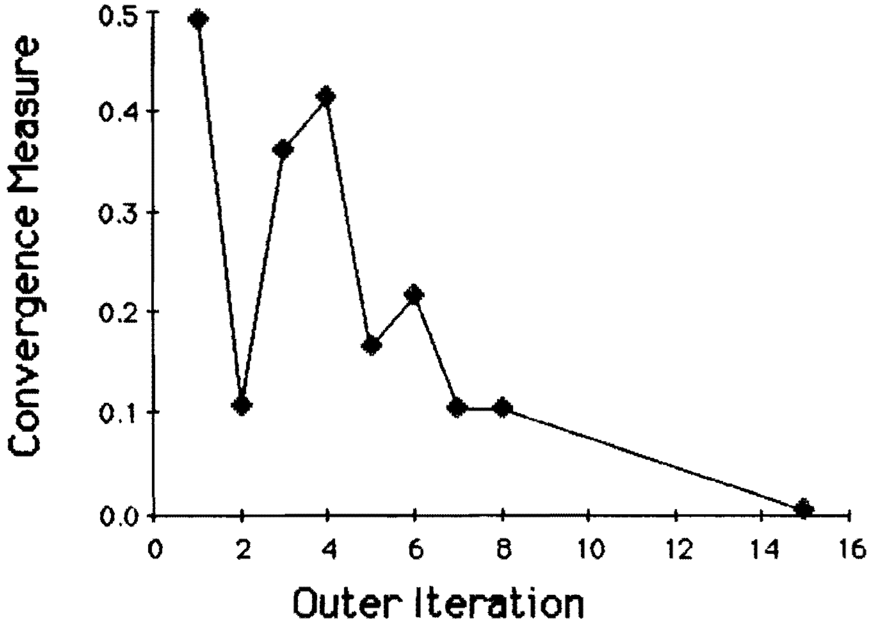

Fig. 3.2.1.11

Table 3.2.1.11- Summary of results for Network 1 User Equilibrium

Maximum Number of Internal Iterations	CPU - Time (Seconds)	Total Number of Internal Iterations required for Convergence
1	9.994	31
2	13.403	43
3	15.944	52
4	18.624	61
5	18.367	60
6	23.161	77
7	25.870	86
8	29.105	97
9	32.699	107
10	33.304	111

Network #1 - System Optimum CPU = 28.417 seconds

Maximum # of Internal Iterations = 1

Table 3.2.1.12

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	1	1	.384
2	1	2	.175
3	1	3	.172
4	1	4	.010
5	1	5	.066
6	1	6	.103
7	1	7	.057
8	1	8	.091
30	1	30	.017
60	1	60	.011
65	1	65	.005

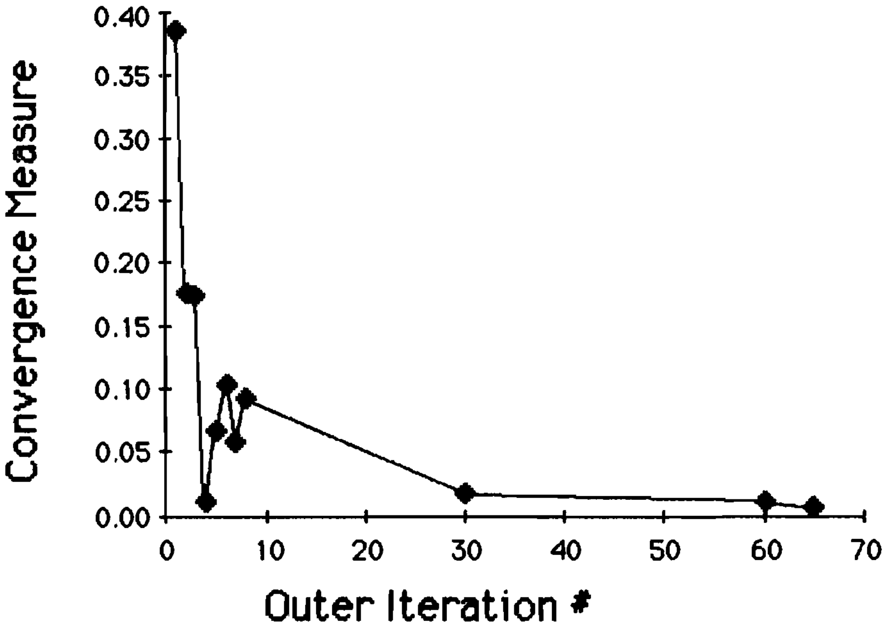

Fig. 3.2.1.12

Network #1 - System Optimum CPU =24.110 seconds

Maximum # of Internal Iterations = 2

Table 3.2.1.13

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	2	2	.451
2	2	4	.278
3	2	6	.108
4	2	8	.123
5	2	10	.063
6	2	12	.100
7	2	14	.083
8	2	16	.071
28	1	55	.005

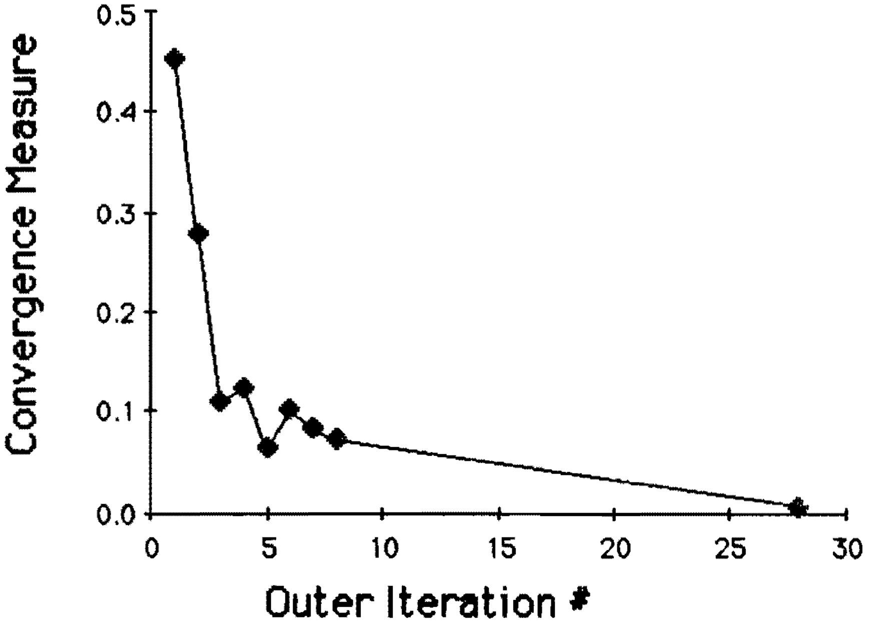

Fig. 3.2.1.13

Network #1 - System Optimum CPU =32.481 seconds

Maximum # of Internal Iterations = 3

Table 3.2.1.14

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	3	3	.492
2	3	6	.453
3	3	9	.231
4	3	12	.427
5	3	15	.159
6	3	18	.116
7	3	21	.077
8	3	24	.096
28	1	75	.004

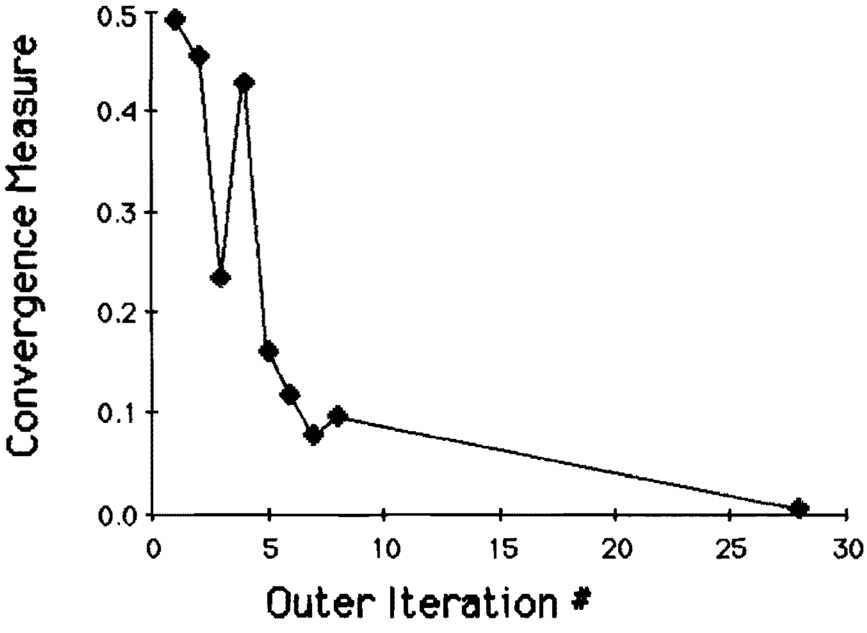

Fig. 3.2.1.14

Network #1 - System Optimum CPU =42.798 seconds

Maximum # of Internal Iterations =4

Table 3.2.1.15

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	4	4	.596
2	4	8	.594
3	4	12	.249
4	4	16	.301
5	4	20	.140
6	4	24	.160
7	4	28	.082
8	4	32	.095
29	1	100	.004

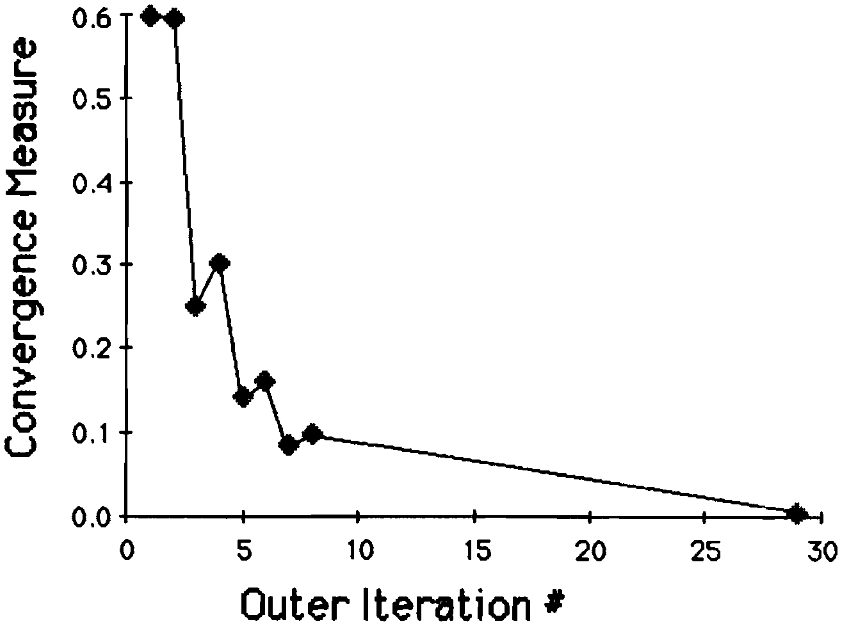

Fig. 3.2.1.15

Network #1 - System Optimum CPU =38.908 seconds

Maximum # of Internal Iterations =5

Table 3.2.1.16

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	5	5	.617
2	5	10	.674
3	5	15	.274
4	5	20	.336
5	5	25	.192
6	5	30	.270
7	5	35	.160
8	5	40	.187
19	1	91	.005

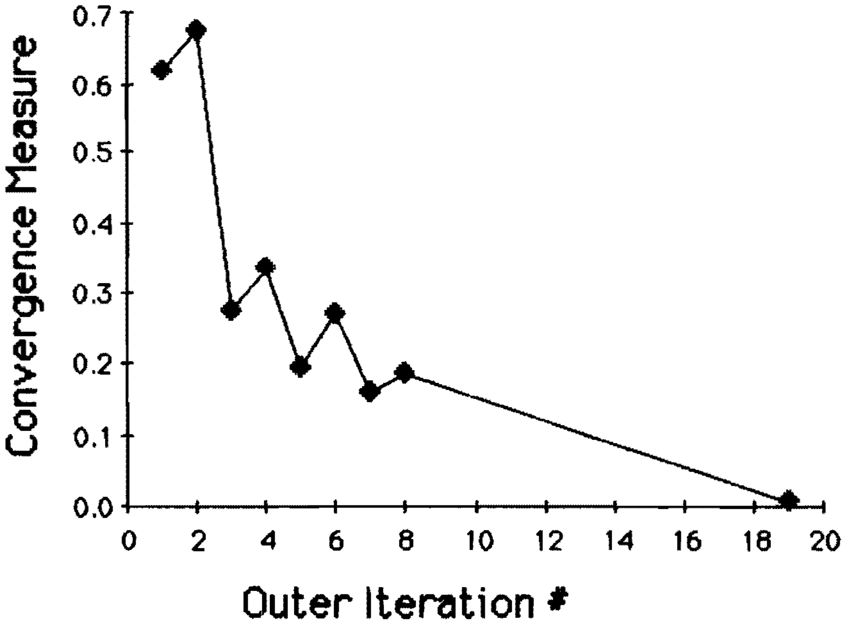

Fig. 3.2.1.16

Network #1 - System Optimum CPU =44.263 seconds

Maximum # of Internal Iterations =6

Table 3.2.1.17

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	6	6	.627
2	6	12	.712
3	6	18	.290
4	6	24	.372
5	6	30	.182
6	6	36	.218
7	6	42	.116
8	6	48	.127
20	1	103	.003

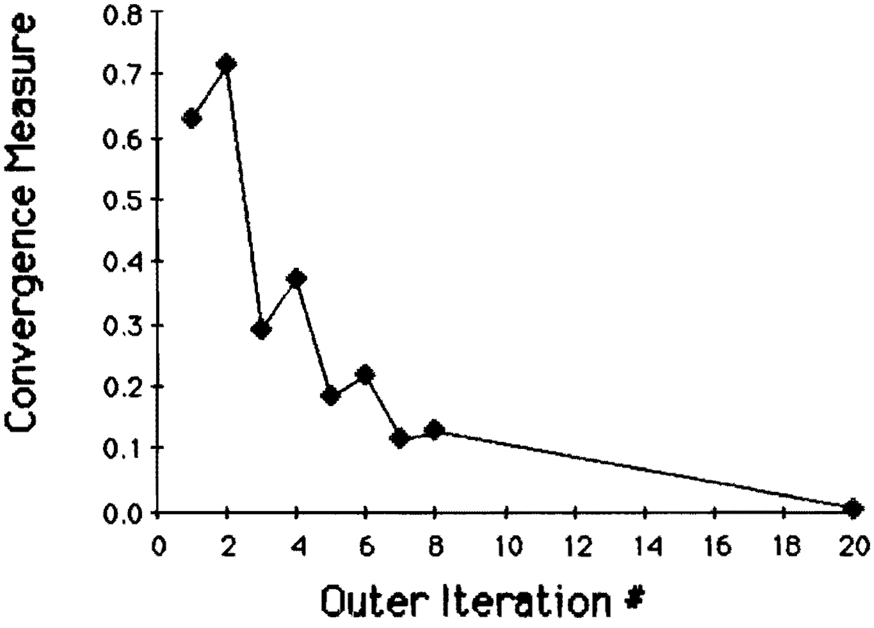

Fig. 3.2.1.17

Network #1 - System Optimum CPU =52.259 seconds

Maximum # of Internal Iterations =7

Table 3.2.1.18

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	7	7	.690
2	7	14	.716
3	7	21	.304
4	7	28	.400
5	7	35	.203
6	7	42	.274
7	7	49	.150
8	7	56	.201
19	1	122	.004

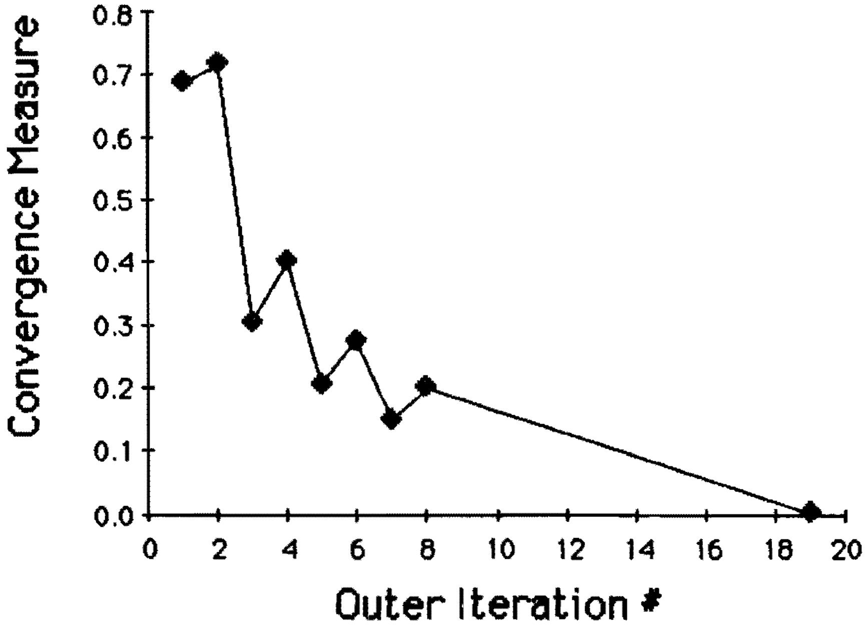

Fig. 3.2.1.18

Network #1 - System Optimum CPU =69.464 seconds

Maximum # of Internal Iterations =8

Table 3.2.1.19

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	8	8	.712
2	8	16	1.012
3	8	24	.375
4	8	32	.608
5	8	40	.286
6	8	48	.460
7	8	56	.230
8	8	64	.378
23	1	163	.005

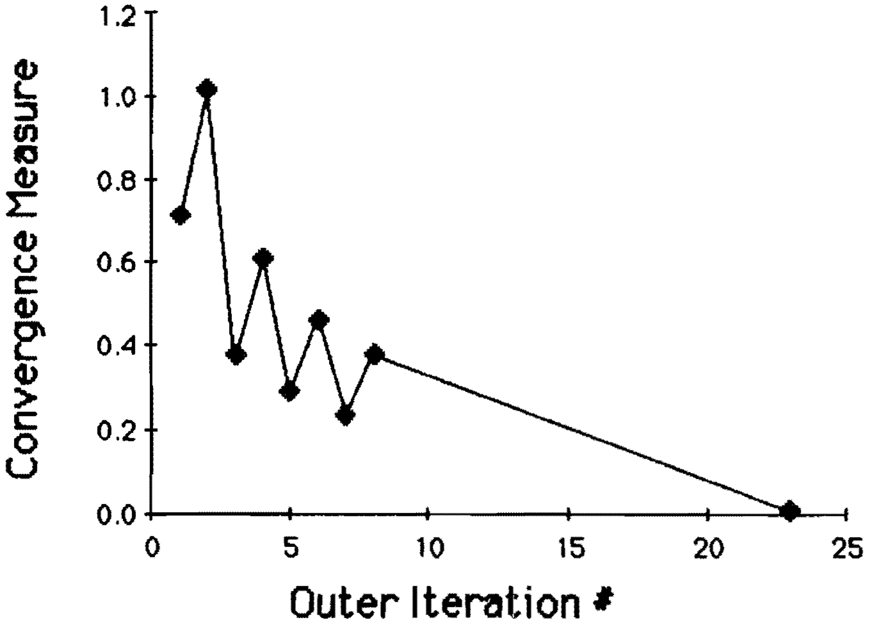

Fig. 3.2.1.19

Network #1 - System Optimum CPU =75.347 seconds

Maximum # of Internal Iterations =9

Table 3.2.1.20

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	9	9	.798
2	9	18	1.016
3	9	27	.372
4	9	36	.623
5	9	45	.271
6	9	54	.440
7	9	63	.231
8	9	72	.374
22	1	177	.004

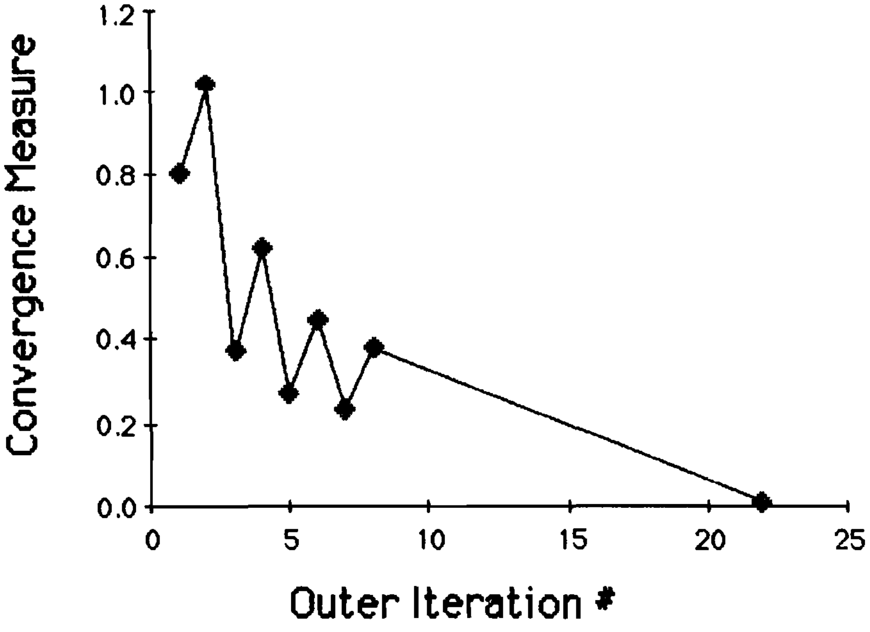

Fig. 3.2.1.20

Network #1 - System Optimum CPU =87.126 seconds

Maximum # of Internal Iterations =10

Table 3.2.1.21

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	10	10	.816
2	10	20	1.237
3	10	30	.397
4	10	40	.726
5	10	50	.311
6	10	60	.511
7	10	70	.230
8	10	80	.396
24	1	205	.004

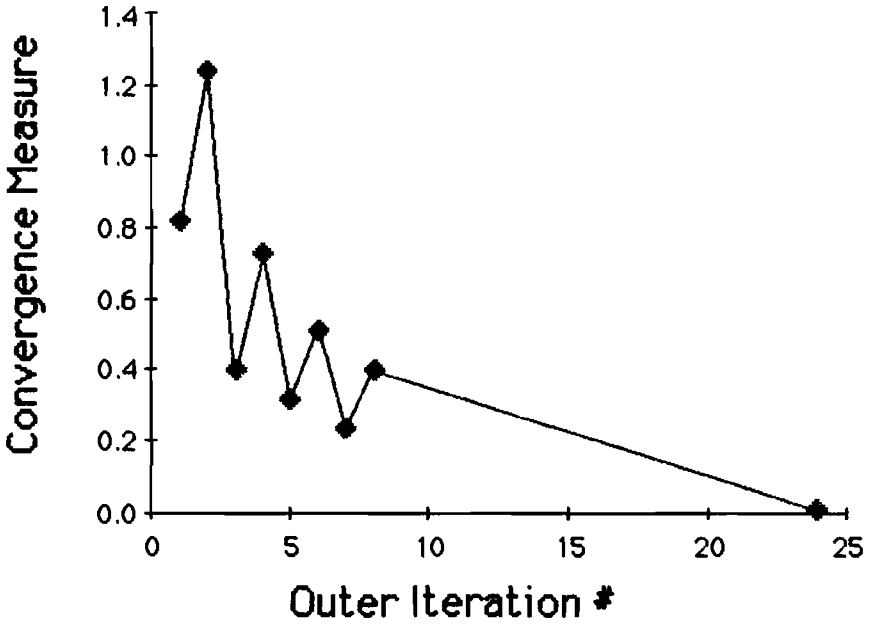

Fig. 3.2.1.21

Table 3.2.1.22- Summary of results for Network 1 System Optimum

Maximum Number of Internal Iterations	CPU - Time (Seconds)	Total Number of Internal Iterations required for Convergence
1	28.417	65
2	24.110	55
3	32.481	75
4	42.798	100
5	38.908	91
6	44.263	103
7	52.259	122
8	69.464	163
9	75.347	177
10	87.126	205

the Texas highway network. Its structure differs from Network 1 in that it is composed of exclusive passenger car lanes, truck lanes and common user lanes. The car user has the option of choosing the exclusive car lane or the common lane while the truck user can use either the exclusive truck lane or the common lane. The option for both classes of using either the exclusive or common lane is controlled with the use of dummy links. If the option is open then the dummy link is assigned a zero cost or if closed then the dummy link is assigned a very high positive cost. This particular structure was chosen due to the possible future construction of exclusive facilities for special categories of the traffic stream (in this study passenger cars and trucks). A more detailed description of this structure is given in Appendix B.

Network 2 depicts the following features:

Passenger car network coding:

Number of centroids: 14
Number of O-D pairs: 182
Number of centroid connectors: 14
Number of egress links: 14
Number of access links: 14
Number of one-way highway links: 126
Origin nodes: from 1 to 14
Destination nodes: from 15 to 28
Highway nodes: from 29 to 64

Truck network coding:

Origin nodes: from 65 to 78
Destination nodes: from 79 to 92
Highway nodes: from 93 to 128

As before, the truck network is a replica of the passenger car network. The combined features are the following:

Total number of O-D pairs: 364
Total number of nodes: 128
Total number of links: 336

For this network two different groups of tests have been carried out. In the first group the exclusive link option was open for both the passenger cars and trucks, meaning that the dummy links had been assigned a zero cost. In the second test the options of exclusive links were closed, meaning the

the dummy links had been assigned a very high positive cost. These two different test groups were intended to capture any differences in the performance of the diagonalization algorithm, with regard to the use of exclusive links as opposed to the standard common link network configuration.

A series of tests was performed for both groups, for the User Equilibrium and System Optimum assignment policies. However, for the UE a series of runs corresponding to four different capacity levels (see section 3.1) have been conducted to examine the performance of the algorithm under different levels of congestion. The results for the first group are summarized in table 3.2.2.1 and for the second group in table 3.2.2.2 for the User Equilibrium. For illustrative purposes the detailed results for capacity levels 1.0C and 0.5C are presented in tables 3.2.2.3 to 3.2.2.12 and 3.2.2.13 to 3.2.2.22 respectively for the first group only. Additionally, the corresponding graphs of the convergence measure versus the outer iteration number are presented in figures 3.2.2.2 to 3.2.2.11 and 3.2.2.12 to 3.2.2.21 for the 1.0C and 0.5C capacity levels respectively. The results for the System Optimum are summarized in Tables 3.2.2.23 and 3.2.2.24 for both groups respectively. The detailed results for group one for the SO are presented in Tables 3.2.2.25 to 3.2.2.34 and the corresponding graphs in figures 3.2.2.22 to 3.2.2.31. Following, the description of the Texas network developed for this study is presented.

3.2.3 The Texas Network-Network 3

One of the objectives of this study was to test the diagonalization algorithm on a large scale network, in view of its applicability to analyze truck-related improvements to the Texas highway network. A subset of the Texas highway network was developed and tested in this study. The structure of the developed Texas network has the same form as Network 2, thus allowing also the inclusion of the design element. The basic features of Network 3 are given below while a description of the database development is given in Appendix B.3.

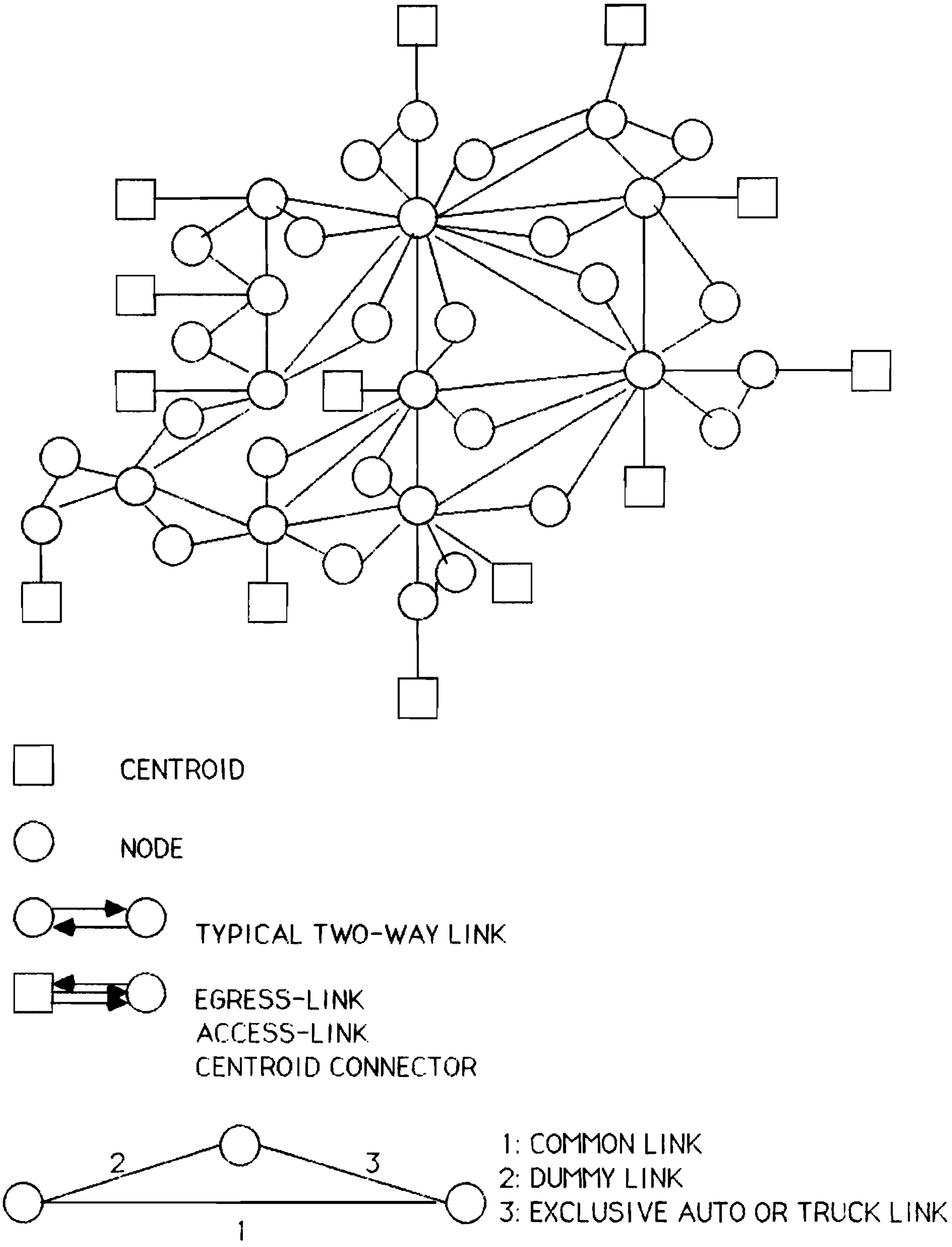

FIG.3.2.2.1 - NETWORK 2

Table 3.2.2.1 - Summary of results for Network 2
User Equilibrium - Options Open

Maximum Number of Internal Iterations	CPU - Time (Seconds)			
	1.0C	0.8C	0.5C	4.0C
1	10.049	8.428	9.032	43.401
2	9.464	7.339	12.646	27.852
3	14.257	9.940	8.337	41.451
4	16.390	15.924	11.570	14.454
5	12.568	14.223	12.625	41.386
6	14.161	12.073	13.169	13.432
7	15.320	15.289	15.298	51.316
8	14.692	13.146	15.247	35.507
9	17.539	15.815	14.101	46.011
10	13.664	14.809	15.308	44.034

Mamimum Number of Internal Iterations	Corresponding Total # of Internal Iterations			
	1.0C	0.8C	0.5C	4.0C
1	16	13	14	79
2	15	11	21	49
3	24	16	13	76
4	28	25	19	25
5	21	24	21	76
6	24	20	22	23
7	26	26	26	95
8	25	22	26	65
9	30	27	24	85
10	23	25	26	81

Table 3.2.2.2 - Summary of results for Network 2 User Equilibrium - Options Closed

Maximum Number of Internal Iterations	CPU - Time (Seconds)			
	1.0C	0.8C	0.5C	4.0C
1	11.924	13.926	13.451	6.360
2	7.873	9.915	9.934	3.687
3	15.364	14.389	18.362	3.161
4	20.387	16.371	18.858	13.666
5	23.291	22.303	26.803	7.157
6	25.779	22.293	35.264	9.141
7	24.815	24.305	22.344	7.107
8	20.850	32.761	21.863	7.087
9	24.811	27.352	30.758	7.096
10	30.801	33.792	32.335	7.116

Mamimum Number of Internal Iterations	Corresponding Total # of Internal Iterations			
	1.0C	0.8C	0.5C	4.0C
1	21	25	24	10
2	13	17	17	5
3	28	26	34	4
4	38	30	35	25
5	44	42	51	12
6	49	42	68	16
7	47	46	42	12
8	39	63	41	12
9	47	52	59	12
10	59	65	62	12

Network *2 - User Equilibrium
Options Open
Maximum * of Internal Iterations = 1

CPU = 10.049 seconds
Capacity = 1*C

Table 3.2.2.3

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	1	1	.739
2	1	2	.253
3	1	3	.155
4	1	4	.112
5	1	5	.105
6	1	6	.044
16	1	16	.005

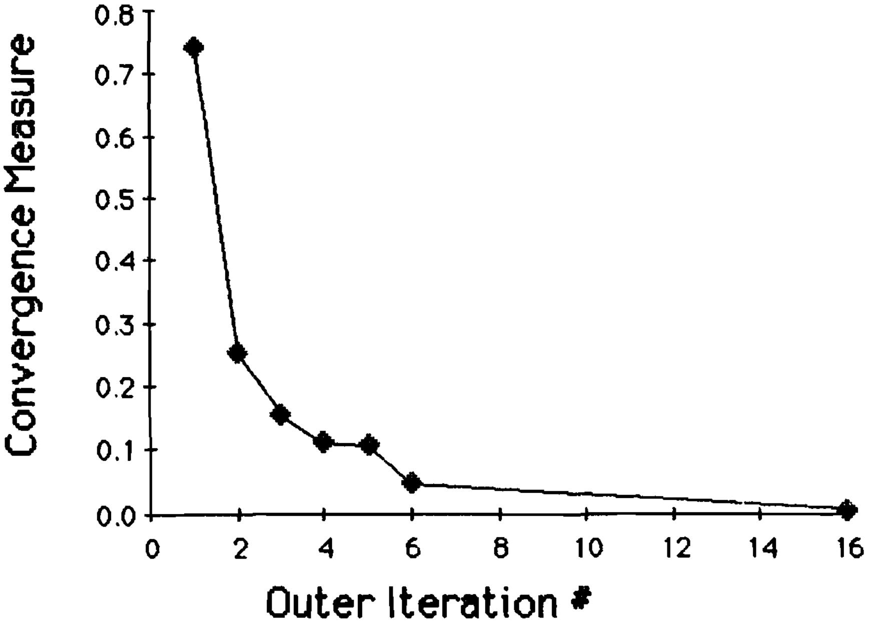

Fig. 3.2.2.2

Network *2 - User Equilibrium
Options Open
Maximum * of Internal Iterations = 2

CPU = 9.464 seconds
Capacity = 1*C

Table 3.2.2.4

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	2	2	.816
2	2	4	.242
3	2	6	.121
4	2	8	.071
5	2	10	.032
6	2	12	.020
8	1	15	.002

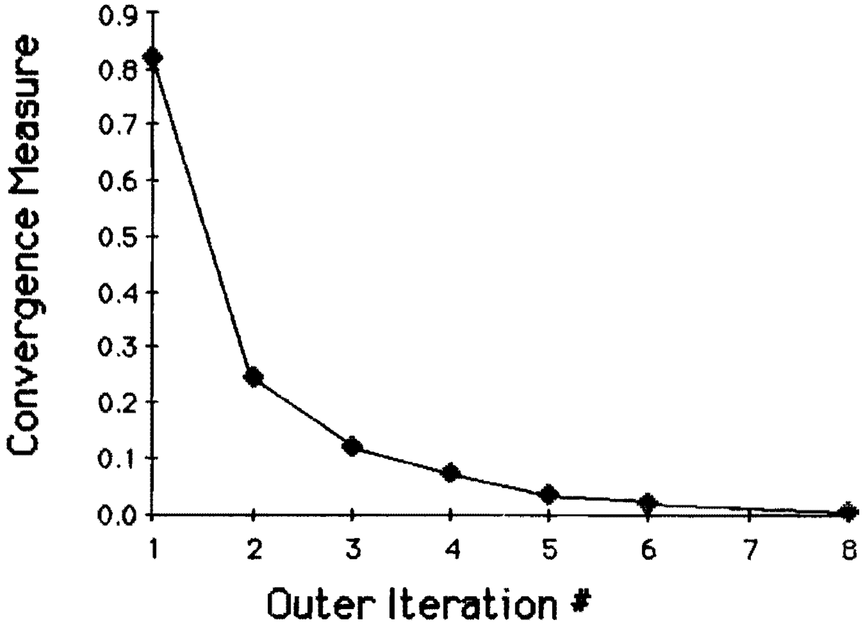

Fig. 3.2.2.3

Network *2 - User Equilibrium
Options Open
Maximum * of Internal Iterations = 3

CPU = 14.257 seconds
Capacity = 1*C

Table 3.2.2.5

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	3	3	.865
2	3	6	.200
3	3	9	.125
4	3	12	.073
5	3	15	.037
6	3	18	.018
9	1	24	.003

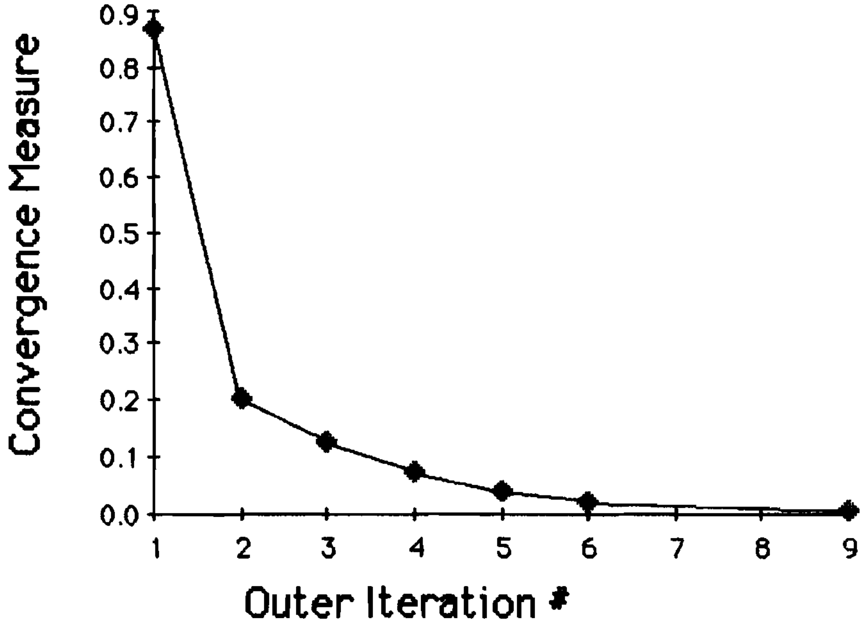

Fig. 3.2.2.4

Network *2 - User Equilibrium
Options Open
Maximum * of Internal Iterations = 4

CPU = 16.390 seconds
Capacity = 1*C

Table 3.2.2.6

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	4	4	.977
2	4	8	.202
3	4	12	.049
4	4	16	.036
5	3	19	.016
6	3	22	.011
9	1	28	.003

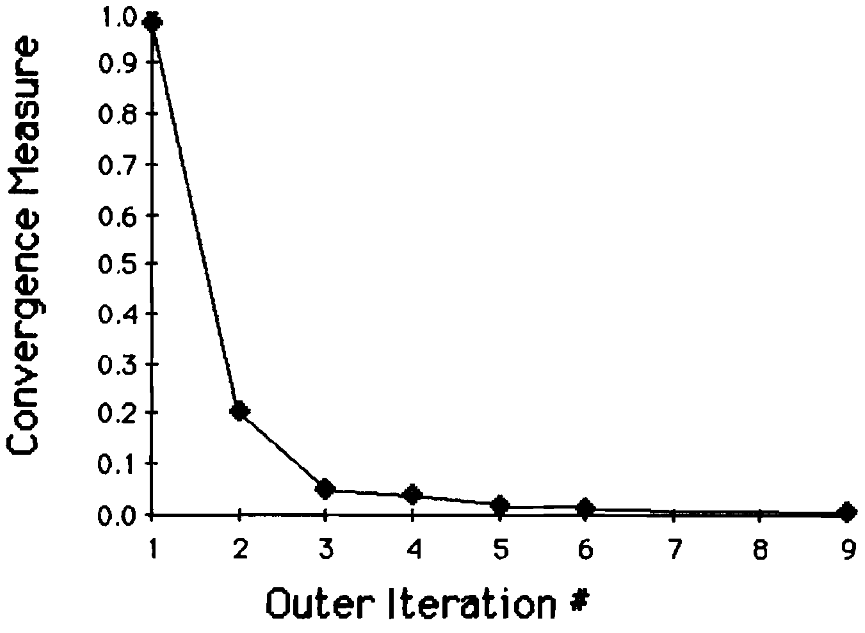

Fig. 3.2.2.5

Network #2 - User Equilibrium
Options Open
Maximum # of Internal Iterations = 5

CPU = 12.568 seconds
Capacity = 1*C

Table 3.2.2.7

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	5	5	1.00
2	5	10	..211
3	5	15	.071
4	5	20	.031
5	1	21	.004

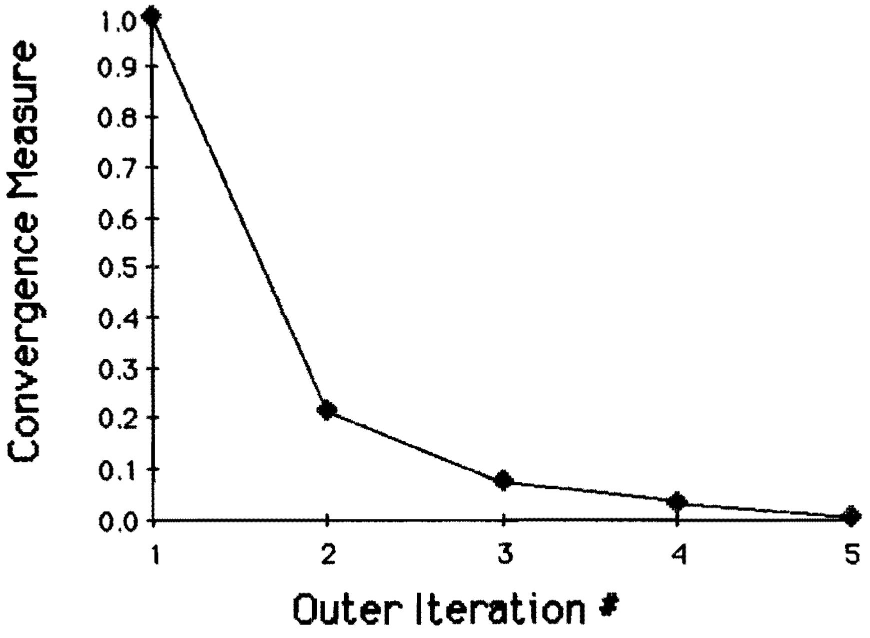

Fig. 3.2.2.6

Network *2 - User Equilibrium CPU = 14.161 seconds
Options Open Capacity = 1*C
Maximum * of Internal Iterations =6

Table 3.2.2.8

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	6	6	1.04
2	6	12	.161
3	6	18	.049
4	3	21	.013
5	2	23	.008
6	1	24	.003

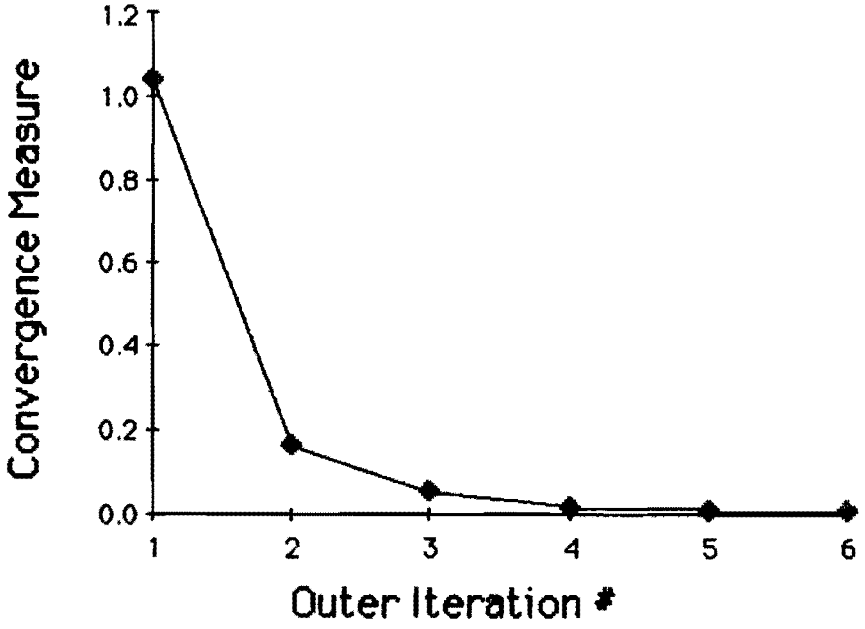

Fig. 3.2.2.7

Network *2 - User Equilibrium
Options Open
Maximum * of Internal Iterations =7

CPU = 15.320 seconds
Capacity = 1*C

Table 3.2.2.9

Outer Iteration #	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	7	7	1.11
2	7	14	.185
3	7	21	.102
4	2	23	.019
5	2	25	.009
6	1	26	.005

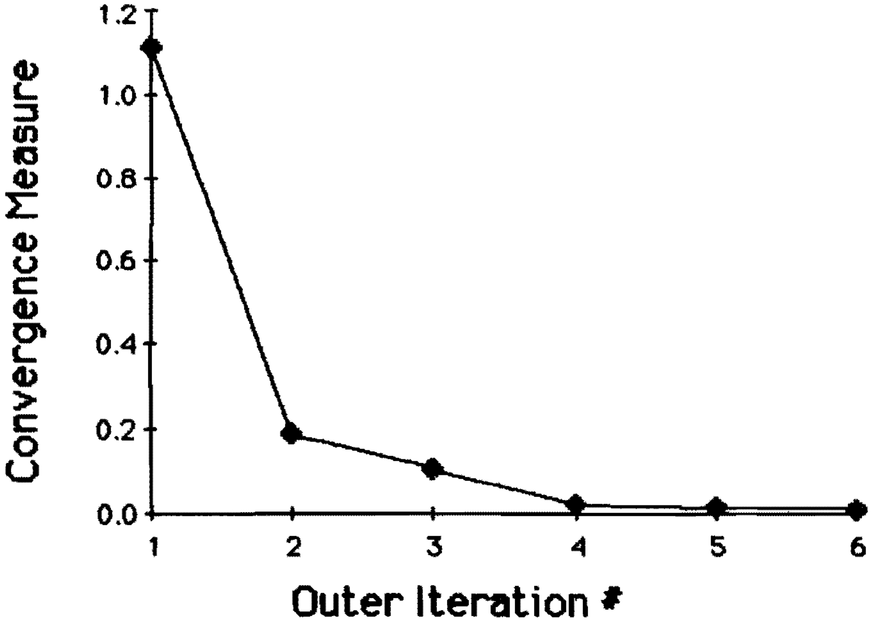

Fig. 3.2.2.8

Network *2 - User Equilibrium CPU = 14.692 seconds
Options Open Capacity = 1*C
Maximum * of Internal Iterations =8

Table 3.2.2.10

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	8	8	1.17
2	8	16	.137
3	8	24	.059
4	1	25	.004

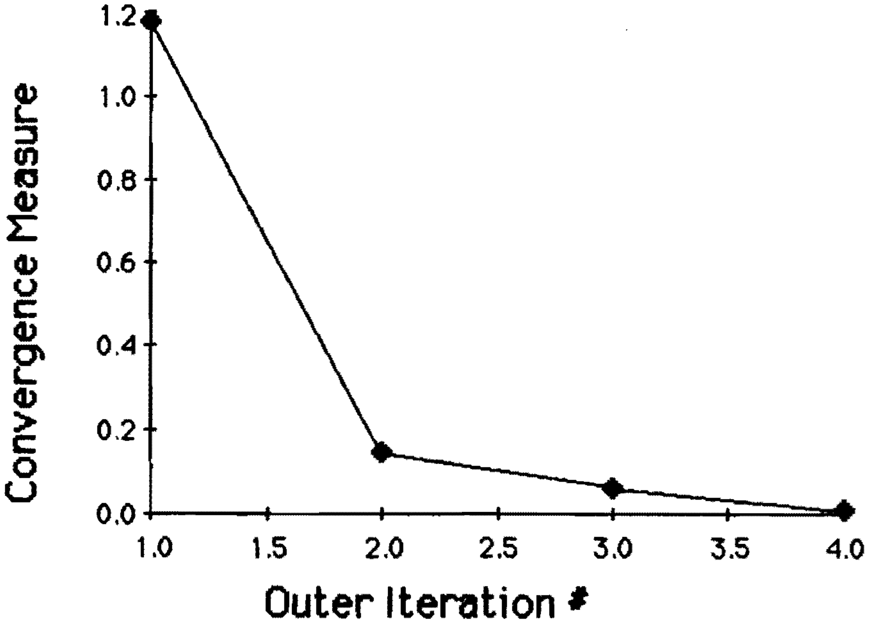

Fig. 3.2.2.9

Network #2 - User Equilibrium CPU = 17.539 seconds
Options Open Capacity = 1*C
Maximum # of Internal Iterations =9

Table 3.2.2.11

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	9	9	1.21
2	7	16	.161
3	9	25	.099
4	2	27	.012
5	2	29	.006
6	1	30	.004

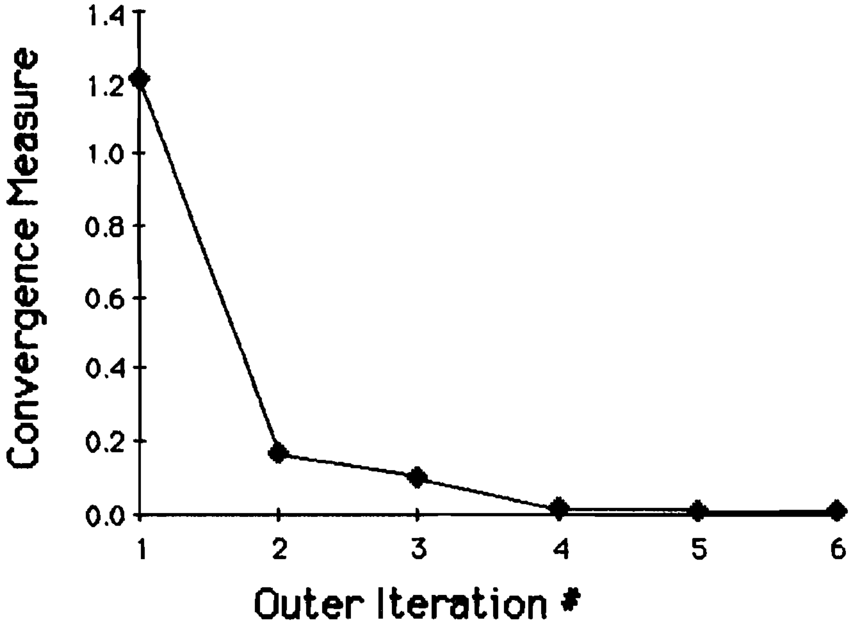

Fig. 3.2.2.10

Network *2 - User Equilibrium CPU = 13.664 seconds
Options Open Capacity = 1*C
Maximum * of Internal Iterations =10

Table 3.2.2.12

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	10	10	1.32
2	7	17	.130
3	5	22	.032
4	1	23	.005

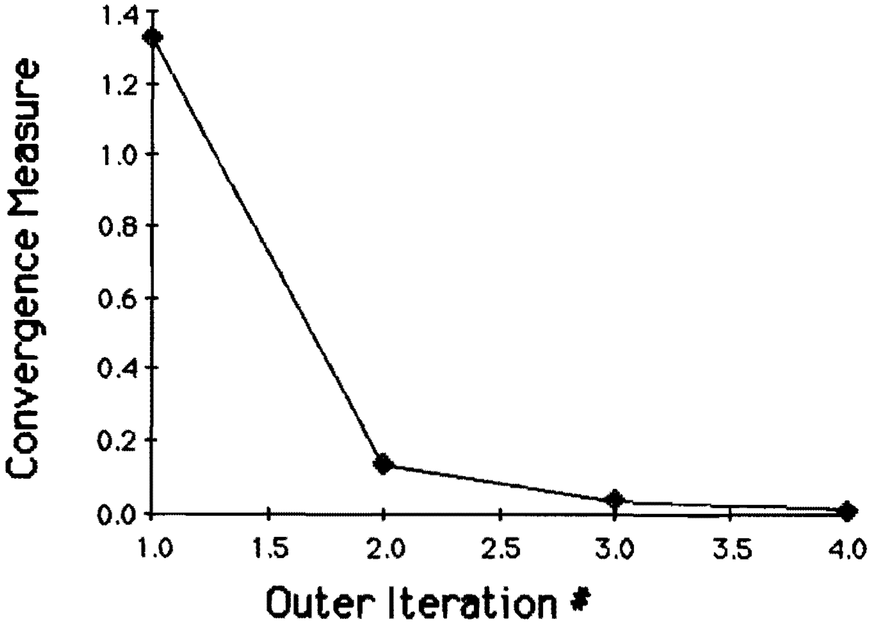

Fig. 3.2.2.11

Network *2 - User Equilibrium
Options Open
Maximum * of Internal Iterations =1

CPU = 9.032 seconds
Capacity =0.5*C

Table 3.2.2.13

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	1	1	.668
2	1	2	.202
3	1	3	.157
4	1	4	.103
5	1	5	.093
6	1	6	.052
14	1	14	.004

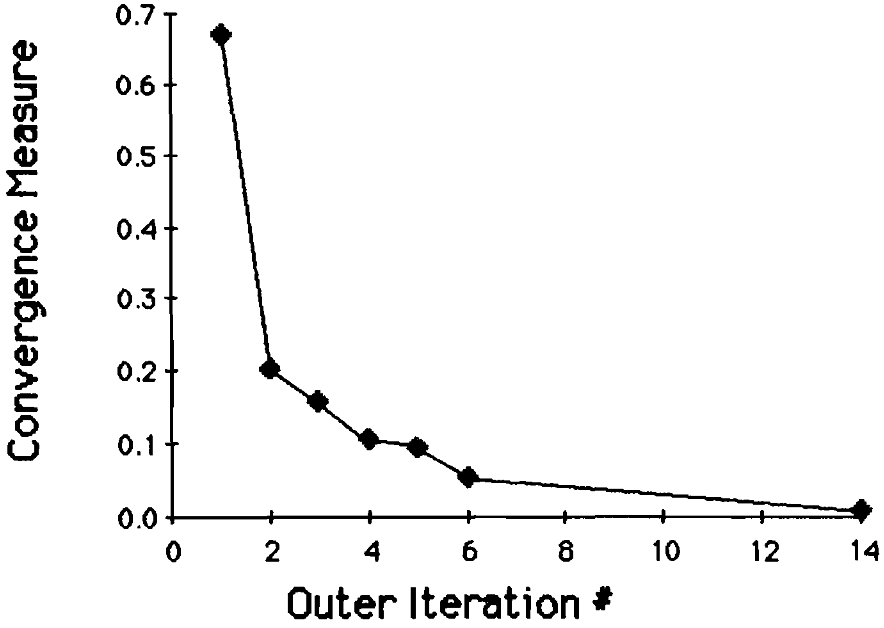

Fig. 3.2.2.12

Network #2 - User Equilibrium
Options Open
Maximum # of Internal Iterations =2

CPU = 12.646 seconds
Capacity =0.5*C

Table 3.2.2.14

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	2	2	.712
2	2	4	.208
3	2	6	.151
4	2	8	.069
5	2	10	.033
6	2	12	.022
11	1	21	.004

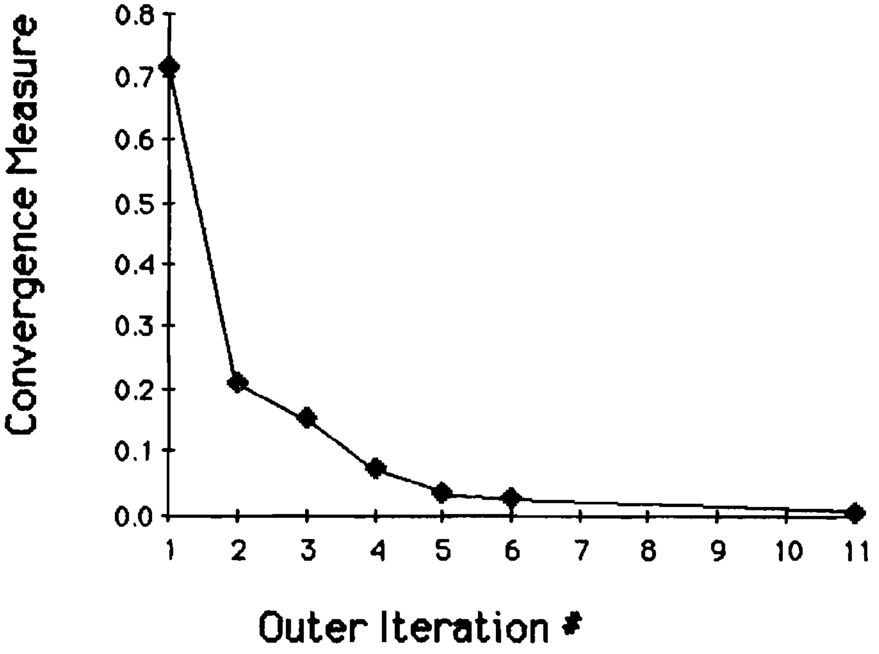

Fig. 3.2.2.13

Network #2 - User Equilibrium
Options Open
Maximum # of Internal Iterations =3

CPU = 8.337 seconds
Capacity =0.5*C

Table 3.2.2.15

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	3	3	.715
2	3	6	.219
3	3	9	.138
4	3	12	.047
5	1	13	.005

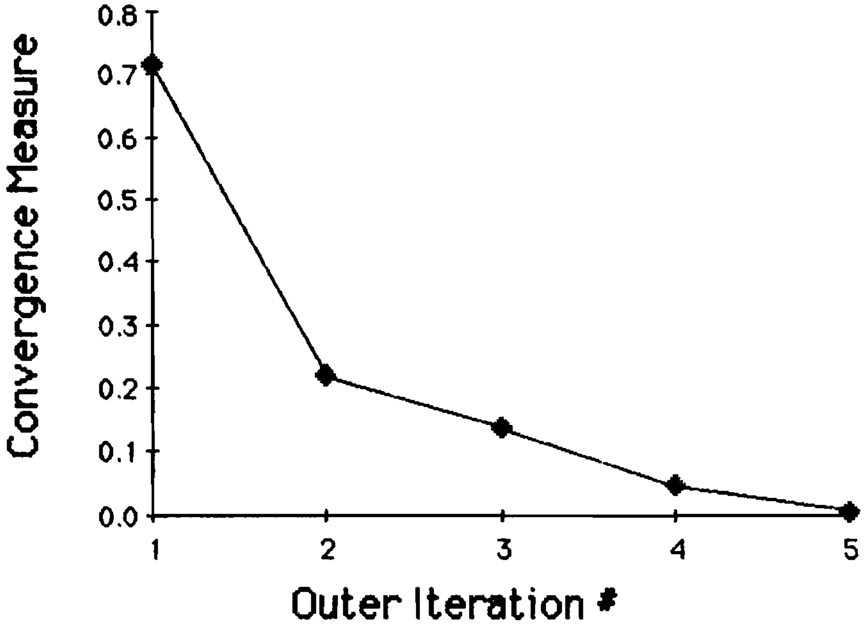

Fig. 3.2.2.14

Network *2 - User Equilibrium
Options Open
Maximum * of Internal Iterations =4

CPU = 11.570 seconds
Capacity =0.5*C

Table 3.2.2.16

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	4	4	.755
2	4	8	.187
3	4	12	.068
4	3	15	.024
5	3	18	.013
6	1	19	.002

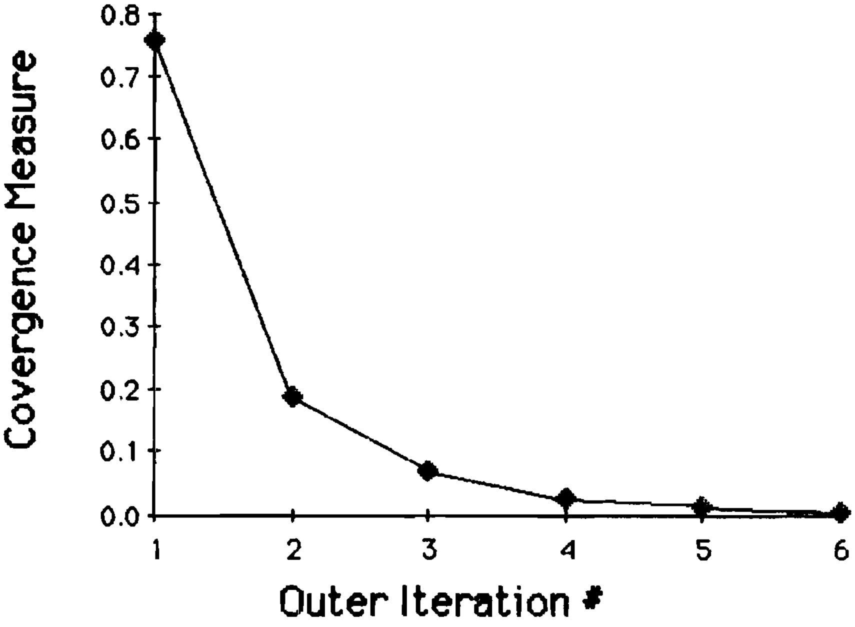

Fig. 3.2.2.15

Network *2 - User Equilibrium
Options Open
Maximum * of Internal Iterations =5

CPU = 12.625 seconds
Capacity =0.5*C

Table 3.2.2.17

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	5	5	.778
2	5	10	.151
3	5	15	.051
4	5	20	.024
5	1	21	.005

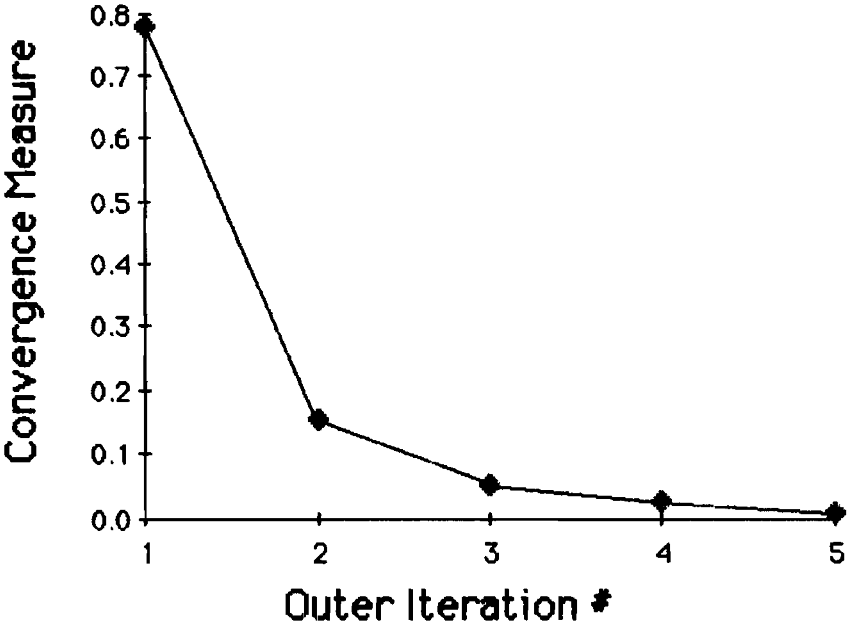

Fig. 3.2.2.16

Network *2 - User Equilibrium CPU = 13.169 seconds
Options Open Capacity =0.5*C
Maximum * of Internal Iterations =6

Table 3.2.2.18

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	6	6	.795
2	6	12	.176
3	5	17	.061
4	2	19	.010
5	2	21	.008
6	1	22	.005

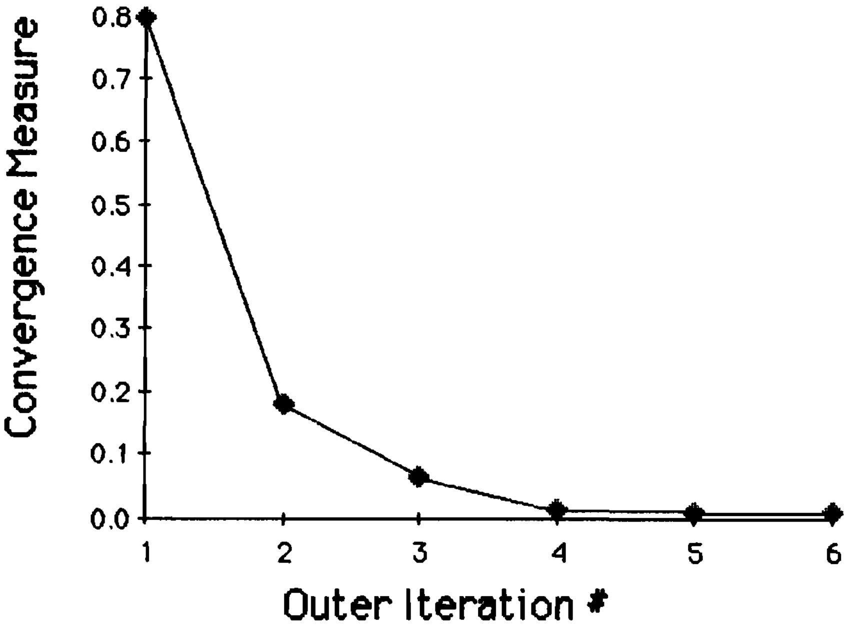

Fig. 3.2.2.17

Network *2 - User Equilibrium
Options Open
Maximum # of Internal Iterations =7

CPU = 15.298 seconds
Capacity =0.5*C

Table 3.2.2.19

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	7	7	.820
2	7	14	.150
3	5	19	.040
4	2	21	.010
5	2	23	.008
6	2	25	.007
7	1	26	.002

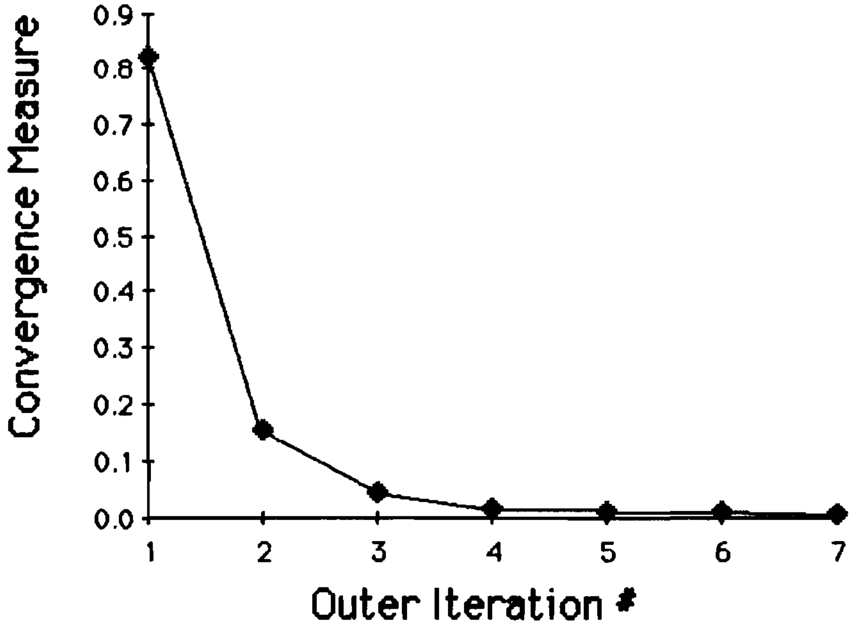

Fig. 3.2.2.18

Network *2 - User Equilibrium CPU = 15.247 seconds
Options Open Capacity =0.5*C
Maximum * of Internal Iterations =8

Table 3.2.2.20

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	8	8	.832
2	8	16	.177
3	7	23	.072
4	2	25	.010
5	1	26	.002

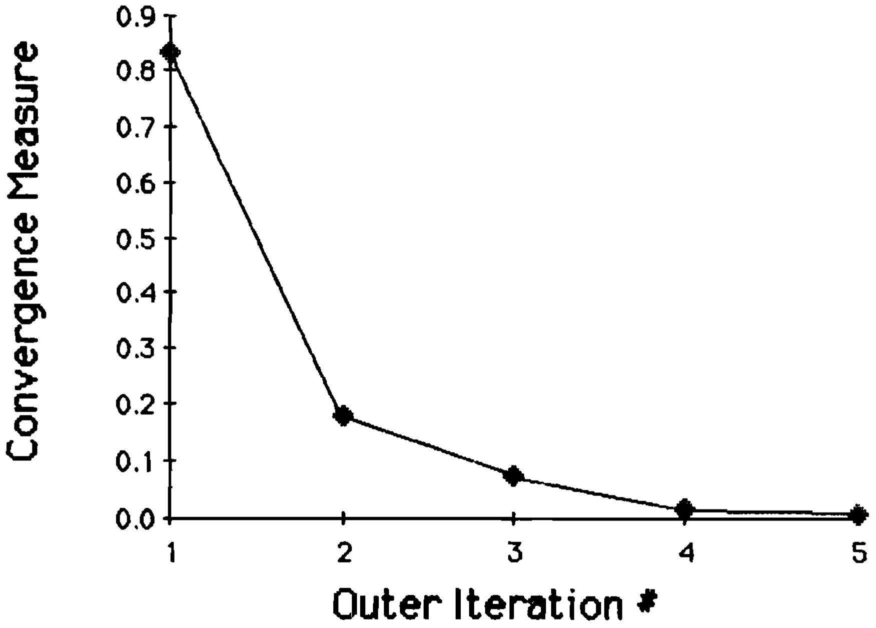

Fig. 3.2.2.19

Network #2 - User Equilibrium
Options Open
Maximum # of Internal Iterations =9

CPU = 14.101 seconds
Capacity =0.5*C

Table 3.2.2.21

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	9	9	.864
2	9	18	.149
3	5	23	.033
4	1	24	.004

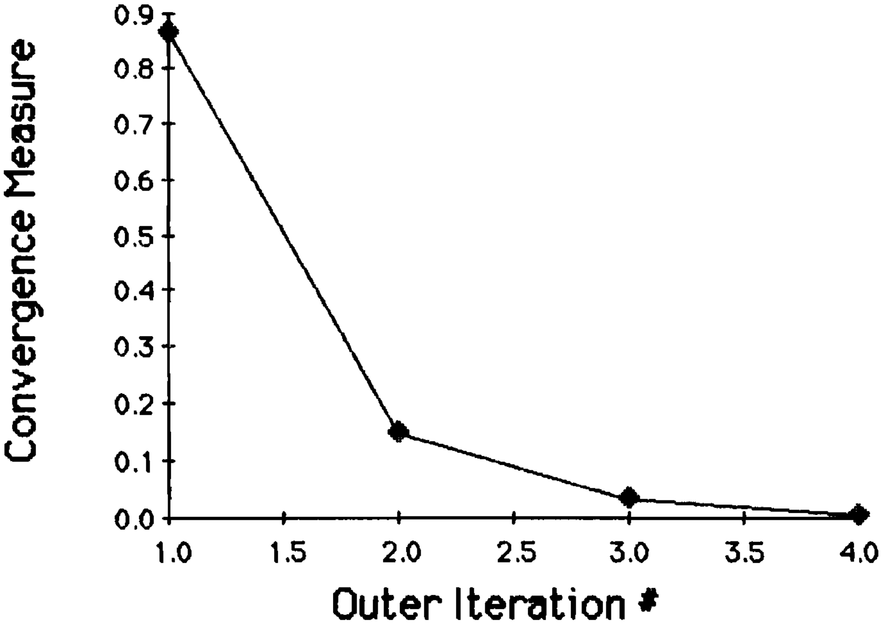

Fig. 3.2.2.20

Network *2 - User Equilibrium
Options Open
Maximum * of Internal Iterations =10

CPU = 15.308 seconds
Capacity =0.5*C

Table 3.2.2.22

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	10	10	.885
2	6	16	.151
3	7	23	.062
4	2	25	.007
5	1	26	.004

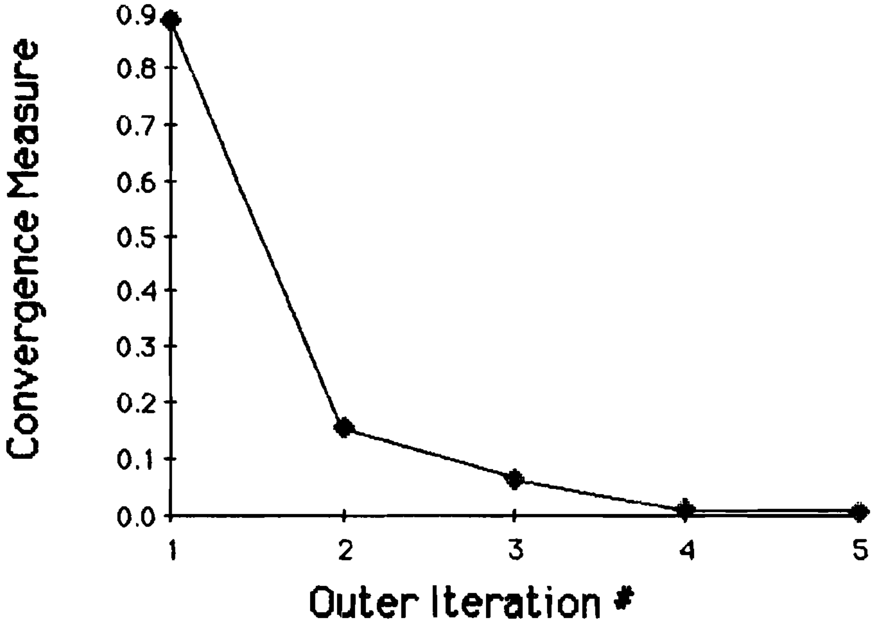

Fig. 3.2.2.21

Table 3.2.2.23 - Summary of results for Network 2 System Optimum - Options Open

Maximum Number of Internal Iterations	CPU - Time (Seconds)	Total Number of Internal Iterations required for Convergence
1	37.258	47
2	26.587	33
3	27.360	34
4	33.913	43
5	35.526	45
6	25.734	32
7	23.218	29
8	30.087	38
9	22.603	28
10	32.290	41

Table 3.2.2.24 – Summary of results for Network 2
System Optimum – Options Closed

Maximum Number of Internal Iterations	CPU - Time (Seconds)	Total Number of Internal Iterations required for Convergence
1	18.940	25
2	17.493	23
3	24.077	33
4	26.134	36
5	30.323	42
6	31.593	44
7	27.450	38
8	28.878	40
9	34.365	48
10	50.069	71

Network *2 -System Optimum CPU =37.258 seconds
Options Open
Maximum * of Internal Iterations =1

Table 3.2.2.25

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	1	1	.600
2	1	2	.180
3	1	3	.196
4	1	4	.066
5	1	5	.047
6	1	6	.045
7	1	7	.027
8	1	8	.026
30	1	30	.009
47	1	47	.004

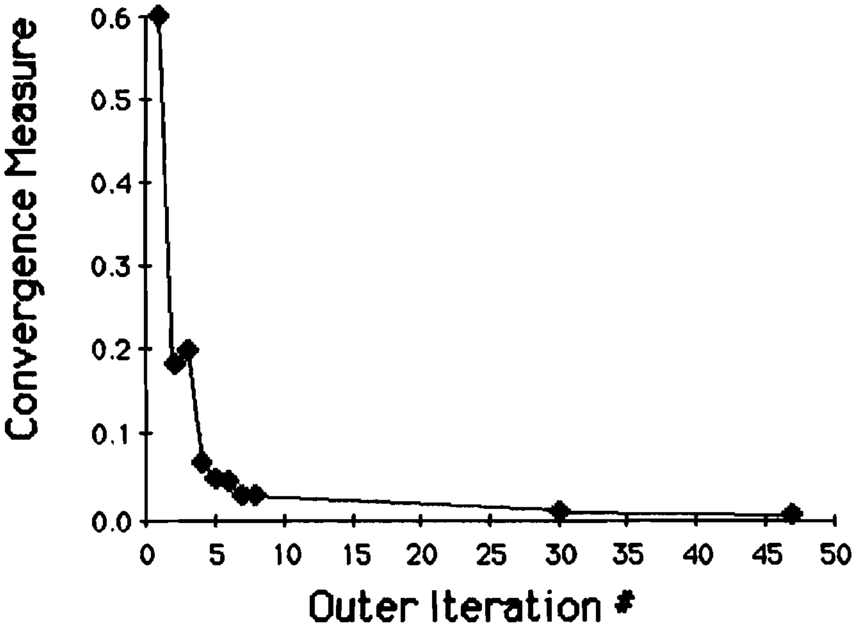

Fig. 3.2.2.22

Network *2 -System Optimum CPU =26.587 seconds
Options Open
Maximum * of Internal Iterations =2

Table 3.2.2.26

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	2	2	.648
2	2	4	.210
3	2	6	.101
4	2	8	.059
5	2	10	.037
6	2	12	.033
7	2	14	.016
8	2	16	.015
17	1	33	.003

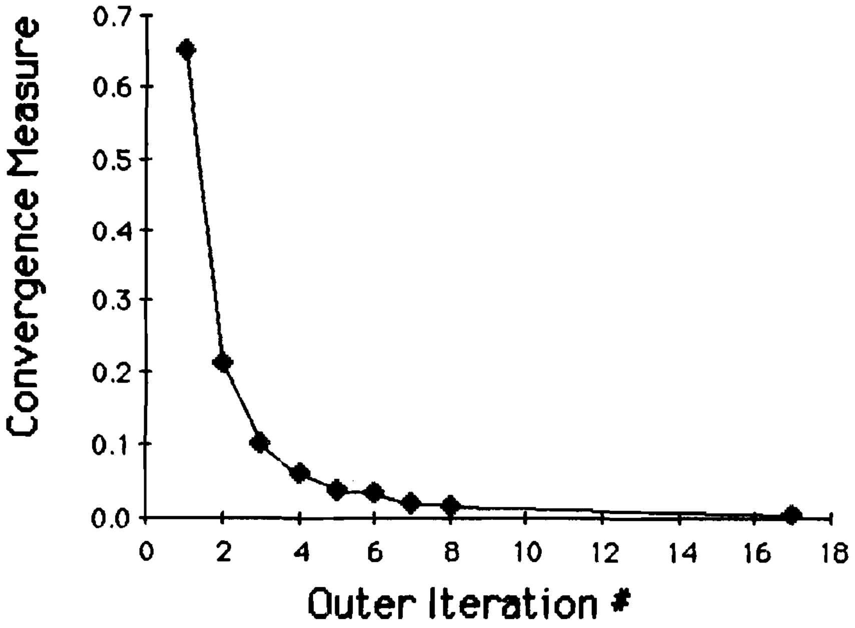

Fig. 3.2.2.23

Network #2 -System Optimum CPU = 27.360 seconds
Options Open
Maximum # of Internal Iterations =3

Table 3.2.2.27

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	3	3	.656
2	3	6	.182
3	3	9	.088
4	3	12	.048
5	3	15	.026
6	3	18	.019
7	3	21	.013
8	3	24	.011
12	1	34	.005

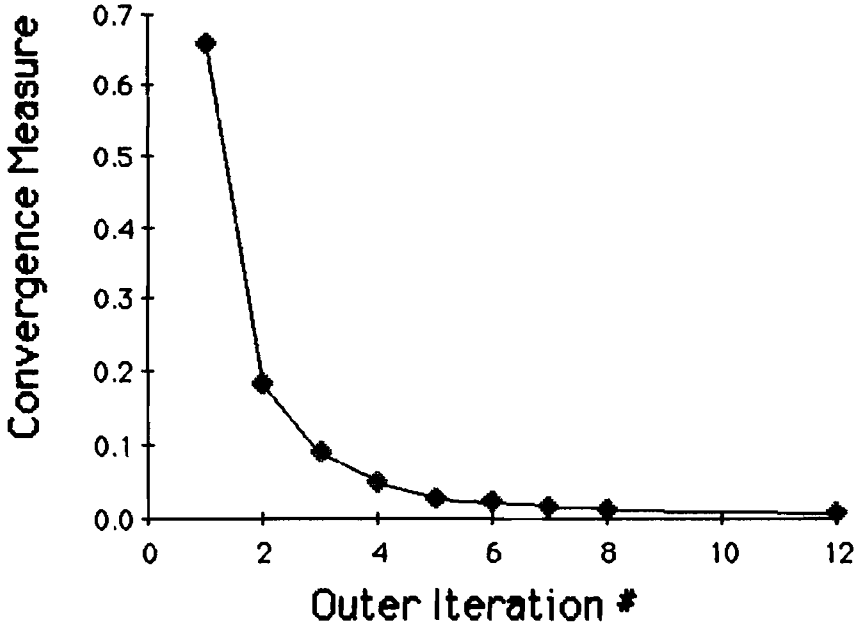

Fig. 3.2.2.24

Network *2 -System Optimum CPU =33.913 seconds
Options Open
Maximum * of Internal Iterations =4

Table 3.2.2.28

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	4	4	.664
2	4	8	.167
3	4	12	.071
4	4	16	.036
5	4	20	.023
6	4	24	.013
7	4	28	.011
8	4	32	.013
13	1	43	.005

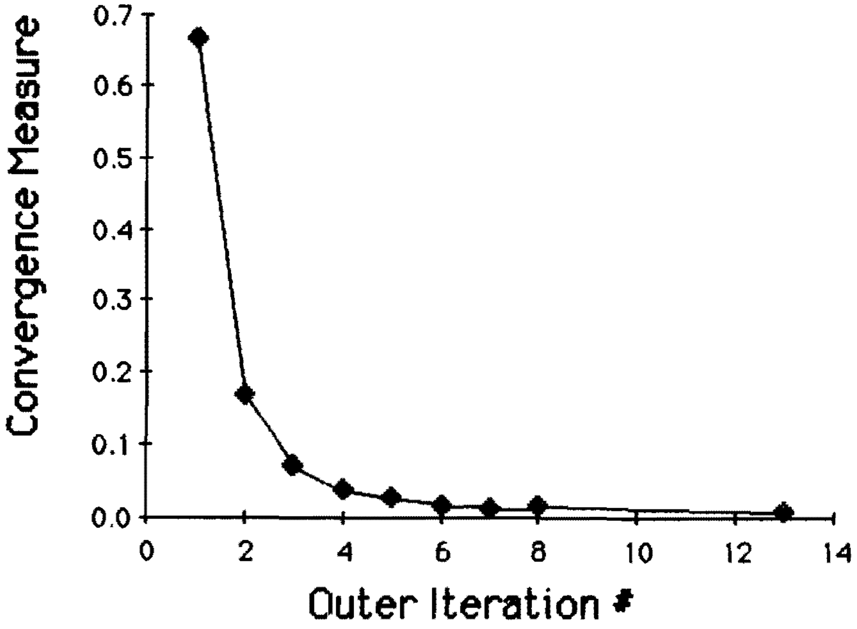

Fig. 3.2.2.25

Network #2 -System Optimum CPU =35.520 seconds
Options Open
Maximum # of Internal Iterations =5

Table 3.2.2.29

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	5	5	.684
2	5	10	.163
3	5	15	.046
4	5	20	.023
5	5	25	.015
6	3	28	.010
7	2	30	.008
8	4	34	.010
13	1	45	.004

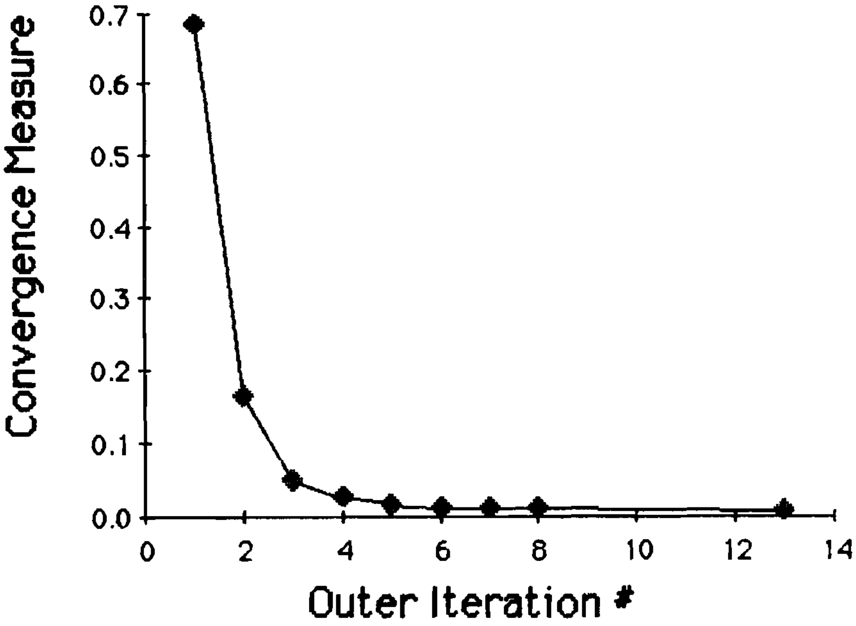

Fig. 3.2.2.26

Network *2 -System Optimum CPU =25.734 seconds
Options Open
Maximum * of Internal Iterations =6

Table 3.2.2.30

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	6	6	.686
2	6	12	.129
3	6	18	.039
4	4	22	.013
5	6	28	.017
6	3	31	.011
7	1	32	.005

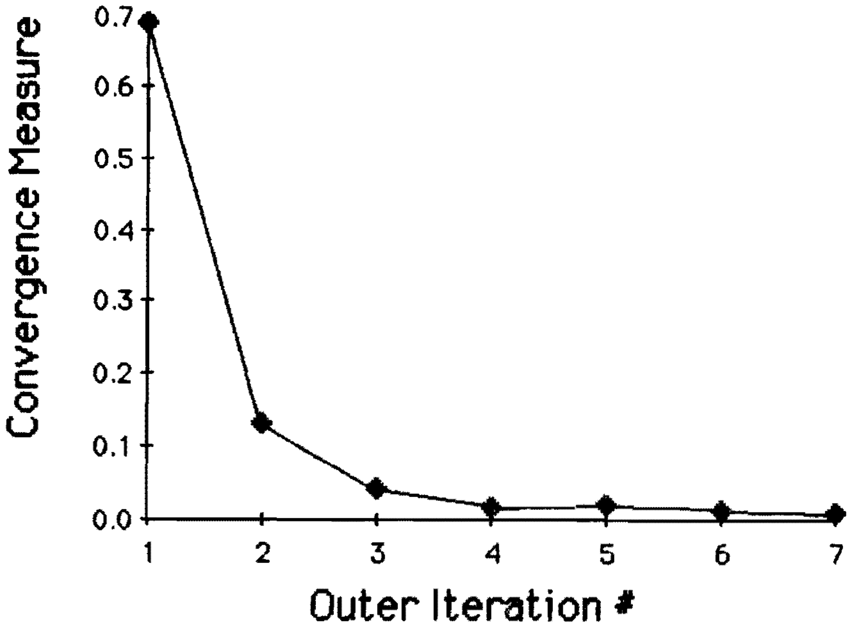

Fig. 3.2.2.27

Network *2 -System Optimum CPU =23.218 seconds
Options Open
Maximum # of Internal Iterations =7

Table 3.2.2.31

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	7	7	.694
2	7	14	.143
3	7	21	.042
4	7	28	.022
5	1	29	.004

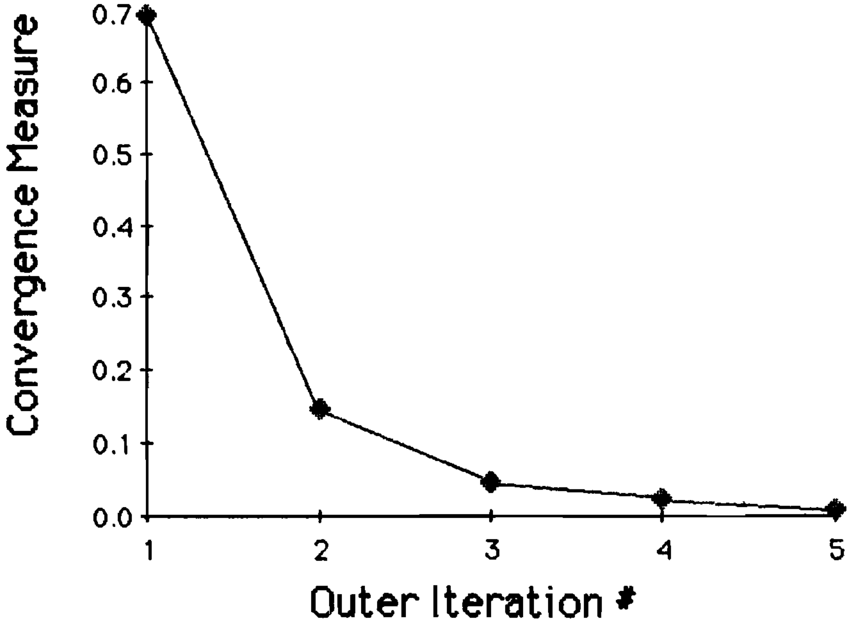

Fig. 3.2.2.28

Network *2 -System Optimum CPU =30.087 seconds
Options Open
Maximum * of Internal Iterations =8

Table 3.2.2.32

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	8	8	.702
2	8	16	.133
3	8	24	.037
4	2	26	.008
5	3	29	.008
6	3	32	.008
7	3	35	.009
8	2	37	.005
9	1	38	

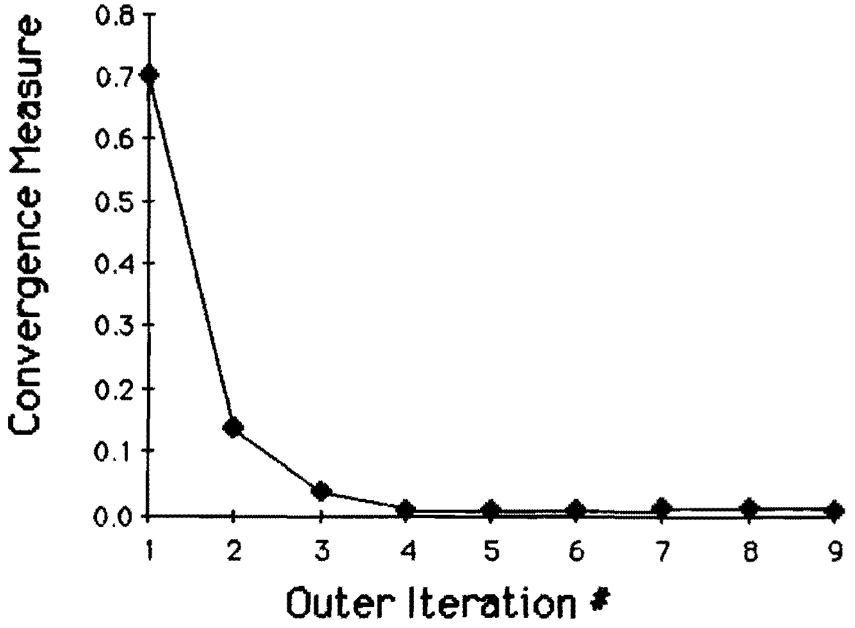

Fig. 3.2.2.29

Network #2 -System Optimum CPU =22.603 seconds
Options Open
Maximum # of Internal Iterations =9

Table 3.2.2.33

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	9	9	.716
2	9	18	.122
3	5	23	.024
4	4	27	.014
5	1	28	.005

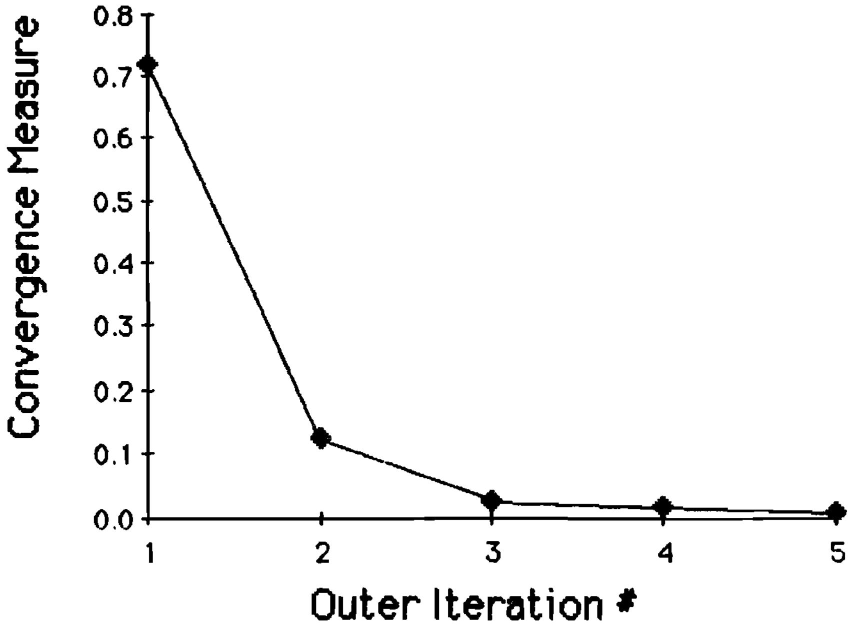

Fig. 3.2.2.30

Network *2 -System Optimum CPU =32.290 seconds
Options Open
Maximum * of Internal Iterations =10

Table 3.2.2.34

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	10	10	.726
2	10	20	.130
3	9	29	.030
4	3	32	.010
5	3	35	.008
6	3	38	.009
7	2	40	.007
8	1	41	.004

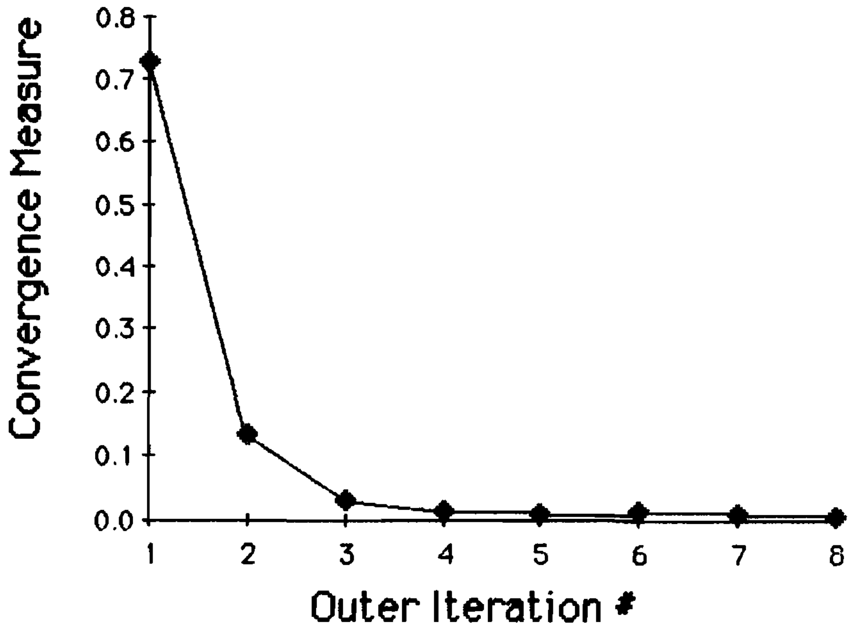

Fig. 3.2.2.31

Passenger car network coding

Number of centroids: 14

Number of O-D pairs: 182

Number of centroid connectors: 14

Number of egress links: 14

Number of access links: 14

Number of one-way highway links: 1914

Origin nodes: from 1 to 14

Destination nodes: from 76 to 89

Highway nodes: from 851 to 1368

Network 3 depicts the following combined characteristics:

Total number of O-D pairs: 364

Total number of nodes: 1400

Total number of links: 3912

Due to the size of the network it was decided to test the algorithm only in the range from 1 to 5 maximum number of internal iterations. This decision was based on the results of the two previous networks, where a higher number of maximum number of internal iterations was less efficient than the lower ones in most of the cases. Again two groups of tests were carried out in a similar way as in Network 2. In the first group, the dummy links had a cost of zero and in the second group they had a very large positive number.

The results for group one are presented in tables 3.2.3.1 to 3.2.3.5 for the User Equilibrium, summarized in Table 3.2.3.6. Figures 3.2.3.2 to 3.2.3.6 present the convergence measure versus the number of outer iterations. The results for the System Optimum are given in Tables 3.2.3.7 to 3.2.3.11, summarized in Table 3.2.3.12. The convergence measure versus the number of outer iterations is presented in Figures 3.2.3.7 to 3.2.3.11. The results for group two are presented in Tables 3.2.3.13 to 3.2.3.17 for the UE while Table 3.2.3.18 summarizes the results. Tables 3.2.3.19 to 3.2.3.23 report the results from the SO while Table 3.2.3.24 summarizes the results. Figures 3.2.3.12 to 3.2.3.16 and Figures 3.2.3.17 to 3.2.3.21 present the plots of the convergence measure versus the number of outer iteration for the UE and SO respectively. The next section presents a summary of the results for all networks.

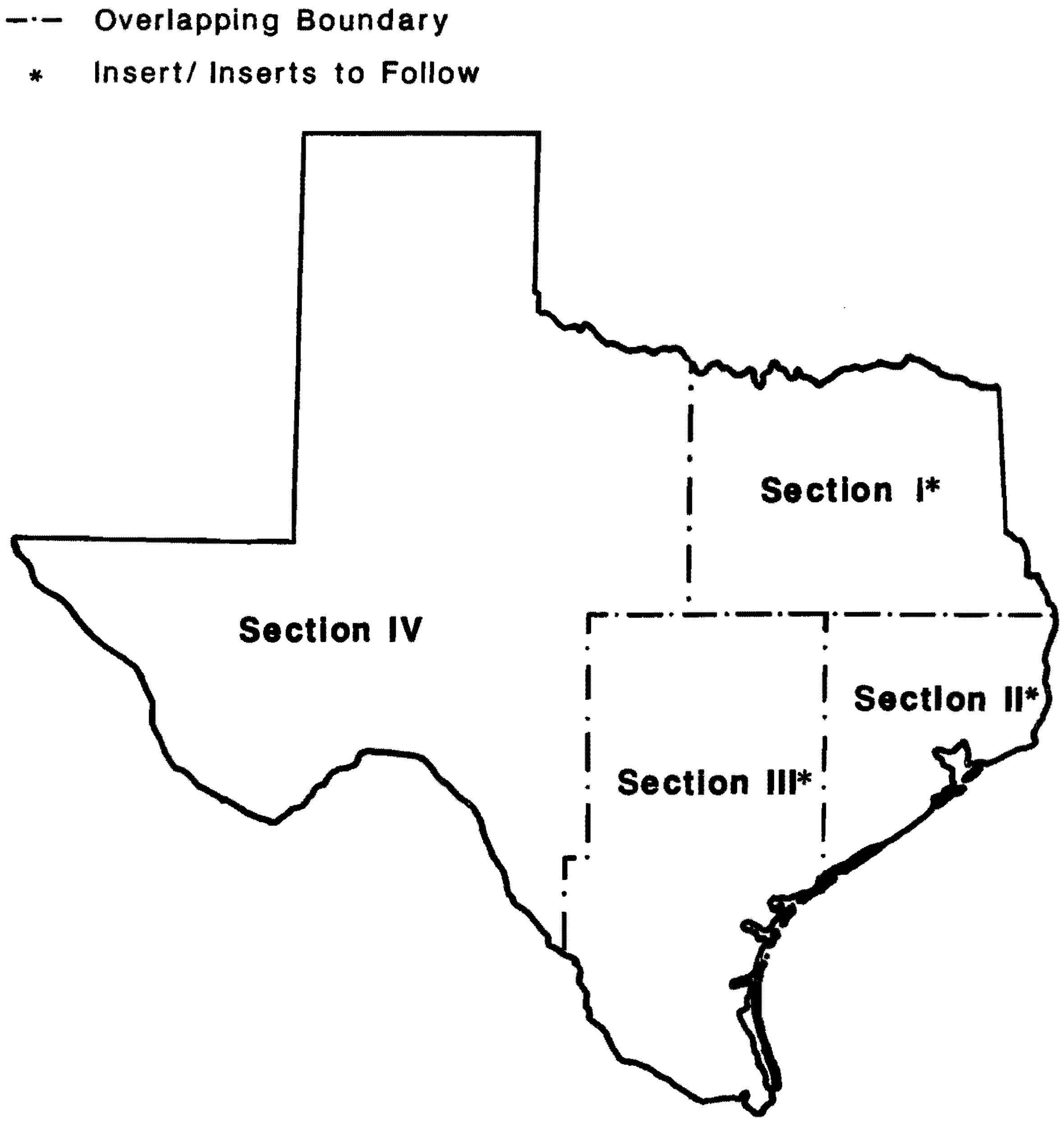

Figure 3.2.3 Texas Map

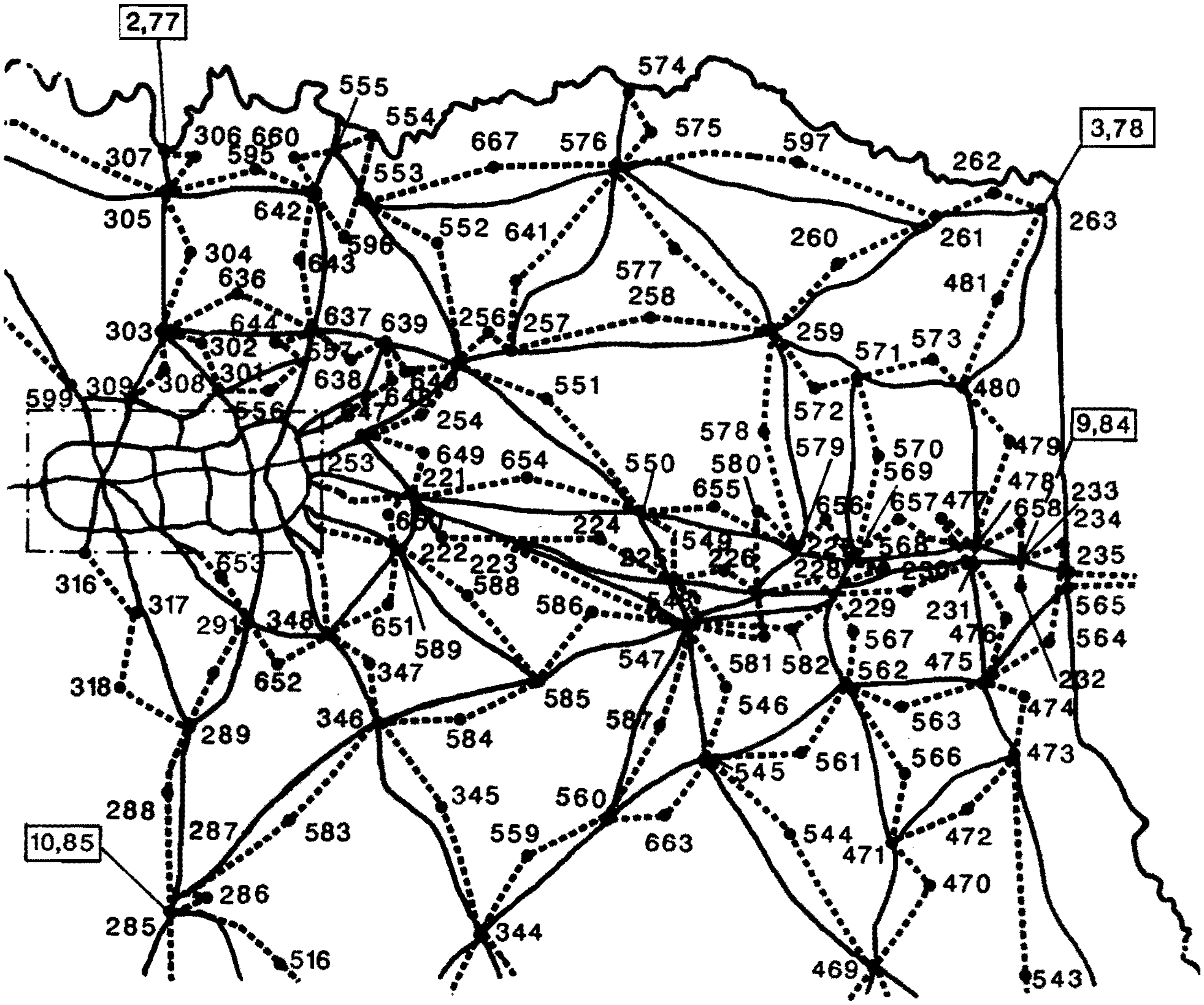

Figure 3.2.3.a - Section I

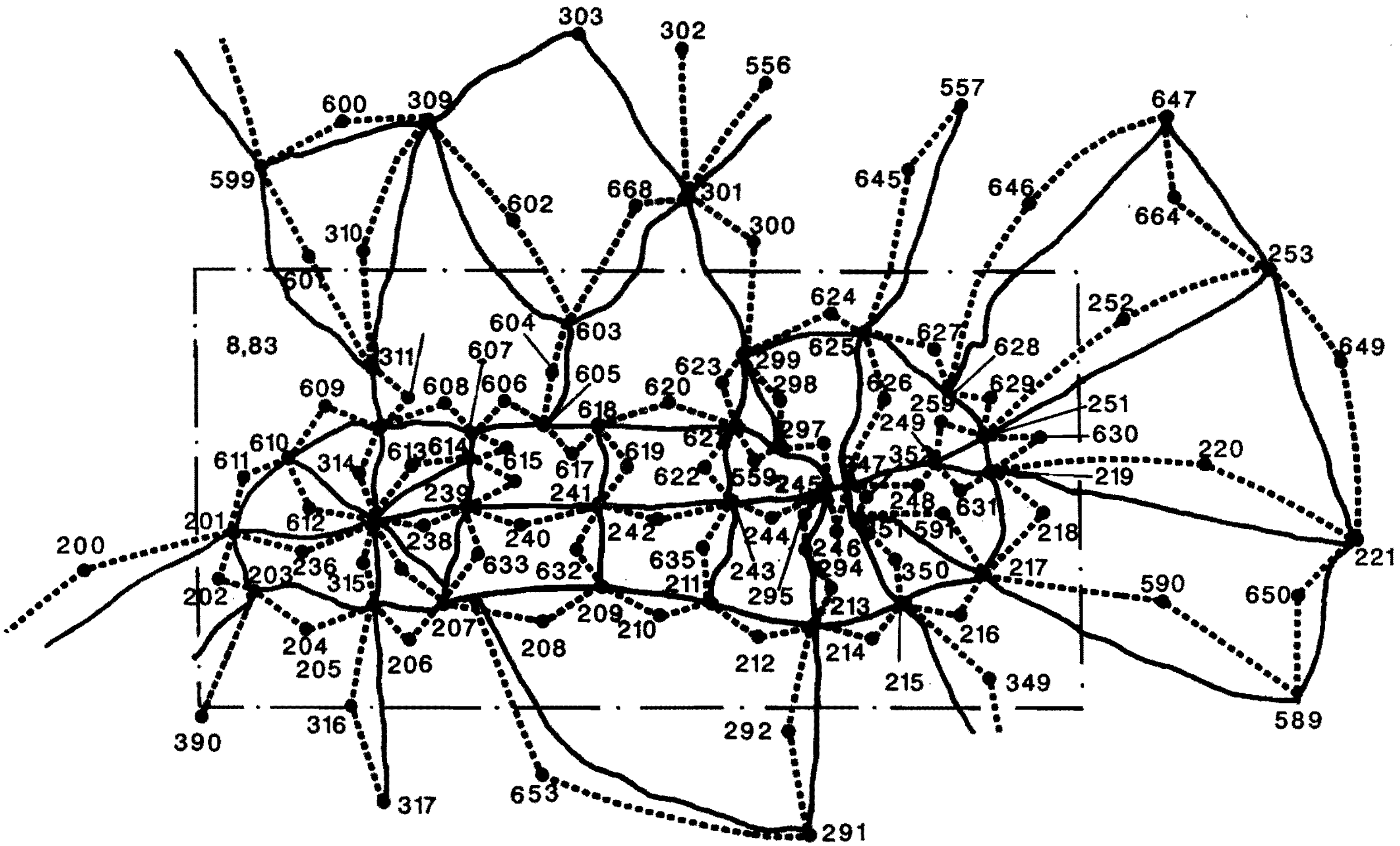

Figure 3.2.3.a.1 Dallas-Ft. Worth

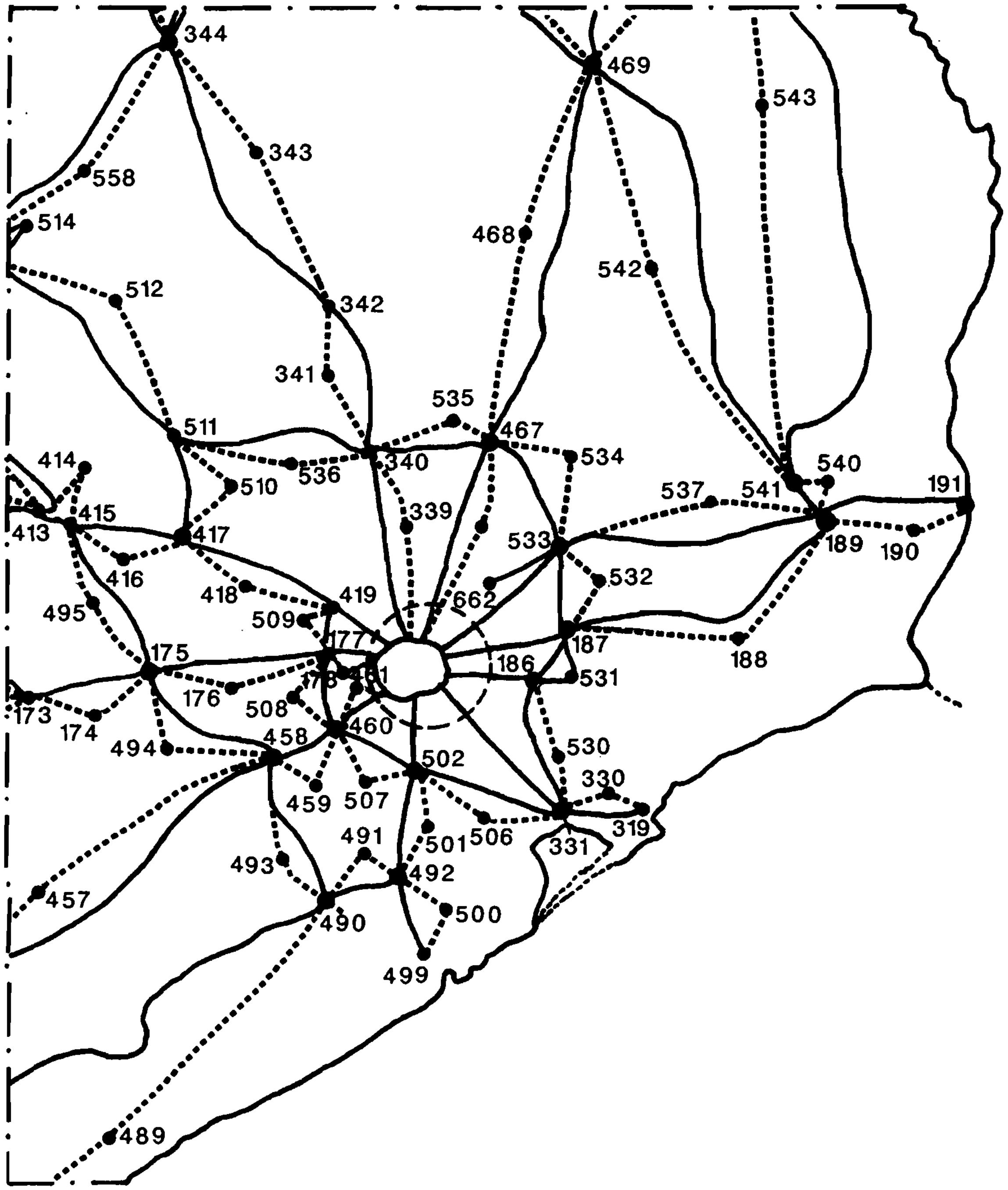

Figure 3.2.3.b Section II

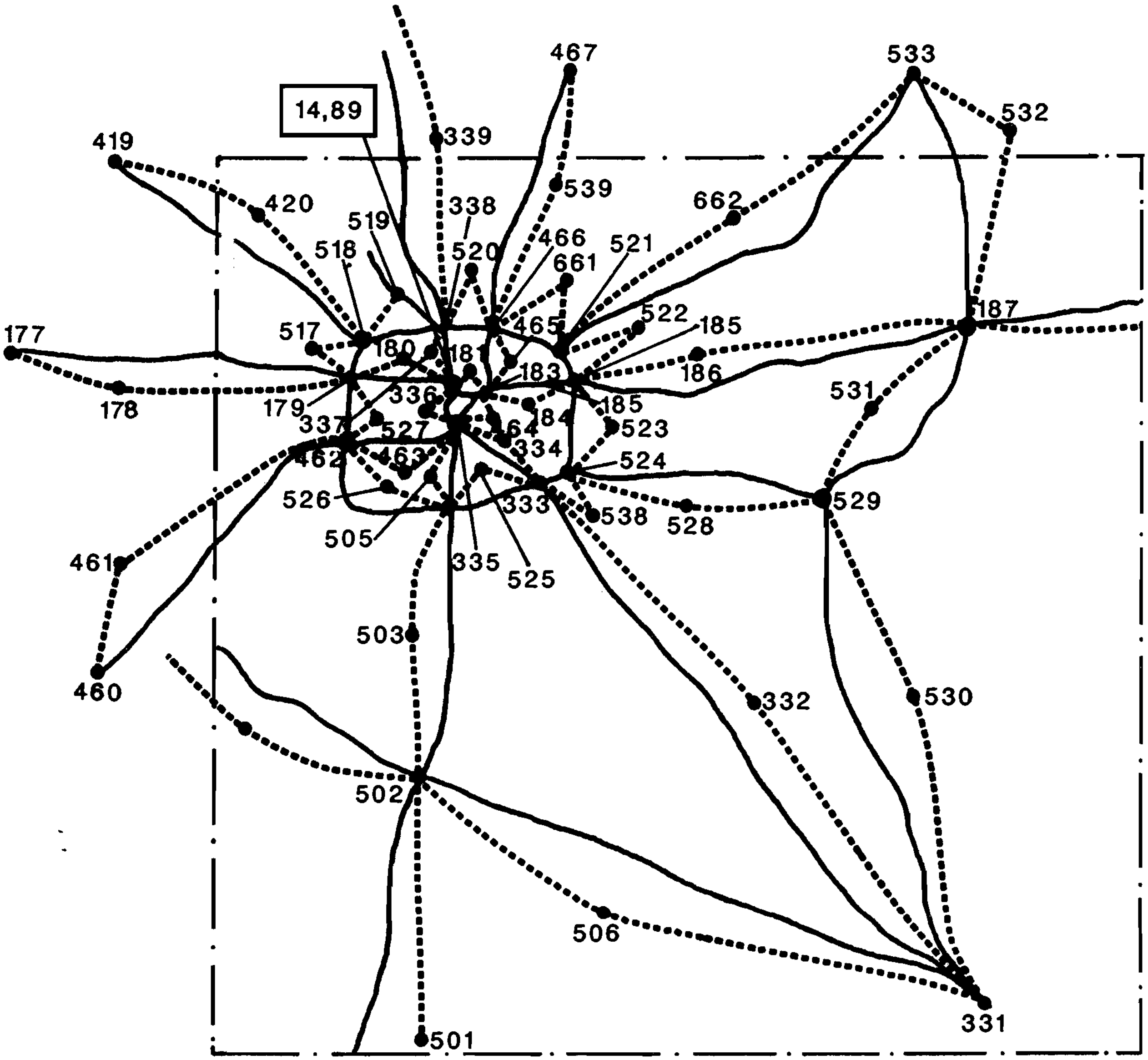

Figure 3.2.3.b.1 Houston

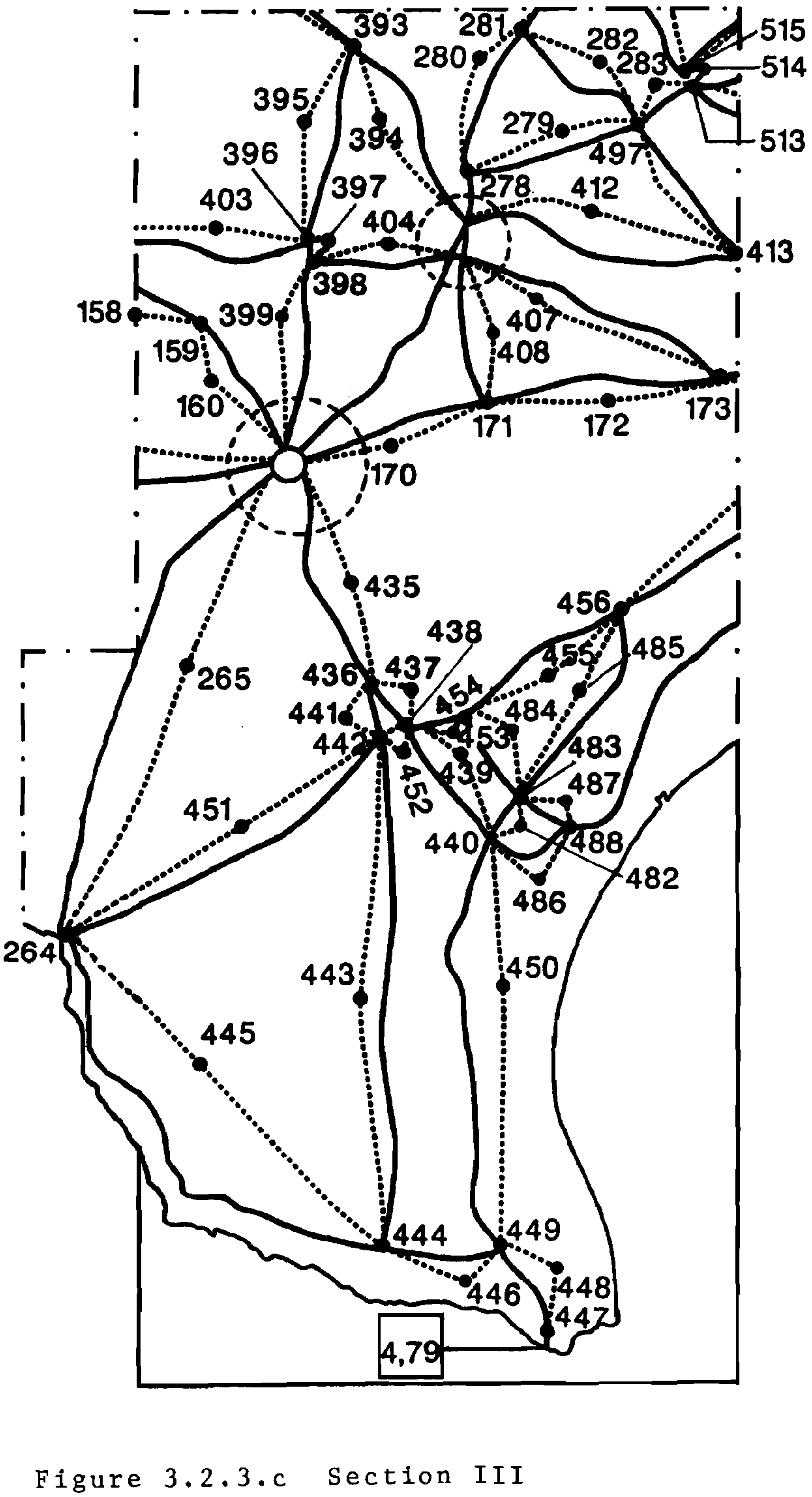

Figure 3.2.3.c Section III

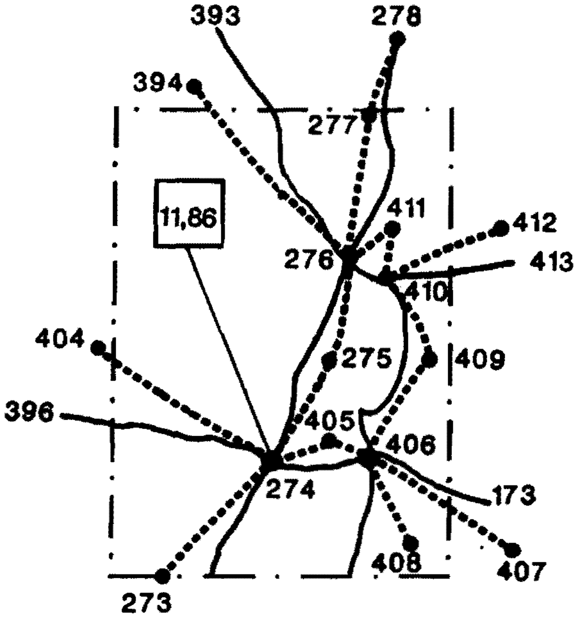

Fig. 3.2.3.c.1 Austin

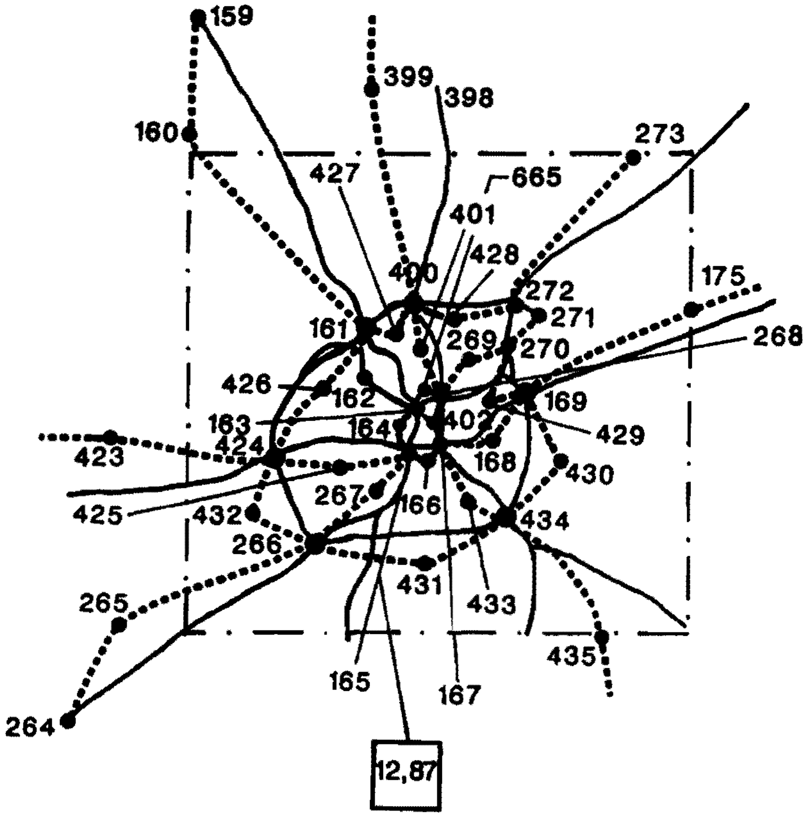

Fig. 3.2.3.c.2 San Antonio

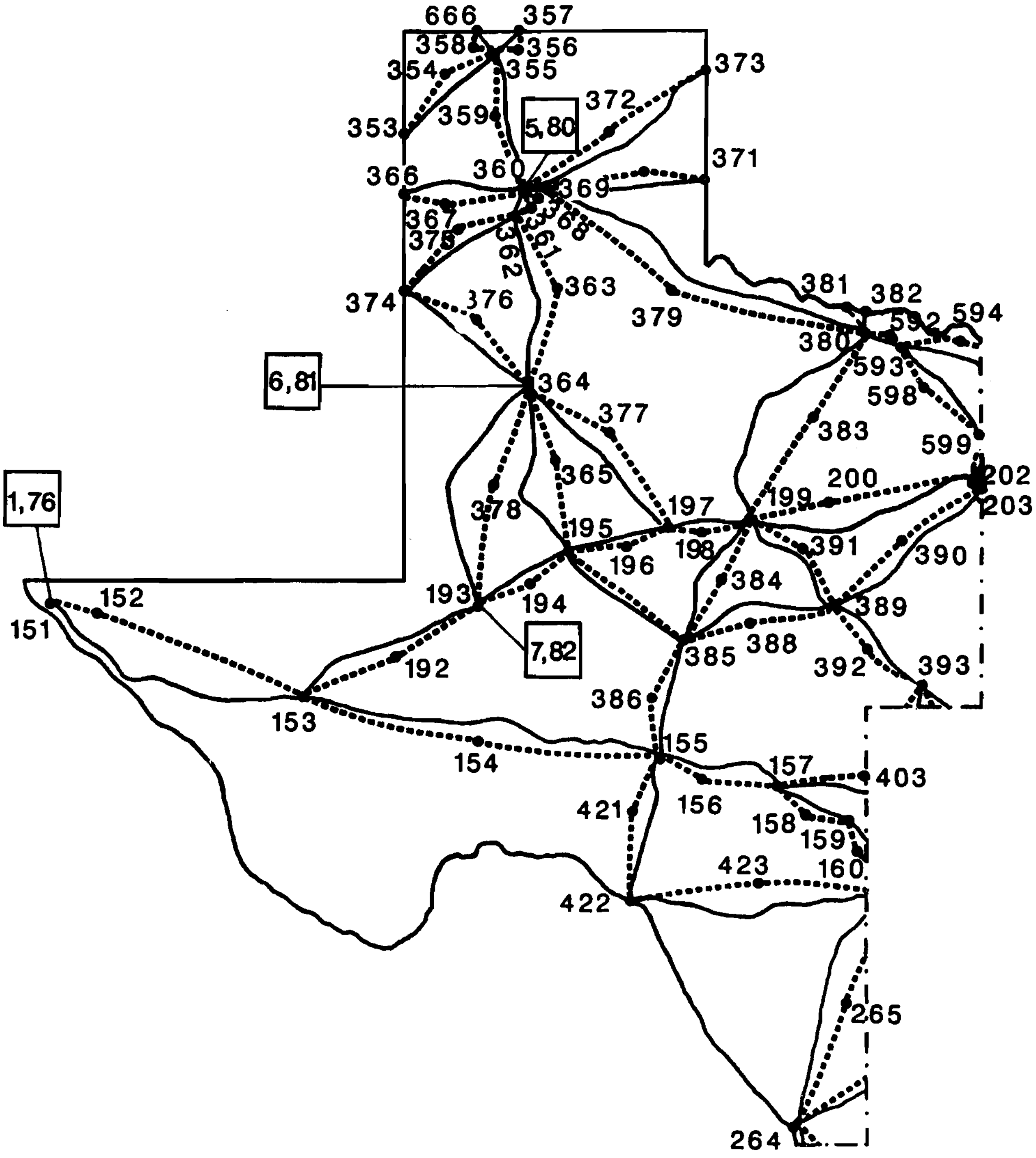

Figure 3.2.3.d Section IV

Network #3 (The Texas Network) - UE CPU =233.188 seconds
Options Open
Maximum # of Internal Iterations =1

Table 3.2.3.1

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	1	1	.258
2	1	2	.143
3	1	3	.116
4	1	4	.094
5	1	5	.112
6	1	6	.073
7	1	7	.053
8	1	8	.045
30	1	30	.006
32	1	32	.005

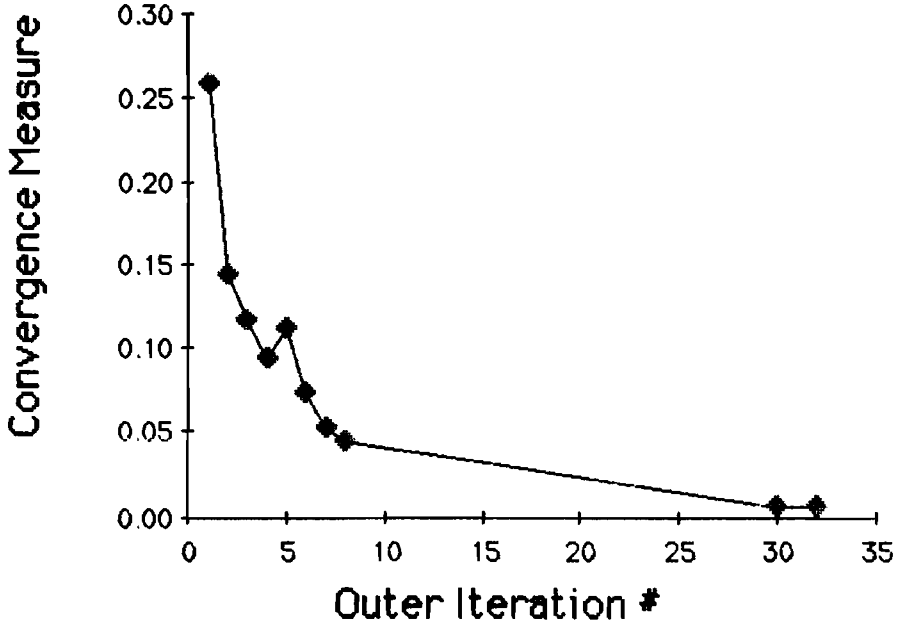

Fig. 3.2.3.2

Network #3 (The Texas Network) - UE CPU =186.607 seconds
Options Open
Maximum # of Internal Iterations =2

Table 3.2.3.2

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	2	2	.354
2	2	4	.258
3	2	6	.254
4	2	8	.149
5	2	10	.069
6	2	12	.046
7	2	14	.027
8	2	16	.021
13	1	25	.005

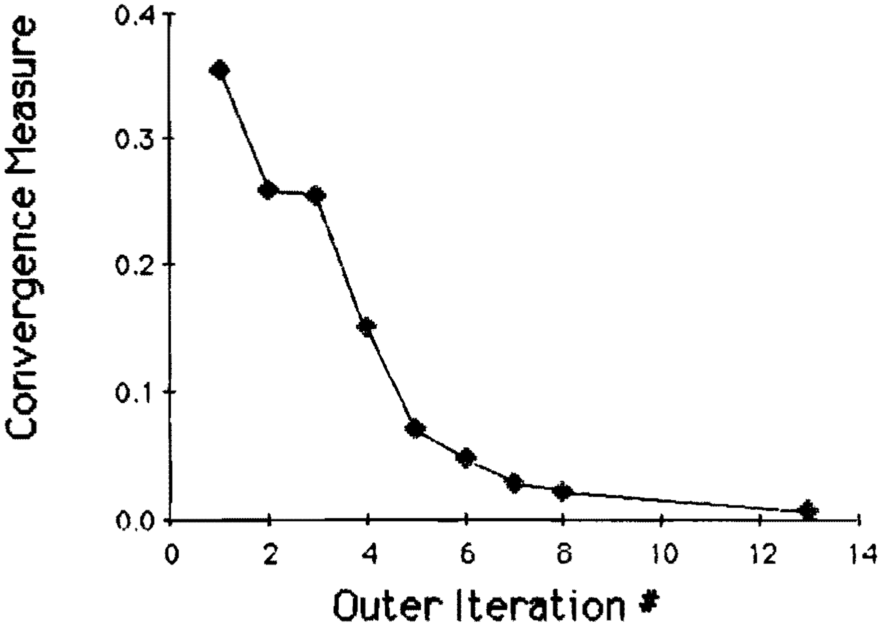

Fig. 3.2.3.3

Network *3 (The Texas Network) - UE CPU =297.713 seconds
Options Open
Maximum * of Internal Iterations =3

Table 3.2.3.3

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	3	3	.465
2	3	6	.246
3	3	9	.130
4	3	12	.073
5	3	15	.048
6	3	18	.036
7	3	21	.032
8	3	24	.023
15	1	42	.003

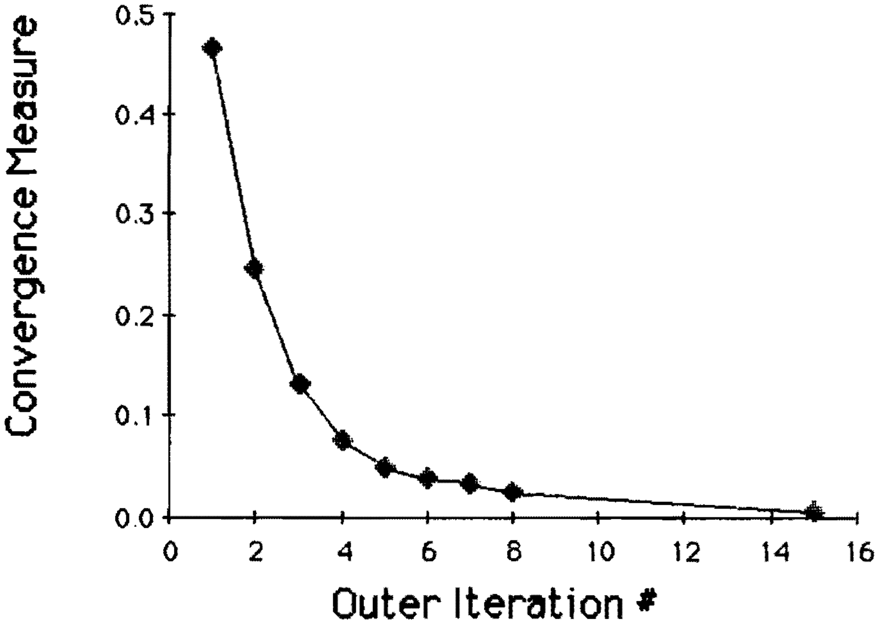

Fig. 3.2.3.4

Network *3 (The Texas Network) - UE CPU =328.196 seconds
Options Open
Maximum * of Internal Iterations =4

Table 3.2.3.4

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	4	4	.543
2	4	8	.394
3	4	12	.172
4	4	16	.094
5	4	20	.055
6	4	24	.030
7	4	28	.024
8	4	32	.019
13	1	47	.005

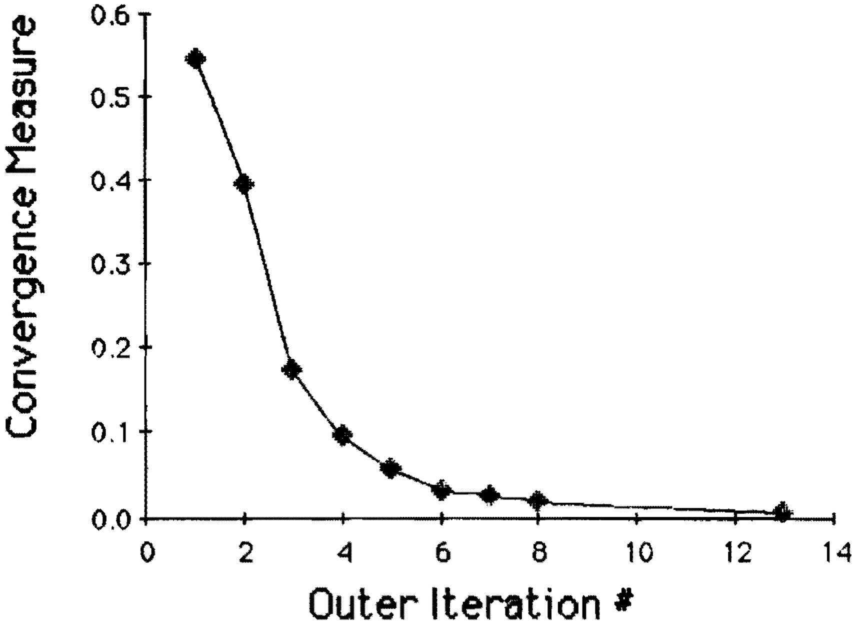

Fig. 3.2.3.5

Network *3 (The Texas Network) - UE CPU =316.117 seconds
Options Open
Maximum * of Internal Iterations =5

Table 3.2.3.5

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	5	5	.613
2	5	10	.288
3	5	15	.158
4	5	20	.087
5	5	25	.055
6	5	30	.031
7	5	35	.025
8	5	38	.012
11	1	45	.004

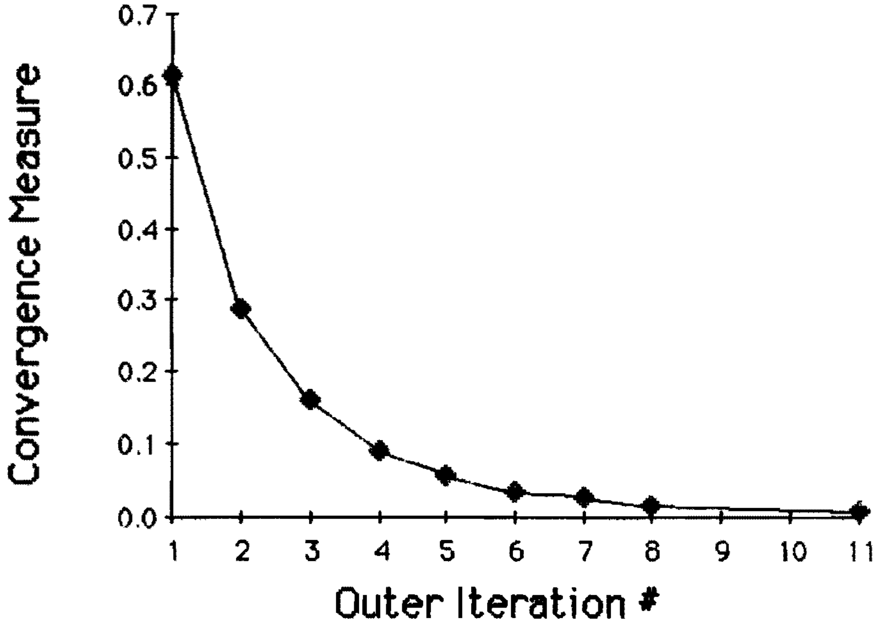

Fig. 3.2.3.6

Table 3.2.3.6 - Summary of results for Network 3 (The Texas Network) User Equilibrium - Options Open

Maximum Number of Internal Iterations	CPU - Time (Seconds)	Total Number of Internal Iterations required for Convergence
1	135.287	16
2	187.868	23
3	210.743	26
4	196.436	24
5	225.033	28

Network *3 (The Texas Network) - SO CPU =253.871 seconds
Options Open
Maximum * of Internal Iterations =1

Table 3.2.3.7

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	1	1	.348
2	1	2	.191
3	1	3	.131
4	1	4	.096
5	1	5	.075
6	1	6	.091
7	1	7	.053
8	1	8	.034
27	1	27	.005

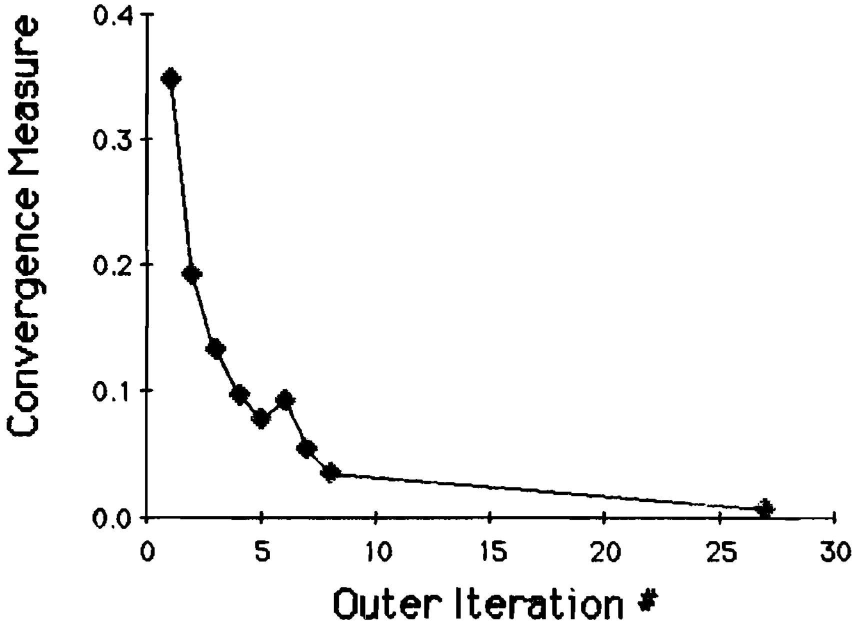

Fig. 3.2.3.7

Network #3 (The Texas Network) - SO CPU =356.381 seconds
Options Open
Maximum # of Internal Iterations =2

Table 3.2.3.8

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	2	2	.531
2	2	4	.261
3	2	6	.105
4	2	8	.074
5	2	10	.054
6	2	12	.036
7	2	14	.037
8	2	16	.025
20	1	39	.005

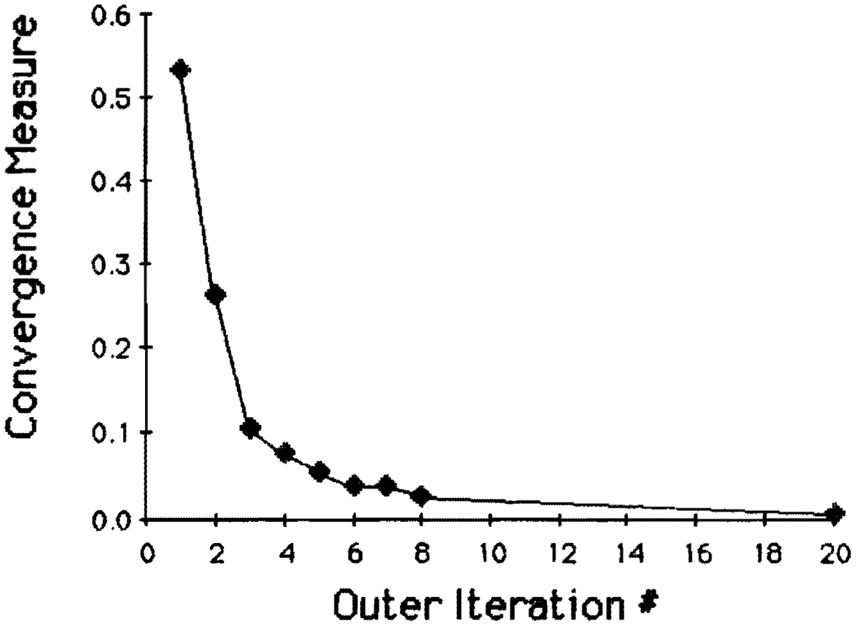

Fig. 3.2.3.8

Network *3 (The Texas Network) - SO CPU =403.483 seconds
Options Open
Maximum * of Internal Iterations =3

Table 3.2.3.9

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	3	3	.707
2	3	6	.274
3	3	9	.159
4	3	12	.100
5	3	15	.045
6	3	18	.036
7	3	21	.021
8	3	24	.022
16	1	44	.004

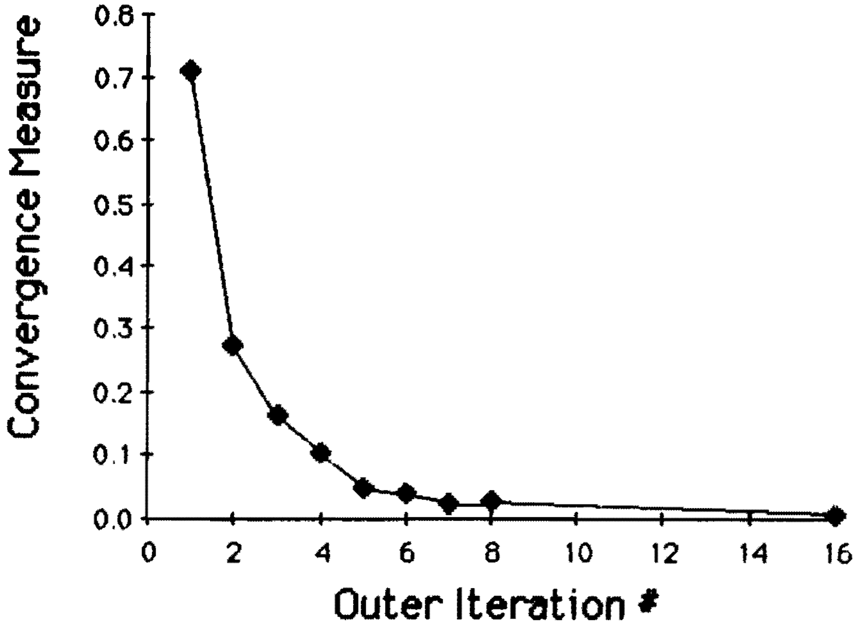

Fig. 3.2.3.9

Network *3 (The Texas Network) - SO CPU =324.542 seconds
Options Open
Maximum * of Internal Iterations =4

Table 3.2.3.10

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	4	4	.797
2	4	8	.249
3	4	12	.119
4	4	16	.077
5	4	20	.036
6	4	24	.021
7	4	28	.022
8	4	32	.017
10	1	35	.004

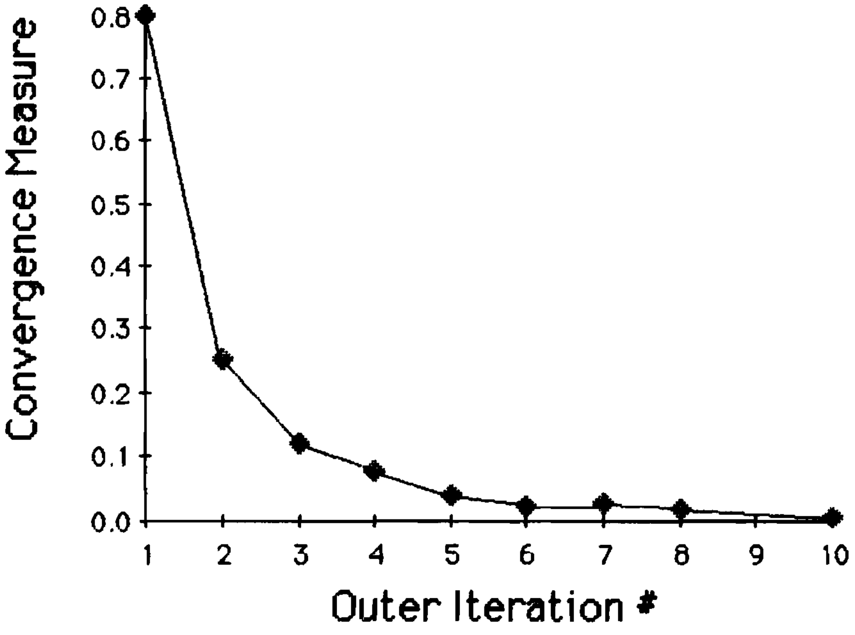

Fig. 3.2.3.10

Network #3 (The Texas Network) - SO CPU =469.369 seconds
Options Open
Maximum # of Internal Iterations =5

Table 3.2.3.11

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	5	5	.999
2	5	10	.410
3	5	15	.173
4	5	20	.086
5	5	25	.058
6	5	30	.035
7	5	35	.025
8	3	38	.014
13	1	51	.003

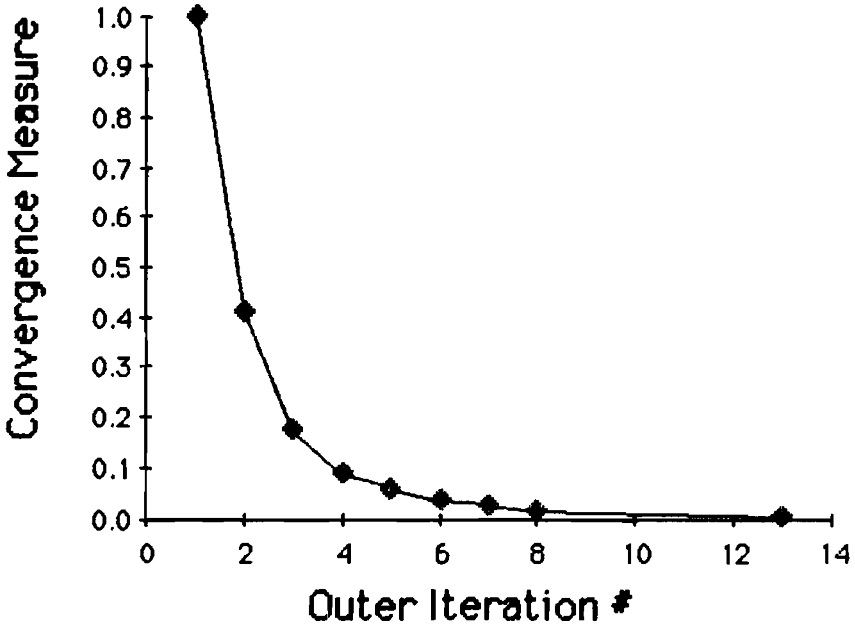

Fig. 3.2.3.11

Table 3.2.3.12 - Summary of results for Network 3 (The Texas Network) System Optimum - Options Open

Maximum Number of Internal Iterations	CPU - Time (Seconds)	Total Number of Internal Iterations required for Convergence
1	121.413	11
2	180.609	17
3	169.986	16
4	209.103	20
5	209.528	20

Network *3 (The Texas Network) - UE CPU =135.287 seconds
Options Closed
Maximum * of Internal Iterations =1

Table 3.2.3.13

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	1	1	.061
2	1	2	.057
3	1	3	.086
4	1	4	.063
5	1	5	.033
6	1	6	.043
7	1	7	.032
8	1	8	.032
16	1	16	.002

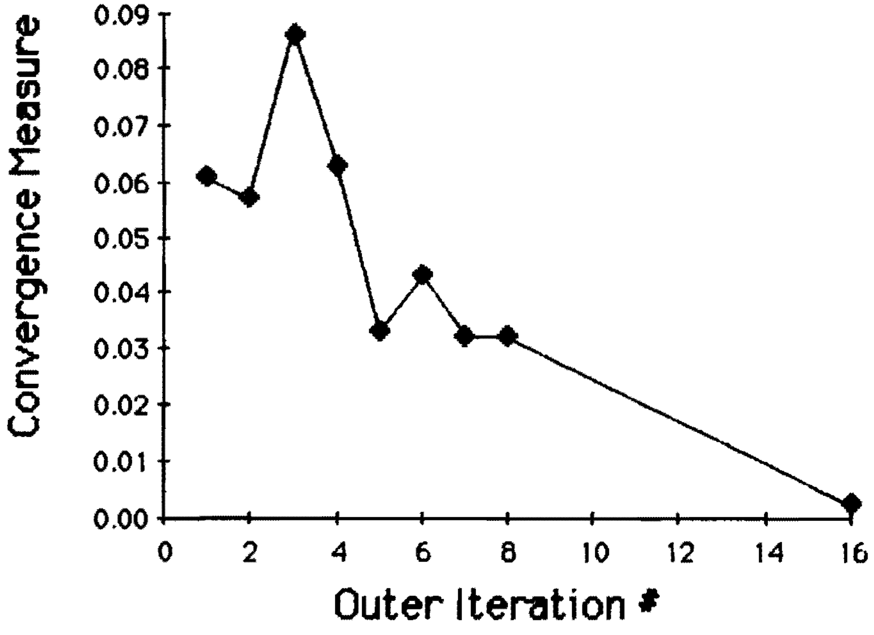

Fig. 3.2.3.12

Network *3 (The Texas Network) - UE CPU =187.868 seconds
Options Closed
Maximum * of Internal Iterations =2

Table 3.2.3.14

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	2	2	.132
2	2	4	.047
3	2	6	.035
4	2	8	.036
5	2	10	.031
6	2	12	.013
7	2	14	.017
8	2	16	.016
12	1	23	.004

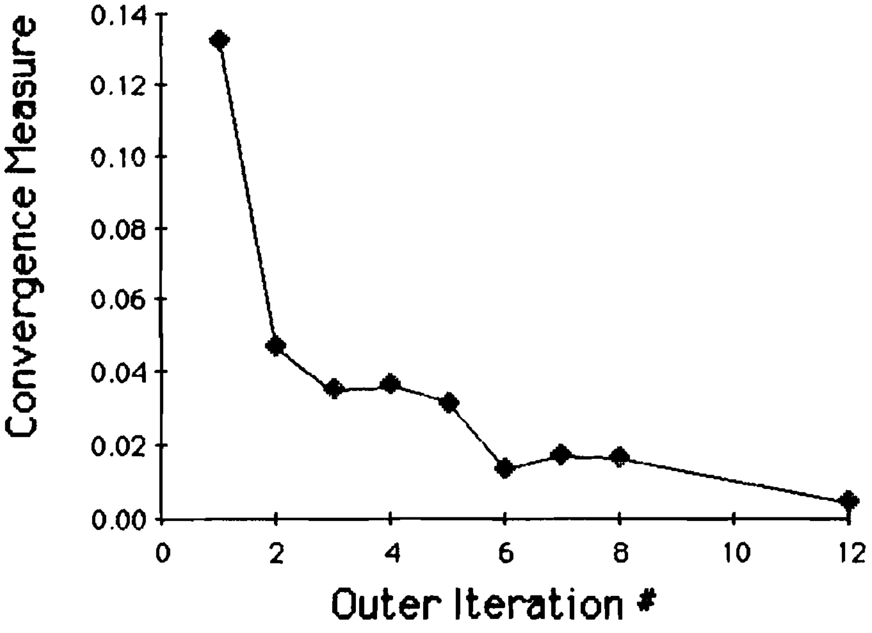

Fig. 3.2.3.13

Network *3 (The Texas Network) - UE CPU =210.743 seconds
Options Closed
Maximum * of Internal Iterations =3

Table 3.2.3.15

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	3	3	.179
2	3	6	.066
3	3	9	.066
4	3	12	.030
5	2	14	.015
6	3	17	.016
7	2	19	.015
8	2	21	.009
11	1	26	.004

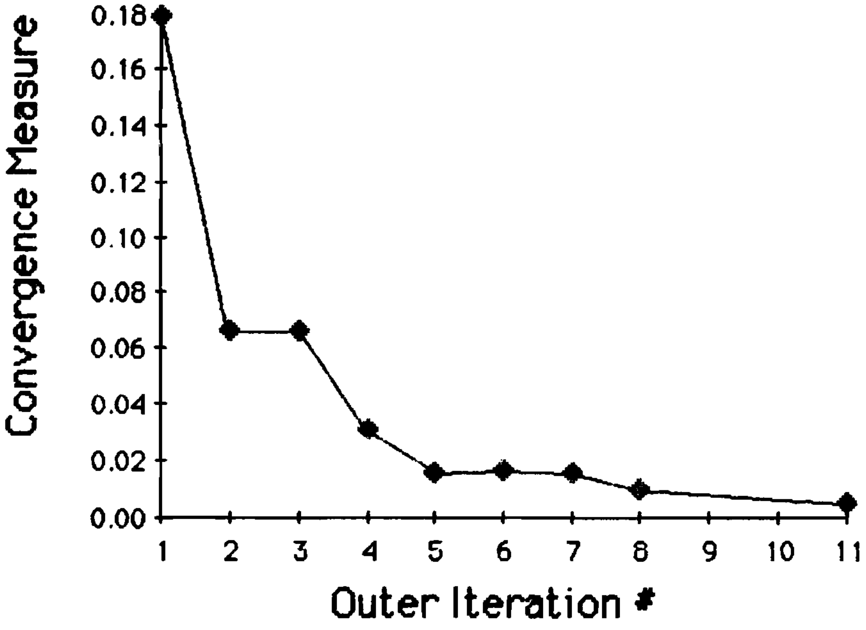

Fig. 3.2.3.14

Network #3 (The Texas Network) - UE CPU =196.436 seconds
Options Closed
Maximum # of Internal Iterations =4

Table 3.2.3.16

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	4	4	.169
2	4	8	.063
3	4	12	.069
4	2	14	.016
5	3	17	.016
6	2	19	.020
7	2	21	.015
8	2	23	.012
9	1	24	.004

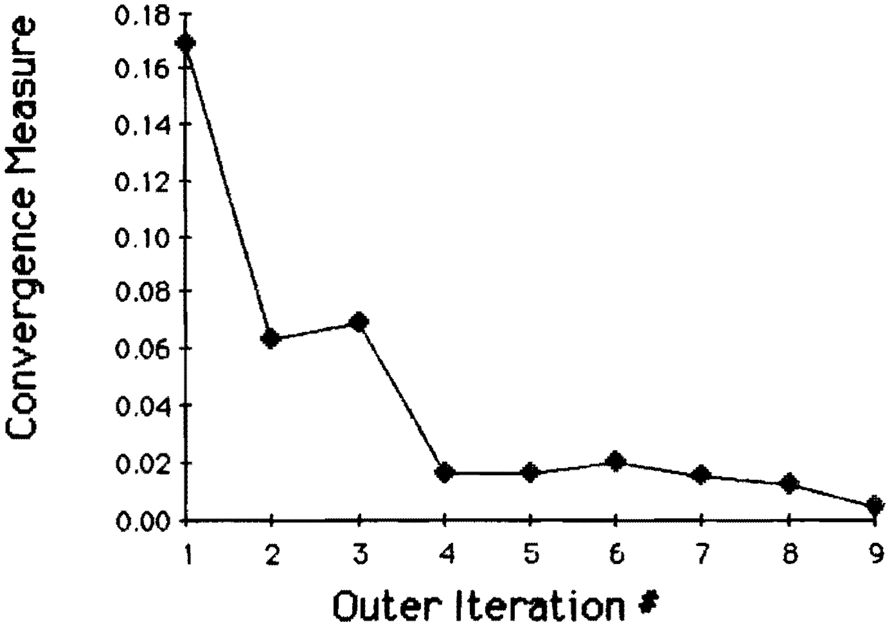

Fig. 3.2.3.15

Network *3 (The Texas Network) - UE CPU =225.033 seconds
Options Closed
Maximum * of Internal Iterations =5

Table 3.2.3.17

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	5	5	.238
2	5	10	.069
3	5	15	.045
4	4	19	.030
5	2	21	.012
6	2	23	.012
7	2	25	.010
8	2	27	.007
9	1	28	.005

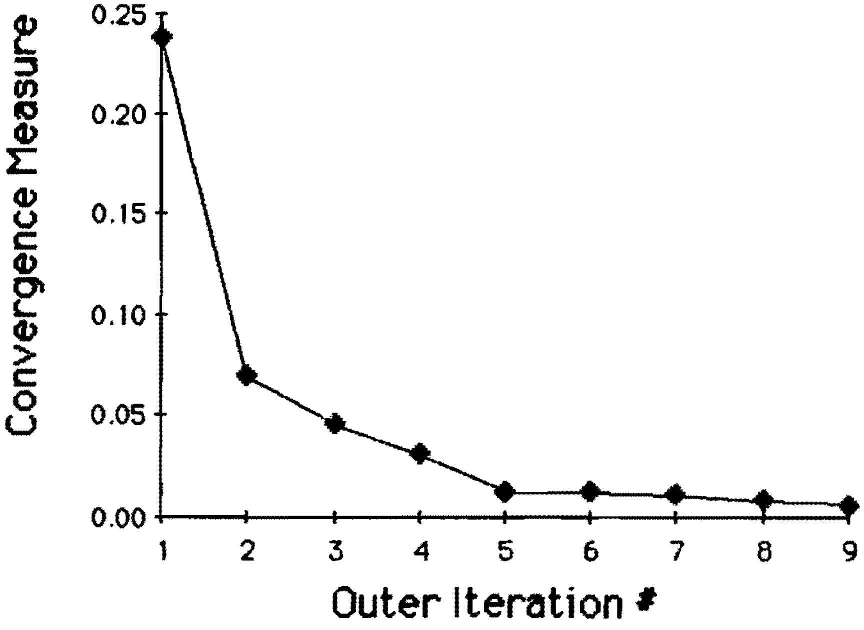

Fig. 3.2.3.16

Table 3.2.3.18 - Summary of results for Network 3
(The Texas Network)
User Equilibrium - Options Closed

Maximum Number of Internal Iterations	CPU - Time (Seconds)	Total Number of Internal Iterations required for Convergence
1	233.188	32
2	186.607	25
3	297.713	42
4	328.196	47
5	316.117	45

Network *3(The Texas Network) - SO CPU =121.413 seconds
Options Closed
Maximum * of Internal Iterations =1

Table 3.2.3.19

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	1	1	.057
2	1	2	.041
3	1	3	.047
4	1	4	.039
5	1	5	.039
6	1	6	.022
7	1	7	.015
8	1	8	.015
11	1	11	.005

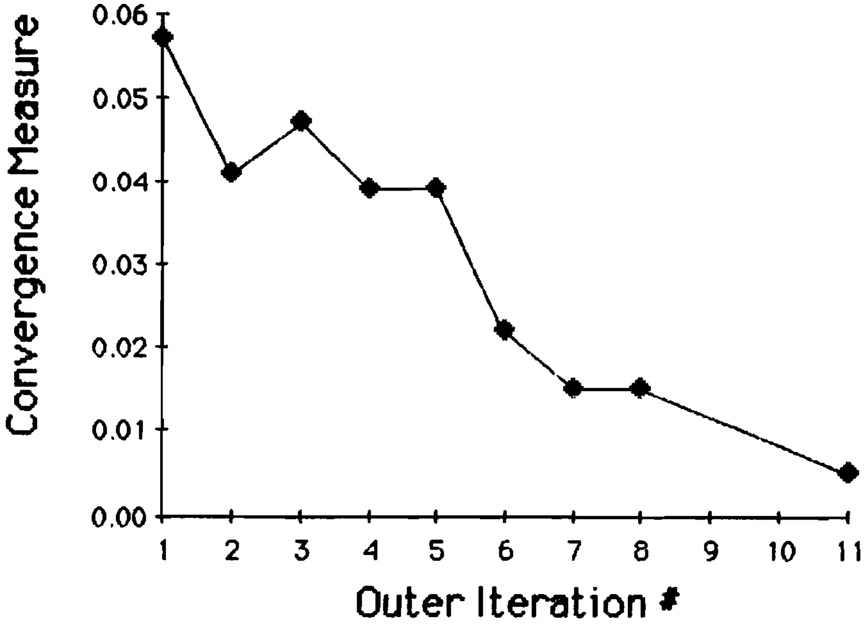

Fig. 3.2.3.17

Network #3 (The Texas Network) - SO CPU =180.609 seconds
Options Closed
Maximum # of Internal Iterations =2

Table 3.2.3.20

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	2	2	.093
2	2	4	.060
3	2	6	.029
4	2	8	.022
5	2	10	.016
6	2	12	.019
7	2	14	.010
8	2	16	.009
9	1	17	.005

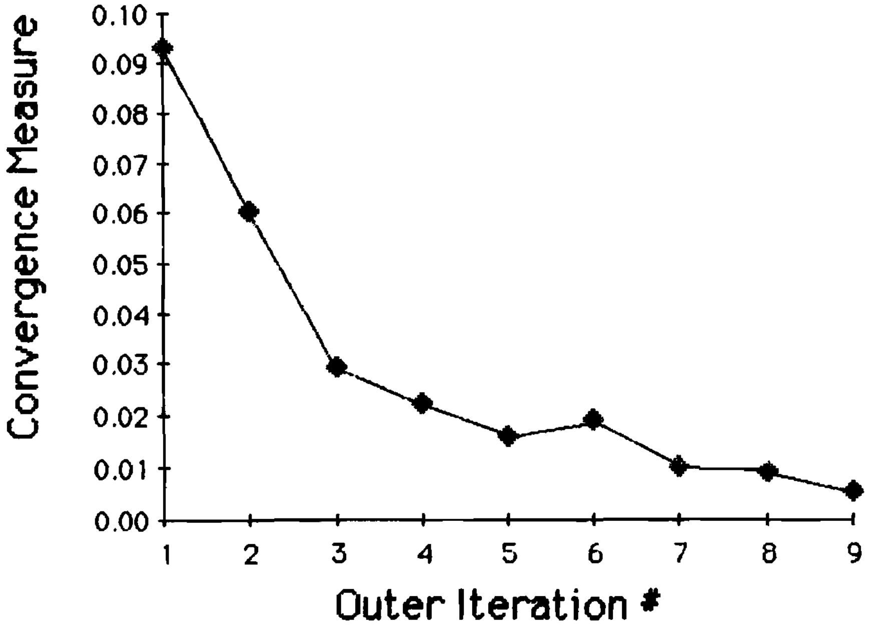

Fig. 3.2.3.18

Network *3(The Texas Network) - SO CPU =169.986 seconds
Options Closed
Maximum * of Internal Iterations =3

Table 3.2.3.21

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	3	3	.133
2	3	6	.063
3	3	9	.068
4	3	12	.028
5	3	15	.019
6	1	16	.004

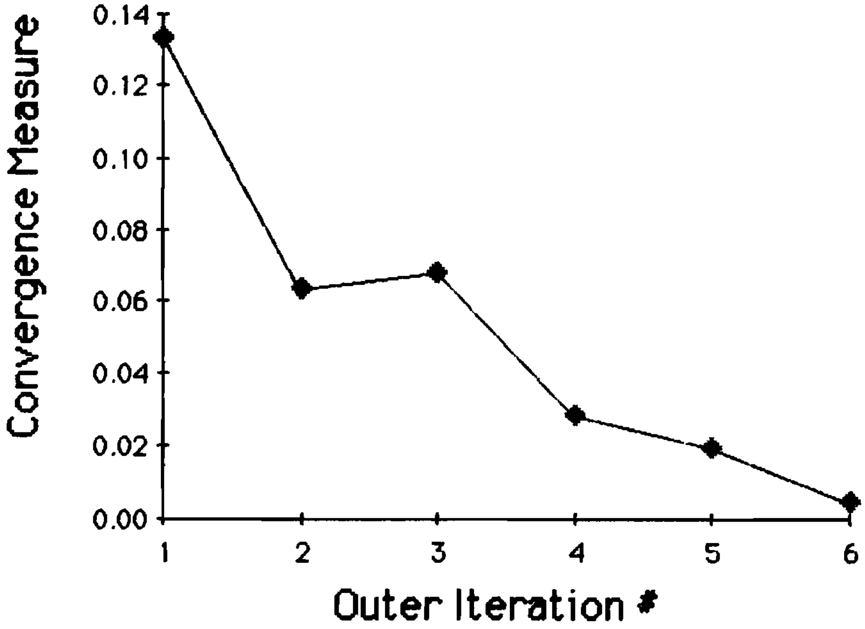

Fig. 3.2.3.19

Network *3 (The Texas Network) - SO CPU =209.103 seconds
Options Closed
Maximum * of Internal Iterations =4

Table 3.2.3.22

Outer Iteration *	Required * of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	4	4	.141
2	4	8	.063
3	4	12	.073
4	4	16	.023
5	3	19	.014
6	1	20	.002

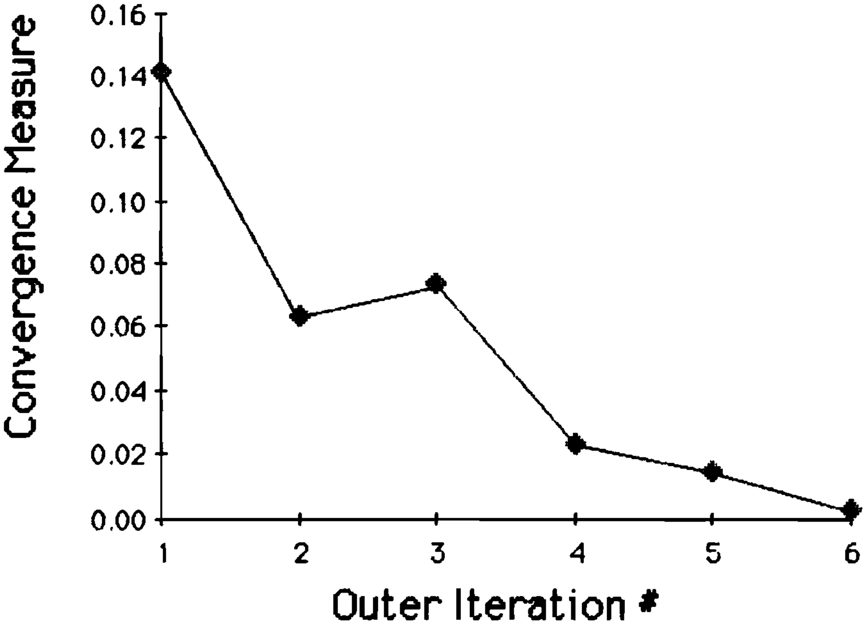

Fig. 3.2.3.20

Network *3 (The Texas Network) - SO CPU =209.528 seconds
Options Closed
Maximum * of Internal Iterations =5

Table 3.2.3.23

Outer Iteration #	Required # of Internal Iterations	Sum of Internal Iterations	Convergence Measure
1	5	5	.153
2	5	10	.059
3	5	15	.081
4	2	17	.012
5	2	19	.011
6	1	20	.002

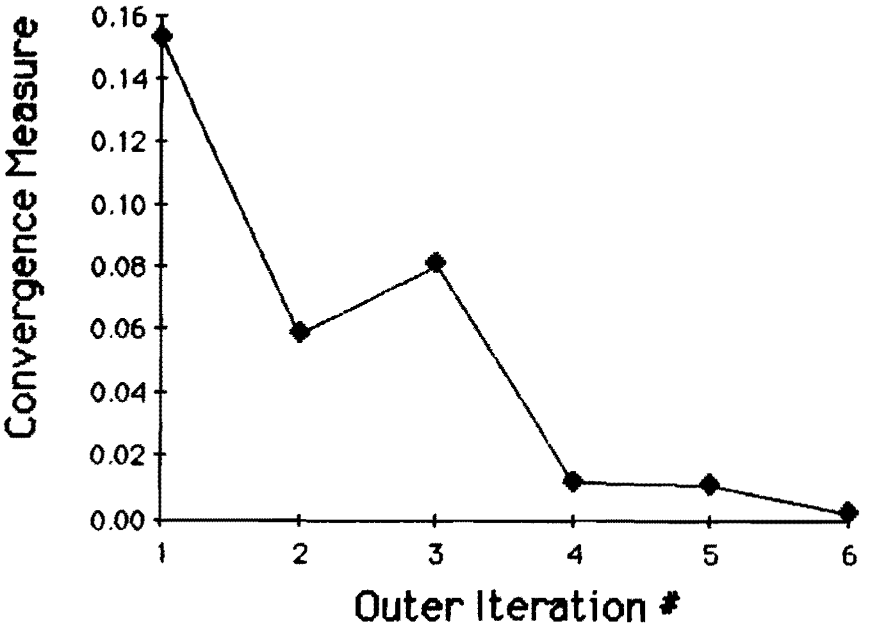

Fig. 3.2.3.21

Table 3.2.3.24 - Summary of results for Network 3 (The Texas Network) System Optimum - Options Closed

Maximum Number of Internal Iterations	CPU - Time (Seconds)	Total Number of Internal Iterations required for Convergence
1	253.871	27
2	356.381	39
3	403.483	44
4	324.542	35
5	469.369	51

3.3 Summary of Results

The first important observation is that convergence was reached in all tests conducted, on all networks and for both traffic assignment rules. As mentioned in Chapter two, the diagonalization algorithm is not based on a mathematical programming formulation, but rather on an iterative process, where if convergence is achieved, the solution is the equilibrium flow pattern in the network. The necessary conditions for the algorithm to converge are not yet established; however, some authors have provided rather restrictive sufficient conditions which, if met, guarantee that the algorithm will converge. (Dafermos, 1982). The basic condition is that the link performance functions need to be such that the travel cost of one class depends mainly on the load of that class on the link. However, this condition is not likely to hold in the case of cars and trucks sharing the same right of way. Despite the fact that this condition is not met in this study, convergence was achieved in all tests, thereby confirming the applicability of this algorithm in applications involving asymmetric interactions between cars and trucks on highway links.

The second major objective of this study was to examine possible shortcuts of the algorithm. The results show that considerable savings can be achieved by adopting some streamlining techniques. In most of the test results, using a maximum number of internal iterations of 1, 2 or 3, required much less computational effort than the original algorithm, as shown in the summary tables, presented hereafter. Tables 3.3.1 and 3.3.2 refer to the User Equilibrium assignment and tables 3.3.3 and 3.3.4 refer to the system optimum assignment rules. In tables 3.3.1 and 3.3.3, the performance of each of the maximum number of internal iterations for each test is ranked, with the best ranked first. Additionally, in parentheses, the difference between the total number of (internal) iterations required for convergence (for each maximum allowable number of iterations) and the optimum (minimum) total number of internal iterations obtained for each test is given. Tables 3.3.2 and 3.3.4 present the frequencies of the rank order position that each maximum allowable number of internal iterations was placed over all the conducted tests. Since the tests for the large Texas Network (Network 3) were carried only from 1 to 5 maximum allowable number of internal iterations, their results are considered separately from the tests performed on Networks 1 and 2, their corresponding frequencies are given in parentheses.

The basic conclusion of table 3.3.2, pertaining to the UE results is that out of nine tests conducted on Networks 1 and 2, using a maximum number of 2 internal iterations performed best five times, using 3 performed best twice while using 6 and 1 performed best once respectively. For the two tests conducted for the large Texas network, using a maximum number of 1 and 2 internal iterations, performed best once each. It can further be observed that using a maximum number of one internal iteration was ranked second six times, with a corresponding deviation from the "best" ranging from 1 to 8 total iterations. Overall, it can be observed that using a maximum of only 1, 2 or 3 internal iterations, yielded results that were ranked in the first three positions in most of the test cases. When not ranked best, the deviation (in terms of the total number of iterations) from the "best" strategy when using a maximum of two internal iterations ranged from 1 to 26 iterations; similarly, the deviation ranged from 5 to 53 iterations when using a maximum of three internal iterations. The deviations of 26 and 53 were observed in only one experiment. Excluding that experiment, the maximum deviations observed were 17 and 25 for the 2 and 3 maximum numbers of internal iterations, respectively. In general, for the higher maximum number of internal iterations tested, namely from 4 to 10, a range of difference from the best of 8 to 80 iterations was observed; further details can be found in Table 3.3.1. Similar results were observed for the SO assignment, though only 3 tests were conducted for the first two networks and two for Network 3. Combining all tests together for both the UE and the SO rules it can still be observed that using a maximum of 2 internal iterations performed best in most of the cases, followed by 1 and 3.

It can further be observed that although a maximum number of internal iterations was specified, internal convergence was attained with a lower (than that maximum) number of internal iterations after the first few corresponding outer iterations. However, clear patterns were evident in that regard with different results for the various tests. By examining the convergence measure versus the number of outer iterations, it can be seen that the shape of the curves is different. In some cases, especially for the lower values of the maximum number of internal iterations, a divergence pattern was observed in the first few iterations. This was followed by a sudden drop toward the convergence target, followed by a long tail until convergence was reached. This general shape, consisting of a sudden drop, followed by a long tail, was the general characteristic of all the tests performed. It can also be noted that those curves are not smooth, and some "flip-flopping" is observed until convergence is reached.

Another factor that was tested in these experiments is the effect of different capacity levels (1.0C, 0.8C, 0.5C, 4.0C) which effectively determine the level of congestion in the network, on the performance of the algorithm. This was tested only on Network 2, for the User Equilibrium assignment rule only. In the first experiment, where exclusive lane options were open, the number of total iterations required at level of capacity 4.0C (i.e. very low congestion levels) was significantly higher than that required for the other three levels, where no significant differences were observed. For levels of capacity 1.0C, 0.8C and 0.5C, a maximum difference of 11 total iterations was observed between these levels for any given maximum number of internal iterations. However, out of the ten tests, eight at the 4.0C capacity level exhibited differences (in the number of internal iterations) ranging between 28 and 69 iterations relative to the other three capacity levels. A somewhat contradictory result was observed for the second experiment (no additional lanes open) where the 4.0C capacity level performed better than the other three in all ten tests (Table 3.2.2.2). Additional experiments may be needed to further examine this aspect.

This algorithm was tested on a CDC 6600 computer, and the CPU time was recorded for each test. The CPU time is directly related to the total number of internal iterations. The corresponding relationships are presented in table 3.3.5. For all tests, there is a perfect linear relationship between the CPU time versus the number of internal iterations. Each iteration involves solving a shortest path problem for all O-D pairs, determining the current move size using the bisection line-search method, and updating the flows. Therefore, the more O-D pairs and the larger the network (with regard to number of links and nodes), the higher will be the cost per iteration. This can be seen in the results from each test network. A marked difference between Network 3 and the other two networks can be noted in these results. The cost per iteration for the UE in Network 3 is 7.508 seconds (TM time), whereas for Networks 1 and 2, it is .292 and .533 respectively. This is to be expected given the considerably larger size of Network 3 (the Texas

Network).

While the CPU per iteration was higher for the larger Texas Network, the total number of iterations required for this network was not significantly different from network 2, the reduced abstracted version of the Texas Network. The maximum difference observed between the two networks for the UE was 12 iterations for the case where additional exclusive lanes were closed and 8 iterations when they were open. A larger difference was observed for the System Optimal assignment, where maximum difference of 36 iterations was observed for the case where the exclusive lane options were open, with Network 3 converging faster than Network 2.

Another observation is that in all tests, the CPU time per iteration for the System Optimal (SO) assignment rule was higher than the corresponding value for the UE assignment. This is due to the more complicated link functions derived (Appendix A) to adapt the algorithm for solving the SO problems.

In conclusion, it was demonstrated that considerable savings can be achieved by using short cut techniques for the diagonalization algorithm. The so-called "streamlined" version of the algorithm, which uses only 1 internal iteration, as first suggested by Sheffi(1984), was shown to perform better than the unmodified algorithm and some of the other shortcut values tested. However, it was not uniformly the best streamlining strategy. In the tests reported in this study, using a maximum number of two internal iterations performed best in most cases. It is difficult to predict a priori which one will perform better in a particular situation. For the Texas study of truck facilities in the highway network, a maximum value of 2 internal iterations is recommended. This study is concluded in Chapter 4, where a summary of the overall work is given, followed by the conclusions and recommendations.

User Equilibrium

Table 3.3.1 - Rank order position of maximum number of internal iterations and relative difference of the total number of internal iterations from the best one performed (given in parenthesis)

Maximum Number of Internal Iterations	Network 1	Network 2								Network 3	
		Options Open				Options Closed				Options Open	Options Closed
		1.0C	.8C	.5C	4.0C	1.0C	.8C	.5C	4.0C		
1	1(0)	2(1)	2(2)	2(1)	7(56)	2(8)	2(8)	2(7)	3(6)	1(0)	2(7)
2	2(12)	1(0)	1(0)	4(8)	3(26)	1(0)	1(0)	1(0)	2(1)	2(7)	1(0)
3	3(21)	5(9)	3(5)	1(0)	5(53)	3(15)	3(9)	3(17)	1(0)	4(10)	3(17)
4	5(30)	8(13)	7(14)	3(6)	2(2)	4(25)	4(13)	4(18)	6(21)	3(8)	5(22)
5	4(29)	3(6)	6(13)	4(8)	6(53)	6(31)	5(25)	7(34)	4(8)	5(12)	4(20)
6	6(46)	5(9)	4(9)	5(9)	1(0)	8(36)	5(25)	10(51)	5(12)		
7	7(55)	7(11)	8(15)	7(13)	10(72)	7(34)	6(29)	6(25)	4(8)		
8	8(66)	6(10)	5(11)	7(13)	4(42)	5(26)	8(46)	5(24)	4(8)		
9	9(78)	9(15)	9(16)	6(11)	9(62)	7(34)	7(39)	8(42)	4(8)		
10	10(80)	4(8)	7(13)	7(13)	8(58)	9(46)	9(48)	9(45)	4(8)		

User Equilibrium

Table 3.3.2- Frequencies of rank order position for each maximum number of internal iterations with respect to the performance of each in all tests (the corresponding results for Network 3 are given in parenthesis)

Maximum number of internal iterations	Rank order position (from 1 to 10)									
	1	2	3	4	5	6	7	8	9	10
1	(1)1	(1)5	2				1			
2	(1)6	(1)2		1						
3	1	1	(1)5	(1) 1	1					
4		1	(1)	3	(1)3	1	1	2	1	2
5			2	(1) 1	(1)1	4		3		
6				1	3	1	2	1	1	
7	1			1	1		4	1	2	2
8				1		1	1	2	1	1
9									4	3
10						1	1			1

System Optimum

Table 3.3.3 – Rank order position of maximum number of internal iterations and relative difference of the total number of internal iterations from the best one perfomed (given in parenthesis)

Maximum Number of Inter. Iterations	Network 1	Network 2		Network 3	
		Options Open	Options Closed	Options Open	Options Closed
1	2(10)	10(19)	2(2)	1(0)	1(0)
2	1(0)	4(5)	1(0)	3(6)	3(12)
3	3(20)	5(6)	3(10)	2(5)	4(17)
4	5(45)	8(15)	4(13)	4(3)	2(8)
5	4(36)	9(17)	7(19)	4(3)	5(24)
6	6(48)	3(4)	8(21)		
7	7(67)	2(1)	5(15)		
8	8(108)	6(10)	6(17)		
9	9(122)	1(0)	9(25)		
10	10(150)	7(13)	10(48)		

System Optimum

Table 3.3.4 - Frequencies of rank order position for each maximum number of internal iterations with respect to the performance of each in all tests (the corresponding results for Network 3 are given in parenthesis)

Maximum number of internal iterations	Rank order position (from 1 to 10)									
	1	2	3	4	5	6	7	8	9	10
1	(2)	2							1	
2	2		(1)	(1)			1			
3		(2)	2			1				
4		1	(1)	(1)1	(1)1					
5			1	1	(1)		(1)			
6						1		2		
7					1		1			1
8				1		1		1	1	
9					1				1	
10										2

Table 3.3.5 - CPU Vs ITERATION

CPU= a + b*ITERATION

(UE)		CPU	=	.847	+	.292*ITER	r=1.00	**NET 1**
(SO)		CPU	=	.924	+	.420*ITER	r=1.00	
(UE)	1.0C	CPU	=	1.435	+	.533*ITER	r=1.00	
(UE)	0.8C	CPU	=	1.268	+	.547*ITER	r=.993	
(UE)	0.5C	CPU	=	1.564	+	.527*ITER	r=1.00	**NET 2**
(UE)	4.0C	CPU	=	1.490	+	.526*ITER	r=1.00	Options
(SO)	1.0C	CPU	=	1.326	+	.760*ITER	r=1.00	Open
(UE)	1.0C	CPU	=	1.443	+	.497*ITER	r=1.00	
(UE)	0.8C	CPU	=	1.465	+	.497*ITER	r=1.00	**NET2**
(UE)	0.5C	CPU	=	1.496	+	.497*ITER	r=1.00	Options
(UE)	4.0C	CPU	=	1.181	+	.498*ITER	r=1.00	Closed
(SO)	1.0C	CPU	=	1.792	+	.679*ITER	r=1.00	
(UE)		CPU	=	15.384	+	7.508*ITER	r=1.00	Options
(SO)		CPU	=	13.943	+	9.773*ITER	r=1.00	Open
								NET 3
(UE)		CPU	=	26.258	+	6.443*ITER	r=1.00	Options
(SO)		CPU	=	10.578	+	8.953*ITER	r=1.00	Closed

CHAPTER 4 CONCLUSIONS AND RECOMMENDATIONS

4.1 Summary of Conclusions

The basic objective of this study was to investigate the applicability of the diagonalization algorithm for network assignment with asymmetric interactions between vehicle classes to the analysis and design of truck-related highway improvements in a statewide network. In particular, it was intended to test whether this algorithm would converge to the desired solution in this type of application, as well as to determine shortcut strategies that would enhance its efficiency and reduce its computational requirements. These questions were investigated for both the User Equilibrium and the System Optimum rules of traffic assignment in three networks.

The algorithm was implemented in the form of two computer programs, developed for the UE and SO assignment rules, respectively. The algorithm was tested on three networks, including a medium-sized abstraction of the Texas network (network 2), developed for extensive testing purposes, as well as a detailed full-scale Texas network (network 3). The necessary input for each network, mainly the O-D matrices for both cars and trucks, were developed, as well as information on each link's characteristics. This study was conducted in conjunction with the development of an overall design and analysis methodology (Mahmassani et. al., 1985) for the assessment and evaluation of truck lane needs and improvements in the Texas highway network. An accomplishment of this study was the special structure devised to represent different types of link improvements. This representation can handle not only lane additions to existing links, but also more general improvements consisting of capacity expansion jointly with operational strategies in terms of lane access restrictions to either class of vehicles. As such, one can analyze the effect of exclusive truck lanes, exclusive car lanes, shared-use lanes, as well as lane use restrictions effecting certain truck categories. The interaction between the two vehicle classes was handled through a modification of the BPR curves, as described in Section 3.1.

The results of the tests performed in this study indicated that convergence was achieved in all cases, confirming the algorithm's appropriateness for this type of application. This is an encouraging result, since the convergence of the algorithm is not guaranteed, as discussed in Chapter two. Also tested was a streamlined version (Section 2.1) of the algorithm, as suggested by Sheffi (1984). In addition, a more general streamlining strategy was devised, consisting of varying the maximum allowable number of internal iterations in the algorithm from 1 to 10. The tests conducted revealed that for the type of application context under consideration the use of a maximum number of two internal iterations performed best in most of the cases addressed. Using one internal iteration (Sheffi's proposal) followed, consistently ranking as the second best in most tests, and the best in some. In general, it is not recommended to allow more than four internal iterations given their poor performance observed in almost all tests.

Overall, the results are quite positive, since the algorithm performed well on a large highway network. Using the special structure devised, it can be implemented to determine flows on the Texas network for both vehicle classes, and to further examine the effects of different improvement options. As mentioned earlier, both the UE and SO assignment rules were tested. While the user equilibrium principle is recognized as more realistic in terms of describing individual route choice behavior, thereby making it a more appropriate descriptive tool than the system optimal model, the latter is quite important because of its role in network design models, where it is used to provide a lower bound for the minimum travel time.

4.2 Limitations and Recommendations for Further Research

This study focused on implementing the diagonalization algorithm to networks where two classes of users are operating. Therefore the link performance functions modified for this purpose take into account only two vehicle classes: passenger cars vs. trucks. A natural extension would be to explicitly define different truck classes. The principal difficulty in this regard is empirical, requiring data for calibration. It should be noted in this regard that the functions used in this study were developed based on engineering considerations, and intended principally for testing the algorithm; as such, they are to be used with caution, and should not be considered as final observationally-based functions. Future implementation of the algorithm for actual policy and engineering studies would require actual observations of traffic interactions of the different vehicle classes in the traffic stream. Preliminary efforts in this direction are underway in a Center for Transportation Research study aimed at calibrating travel cost functions, which explicitly recognize different classes of vehicles, using available national data collected by FHWA. However, it would be highly desirable to systematically develop such travel cost functions for different link types in the Texas Network or any other network for which it is intended to apply the algorithm.

Another important element is the development of the O-D matrix for the preliminary Texas Network. As noted, the one used in this study was intended mostly for methodological development and testing, and not for direct policy and planning decisions. Additional research is needed to develop more detailed and representative origin-destination patterns for passenger cars, commodity flows and truck movement. Some algorithmic approaches have been suggested in the literature for the development of O-D matrices. Modifications of these methods for adaptation to the objectives of a particular application may be useful. Coupled with improved travel cost functions, the procedures developed and tested in this study can provide a valuable tool to obtain the distribution of both truck and passenger car flows (or other different classes that the traffic stream might be divided into) on the highway links.

Although the algorithm performed well on the tests conducted it should be noted that further improvements can be implemented. Both programs developed can be improved for more efficiency, and the design structure of the network can also be improved. Although the computational time required for the algorithm to converge seems to be rather encouraging, its use within a formal network design optimal improvement search technique could be a burden, given the need for repeated application of the assignment algorithm (Mahmassani et. al., 1985).

This study was entirely focused on the diagonalization algorithm for solving for the User Equilibrium assignment problem with asymmetric link interactions. As noted in the introduction, another approach proposed in the literature is the so-called projection method. This method can also be implemented and compared to the results of this study. This will provide a more complete analysis of deterministic algorithmic approaches for solving the user equilibrium assignment problem. In addition, one can introduce stochastic elements in this problem; however, operationally viable approaches to solve such a problem remain in their infancy at this stage.

REFERENCES

1. Abdulaal, M., and L.J. LeBlanc (1979). Methods for Combining Modal Split and Equilibrium Assignment Models.*Transportation Science* *13*(4), pp. 292-314.

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

3. Branston, D. (1976). Link Capacity Functions: A Review. *Transportation Research* *10* (4), pp. 223-236.

4. Dafermos, S.C. (1971). An Extended Traffic Assignment Model with Application to Two-Way Traffic. *Transportation Science* *5*(4), pp. 366-389.

5. Dafermos, S.C. (1976). Integrated Equilibrium Flow Models for Transportation Planning. In M. Florian (Ed.), *Traffic Equilibrium Methods*, Lecture Notes in Economics and Mathematical Systems 118, Springer-Verlag, New York, pp. 106-118.

6. Dafermos, S.C. (1980). Traffic Equilibrium and Variational Inequalities. *Transportation Science* *14* (1), pp. 42-54.

7. Dafermos, S.C. (1982). Relaxation Algorithms for the General Asymmetric Traffic Equilibrium Problem. *Transportation Science* *16* (2), pp. 231-240.

8. Davidson, K.B. (1966). A Flow-Travel Time Relationship for Use in Transportation Planning. *Proceedings, Australian Road Research Board*, Melbourne, Vol. 3, pp. 183-194.

9. Fisk, C., and S. Nguyen (1982). Solution Algorithms for Network Equilibrium Models with Asymmetric User Costs. *Transportation Science* *16* (3), pp. 361-381.

10. Florian, M., et. al (1979) Validation and Application of EMME: An Equilibrium Based Two-Mode Urban Transportation Planning Method, Centre de Reserche surles Transports, Universite of Montreal, Canada.

11. Frank, M., and P. Wolfe (1956), An Algorithm for Quadratic Programming. *Naval Research Logistics Quarterly* *3* (1-2), pp. 95-110.

12. Mahmassani, H.S., et.al (1985). A Methodology for the Assessment of Truck Lane Needs in the Texas Highway Network, CTR Report 356-3F, University of Texas at Austin.

13. Mosher, W.W. (1963). A Capacity Restraint Algorithm for Assigning Flow to a Transport Network, Highway Research Record 6, pp. 41-70.

14. Nagurney, A.B. (1983). Comparative Tests of Multimodal Traffic Equilibrium Methods. U.S. DOT. Program of University Research Report, Brown University, Providence, Rhode Island.

15. Nguyen, S. (1974a). A Unified Approach to Equilibrium Methods for Traffic Assignment. In M. Beckmann and H.P. Kunze (Eds.), Traffic Equilibrium Methods, Lecture Notes in Economics and Mathematical Systems. Springer-Verlag, New York.

16. Nguyen, S. (1974b). An Algorithm for the Traffic Assignment Problem. Transportation Science 8 (3), pp. 203-216.

17. Sheffi, Y. (1984). Urban Transportation Networks. Prentice-Hall, Englewood Cliffs, New Jersey, pp. 204-230.

18. Sheffi, Y. (1984). Urban Transportation Networks. Prentice-Hall, Englewood Cliffs, New Jersey, pp. 216-217.

19. Sheffi, Y. (1984). Urban Transportation Networks. Prentice-Hall, Englewood Cliffs, New Jersey, pp. 220-227.

20. Smith, M.J. (1979). The Existence, Uniqueness and Stability of Traffic Equilibria. Transportation Research 13B (4), pp. 295-304.

21. U.S. Bureau of Public Roads (1964). Traffic Assignment Manual. U.S. Department of Commerce, Washington, D.C.

22. Wardrop, J.G. (1952). Some Theoretical Aspects of Road Traffic Research. Proceedings, Institution of Civil Engineers II (1), pp. 325-378.

23. Wardrop, J.G. (1968) Journey Speed and Flow in Central London. Traffic Engineering and Control, 9, pp. 528-532.

APPENDIX A

DOCUMENTATION OF THE USER EQUILIBRIUM AND SYSTEM OPTIMAL TRAFFIC ASSIGNMENT COMPUTER PROGRAMS

In this appendix, a brief description of the User Equilibrium and System Optimum programs is given. Listings of the computer programs are included in Appendix C. A description of the input data variables is given in Appendix B.

A.1. The User Equilibrium Computer Program

The diagonalization algorithm was presented in Chapter two along with the basic steps of the Frank-Wolfe algorithm, which is a method for solving problem [2.1.1] of STEP 1 of the algorithm. This iterative procedure is followed in this computer program.

The iterative process is taking place in subroutine UED, where all intermediate steps needed for the application of the convex combinations algorithm are called from the appropriate subprograms.

The initialization STEP 0 takes place from DO LOOP 27 to DO LOOP 29. All paths at first from all origins to all destinations are assigned the value of zero. Then all the flows on the links are also initialized to zero by calling subroutine AONUED. This is achieved with DO LOOP 10 of subroutine AONUED. In addition, DO LOOP 10 assigns all the parameters needed to calculate the travel cost of each link, which is also calculated there. Subroutine AONUED then calls subroutine SHPUED, which calculates the shortest path between an origin and a destination. This is continued iteratively to cover all origins and destinations. Then subroutine AONUED continues to identify all links which are included in each shortest path. Finally, an all-or-nothing assignment takes place, by assigning the demand of an O-D pair on the links that are included in the shortest path. The above procedure takes place in DO LOOP 20, ending STEP 0 yielding a feasible link-flow pattern vector X^0.

The next step, STEP 1, involves solving problem [2.1.1], also called the diagonalization step. STEP 1 and STEP 2 are included in DO LOOP 11 in subroutine UED. Using the link-flow vector X^0 subroutine AONUED is called, which updates the travel cost on each link based on the new link flows, and again performs an all-or-nothing assignment yielding a set of (auxiliary) flows Y^n. Then subroutine BISUED is called where the move size a_n is determined, which then is used to update the link flows.

Subroutine BISUED uses the bisection method to find the scalar a_n which minimizes the objective function shown in STEP 3 of the streamlined version of the algorithm. Then the updated flows are checked for convergence; the process is continued iteratively until internal convergence is achieved, or a prespecified number of iterations is reached. This yields a link-flow vector X^1. This process is controlled under DO LOOP 14.

Next, STEP 2 of the diagonalization algorithm is performed by checking the convergence criterion, comparing the closeness of the link-flow vector X^1 with X^0. If convergence is not achieved, then the process is continued iteratively under DO LOOP 11 until convergence is achieved or a maximum number of outer iterations is reached.

The printout of information generated in the intermediate iterations, as specified by the model user, is controlled by calling subroutine DUMPUED. A description of the printed output is given hereafter.

First, an "echo check" on the number of O-D pairs is given. Second, the convergence criterion required in STEP 2 of the diagonalization algorithm is printed, followed by the internal convergence criterion required in STEP 1 of the diagonalization algorithm. The fourth item printed is accuracy of the move size used in subroutine BISUED, followed by the maximum number of allowable internal iterations. All the above items are initially specified by the model user to control the execution of the program.

The second category of information consists of the series of intermediate iteration outputs as specified in the input data through the variable DMP. These intermediate outputs contain the following information.

1. The current iteration number
2. The number of required internal iterations
3. The convergence measure level
4. The passenger car travel time
5. The truck travel time

When convergence is achieved or the maximum number of outer iterations is reached, the auto-link flows and truck-link flows are printed. Also the volume to capacity ratio is given for each link as well as the percent of trucks, the total AADT and the link identification number. In this output, only the highway links are presented. The centroid connectors and the access and egress links are not printed. If it is desired to print the link flows in the intermediate outputs, then the following line should be deleted from subroutine DUMPUED:

```
IF (CONV. GT. EPS) RETURN
```

In the next section, the modifications needed for the System Optimum program are presented.

A.2 The System Optimum Program

This program follows the same procedure discussed in the User Equilibrium program. The basic difference lies in the specification of the link travel cost functions. The User Equilibrium uses the actual travel cost functions, while the System Optimum uses the modified marginal "travel cost" functions as derived in Chapter 2, given by expression [2.2.7]. The form of these modified functions, derived for the particular actual travel cost functions used in this study, is given hereafter.

As presented in Chapter 3, the actual travel cost functions have the following form:

$$t_{aA}\ (X_{aA},X_{aT}) = t_o \left[1.0 + A1\left(\frac{X_{aA} + E \,.\, X_{aT}}{C_a}\right)^{B1}\right]$$

$$t_{aT}\ (X_{aA},X_{aT}) = t_o \left[1.0 + A2\left(\frac{X_{aA} + E \quad X_{aT}}{C_a}\right)^{B2}\right]$$

The modified functions for the SO program then are as follows:

$$\tilde{t}_{aA}\ (X_{aA},X_{aT}) = t_{aA}\ (X_{aA}X_{aT}) + X_{aA}\,\frac{dt_{aA}(X_{aA},X_{aT})}{dX_{aA}}$$

$$+ X_{aT}\,\frac{dt_{aT}(X_{aA}X_{aT})}{dX_{aA}}$$

$$= t_{aA}\ (X_{aA},\ X_{aT}) +$$

$$+ t_o\, X_{aA} + t_o\left(\frac{X_{aA}\ A_1}{B1-1}\right)\left(\frac{1}{C_a}\right)^{B1} \cdot \left(X_{aA} + E\ .\ X_{aT}\right)^{B1-1}$$

$$+ t_o\, X_{aT} + t_o\left(\frac{X_{aT}\ A_2}{B2-1}\right) \cdot \left(\frac{1}{C_a}\right)^{B2} \cdot \left(X_{aA} + E\ .\ X_{aT}\right)^{B2-1} \cdot E$$

$$= t_{aA}\ (X_{aA},\ X_{aT}) +$$

$$+ t_{aA} (X_{aA}, X_{aT}) \cdot X_{aA} \cdot \frac{1}{X_{aA} + E \cdot X_{aT}} \cdot \frac{1}{B1-1} +$$

$$+ t_{aT} (X_{aA}, X_{aT}) \cdot X_{aT} \cdot \frac{1}{X_{aA} + E \cdot X_{aT}} \cdot \frac{E}{B2-1}$$

This expression is used in subprogram FUNCTION COSTFN. This is the only substantial change from the User Equilibrium program. Some minor changes which might be observed do not affect any of the steps of the diagonalization algorithm. The output has the same form as that used in the User Equilibrium. A listing of the two computer programs is included in Appendix C. Program UETRDIA refers to the UE computer program and program SOTRDIA refers to the SO computer program. A sample of the output is also included in Appendix C.

APPENDIX B

INPUT DATA

Appendix B.1

Description of Sample Network

In this appendix, a small network is generated in order to provide an illustration of the required data needed for running the programs described in Appendix A. In addition to this, a description is given, (in connection with the network design methodology) of the network structure chosen to represent the different options that the proposed model can handle. In section B-1 a listing of the input data for Network 2 is given. In Section B.2 a description of the data base development of Network 3 (the large Texas Network), is also provided. Following, the network inputs are described.

A graph representation of the example network is shown in Fig. (B-1). The origins and destinations (centroids) are represented by a square whereas nodes where no flow is generated or destinated to, and which serve as link start and end nodes, are represented by a circle. As mentioned previously, the traffic stream is divided into passenger cars and trucks. In Fig. (B-1) numbers 1 to 12 represent the passenger car network, whereas the truck network, which is a replica of it, is denoted by numbers 13 to 24. Note that these two networks are a conceptual representation of the common physical highway network shared by the two vehicle classes.

Nodes 1, 2 and 3 represent the origins of the passenger cars and nodes 4, 5 and 6 represent the destination. Links 1-7, 2-9, and 3-11 represent the centroid connectors for passenger cars. Links 4-7, 5-9, 6-11 and 7-4, 9-5, 11-6 represent the access and egress links of the passenger car network respectively. Links 7-9, 9-11, 11-7 and 9-7, 11-9, 9-7 represent the physical links of the network. Links 7-9, 9-10, 7-12 and 8-7, 10-9, 12-7 represent dummy links which act in an on and off operation by assigning a zero or very high cost respectively, thus traffic is either allowed to enter, or not to enter this link. Links 8-9, 10-11, 11-12 and 9-8, 11-10-12-11 represent improvements of the physical links in form of a lane addition. The truck network follows the same structure, with the corresponding numbering as listed below.

Origins: 13, 14, 15

Destinations: 16, 17, 18

Physical links: 19-21, 21-19, 21-23, 23-21, 23-19, 19-23

Dummy links: 19-20, 20-19, 21-22, 22-21, 19-24, 24-19

Improvement links: 20-21, 21-20, 22-23, 23-22, 23-24, 24-23

For the network design purposes, the following link numbering is used:

Physical links (cars): 1,2,3 in one direction

Physical links (trucks): 10,11,12 in one direction

Physical links (cars): 19,20,21 in opposite direction

Physical links (trucks): 28,29,30 in opposite direction

Dummy links (cars): 4,5,6, in one direction

Dummy links (trucks): 13,14,15 in one direction

Dummy links (cars): 22,23,24 in opposite direction

Dummy links (trucks): 31,32,33 in opposite direction

Improvement links (cars): 7,8,9 in one direction

Improvement links (trucks): 16,17,18 in one direction

Improvement links (cars): 25,26,27, in opposite direction

Improvement links (trucks): 34, 35, 36 in opposite direction

Under the above description of the network, the following options of link improvements can be achieved:

Option 1: Open a new lane for both cars and trucks. Assign zero cost to both car and truck dummy links.

Option 2: Open an exclusive lane for cars. Assign a zero cost to the dummy link for cars and a very high cost to the dummy link for trucks.

Option 3: Open an exclusive lane for trucks. Assign a very high cost to the dummy link for cars and a zero cost to the dummy link for trucks.

Option 4: Divide the truck and car traffic. Assign a high cost to the physical links for both car and truck links, and assign a zero cost to the dummy links of both classes.

Option 5: Restrict cars from using some lanes of the road. Assign a high cost to the car physical link and assign a zero cost to both dummy links for cars and trucks respectively.

Option 6: Restrict trucks from using some lanes of the road. Repeat same policy of option 5 in the reverse form.

It should be noted that the capacity of the links to be improved, employing any form of the above mentioned options, should change accordingly. Also, any combination of the options can be used in the network model design.

Following, the input data are given for the sample network as they appear in the User Equilibrium and System Optimum Programs, presented in Appendix A.

READ INTI, EPS, CCINTI, ACCBIS

where

INTI is the maximum number of internal interations; used in subroutine UED, referring to STEP 1 of the diagonalization algorithm.

EPS is the convergence criterion, referring to STEP 2 of the algorithm

CCINTI is the convergence criterion used as a measure for solving STEP 1.

ACCBIS is the accuracy of the move size, used in subroutine BISUED calculated in STEP 1 of the algorithm.

The measure used for both EPS and CCINTI is the following:

$$\frac{1}{A} \sum_{i} \frac{X_i^{n+1} - X_i^n}{X_i^{n+1}} \leq \text{EPS or CCINTI.} \quad \forall i$$

where X_i^{n+1} and X_i^n denote the updated flows on link i of iteration (n + 1) and the flow of iteration (n) respectively, and

A represents the total number of links.

In this study INTI ranged from 1 to 10 iterations, EPS and CCINTI had a value of .005 respectively and ACCBIS had a value of .00005.

READ NARC, NCENT, NNODA, NNODT, NOD, NDMP

where NARC is the total number of arcs

NCENT is the number of centroids

NNODA is the number of passenger car nodes plus one

NNODT is the total number of nodes plus one

NOD is the number of O-D pairs

NDMP is the number of printed iterations for the output.

For the sample network the above variables have the following values.

NARC = NCENT x 3x2 + 9x4 = 54

NCENT = 3

NNODA = 12+1 = 13

NNODT = 24+1 = 25

NOD = 2x6 = 12

NDMP = any number of intermediate iterations one wishes to have on the output

READ ALP(I), BET(I), ALP1(I), E(I)

where

ALP is the a parameter in the travel cost values

BET is the b parameter in the travel cost function.

ALP1 is used for the single class User Equilibrium program; not needed in this study.

E is the passenger car equivalent parameter used in the travel cost functions.

READ DMP(I)

where DMP is an array of the intermediate iterations one wishes to be printed in the output. The maximum number is controlled by NDMP.

READ TOO(I), L(I), C(I), UEL(I), TYP(I), T(I), LINK(I), RL(I)

where

TOO(I) is the end node of each link. The start node of each link. The start node of each link is represented by a forward star array j. An example is given below using the sample network.

Start Node		End Node		
Cars	Trucks	Cars	Trucks	Forward Star
1	13	7	19	1
2	14	9	21	2
3	15	11	23	3
4	16	7	19	4
5	17	9	21	5
6	18	11	23	6
7	19	4	16	7
7	19	8	20	12
7	19	9	21	14
7	19	11	23	19
7	19	12	24	21
8	20	7	19	26
8	20	9	21	28
9	21	7	19	29
9	21	8	20	30
9	21	5	17	31
9	21	10	22	32
9	21	11	12	33
10	22	9	21	34
10	22	11	23	39
11	23	10	24	41
11	23	9	21	46
11	23	6	18	48
11	23	12	24	53
11	23	7	19	55
12	24	7	19	
12	24	11	23	

It should be noted here that the order in which the car node end array is set should be followed symmetrically with the corresponding truck node end. The forward star array always starts with number one. The start nodes should be followed sequentially. If a number is missing from the sequence, then the last number that was calculated in the forward star array appears one more time. If many numbers are missing in sequence, then the last number appears as many times as the sum of the missing numbers and the forward star continues to calculate all the remaining start nodes. Taking the forward star array, one should be able to extract the end nodes that correspond to the start nodes of all links of the network. This is achieved by subtracting two sequential numbers of the forward star vector, where their difference corresponds to the number of links the start node is connected to. If the difference is zero then that particular start node is not included in the network. The last number of the forward star vector should be the number of links plus one.

L(I) is the link length array

C(I) is the link capacity array

VEL(I) is the free flow speed

TYP(I) is the I value which corresponds to the ALP(I), BET(I) and E(I) arrays for each link. This value is a function of the capacity of the link and the free flow speed.

T(I) is a zero-one variable which controls the interaction between cars and trucks. If there is an interaction then both car and truck links have a value of one. If there is no interaction then the truck link takes the value of zero. T(I) appears in the cost functions.

LINK(I) is the array of the numbers assigned to each link as explained earlier.

RL(I) is an array which takes the value of zero, one and a very high value, assigning no cost, the travel cost or a very high cost respectively to each link as desired.

READ TOD(I), AMT(I)

where

TOD(I) is the array of the destinations of the O-D matrix AMT(I) is the demand for each O-D pair.

READ ODLK(I)

where ODLK(I) is the forward star array of the origins of the O-D matrix.

READ FS(I)

where FS(I) is the forward star array of the start node of the links as given in the example earlier.

The input data of network 2 are listed hereafter.

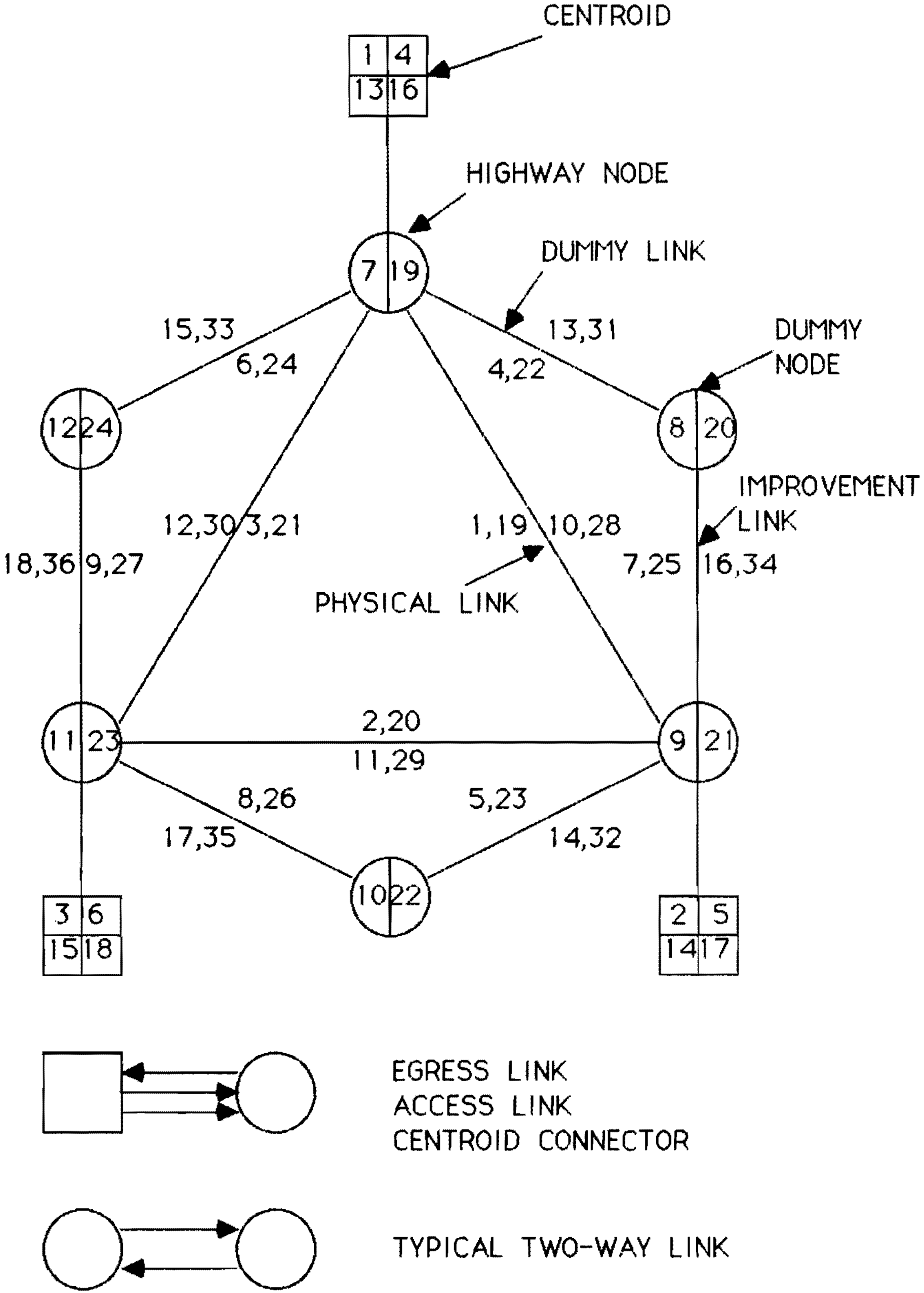
CENTROID
1
4
13
16
HIGHWAY NODE
7
19
DUMMY LINK
15,33
6,24
13,31
4,22
DUMMY NODE
12
24
8
20
IMPROVEMENT LINK
12,30
3,21
1,19
10,28
18,36
9,27
7,25
16,34
PHYSICAL LINK
11
23
2,20
11,29
9
21
8,26
5,23
17,35
14,32
10
22
3
6
15
18
2
5
14
17
EGRESS LINK
ACCESS LINK
CENTROID CONNECTOR
TYPICAL TWO-WAY LINK

Fig. B-1

APPENDIX B.2

DATABASE DEVELOPMENT FOR NETWORK 3 (THE TEXAS NETWORK)

A major task of this study was to test the diagonalization algorithm on a large scale network, in view of its applicability to analyze truck-related improvements to the Texas highway network. The development of the database needed to perform the traffic assignment using the diagonalization algorithm, as modified for two classes of users, is described below.

The first step in the development of the database was to represent the Texas highway system in network form. The road segments are represented with links whereas the intersection of road segments are represented with nodes. The sources of attractions (destinations) and productions (origins) are represented with centroid nodes, which serve both as origins and destinations. The link characteristics were identified and coded, and an O-D (origin-destination) matrix was generated for each class to test the algorithm.

Since the test was concerned with trucks, a separate class of users, in addition to passenger cars, the network developed had to include those links where truck traffic would be considered significant either presently or in the future. Thus as a first basis, links were included in the network if they experienced truck traffic in excess of 200 maximum hourly flow. This information was obtained from the HPMS (Highway Performance Monitoring System) data files for the year 1982. Additional information on the methodology for obtaining these data can be found in Mahmassani et. al. (1985). The following information could also be identified from the HPMS data file: speed limit, number of lanes, route number, route classification (freeway, rural arterial, urban arterial) and the HPMS highway section. In addition, several other links were included in the network, either to ensure connectivity or because they were judged to be potentially significant for future route shifting.

Having identified the appropriate highway sections to be included, the next step involved specifying the link in the corresponding network representation. It is desirable for such a link to comprise sections exhibiting the same basic characteristics, mainly: speed limit, number of lanes, and route classification. In most cases, a link was defined as the highway segment between two nodes; a node was defined at the intersection between two highways or at the point where the characteristics of a given highway segment were changing. It should be noted that some of the links ultimately specified are not homogenous throughout their length, meaning that the characteristics mentioned above are not the same for all highway sections comprising the link. In those cases, the most critical sections governed the characteristics of the link. The primary reason for not breaking the link into several component links was concern for the size of the network and the interest to simplify non-essential detail. Having specified the links and nodes of the network, the length of each link was measured. It should be noted here that the HPMS links are not the same with the ones developed. A link of the developed network may include several HPMS links or it may be part of an HPMS link. The length was measured using a set of detailed highway maps which included the section identification number.

The resulting network was completed by including the dummy nodes and links and the auxiliary links necessary for the representation of link improvements in terms of capacity expansion (lane addition) and/or lane access restrictions to either of the two vehicle classes. A more detailed description of the representation of the link improvements was given earlier.

The last step involved the development of the O-D matrix for each of the two classes under consideration, the passenger car O-D matrix and the truck O-D matrix. A total of 14 centroids of trip attractions and productions, for both user classes, were defined, corresponding to the major urban areas in Texas. It can be noted that this matrix provides a rather coarse aggregation that may not accurately convey the detailed spatial patterns of productions and attractions that generate Texas highway traffic. Therefore, it is not intended in its current form for accurate forecasting in the context of policy planning decisions. Future studies can develop more accurate and detailed O-D matrices for the two major vehicle classes of vehicles in the traffic stream, and the diagonalization algorithm can be used to distribute the flows, permitting the comparison of the resulting flows with observed flows. At this stage, the developed O-D matrices were intended to test the diagonalization algorithm on a large network.

APPENDIX C

SAMPLE LISTINGS OF INPUT DATA, THE COMPUTER PROGRAMS AND THE OUTPUT

APPENDIX C-1

INPUT DATA FOR NETWORK 2

```
 6   .005     .005       .0005
 336  14  65 129 364   12
.000 1.10 .000 4.0
.731 3.66 .157 4.0
.613 3.5  .136 4.0
.877 4.46 .161 4.0
.69  5.16 .111 4.0
1.15 4.42 .212 4.0
.62  3.65 .133 4.0
.67  4.94 .112 4.0
.62  5.14 .101 4.0
1.03 5.52 .158 4.0
.66  5.09 .109 4.0
.54  5.79 .080 4.0
1.0  6.59 .133 4.0
.88  4.93 .148 4.0
.77  5.34 .121 4.0
1.15 6.87 .146 4.0
1.20 7.00 .140 4.0
  1
  2
  3
  4
  5
  6
  7
  8
 30
 60
 80
 99
  29 1.010000.25.  1 1   0      1.E+00
  33 1.010000.25.  1 1   0      1.E+00
  35 1.010000.25.  1 1   0      1.E+00
  39 1.010000.25.  1 1   0      1.E+00
  41 1.010000.25.  1 1   0      1.E+00
  37 1.010000.25.  1 1   0      1.E+00
  45 1.010000.25.  1 1   0      1.E+00
  47 1.010000.25.  1 1   0      1.E+00
  51 1.010000.25.  1 1   0      1.E+00
  49 1.010000.25.  1 1   0      1.E+00
  55 1.010000.25.  1 1   0      1.E+00
  57 1.010000.25.  1 1   0      1.E+00
  59 1.010000.25.  1 1   0      1.E+00
  61 1.010000.25.  1 1   0      1.E+00
  29 1.010000.25.  1 1   0      1.E+00
  33 1.010000.25.  1 1   0      1.E+00
  35 1.010000.25.  1 1   0      1.E+00
  39 1.010000.25.  1 1   0      1.E+00
  41 1.010000.25.  1 1   0      1.E+00
  37 1.010000.25.  1 1   0      1.E+00
  45 1.010000.25.  1 1   0      1.E+00
  47 1.010000.25.  1 1   0      1.E+00
  51 1.010000.25.  1 1   0      1.E+00
  49 1.010000.25.  1 1   0      1.E+00
  55 1.010000.25.  1 1   0      1.E+00
```

```
57 1.010000.25.  1 1   0    1.E+00
59 1.010000.25.  1 1   0    1.E+00
61 1.010000.25.  1 1   0    1.E+00
 15 1.010000.25.  1 1   0    1.E+00
30 1.010000.25.  1 1  22    1.E-20
31170. 4000.55.14 1   1    1.E+00
29 1.010000.25.  1 1 148    1.E-20
31170. 2000.55.16 1  43    1.E+00
30170. 2000.55.16 1 169    1.E+00
29170. 4000.55.14 1 127    1.E+00
32 1.010000.25.  1 1  23    1.E-20
33115. 4000.55.14 1   2    1.E+00
59390. 4000.55.14 1  21    1.E+00
64390. 2000.55.16 1  63    1.E+00
31 1.010000.25.  1 1 149    1.E-20
33115. 2000.55.16 1  44    1.E+00
32115. 2000.55.16 1 170    1.E+00
31115. 4000.55.14 1 128    1.E+00
 16 1.010000.25.  1 1   0    1.E+00
34 1.010000.25.  1 1  24    1.E-20
35145. 4000.55.14 1   3    1.E+00
36 1.010000.25.  1 1  25    1.E-20
37350. 4000.55.14 1   4    1.E+00
33 1.010000.25.  1 1 150    1.E-20
35145. 2000.55.16 1  45    1.E+00
33145. 4000.55.14 1 129    1.E+00
34145. 2000.55.16 1 171    1.E+00
 17 1.010000.25.  1 1   0    1.E+00
38 1.010000.25.  1 1  26    1.E-20
39120. 4000.55.14 1   5    1.E+00
33 1.010000.25.  1 1 151    1.E-20
37350. 2000.55.16 1  46    1.E+00
33350. 4000.55.14 1 130    1.E+00
36350. 2000.55.16 1 172    1.E+00
39365. 4000.55.14 1   6    1.E+00
40365. 2000.55.16 1  48    1.E+00
41 50. 4000.55.14 1   7    1.E+00
42 1.010000.25.  1 1  28    1.E-20
20 1.010000.25.  1 1   0    1.E+00
44 1.010000.25.  1 1  29    1.E-20
45185. 4000.55.14 1   8    1.E+00
43 1.010000.25.  1 1  31    1.E-20
47150. 4000.55.14 1  10    1.E+00
52 1.010000.25.  1 1  33    1.E-20
49240. 4000.55.14 1  12    1.E+00
63 1.010000.25.  1 1  41    1.E-20
61200. 4000.55.14 1  20    1.E+00
35 1.010000.25.  1 1 152    1.E-20
39120. 2000.55.16 1  47    1.E+00
35120. 4000.55.14 1 131    1.E+00
38120. 2000.55.16 1 173    1.E+00
 18 1.010000.25.  1 1   0    1.E+00
40 1.010000.25.  1 1  27    1.E-20
37365. 4000.55.14 1 132    1.E+00
39 1.010000.25.  1 1 153    1.E-20
37365. 6000.55.16 1 174    1.E+00
 19 1.010000.25.  1 1   0    1.E+00
```

```
42 50. 2000.55.16 1  49     1.E+00
37 50. 4000.55.14 1 133     1.E+00
41 50. 2000.55.16 1 175     1.E+00
37 1.010000.25. 1 1 154     1.E-20
37 1.010000.25. 1 1 157     1.E-20
47150. 2000.55.16 1  52     1.E+00
37 1.010000.25. 1 1 155     1.E-20
45185. 2000.55.16 1  50     1.E+00
37185. 4000.55.14 1 134     1.E+00
44185. 2000.55.16 1 176     1.E+00
21 1.010000.25. 1 1   0     1.E+00
46 1.010000.25. 1 1  30     1.E-20
47 70. 4000.55.14 1   9     1.E+00
45 1.010000.25. 1 1 156     1.E-20
47 70. 2000.55.16 1  51     1.E+00
46 70. 2000.55.16 1 177     1.E+00
45 70. 4000.55.14 1 135     1.E+00
37150. 4000.55.14 1 136     1.E+00
43150. 2000.55.16 1 178     1.E+00
22 1.010000.25. 1 1   0     1.E+00
48 1.010000.25. 1 1  32     1.E-20
49225. 4000.55.14 1  11     1.E+00
47 1.010000.25. 1 1 158     1.E-20
49225. 2000.55.16 1  53     1.E+00
48225. 2000.55.16 1 179     1.E+00
47225. 4000.55.14 1 137     1.E+00
52240. 2000.55.16 1  54     1.E+00
37240. 4000.55.14 1 138     1.E+00
53 1.010000.25. 1 1  37     1.E-20
61160. 4000.55.14 1  16     1.E+00
62 1.010000.25. 1 1  36     1.E-20
59200. 4000.55.14 1  15     1.E+00
55210. 4000.55.14 1  14     1.E+00
54 1.010000.25. 1 1  35     1.E-20
24 1.010000.25. 1 1   0     1.E+00
51 85. 4000.55.14 1  13     1.E+00
50 1.010000.25. 1 1  34     1.E-20
49 1.010000.25. 1 1 160     1.E-20
51 85. 2000.55.16 1  55     1.E+00
50 85. 2000.55.16 1 181     1.E+00
49 85. 4000.55.14 1 139     1.E+00
23 1.010000.25. 1 1   0     1.E+00
37 1.010000.25. 1 1 159     1.E-20
49240. 2000.55.16 1 180     1.E+00
49 1.010000.25. 1 1 163     1.E-20
61160. 2000.55.16 1  58     1.E+00
49 1.010000.25. 1 1 161     1.E-20
55210. 2000.55.16 1  56     1.E+00
25 1.010000.25. 1 1   0     1.E+00
49210. 4000.55.14 1 140     1.E+00
54210. 2000.55.16 1 182     1.E+00
58 1.010000.25. 1 1  39     1.E-20
59145. 4000.55.14 1  18     1.E+00
56 1.010000.25. 1 1  40     1.E-20
57155. 8000.55.14 1  19     1.E+00
55 1.010000.25. 1 1 166     1.E-20
57155. 2000.55.16 1  61     1.E+00
```

```
 26 1.010000.25.  1 1   0     1.E+00
 56155. 2000.55.16 1 187     1.E+00
 55155. 8000.55.14 1 145     1.E+00
 55 1.010000.25.  1 1 165     1.E-20
 59145. 2000.55.16 1  60     1.E+00
 58145. 2000.55.16 1 186     1.E+00
 55145. 4000.55.14 1 144     1.E+00
 27 1.010000.25.  1 1   0     1.E+00
 64 1.010000.25.  1 1  42     1.E-20
 31390. 4000.55.14 1 147     1.E+00
 60 1.010000.25.  1 1  38     1.E-20
 61 80. 4000.55.14 1  17     1.E+00
 62200. 2000.55.16 1  57     1.E+00
 49200. 4000.55.14 1 141     1.E+00
 59 1.010000.25.  1 1 164     1.E-20
 61 80. 2000.55.16 1  59     1.E+00
 59 80. 4000.55.14 1 143     1.E+00
 60 80. 2000.55.16 1 185     1.E+00
 28 1.010000.25.  1 1   0     1.E+00
 37200. 4000.55.14 1 146     1.E+00
 63200. 2000.55.16 1  62     1.E+00
 53160. 2000.55.16 1 184     1.E+00
 49160. 4000.55.14 1 142     1.E+00
 49 1.010000.25.  1 1 162     1.E-20
 59200. 2000.55.16 1 183     1.E+00
 37 1.010000.25.  1 1 167     1.E-20
 61200. 2000.55.16 1 188     1.E+00
 31390. 2000.55.16 1 189     1.E+00
 59 1.010000.25.  1 1 168     1.E-20
 93 1.010000.25.  1 1   0     1.E+00
 97 1.010000.25.  1 1   0     1.E+00
 99 1.010000.25.  1 1   0     1.E+00
103 1.010000.25.  1 1   0     1.E+00
105 1.010000.25.  1 1   0     1.E+00
101 1.010000.25.  1 1   0     1.E+00
109 1.010000.25.  1 1   0     1.E+00
111 1.010000.25.  1 1   0     1.E+00
115 1.010000.25.  1 1   0     1.E+00
113 1.010000.25.  1 1   0     1.E+00
119 1.010000.25.  1 1   0     1.E+00
121 1.010000.25.  1 1   0     1.E+00
123 1.010000.25.  1 1   0     1.E+00
125 1.010000.25.  1 1   0     1.E+00
 93 1.010000.25.  1 1   0     1.E+00
 97 1.010000.25.  1 1   0     1.E+00
 99 1.010000.25.  1 1   0     1.E+00
103 1.010000.25.  1 1   0     1.E+00
105 1.010000.25.  1 1   0     1.E+00
101 1.010000.25.  1 1   0     1.E+00
109 1.010000.25.  1 1   0     1.E+00
111 1.010000.25.  1 1   0     1.E+00
115 1.010000.25.  1 1   0     1.E+00
113 1.010000.25.  1 1   0     1.E+00
119 1.010000.25.  1 1   0     1.E+00
121 1.010000.25.  1 1   0     1.E+00
123 1.010000.25.  1 1   0     1.E+00
125 1.010000.25.  1 1   0     1.E+00
```

```
 79 1.010000.25.  1 1   0     1.E+00
 94 1.010000.25.  1 0  85     1.E-20
 95170.  4000.55.15 1  64     1.E+00
 93 1.010000.25.  1 0 211     1.E-20
 95170.  1000.55.17 0 106     1.E+00
 94170.  1000.55.17 0 232     1.E+00
 93170.  4000.55.15 1 190     1.E+00
 96 1.010000.25.  1 0  86     1.E-20
 97115.  4000.55.15 1  65     1.E+00
123390.  4000.55.15 1  84     1.E+00
128390.  1000.55.17 0 126     1.E+00
 95 1.010000.25.  1 0 212     1.E-20
 97115.  1000.55.17 0 107     1.E+00
 96115.  1000.55.17 0 233     1.E+00
 95115.  4000.55.15 1 191     1.E+00
 80 1.010000.25.  1 0   0     1.E+00
 98 1.010000.25.  1 0  87     1.E-20
 99145.  4000.55.15 1  66     1.E+00
100 1.010000.25.  1 0  88     1.E-20
101350.  4000.55.15 1  67     1.E+00
 97 1.010000.25.  1 0 213     1.E-20
 99145.  1000.55.17 0 108     1.E+00
 97145.  4000.55.15 1 192     1.E+00
 98145.  1000.55.17 0 234     1.E+00
 81 1.010000.25.  1 0   0     1.E+00
102 1.010000.25.  1 0  89     1.E-20
103120.  4000.55.15 1  68     1.E+00
 97 1.010000.25.  1 0 214     1.E-20
101350.  1000.55.17 0 109     1.E+00
 97350.  4000.55.15 1 193     1.E+00
100350.  1000.55.17 0 235     1.E+00
103365.  4000.55.15 1  69     1.E+00
104365.  1000.55.17 0 111     1.E+00
105 50.  4000.55.15 1  70     1.E+00
106 1.010000.25.  1 0  91     1.E-20
 84 1.010000.25.  1 0   0     1.E+00
108 1.010000.25.  1 0  92     1.E-20
109185.  4000.55.15 1  71     1.E+00
107 1.010000.25.  1 0  94     1.E-20
111150.  4000.55.15 1  73     1.E+00
116 1.010000.25.  1 0  96     1.E-20
113240.  4000.55.15 1  75     1.E+00
127 1.010000.25.  1 0 104     1.E-20
125200.  4000.55.15 1  83     1.E+00
 99 1.010000.25.  1 0 215     1.E-20
103120.  1000.55.17 0 110     1.E+00
 99120.  4000.55.15 1 194     1.E+00
102120.  1000.55.17 0 236     1.E+00
 82 1.010000.25.  1 0   0     1.E+00
104 1.010000.25.  1 0  90     1.E-20
101365.  4000.55.15 1 195     1.E+00
103 1.010000.25.  1 0 216     1.E-20
101365.  1000.55.17 0 237     1.E+00
 83 1.010000.25.  1 0   0     1.E+00
106 50.  1000.55.17 0 112     1.E+00
101 50.  4000.55.15 1 196     1.E+00
105 50.  1000.55.17 0 238     1.E+00
```

```
101 1.010000.25.  1 0 217     1.E-20
101 1.010000.25.  1 0 220     1.E-20
111150.  1000.55.17 0 115     1.E+00
101 1.010000.25.  1 0 218     1.E-20
109185.  1000.55.17 0 113     1.E+00
101185.  4000.55.15 1 197     1.E+00
108185.  1000.55.17 0 239     1.E+00
 85 1.010000.25.  1 0   0     1.E+00
110 1.010000.25.  1 0  93     1.E-20
111 70.  4000.55.15 1  72     1.E+00
109 1.010000.25.  1 0 219     1.E-20
111 70.  1000.55.17 0 114     1.E+00
110 70.  1000.55.17 0 240     1.E+00
109 70.  4000.55.15 1 198     1.E+00
101150.  4000.55.15 1 199     1.E+00
107150.  1000.55.17 0 241     1.E+00
 86 1.010000.25.  1 0   0     1.E+00
112 1.010000.25.  1 0  95     1.E-20
113225.  4000.55.15 1  74     1.E+00
111 1.010000.25.  1 0 221     1.E-20
113225.  1000.55.17 0 116     1.E+00
112225.  1000.55.17 0 242     1.E+00
111225.  4000.55.15 1 200     1.E+00
116240.  1000.55.17 0 117     1.E+00
101240.  4000.55.15 1 201     1.E+00
117 1.010000.25.  1 0 100     1.E-20
125160.  4000.55.15 1  79     1.E+00
126 1.010000.25.  1 0  99     1.E-20
123200.  4000.55.15 1  78     1.E+00
119210.  4000.55.15 1  77     1.E+00
118 1.010000.25.  1 0  98     1.E-20
 88 1.010000.25.  1 0   0     1.E+00
115 85.  4000.55.15 1  76     1.E+00
114 1.010000.25.  1 0  97     1.E-20
113 1.010000.25.  1 0 223     1.E-20
115 85.  1000.55.17 0 118     1.E+00
114 85.  1000.55.17 0 244     1.E+00
113 85.  1000.55.15 1 202     1.E+00
 87 1.010000.25.  1 0   0     1.E+00
101 1.010000.25.  1 0 222     1.E-20
113240.  1000.55.17 0 243     1.E+00
113 1.010000.25.  1 0 226     1.E-20
125160.  1000.55.17 0 121     1.E+00
113 1.010000.25.  1 0 224     1.E-20
119210.  1000.55.17 0 119     1.E+00
 89 1.010000.25.  1 0   0     1.E+00
113210.  4000.55.15 1 203     1.E+00
118210.  1000.55.17 0 245     1.E+00
122 1.010000.25.  1 0 102     1.E-20
123145.  4000.55.15 1  81     1.E+00
120 1.010000.25.  1 0 103     1.E-20
121155.  8000.55.15 1  82     1.E+00
119 1.010000.25.  1 0 229     1.E-20
121155.  1000.55.17 0 124     1.E+00
 90 1.010000.25.  1 0   0     1.E+00
120155.  1000.55.17 0 250     1.E+00
119155.  8000.55.15 1 208     1.E+00
```

```
119 1.010000.25.  1 0 228     1.E-20
123145.  1000.55.17 0 123     1.E+00
122145.  1000.55.17 0 249     1.E+00
119145.  4000.55.15 1 207     1.E+00
 91 1.010000.25.  1 0   0     1.E+00
128 1.010000.25.  1 0 105     1.E-20
 95390.  4000.55.15 1 221     1.E+00
124 1.010000.25.  1 0 101     1.E-20
125 80.  4000.55.15 1  80     1.E+00
126200.  1000.55.17 0 120     1.E+00
113200.  4000.55.15 1 204     1.E+00
123 1.010000.25.  1 0 227     1.E-20
125 80.  1000.55.17 0 122     1.E+00
123 80.  4000.55.15 1 206     1.E+00
124 80.  1000.55.17 0 248     1.E+00
 92 1.010000.25.  1 0   0     1.E+00
101200.  4000.55.15 1 209     1.E+00
127200.  1000.55.17 0 125     1.E+00
117160.  1000.55.17 0 247     1.E+00
113160.  4000.55.15 1 205     1.E+00
113 1.010000.25.  1 0 225     1.E-20
123200.  1000.55.17 0 246     1.E+00
101 1.010000.25.  1 0 230     1.E-20
125200.  1000.55.17 0 251     1.E+00
 95390.  1000.55.17 0 252     1.E+00
123 1.010000.25.  1 0 231     1.E-20
16625.
17351.
18190.
19 37.
201508.
21 28.
22 28.
23 28.
241500.
25115.
26 76.
27625.
28281.
15625.
17781.
18347.
19 63.
202302.
21 40.
22 40.
23 20.
24838.
25105.
26 64.
27500.
28190.
15351.
16781.
18781.
19 40.
201207.
```

```
21 20.
22 28.
23 16.
24615.
25 49.
26 35.
27301.
28156.
15190.
16347.
17781.
19 63.
202302.
21 40.
22 40.
23 20.
24838.
25 49.
26 35.
27301.
28115.
15 37.
16 63.
17 40.
18 63.
209600.
21 80.
22 80.
23 20.
243000.
25 40.
26 48.
27400.
28280.
151508.
162302.
171207.
182302.
199600.
212414.
22170.
231072.
246000.
251900.
261200.
273000.
283000.
15 28.
16 40.
17 20.
18 40.
19 80.
202414.
22320.
23 20.
24800.
25 40.
```

```
26 20.
27160.
28100.
15 28.
16 40.
17 28.
18 40.
19 80.
20170.
21320.
23 35.
241000.
25 60.
26 28.
27250.
28150.
15 28.
16 20.
17 16.
18 20.
19 20.
201072.
21 201.
22 35.
248000.
25100.
26 50.
27400.
28280.
151500.
16838.
17615.
18838.
193000.
206000.
21800.
221000.
238000.
251000.
261000.
273000.
286000.
15115.
16105.
17 49.
18 49.
19 40.
201900.
21 40.
22 60.
23100.
241000.
26900.
277000.
281400.
15 76.
16 64.
```

```
 17 35.
 18 35.
 19 28.
201200.
21 20.
22 28.
23 50.
241000.
25900.
271200.
28350.
 15625.
 16500.
 17301.
 18301.
 19400.
203000.
21160.
22250.
23400.
243000.
257000.
261200.
288000.
 15281.
 16190.
 17156.
 18115.
 19280.
203000.
21100.
22 50.
23280.
246000.
251400.
26350.
273000.
800.
810.
820.
830.
840.
851.
860.
870.
880.
890.
900.
911.
923.
790.
810.
830.
830.
840.
850.
86246.
```

```
870.
880.
890.
900.
910.
920.
790.
810.
82198.
830.
840.
850.
8642.
870.
880.
890.
900.
910.
920.
790.
810.
81198.
8387.
84221.
850.
8674.
8725.
88110.
890.
9032.
914.
9210.
790.
810.
810.
8287.
840.
850.
860.
870.
880.
890.
900.
910.
920.
790.
800.
810.
82221.
830.
85329.
8693.
87109.
88340.
89472.
900.
9126.
```

```
9277.
791.
800.
810.
820.
830.
84329.
860.
870.
88269.
8960.
9071.
9168.
9215.
795.
80246.
8142.
8274.
830.
8493.
850.
87136.
88316.
890.
9060.
910.
925.
790.
800.
810.
8225.
830.
84109.
850.
86136.
882599.
890.
900.
9123.
9252.
790.
800.
810.
82110.
830.
84340.
85269.
86316.
872599.
8929.
900.
9166.
92165.
790.
800.
810.
820.
```

```
830.
84472.
8560.
860.
870.
8829.
900.
9169.
920.
790.
800.
810.
8232.
830.
840.
8571.
8660.
870.
880.
890.
9128.
920.
791.
800.
810.
824.
830.
8426.
8568.
860.
8723.
8866.
8969.
9028.
92271.
793.
800.
810.
8210.
830.
8477.
8515.
865.
8752.
88165.
890.
900.
91271.
   1
  14
  27
  40
  53
  66
  79
  92
 105
```

```
118
131
144
157
170
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
183
196
```

209
222
235
248
261
274
287
300
313
326
339
352
365
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
32
34
40
42
49
51
56
58
73
75
80
82
85
87
89

91
96
98
105
107
120
122
125
127
129
131
138
140
143
145
154
156
163
165
167
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
200
202
208
210
217
219
224
226

241
243
248
250
253
255
257
259
264
266
273
275
288
290
293
295
297
299
306
308
311
313
322
324
331
333
335
337

APPENDIX C-2

THE USER EQUILIBRIUM COMPUTER PROGRAM

```
      PROGRAM UETRDIA(INPUT,OUTPUT,TTY,TAPE1=INPUT,TAPE4=OUTPUT,
     $TAPE5=TTY)
      COMMON /ENA/ NFLI,INTI,EPS,CCINTI,ACCBIS
      COMMON/COM4/VALUEUP,NARC,LINK,AADT,VCAT,TRUP
      COMMON /ARCDT/ TOO,L,C,VEL,FL,COST,T,RL
      COMMON /ODDT/ TOD,AMT
      COMMON /FST/ FS
      COMMON/ODL/ ODLK
      COMMON /DMPDT/ NDMP,DMP
      COMMON /ALBET/ ALP,BET,ALP1,TYP,E
      REAL L(4000),C(4000),VEL(4000),FL(4000),COST(4000),AMT(500),
     #TITLE(20),ALP(17),BET(17),ALP1(17),E(17),RL(4000),
     $TRUP(4000),AADT(4000),VCAT(4000),EPS,CCINTI,ACCBIS,NFLI(4000)
       INTEGER TOO(4000),TOD(4000),FS(2002),ODLK(801),DMP(50),TYP(4000),
     $T(4000),LINK(4000)
      READ(1,106)INTI,EPS,CCINTI,ACCBIS
  106 FORMAT(I2,3F10.5)
      READ(1,108) NARC,NCENT,NNODA,NNODT,NOD,NDMP
  108 FORMAT(6I4)
      DO 26 I=1,17
      READ(1,109) ALP(I),BET(I),ALP1(I),E(I)
  109 FORMAT(4F5.3)
   26 CONTINUE
      WRITE(5,*)ALP(1),BET(1),E(1)
      WRITE(4,1)
    1 FORMAT(T20,'USER EQUILIBRIUM ASSIGNMENT - PSU')
      DO 20 I=1,NDMP
      READ(1,111) DMP(I)
  111 FORMAT(I2)
   20 CONTINUE
      DO 21 I=1,NARC
      READ(1,100) TOO(I),L(I),C(I),VEL(I),TYP(I),T(I),LINK(I),RL(I)
  100 FORMAT(I4,F5.1,F6.1,F3.0,2I2,I4,E7.0)
   21 CONTINUE
      WRITE(5,*)TOO(NARC),L(NARC),C(NARC),VEL(NARC),TYP(NARC),T(NARC),LINK(NARC)
,AL(NARC)
      WRITE(4,7) NOD
    7 FORMAT( '     # OF OD PAIRS =',I6)
      IF (NOD.EQ.0) GO TO 10
      DO 22 I=1,NOD
      READ(1,101) TOO(I),AMT(I)
  101 FORMAT(I4,F8.0)
   22 CONTINUE
      NCNT1=NCENT+NNODA
      DO 23 I=1,NCNT1
      READ(1,102) ODLK(I)
  102 FORMAT(I4)
   23 CONTINUE
      DO 24 I=1,NNODT
      READ(1,102) FS(I)
   24 CONTINUE
      WRITE(5,*)NNODT,NCNT1
      WRITE(5,*)ODLK(NCNT1)
   10 WRITE(4,9) EPS,CCINTI,ACCBIS,INTI
    9 FORMAT(/'     CONVERGENCE CRITERION =',F7.4/
     $       '     INTERNAL CONVERGENCE  =',F7.4/
     $        '    ACCURACY OF MOVE SIZE =',E18.9/
     $        '     INTERNAL ITERATIONS   =',I4)
      IF (NOD.NE.0) CALL UED(NNODA,NNODT,NOD,NCENT,ITER,ITERIN)
      STOP
      END
C
      SUBROUTINE UED(NNODA,NNODT,NOD,NCENT,ITER,ITERIN)
      COMMON /ENA/ NFLI,INTI,EPS,CCINTI,ACCBIS
      COMMON /ARCDT/ TOO,L,C,VEL,FL,COST,T,AL
      COMMON /ODDT/ TOD,AMT
      COMMON /DMPDT/ NDMP,DMP
      COMMON /FST/ FS
      COMMON /ODL/ ODLK
      COMMON /ALBET/ ALP,BET,ALP1,TYP,E
      COMMON/COM4/VALUEUP,NARC,LINK,AADT,VCRT,TRUP
      REAL L(4000),C(4000),VEL(4000),FL(4000),COST(4000),AMT(500)
     *,NFL(4000),ALP(17),BET(17),ALP1(17),E(17)
     $,AL(4000),TRUP(4000),AADT(4000),VCRT(4000),EPS,NFLI(4000),CCINTI,ACCBIS
      INTEGER TOO(4000),TOD(4000),FS(2002),ODLK(801),DMP(50),TYP(4000),
     $T(4000),LINK(4000),INTI
      DO 30 I=1,NARC
   30 NFLI(I)=0.
      CALL AONUED(FL,NNODA,NNODT,NOD,NCENT)
      DO 29 I=1,NARC
      NFLI(I)=FL(I)
  29  CONTINUE
      FOBJ=0
      CONV=2.*EPS
      WRITE(5,*)40HINITIALIZATION COMPLETED
      ITER=1
      ITOT=0
      ITERIN=1
      J=1
      CALL DUMPUED(ITER,ITERIN,CONV,FOBJ,ITOT,EPS,NNODA,NCENT)
      DO 11 I=ITER,50
      ITER = I
      DO 14 II=1,INTI
      ITERIN=II
      CALL AONUED(NFL,NNODA,NNODT,NOD,NCENT)
      CALL BISUED(NFL)
      CONVI=0.
      DO 13 NI=1,NARC
      XNI=ABS(NFL(NI)-NFLI(NI))
      IF(XNI.EQ.0.)GO TO 13
      DI=NFL(NI)
      IF(DI.EQ.0.)DI=NFLI(NI)
      CONVI=CONVI+XNI/DI
      NFLI(NI)=NFL(NI)
   13 CONTINUE
      CONVI=CONVI/FLOAT(NARC)
      IF(CONVI.LT.CCINTI)GO TO 15
   14 CONTINUE
   15 CONV=0.
      ITOT=ITOT+ITERIN
      FOBJ=0.
      DO 20 N=1,NARC
      XN=ABS(NFL(N)-FL(N))
      IF (XN.EQ.0.) GO TO 20
      D=NFL(N)
      IF (D.EQ.0.) D=FL(N)
      CONV=CONV+XN/D
      FL(N)=NFL(N)
   20 CONTINUE
      CONV=CONV/FLOAT(NARC)
      IF(CONV.LT.EPS) GO TO 12
      IF(ITER.EQ.25) GO TO 12
      IF(DMP(J).EQ.ITER) GO TO 22
      GO TO 11
   22 CALL DUMPUED(ITER,ITERIN,CONV,FOBJ,ITOT,EPS,NNODA,NCENT)
      J=J+1
   11 CONTINUE
   12 CALL DUMPUED(ITER,ITERIN,CONV,FOBJ,ITOT,EPS,NNODA,NCENT)
      RETURN
      END
C
      SUBROUTINE AONUED(NFL,NNODA,NNODT,NOO,NCENT)
      COMMON /ARCDT/ TOO,L,C,VEL,FL,COST,T,AL
      COMMON /ENA/ NFLI,INTI,EPS,CCINTI,ACCBIS

      COMMON /OODT/ TOO,AMT
      COMMON /FST/ FS
      COMMON /ODL/ ODLK
      COMMON /ALBET/ ALP,BET,ALP1,TYP,E
      COMMON/COM4/VALUEUP,NARC,LINK,AADT,VCRT,TRUP
      REAL L(4000),C(4000),VEL(4000),FL(4000),COST(4000),AMT(500)
     *,NFL(4000),SP(2001),ALP(17),BET(17),ALP1(17),E(17)
     $,AL(4000),AADT(4000),VCRT(4000),TRUP(4000),EPS,NFLI(4000),CCINTI,ACCBIS
      INTEGER TOO(4000),TOD(4000),FS(2002),ODLK(801),PRED(2001),
     $TYP(4000),T(4000),LINK(4000),INTI
      WRITE(5,*)40HENTER A-O-N
      DO 10 N=1,NARC
      A1=ALP(TYP(N))
      B1=BET(TYP(N))
      E1=E(TYP(N))
      IF(N.LE.NARC/2) THEN
                  K=NARC/2+N
                   NFL(N)=0
                   NFL(K)=0
                   C1=C(N)
                   C2=C(K)
                   FLA=NFLI(N)
                   FLT=FL(K)
                   T1=T(N)
                   T2=T(K)
      ELSE
                   K=N-NARC/2
                   NFL(K)=0
                   NFL(N)=0
                   C1=C(K)
                   C2=C(N)
                   FLA=FL(K)
                   FLT=NFLI(N)
                   T2=T(K)
                   T1=T(N)
      ENDIF
   10 COST(N)=TRCOST(L(N),C1,C2,VEL(N),FLA,FLT,A1,B1,E1,T1,T2,RL(N))
      II=1
      NNA=1
      KK=NCENT
      NNT=NNODA
      GO TO 2
    3 II=NNODA
      NNA=NNODA
      KK=NNODA+NCENT-1
      NNT=NNODT
    2 DO 20 I=II,KK
      I1=ODLK(I)
      I2=ODLK(I+1)-1
      IF (I1.GT.I2) GO TO 20
      CALL SHPVED(I,PRED,SP,NNA,NNT)
      DO 30 K=I1,I2
      J=TOD(K)
   60 J1=PRED(J)
      IF (J1.EQ.0) GO TO 30
      N1=FS(J1)
      N2=FS(J1+1)-1
      DO 40 N=N1,N2
      IF (TOO(N).EQ.J) GO TO 50
   40 CONTINUE
   50 NFL(N)=NFL(N)+AMT(K)
      J=J1
      GO TO 60
   30 CONTINUE
   20 CONTINUE
      IF(II.EQ.1) GO TO 3
      WRITE(5,*)40HLEAVE A-O-N
      RETURN
      END
C
      SUBROUTINE BISUED(NFL)
      COMMON /ENA/ NFLI,INTI,EPS,CCINTI,ACCBIS
      COMMON/COM4/VALUEUP,NARC,LINK,AADT,VCAT,TRUP
      COMMON /ARCDT/ TOO,L,C,VEL,FL,COST,T,RL

      COMMON /FST/ FS
      COMMON /ALBET/ ALP,BET,ALP1,TYP,E
      REAL L(4000),C(4000),VEL(4000),FL(4000),COST(4000),NFL(4000),
     $ALP(17),BET(17),ALP1(17),E(17),RL(4000),
     $TRUP(4000),AADT(4000),VCAT(4000),EPS,NFLI(4000),CCINTI,ACCBIS
      INTEGER TOO(4000),FS(2002),TYP(4000),T(4000),LINK(4000),INTI
      WRITE(5,*)40HENTER BISUED
      AMN=0
      AMX=1
   10 AMD=(AMX+AMN)/2
      IF ((AMX-AMN).LE.ACCBIS) GO TO 20
      D=0
      DO 30 N=1,NARC/2
      X1=NFLI(N)+(NFL(N)-NFLI(N))*AMD
      A1=ALP(TYP(N))
      B1=BET(TYP(N))
      E1=E(TYP(N))
      K=N+NARC/2
      C1=C(N)
      C2=C(K)
      FLA=FL(N)
      FLT=FL(K)
      T1=T(N)
      T2=T(K)
      CST1=TRCOST(L(N),C1,C2,VEL(N),X1,FLT,A1,B1,E1,T1,T2,RL(N))
      X2=NFLI(K)+AMD*(NFL(K)-NFLI(K))
      A2=ALP(TYP(K))
      B2=BET(TYP(K))
      E2=E(TYP(K))
      CST2=TRCOST(L(K),C1,C2,VEL(K),FLA,X2,A2,B2,E2,T2,T1,RL(K))
   30 D=D+CST1*(NFL(N)-NFLI(N))+CST2*(NFL(K)-NFLI(K))
      IF (D.GT.0.) AMX=AMD
      IF (D.LE.0.) AMN=AMD
      GO TO 10
   20 DO 40 N=1,NARC
   40 NFL(N)=NFLI(N)+AMD*(NFL(N)-NFLI(N))
      WRITE(5,*)40HLEAVE BISUED
      RETURN
      END
C
      FUNCTION TRCOST(D,C1,C2,VEL,FLA,FLT,A,B,E,T1,T2,RL)
      TRCOST=D*RL/VEL
      IF (C1.NE.0.AND.C2.NE.0.)
     $TRCOST=TRCOST*(1.+A*(FLA*T1/C1+E*FLT*T2/C2)**B)
      RETURN
      END
C
      SUBROUTINE SHPUED(R,PRED,SP,NNA,NNT)
C
C  THIS SUBROUTINE COMPUTES SHORTEST PATHS FROM R TO ALL OTHER NODES.
C  PRED(I) CONTAINS PREDECESSOR OF NODE I, SP(I) CONTAINS LENGTH OF
C  PATH TO NODE I.
C
      COMMON /ARCDT/ TOO,L,C,VEL,FL,COST,T,RL
      COMMON /FST/ FS
      COMMON/COM4/VALUEUP,NARC,LINK,AADT,VCAT,TRUP

      REAL L(4000),C(4000),VEL(4000),FL(4000),COST(4000),SP(2001)
     $,RL(4000),TRUP(4000),AADT(4000),VCAT(4000)
      INTEGER,TOO(4000),FS(2002),PRED(2001),CL(2001),T(4000),LINK(4000),
     $R
      WRITE(5,*)40HENTER SHPUED
      DO 10 I=NNA,NNT
      SP(I)=1.E20
      PRED(I)=0
   10 CL(I)=0
      SP(R)=0
      CL(R)=NNT+1
      I=R
      NT=R
   20 IA=FS(I+1)-1
      S=SP(I)
      IA1=FS(I)
      IF (IA1.GT.IA) GO TO 30
      DO 40 IR=IA1,IA
      K=TOO(IR)
      SD=S+COST(IR)
       IF (SD.GE.SP(K)) GO TO 40
      PRED(K)=I
      SP(K)=SD
       IF (CL(K)) 50,60,40
   60 CL(NT)=K
      NT=K
      CL(K)=NNT+1
      GO TO 40
   50 CL(K)=CL(I)
      CL(I)=K
   40 CONTINUE
   30 ICL=CL(I)
      CL(I)=-1
      I=ICL
       IF (I.LE.NNT) GO TO 20
      WRITE(5,*)40HLEAVE SHPVED
      RETURN
      END
C
      SUBROUTINE DUMPVED(ITER,ITERIN,CONV,FOBJ,ITOT,EPS,NNODA,NCENT)
      COMMON /ARCDT/ TOO,L,C,VEL,FL,COST,T,AL
      COMMON /FST/ FS
      COMMON /ALBET/ ALP,BET,ALP1,TYP,E
      COMMON/COM4/VALUEUP,NARC,LINK,AADT,VCRT,TRUP
      REAL L(4000),C(4000),VEL(4000),FL(4000),COST(4000),MFL(4000)
     *,ALP(17),BET(17),ALP1(17),VCRT(4000),E(17),TRUP(4000),
     $AADT(4000),AL(4000)
       INTEGER TOO(4000),FS(2002),TYP(4000),VC1(4000),VC2(4000),T(4000),
     $LINK(4000),LLINK(4000)
       WRITE(5,*)40HENTER DUMP
       WRITE(4,101) ITER,ITERIN,ITOT
  101 FORMAT(//'NO. OF ITERATIONS = 'I6/
     *        'NO. OF INT. ITERATIONS= 'I6/
     $        'TOTAL NO. ITERATIONS= 'I6)
       K=0
       VALUEUP=0
       NNOD1=NNODA-1
       NNODMC=2*NCENT+1
       GO TO 4
    5  NNODMC=NNODMC+NNODA-1
       AUTOTT=VALUEUP
       WRITE(5,*)AUTOTT,VALUEUP
       NNOD1=2*NNOD1
    4  DO 10 I=NNODMC,NNOD1
       J1=FS(I)
       J2=FS(I+1)-1
        IF (J1.GT.J2) GO TO 10
       DO 20 J=J1,J2
        IF (TOO(J).LT.NNODMC) GO TO 20
       K=K+1
       MFL(K)=FL(J)
       LLINK(K)=LINK(J)
       VCRT(K)=FL(J)/C(J)
       VC1(K)=I
      VC2(K)=TOO(J)
      A1=ALP(TYP(J))
      B1=BET(TYP(J))
      E1=E(TYP(J))
      IF (J.LE.NARC/2) THEN
                    KLM=J+NARC/2
                    C1=C(J)
                    C2=C(KLM)
                    FLA=FL(J)
                    FLT=FL(KLM)
                    T1=T(J)
                    T2=T(KLM)
                    VCRT(K)=FLA*T1/C1+E1*FLT*T2/C2
                    AADT(K)=FLA*T1+FLT*T2
                    TRUP(K)=FLT*T2*100./AADT(K)
                     IF(AADT(K).EQ.0.)TRUP(K) =0.
      ELSE
                    KLM=J-NARC/2
                    C1=C(KLM)
                    C2=C(J)
                    FLA=FL(KLM)
                    FLT=FL(J)
                    T2=T(KLM)
                    T1=T(J)
                    VCRT(K)=FLA*T1/C1+E1*FLT*T2/C2
                    AADT(K)=FLA*T1+FLT*T2
                    TRUP(K)=FLT*100./AADT(K)
                     IF(AADT(K).EQ.0.) TRUP(K)=0.
      ENDIF
      CST=TRCOST(L(J),C1,C2,VEL(J),FLA,FLT,A1,B1,E1,T1,T2,AL(J))
      VALUEUP=VALUEUP+CST*FL(J)
   20 CONTINUE
   10 CONTINUE
      WRITE(5,*)K,AADT(K),CST,FL(J2)
       IF (NNOD1.EQ.NNODA-1) GO TO 5
      TRUCKTT=VALUEUP-AUTOTT
      WRITE(4,2) ITER,CONV,VALUEUP,AUTOTT,TRUCKTT
    2 FORMAT(/' AFTER ',I3,' ITERATIONS:'/
     *        '     CONVERGENCE MEASURE       =',F7.4/
     *        '     TOTAL TRAVEL TIME         =',E25.19/
     $        '     AUTO  TRAVEL TIME         =',E25.19/
     $        '     TRUCK TRAVEL TIME         =',E25.19///)
      WRITE(5,*)40HLEAVE DUMP
       IF(CONV.GT.EPS)RETURN
      RETURN
      WRITE(4,113)
  113 FORMAT(32X,'ORIG',2X,'DEST')
      WRITE(4,112)
  112 FORMAT('LINK NO.',5X,'FLOW',4X,'V/C RATIO',2X,'NODE',2X,
     4'NODE',2X,' # TRUCKS',2X,'AADT',3X,'LINK ID')
      WRITE(4,114)
  114 FORMAT('-------',5X,'----'4X,'---------',2X,'----',2X,'----'
     $,2X,'---------',2X,'-----',3X,'--------')
      DO 23 I=1,K
      WRITE(4,102) I,NFL(I),VCRT(I),VC1(I),VC2(I),TRUP(I),AADT(I)
     $,LLINK(I)
  102 FORMAT(I5,F13.0,F10.0,2I7,2F9.0,I8)
```

```
23 CONTINUE
   RETURN
   END
```

APPENDIX C-3

THE SYSTEM OPTIMUM COMPUTER PROGRAM

```
      PROGRAM SOTRDIA(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      COMMON /ENA/ NFLI,INTI,EPS,CCINTI,ACCBIS
      COMMON/COM4/VALUE1,NARC,LINK,AADT,VCRT
      COMMON /ARCDT/ TOO,L,C,VEL,FL,COST,T,RL
      COMMON /ODDT/ TOD,AMT
      COMMON /FST/ FS
      COMMON/ODL/ ODLK
      COMMON /DMPDT/ NDMP,DMP
      COMMON /ALBET/ ALP,BET,ALP1,TYP,E
      REAL L(4000),C(4000),VEL(4000),FL(4000),COST(4000),AMT(2000),
     *TITLE(20),ALP(17),BET(17),ALP1(17),E(17),RL(4000),
     $TRUP(4000),AADT(4000),VCRT(4000),EPS,CCINTI,ACCBIS,NFLI(4000)
      INTEGER TOO(4000),TOD(2000),FS(2002),ODLK(1501),DMP(50),TYP(4000),
     $T(4000),LINK(4000)
      READ (5,106)INTI,EPS,CCINTI,ACCBIS
  106 FORMAT(I2,3F10.5)
      READ(5,108) NARC,NCENT,NNODA,NNODT,NOD,NDMP
  108 FORMAT(6I4)
      DO 26 I=1,17
      READ(5,109) ALP(I),BET(I),ALP1(I),E(I)
  109 FORMAT(4F5.3)
   26 CONTINUE
      WRITE(6,1)
    1 FORMAT(T20,'SYSTEM OPTIMUM   ASSIGNMENT -UT')
      DO 20 I=1,NDMP
      READ(5,111) DMP(I)
  111 FORMAT(I2)
   20 CONTINUE
      DO 21 I=1,NARC
      READ(5,100) TOO(I),L(I),C(I),VEL(I),TYP(I),T(I),LINK(I),RL(I)
  100 FORMAT(I4,F5.1,F6.1,F3.0,2I2,I4,E7.0)
   21 CONTINUE
      WRITE(6,7) NOD
    7 FORMAT( '     * OF OD PAIRS =',I6)
      IF (NOD.EQ.0) GO TO 10
      DO 22 I=1,NOD
      READ(5,101) TOD(I),AMT(I)
  101 FORMAT(I4,F8.0)
   22 CONTINUE
      NCNT1=NCENT+NNODA
      DO 23 I=1,NCNT1
      READ(5,102) ODLK(I)
  102 FORMAT(I4)
   23 CONTINUE
      DO 24 I=1,NNODT
      READ(5,102) FS(I)
   24 CONTINUE
   10 WRITE(6,9) EPS,CCINTI,ACCBIS,INTI
    9 FORMAT(/'     CONVERGENCE CRITERION =',E25.19/
     $       '     INTERNAL CONVERGENCE  =',E25.19/
     $       '     ACCURACY OF MOVE SIZE =',E25.19/
     $       '     INTERNAL ITERATIONS   =',I4)
      IF (NOD.NE.0) CALL SOD(NNODA,NNODT,NOD,NCENT,ITER)
      STOP
      END
C
```

```
      SUBROUTINE SOD(NNODA,NNODT,NOD,NCENT,ITER)
      COMMON /ENA/ NFLI,INTI,EPS,CCINTI,ACCBIS
      COMMON /ARCDT/ TOO,L,C,VEL,FL,COST,T,RL
      COMMON /ODDT/ TOD,AMT
      COMMON /DMPDT/ NDMP,DMP
      COMMON /FST/ FS
      COMMON /ODL/ ODLK
      COMMON /ALBET/ ALP,BET,ALP1,TYP,E
      COMMON/COM4/VALUE1,NARC,LINK,AADT,VCAT
      REAL L(4000),C(4000),VEL(4000),FL(4000),COST(4000),AMT(2000)
     *,NFL(4000),ALP(17),BET(17),ALP1(17),E(17)
     $,RL(4000),TRUP(4000),AADT(4000),VCAT(4000),EPS,NFLI(4000),CCINTI,ACCBIS
      INTEGER TOO(4000),TOD(2000),FS(2002),ODLK(1501),DMP(50),TYP(4000),
     $T(4000),LINK(4000),INTI
      DO 30 I=1,NARC
   30 NFLI(I)=0.
      CALL AONSOD(FL,NNODA,NNODT,NOD,NCENT)
      DO 29 I=1,NARC
      NFLI(I)=FL(I)
 29   CONTINUE
      FOBJ=0
      CONV=2.*EPS
      ITER=1
      ITERIN=1
      ITOT=0
      J=1
      CALL DUMPSOD(ITER,ITERIN,ITOT,EPS,CONV,FOBJ,NNODA,NCENT)
      DO 11 I=ITER,100
      ITER = I
      DO 14 II=1,INTI
      ITERIN=II
      CALL AONSOD(NFL,NNODA,NNODT,NOD,NCENT)
      CALL BISSOD(NFL)
      CONVI=0.
      DO 13 NI=1,NARC
      XNI=ABS(NFL(NI)-NFLI(NI))
      IF(XNI.EQ.0.)GO TO 13
      DI=NFL(NI)
      IF(DI.EQ.0.)DI=NFLI(NI)
      CONVI=CONVI+XNI/DI
      NFLI(NI)=NFL(NI)
   13 CONTINUE
      CONVI=CONVI/FLOAT(NARC)
      IF(CONVI.LT.CCINTI)GO TO 15
   14 CONTINUE
   15 CONV=0.
      ITOT=ITOT+ITERIN
      FOBJ=0.
      DO 20 N=1,NARC
      XN=ABS(NFL(N)-FL(N))
      IF (XN.EQ.0.) GO TO 20
      D=NFL(N)
      IF (D.EQ.0.) D=FL(N)
      CONV=CONV+XN/D
      FL(N)=NFL(N)
   20 CONTINUE
      CONV=CONV/FLOAT(NARC)
      IF(CONV.LT.EPS) GO TO 12
      IF(ITER.EQ.50) GO TO 12
      IF(DMP(J).EQ.ITER) GO TO 22
      GO TO 11
   22 CALL DUMPSOD(ITER,ITERIN,ITOT,EPS,CONV,FOBJ,NNODA,NCENT)
      J=J+1
   11 CONTINUE
   12 CALL DUMPSOD(ITER,ITERIN,ITOT,EPS,CONV,FOBJ,NNODA,NCENT)
      RETURN
      END
C
      SUBROUTINE AONSOD(NFL,NNODA,NNODT,NOD,NCENT)
      COMMON /ARCDT/ TOO,L,C,VEL,FL,COST,T,AL
      COMMON /ENA/ NFLI,INTI,EPS,CCINTI,ACCBIS

      COMMON /ODDT/ TOD,AMT
      COMMON /FST/ FS
      COMMON /ODL/ ODLK
      COMMON /ALBET/ ALP,BET,ALP1,TYP,E
      COMMON/COM4/VALUE1,NARC,LINK,AADT,VCRT
      REAL L(4000),C(4000),VEL(4000),FL(4000),COST(4000),AMT(2000)
     *,NFL(4000),SP(2001),ALP(17),BET(17),ALP1(17),E(17)
     $,AL(4000),AADT(4000),VCRT(4000),TRUP(4000),EPS,NFLI(4000),CCINTI,ACCBIS
      INTEGER TOO(4000),TOD(2000),FS(2002),ODLK(1501),PRED(2001),
     $TYP(4000),T(4000),LINK(4000)
      DO 10 N=1,NARC
      E1=E(TYP(N))
      IF(N.LE.NARC/2) THEN
                  K=NARC/2+N
                   A1=ALP(TYP(N))
                   B1=BET(TYP(N))
                   A2=ALP(TYP(K))
                   B2=BET(TYP(K))
                   NFL(N)=0
                   NFL(K)=0
                   C1=C(N)
                   C2=C(K)
                   FLA=NFLI(N)
                   FLT=FL(K)
                   T1=T(N)
                   T2=T(K)
      ELSE
                   K=N-NARC/2
                   A1=ALP(TYP(N))
                   B1=BET(TYP(N))
                   A2=ALP(TYP(N))
                   B2=BET(TYP(K))
                   NFL(K)=0
                   NFL(N)=0
                   C1=C(K)
                   C2=C(N)
                   FLA=FL(K)
                   FLT=NFLI(N)
                   T2=T(K)
                   T1=T(N)
      ENDIF
   10 COST(N)=COSTFN(L(N),C1,C2,VEL(N),FLA,FLT,A1,B1,A2,B2,E1,
     $T1,T2,RL(N))
      II=1
      NNA=1
      KK=NCENT
      NNT=NNODA
      GO TO 2
    3 II=NNODA
      NNA=NNODA
      KK=NNODA+NCENT-1
      NNT=NNODT
    2 DO 20 I=II,KK
      I1=ODLK(I)
      I2=ODLK(I+1)-1
      IF (I1.GT.I2) GO TO 20
      CALL SHPSOD(I,PRED,SP,NNA,NNT)
      DO 30 K=I1,I2
      J=TOD(K)
   60 J1=PRED(J)
      IF (J1.EQ.0) GO TO 30
      N1=FS(J1)
      N2=FS(J1+1)-1
      DO 40 N=N1,N2
      IF (TOD(N).EQ.J) GO TO 50
   40 CONTINUE
   50 NFL(N)=NFL(N)+AMT(K)
      J=J1
      GO TO 60
   30 CONTINUE
   20 CONTINUE
      IF(II.EQ.1) GO TO 3
      RETURN
      END
C
      SUBROUTINE BISSOD(NFL)
      COMMON /ENA/ NFLI,INTI,EPS,CCINTI,ACCBIS
      COMMON/COM4/VALUE1,NARC,LINK,ARDT,VCRT
      COMMON /ARCDT/ TOO,L,C,VEL,FL,COST,T,RL

      COMMON /FST/ FS
      COMMON /ALBET/ ALP,BET,ALP1,TYP,E
      REAL L(4000),C(4000),VEL(4000),FL(4000),COST(4000),NFL(4000),
     $ALP(17),BET(17),ALP1(17),E(17),RL(4000),
     $TRUP(4000),ARDT(4000),VCRT(4000),EPS,NFLI(4000),CCINTI,ACCBIS
      INTEGER TOO(4000),FS(2002),TYP(4000),T(4000),LINK(4000),INTI
      AMN=0
      AMX=1
   10 AMD=(AMX+AMN)/2
      IF ((AMX-AMN).LE.ACCBIS) GO TO 20
      D=0
      DO 30 N=1,NARC/2
      X1=NFLI(N)+AMD*(NFL(N)-NFLI(N))
      A1=ALP(TYP(N))
      B1=BET(TYP(N))
      E1=E(TYP(N))
      K=N+NARC/2
      C1=C(N)
      C2=C(K)
      FLA=FL(N)
      FLT=FL(K)
      T1=T(N)
      T2=T(K)
      X2=NFLI(K)+AMD*(NFL(K)-NFLI(K))
      A2=ALP(TYP(K))
      B2=BET(TYP(K))
      E2=E(TYP(K))
      CST1=COSTFN(L(N),C1,C2,VEL(N),X1,FLT,A1,B1,A2,B2,E1,T1,T2,AL(N))
      CST2=COSTFN(L(K),C1,C2,VEL(K),FLA,X2,A2,B2,A1,B1,E2,T2,T1,AL(K))
   30 D=D+CST1*(NFL(N)-NFLI(N))+CST2*(NFL(K)-NFLI(K))
      IF (D.GT.0. ) AMX=AMD
      IF (D.LE.0. ) AMN=AMD
      GO TO 10
   20 DO 40 N=1,NARC
   40 NFL(N)=NFLI(N)+AMD*(NFL(N)-NFLI(N))
      RETURN
      END
C
      FUNCTION TRCOST(D,C1,C2,VEL,FLA,FLT,A,B,E,T1,T2,AL)
      TRCOST=D*AL/VEL
      IF (C1.NE.0.AND.C2.NE.0.)
     $TRCOST=TRCOST*(1.+A*(FLA*T1/C1+E*FLT*T2/C2)**B)
      RETURN
      END
C
      FUNCTION COSTFN(D,C1,C2,VEL,FLA,FLT,A1,B1,A2,B2,E,T1,T2,AL)
      COSTFN=D*AL/VEL
      IF(C1.NE.0. .AND. C2.NE.0.)THEN
      B11=B1-1
      B22=B2-1
      AT=FLA*T1/C1+E*FLT*T2/C2
      ATT=AT+FLA*T1*B1/C1
      COSTFN=COSTFN*(1.+A1*ATT*AT**B11+A2*B2*FLT*T2*AT**B22/C1)
      ENDIF
      RETURN
      END
C
      FUNCTION FINT(D,C1,C2,VEL,FLA,FLT,FL,A1,B1,A2,B2,E,T1,T2)
      FINT =D/VEL*FL
      B11=B1-1
      B22=B2-1
      IF(C1.NE.0. .AND. C2.NE.0.)
     $AT=FLA*T1/C1+E*FLT*T2/C2
      ATT=AT+FLA*T1*B1/C1
      FINT=FINT*(1.+A1*ATT*AT**B11+A2*B2*FLT*T2*AT**B22/C1)
      RETURN
      END
C
      SUBROUTINE SHPSOD(A,PRED,SP,NNA,NNT)
C
C  THIS SUBROUTINE COMPUTES SHORTEST PATHS FROM A TO ALL OTHER NODES.
C  PRED(I) CONTAINS PREDECESSOR OF NODE I, SP(I) CONTAINS LENGTH OF
C  PATH TO NODE I.
C
      COMMON /ARCDT/ TOO,L,C,VEL,FL,COST,T,AL
      COMMON /FST/ FS
      COMMON/COM4/VALUE1,NARC,LINK,AADT,VCRT

      REAL L(4000),C(4000),VEL(4000),FL(4000),COST(4000),SP(2001)
     $,AL(4000),TRUP(4000),AADT(4000),VCRT(4000)
      INTEGER,TOO(4000),FS(2002),PRED(2001),CL(2001),T(4000),LINK(4000)
     $,R
      DO 10 I=NNA,NNT
      SP(I)=1.E20
      PRED(I)=0
   10 CL(I)=0
      SP(R)=0
      CL(R)=NNT+1
      I=R
      NT=R
   20 IA=FS(I+1)-1
      S=SP(I)
      IA1=FS(I)
      IF (IA1.GT.IA) GO TO 30
      DO 40 IR=IA1,IA
      K=TOO(IR)
      SD=S+COST(IR)
      IF (SD.GE.SP(K)) GO TO 40
      PRED(K)=I
      SP(K)=SD
      IF (CL(K)) 50,60,40
   60 CL(NT)=K
      NT=K
      CL(K)=NNT+1
      GO TO 40
   50 CL(K)=CL(I)
      CL(I)=K
   40 CONTINUE
   30 ICL=CL(I)
      CL(I)=-1
      I=ICL
      IF (I.LE.NNT) GO TO 20
      RETURN
      END
C
      SUBROUTINE DUMPSOD(ITER,ITERIN,ITOT,EPS,CONV,FOBJ,NNODA,NCENT)
      COMMON /ARCDT/ TOO,L,C,VEL,FL,COST,T,AL
      COMMON /FST/ FS
      COMMON /ALBET/ ALP,BET,ALP1,TYP,E
      COMMON/COM4/VALUE1,NARC,LINK,AADT,VCRT
      REAL L(4000),C(4000),VEL(4000),FL(4000),COST(4000),MFL(4000)
     *,ALP(17),BET(17),ALP1(17),VCRT(4000),E(17),TRUP(4000),
     $AADT(4000),AL(4000)
      INTEGER TOO(4000),FS(2002),TYP(4000),VC1(4000),VC2(4000),T(4000),
     $LINK(4000),LLINK(4000)
      WRITE(6,101) ITER,ITERIN,ITOT
  101 FORMAT('NO. OF ITERATIONS = 'I6/
     $       'NO. OF INT.ITERATIONS= 'I6/
     $       'TOTAL NO. ITERATIONS= 'I6//)
      K=0
      VALUE1=0
      NNOD1=NNODA-1
      NNODMC=2*NCENT+1
      GO TO 4
    5 NNODMC=NNODMC+NNODA-1
      AUTOTT=VALUE1
      NNOD1=2*NNOD1
   4  DO 10 I=NNODMC,NNOD1
      J1=FS(I)
      J2=FS(I+1)-1
       IF (J1.GT.J2) GO TO 10
      DO 20 J=J1,J2
       IF (TOO(J).LT.NNODMC) GO TO 20
      K=K+1
      NFL(K)=FL(J)
      LLINK(K)=LINK(J)
      VCRT(K)=FL(J)/C(J)
      VC1(K)=I
      VC2(K)=TOO(J)
      A1=ALP(TYP(J))
      B1=BET(TYP(J))
      E1=E(TYP(J))
       IF (J.LE.NARC/2) THEN
                      KLM=J+NARC/2
                      C1=C(J)
                      C2=C(KLM)
                      FLA=FL(J)
                      FLT=FL(KLM)
                      T1=T(J)
                      T2=T(KLM)
                      VCRT(K)=FLA*T1/C1+E1*FLT*T2/C2
                      AADT(K)=FLA*T1+FLT*T2
                      TRUP(K)=FLT*T2*100./AADT(K)
                       IF(AADT(K).EQ.0.)TRUP(K) =0.
      ELSE
                      KLM=J-NARC/2
                      C1=C(KLM)
                      C2=C(J)
                      FLA=FL(KLM)
                      FLT=FL(J)
                      T2=T(KLM)
                      T1=T(J)
                      VCRT(K)=FLA*T1/C1+E1*FLT*T2/C2
                      AADT(K)=FLA*T1+FLT*T2
                      TRUP(K)=FLT*100./AADT(K)
                       IF(AADT(K).EQ.0.) TRUP(K)=0.
      ENDIF
      CST=TRCOST(L(J),C1,C2,VEL(J),FLA,FLT,A1,B1,E1,T1,T2,AL(J))
      VALUE1=VALUE1+CST*FL(J)
   20 CONTINUE
   10 CONTINUE
       IF (NNOD1.EQ.NNODA-1) GO TO 5
      TRUCKTT=VALUE1-AUTOTT
      WRITE(6,2) ITER,CONV,VALUE1,AUTOTT,TRUCKTT
    2 FORMAT(/'AFTER ',I3,'ITERATIONS:'/
     $        '   CONVERGENCE MEASURE      =',F7.4/
     $        '   TOTAL TRAVEL TIME        =',E25.19/
     $        '   AUTO  TRAVEL TIME        =',E25.19/
     $        '   TRUCK TRAVEL TIME        =',E25.19///)
       IF(CONV.GT.EPS)RETURN
      RETURN
      WRITE(6,113)
  113 FORMAT(32X,'ORIG',2X,'DEST')
      WRITE(6,112)
  112 FORMAT('LINK NO.',5X,'FLOW',4X,'V/C RATIO',2X,'NODE',2X,
     4'NODE',2X,' % TRUCKS',2X,'AADT',3X,'LINK ID')
      WRITE(6,114)
  114 FORMAT('-------',5X,'----'4X,'----------',2X,'----',2X,'----'
     $,2X,'---------',2X,'----',3X,'-------')
      DO 23 I=1,K
      WRITE(6,102) I,NFL(I),VCRT(I),VC1(I),VC2(I),TRUP(I),AADT(I)
     $,LLINK(I)
  102 FORMAT(I5,F13.2,F10.2,2I7,2F9.2,I8)
   23 CONTINUE
      RETURN
      END
```

APPENDIX C-4

SAMPLE OUTPUT OF THE USER EQUILIBRIUM COMPUTER PROGRAM

```
                  USER EQUILIBRIUM ASSIGNMENT - PSU
   # OF OD PAIRS =   220

   CONVERGENCE CRITERION =     .500000000E-02
    INTERNAL CONVERGENCE  =    .500000000E-02
   ACCURACY OF MOVE SIZE =     .500000000E-03
    INTERNAL ITERATIONS   =   1
NO. OF ITERATIONS =       1
NO. OF INT. ITERATIONS=      1
TOTAL NO. ITERATIONS=      0


 AFTER   1 ITERATIONS:
   CONVERGENCE MEASURE        =     .100000000E-01
   OBJECTIVE FUNCTION         =                0
   TOTAL TRAVEL TIME          =     .181192584E+05
NO. OF ITERATIONS =       1
NO. OF INT. ITERATIONS=      1
TOTAL NO. ITERATIONS=      1


 AFTER   1 ITERATIONS:
   CONVERGENCE MEASURE        =     .301931358E+00
   OBJECTIVE FUNCTION         =                0
   TOTAL TRAVEL TIME          =     .118355155E+05
NO. OF ITERATIONS =       2
NO. OF INT. ITERATIONS=      1
TOTAL NO. ITERATIONS=      2


 AFTER   2 ITERATIONS:
   CONVERGENCE MEASURE        =     .512224274E+00
   OBJECTIVE FUNCTION         =                0
   TOTAL TRAVEL TIME          =     .100415127E+05
NO. OF ITERATIONS =       3
NO. OF INT. ITERATIONS=      1
TOTAL NO. ITERATIONS=      3


 AFTER   3 ITERATIONS:
   CONVERGENCE MEASURE        =     .198381402E+00
   OBJECTIVE FUNCTION         =                0
   TOTAL TRAVEL TIME          =     .915627417E+04
NO. OF ITERATIONS =       4
NO. OF INT. ITERATIONS=      1
TOTAL NO. ITERATIONS=      4


 AFTER   4 ITERATIONS:
   CONVERGENCE MEASURE        =     .124651763E+00
   OBJECTIVE FUNCTION         =                0
```

```
     TOTAL TRAVEL TIME          =    .923790807E+04
NO. OF ITERATIONS =      5
NO. OF INT. ITERATIONS=       1
TOTAL NO. ITERATIONS=      5


  AFTER   5 ITERATIONS:
     CONVERGENCE MEASURE        =    .188447533E+00
     OBJECTIVE FUNCTION         =                0
     TOTAL TRAVEL TIME          =    .908713451E+04
NO. OF ITERATIONS =      6
NO. OF INT. ITERATIONS=       1
TOTAL NO. ITERATIONS=      6


  AFTER   6 ITERATIONS:
     CONVERGENCE MEASURE        =    .109827811E+00
     OBJECTIVE FUNCTION         =                0
     TOTAL TRAVEL TIME          =    .902875700E+04
NO. OF ITERATIONS =      7
NO. OF INT. ITERATIONS=       1
TOTAL NO. ITERATIONS=      7


  AFTER   7 ITERATIONS:
     CONVERGENCE MEASURE        =    .641257165E-01
     OBJECTIVE FUNCTION         =                0
     TOTAL TRAVEL TIME          =    .903055287E+04
NO. OF ITERATIONS =      8
NO. OF INT. ITERATIONS=       1
TOTAL NO. ITERATIONS=      8


  AFTER   8 ITERATIONS:
     CONVERGENCE MEASURE        =    .889874080E-01
     OBJECTIVE FUNCTION         =                0
     TOTAL TRAVEL TIME          =    .889080198E+04
NO. OF ITERATIONS =     30
NO. OF INT. ITERATIONS=       1
TOTAL NO. ITERATIONS=     30


  AFTER  30 ITERATIONS:
     CONVERGENCE MEASURE        =    .605009161E-02
     OBJECTIVE FUNCTION         =                0
     TOTAL TRAVEL TIME          =    .886325600E+04
NO. OF ITERATIONS =     31
NO. OF INT. ITERATIONS=       1
TOTAL NO. ITERATIONS=     31
```

```
 AFTER  31 ITERATIONS:
    CONVERGENCE MEASURE        =     .408009063E-02
    OBJECTIVE FUNCTION         =                 0
    TOTAL TRAVEL TIME          =     .886235948E+04
                                         ORIG  DEST
LINK NO.       FLOW     V/C RATIO  NODE  NODE     % TRUCKS  AADT     LINK ID
--------       ----     ---------  ----  ----     --------  ----     -------
     1       2004.60         .56     23      24      23.33  2614.60          0
     2       2706.36         .69     23      25      20.64  3410.08          0
     3        689.04         .18     23      26        .90   695.32          0
     4       2004.60         .56     24      23      23.33  2614.60          0
     5        952.02         .21     24      25       7.99  1034.70          0
     6       1771.52         .56     24      26       6.34  1891.50          0
     7       1385.84         .40     24      30       3.58  1437.24          0
     8       3394.65         .80     24      31      18.21  4150.56          0
     9       2706.36         .69     25      23      20.64  3410.08          0
    10        952.02         .21     25      24       7.99  1034.70          0
    11        376.99         .10     25      26        .35   378.31          0
    12       3741.96         .83     25      27      16.10  4460.08          0
    13        689.04         .18     26      23        .90   695.32          0
    14        376.99         .10     26      25        .35   378.31          0
    15        927.46         .43     26      27      17.92  1129.98          0
    16       2909.36         .74     26      28        .56  2925.83          0
    17       1018.10         .35     26      30       8.27  1109.89          0
    18       1771.52         .56     26      24       6.34  1891.50          0
    19       3963.62         .91     27      37      17.35  4795.66          0
    20       1855.80         .48     27      28        .68  1868.41          0
    21        927.46         .43     27      26      17.92  1129.98          0
    22       3741.96         .83     27      25      16.10  4460.08          0
    23       1855.80         .48     28      27        .68  1868.41          0
    24       2495.41         .70     28      37       3.05  2573.85          0
    25       2265.43         .58     28      36        .52  2277.25          0
    26       2218.55         .64     28      29       3.58  2301.04          0
    27       2909.36         .74     28      26        .56  2925.83          0
    28       2218.55         .64     29      28       3.58  2301.04          0
    29        361.51         .11     29      36       4.49   378.51          0
    30       2657.69         .71     29      34       1.58  2700.37          0
    31       2064.02         .60     29      32       4.05  2151.23          0
    32       2350.87         .64     29      30       2.23  2404.51          0
    33       1385.84         .40     30      24       3.58  1437.24          0
    34       1018.10         .35     30      26       8.27  1109.89          0
    35       2350.87         .64     30      29       2.23  2404.51          0
    36       1964.30         .58     30      31       4.55  2057.96          0
    37       3394.65         .80     31      24      18.21  4150.56          0
    38       1964.30         .58     31      30       4.55  2057.96          0
    39       3480.35         .79     31      32      16.99  4192.59          0
    40       3480.35         .79     32      31      16.99  4192.59          0
    41       2064.02         .60     32      29       4.05  2151.23          0
    42       4016.33         .84     32      33      14.39  4691.36          0
    43       4016.33         .84     33      32      14.39  4691.36          0
    44        600.29         .18     33      34       4.00   625.30          0
    45       3216.62         .64     33      39      12.87  3691.67          0
    46        600.29         .18     34      33       4.00   625.30          0
    47       2657.69         .71     34      29       1.58  2700.37          0
    48        339.15         .12     34      36       9.85   376.23          0
    49       1181.78         .32     34      35       2.30  1209.54          0
    50       1214.78         .33     34      39       2.17  1241.75          0
```

51	2271.97	.42	35	39	10.35	2534.20	0
52	1181.78	.32	35	34	2.30	1209.54	0
53	2878.11	.75	35	36	1.00	2907.08	0
54	2144.71	.44	35	38	13.90	2490.95	0
55	361.51	.11	36	29	4.49	378.51	0
56	2265.43	.58	36	28	.52	2277.25	0
57	590.31	.18	36	37	5.93	627.54	0
58	2878.11	.75	36	35	1.00	2907.08	0
59	339.15	.12	36	34	9.85	376.23	0
60	4568.71	.89	37	38	12.27	5207.96	0
61	590.31	.18	37	36	5.93	627.54	0
62	2495.41	.70	37	28	3.05	2573.85	0
63	3963.62	.91	37	27	17.35	4795.66	0
64	4568.71	.89	38	37	12.27	5207.96	0
65	2144.71	.44	38	35	13.90	2490.95	0
66	3216.62	.64	39	33	12.87	3691.67	0
67	1214.78	.33	39	34	2.17	1241.75	0
68	2271.97	.42	39	35	10.35	2534.20	0
69	610.00	.56	62	63	23.33	2614.60	0
70	703.72	.69	62	64	20.64	3410.08	0
71	6.28	.18	62	65	.90	695.32	0
72	610.00	.56	63	62	23.33	2614.60	0
73	82.69	.21	63	64	7.99	1034.70	0
74	119.98	.56	63	65	6.34	1891.50	0
75	51.39	.40	63	69	3.58	1437.24	0
76	755.91	.80	63	70	18.21	4150.56	0
77	703.72	.69	64	62	20.64	3410.08	0
78	82.69	.21	64	63	7.99	1034.70	0
79	1.32	.10	64	65	.35	378.31	0
80	718.13	.83	64	66	16.10	4460.08	0
81	6.28	.18	65	62	.90	695.32	0
82	1.32	.10	65	64	.35	378.31	0
83	202.53	.43	65	66	17.92	1129.98	0
84	16.47	.74	65	67	.56	2925.83	0
85	91.79	.35	65	69	8.27	1109.99	0
86	119.98	.56	65	63	6.34	1891.50	0
87	832.04	.91	66	76	17.35	4795.66	0
88	12.61	.48	66	67	.68	1868.41	0
89	202.53	.43	66	65	17.92	1129.98	0
90	718.13	.83	66	64	16.10	4460.08	0
91	12.61	.48	67	66	.68	1868.41	0
92	78.44	.70	67	76	3.05	2573.85	0
93	11.82	.58	67	75	.52	2277.25	0
94	82.49	.64	67	68	3.58	2301.04	0
95	16.47	.74	67	65	.56	2925.83	0
96	82.49	.64	68	67	3.58	2301.04	0
97	17.00	.11	68	75	4.49	378.51	0
98	42.68	.71	68	73	1.58	2700.37	0
99	87.22	.60	68	71	4.05	2151.23	0
100	53.64	.64	68	69	2.23	2404.51	0
101	51.39	.40	69	63	3.58	1437.24	0
102	91.79	.35	69	65	8.27	1109.89	0
103	53.64	.64	69	68	2.23	2404.51	0
104	93.66	.58	69	70	4.55	2057.96	0
105	755.91	.80	70	63	18.21	4150.56	0
106	93.66	.58	70	69	4.55	2057.96	0
107	712.24	.79	70	71	16.99	4192.59	0

108	712.24	.79	71	70	16.99	4192.59	0
109	87.22	.60	71	68	4.05	2151.23	0
110	675.03	.84	71	72	14.39	4691.36	0
111	675.03	.84	72	71	14.39	4691.36	0
112	25.02	.18	72	73	4.00	625.30	0
113	475.05	.64	72	78	12.87	3691.67	0
114	25.02	.18	73	72	4.00	625.30	0
115	42.68	.71	73	68	1.58	2700.37	0
116	37.07	.12	73	75	9.85	376.23	0
117	27.76	.32	73	74	2.30	1209.54	0
118	26.97	.33	73	78	2.17	1241.75	0
119	262.23	.42	74	78	10.35	2534.20	0
120	27.76	.32	74	73	2.30	1209.54	0
121	28.97	.75	74	75	1.00	2907.08	0
122	346.24	.44	74	77	13.90	2490.95	0
123	17.00	.11	75	68	4.49	378.51	0
124	11.82	.58	75	67	.52	2277.25	0
125	37.23	.18	75	76	5.93	627.54	0
126	28.97	.75	75	74	1.00	2907.08	0
127	37.07	.12	75	73	9.85	376.23	0
128	639.25	.89	76	77	12.27	5207.96	0
129	37.23	.18	76	75	5.93	627.54	0
130	78.44	.70	76	67	3.05	2573.85	0
131	832.04	.91	76	66	17.35	4795.66	0
132	639.25	.89	77	76	12.27	5207.96	0
133	346.24	.44	77	74	13.90	2490.95	0
134	475.05	.64	78	72	12.87	3691.67	0
135	26.97	.33	78	73	2.17	1241.75	0
136	262.23	.42	78	74	10.35	2534.20	0