Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

NJIT New Jersey's Science & Technology University

The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
313/761-4700 800/521-0600

Order Number 1360138

A GIS based multi-modal multiple optimal path transit advanced traveler information system

Koncz, Nicholas Alex, M.S.

New Jersey Institute of Technology, 1994

Copyright ©1994 by Koncz, Nicholas Alex. All rights reserved.

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

ABSTRACT

A GIS BASED MULTI-MODAL MULTIPLE OPTIMAL PATH TRANSIT ADVANCED TRAVELER INFORMATION SYSTEM

by
Nicholas A. Koncz

A method for the design and use of a Transit Advanced Traveler Information System (TATIS) using an off-the-shelf Geographic Information System (GIS) is developed in this thesis. The research design included: 1) representing multi-modal transit networks in a digital form with schedule databases; 2) development of a multiple "optimal" path algorithm that takes into account walking transfers using published time schedules; 3) incorporating user preferences and penalties in the algorithm; 4) development of a user-interface with suitable output capabilities; 5) using the prototype for sample inquiries giving performance measures.

This prototype was developed using the Arc/Info GIS developed by ESRI, Inc. The principal results of the research demonstrated the effectiveness and robustness of the TATIS prototype with respect to the five previously mentioned issues. Areas of future improvement and research focus on performance measures and added functionality.

A GIS BASED MULTI-MODAL MULTIPLE OPTIMAL PATH TRANSIT ADVANCED TRAVELER INFORMATION SYSTEM

by
Nicholas A. Koncz

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Transportation

Committee for the Interdisciplinary Program in Transportation

October 1994

Copyright © 1994 by Nicholas A. Koncz
ALL RIGHTS RESERVED

APPROVAL PAGE

A GIS BASED MULTI-MODAL MULTIPLE OPTIMAL PATH TRANSIT ADVANCED TRAVELER INFORMATION SYSTEM

Nicholas A. Koncz

Dr. Kyriacos Mouskos, Thesis Co-Adviser — Date
Assistant Professor, Civil and Environmental Engineering,
Institute for Transportation, NJIT

Dr. Joshua Greenfeld, Thesis Co-Adviser — Date
Associate Professor, Civil and Environmental Engineering

Dr. Louis J. Pignataro, Committee Member — Date
Distinguished Professor, Transportation Engineering and
Director, Institute for Transportation, NJIT

BIOGRAPHICAL SKETCH

Author: Nicholas A. Koncz

Degree: Master of Science in Transportation

Date: October 1994

Undergraduate and Graduate Education:

- Master of Science in Transportation,
  New Jersey Institute of Technology, Newark, NJ, 1994
- Bachelor of Science in Civil Engineering, Summa Cum Laude
  New Jersey Institute of Technology, Newark, NJ, 1987

Major: Transportation

Professional Positions:

- Design Engineer, Boswell Engineering,
  South Hackensack, NJ, 1987-1993
- Research Assistant, New Jersey Institute of
  Technology, Newark, NJ 1993-1994

This thesis is dedicated to
Keith Green, 1954-1982 who had a profound
impact on my life and demonstrated a life of No Compromise
before the Lion of the Tribe of Judah

ACKNOWLEDGEMENT

The author would like to express his sincere gratitude and appreciation to all who provided valuable assistance and advice throughout the course of this research. Special thanks are due to Professor Kyriacos Mouskos and Professor Joshua Greenfeld, my thesis advisors, for their generous advice and guidance and to Dr. Louis J. Pignataro, Director of the Center for Transportation Research and Studies (CTSR) at New Jersey Institute of Technology, for his generous efforts in providing funding for the support of this research. Support for this research was provided from a grant from the Federal Highway Administration (FHWA Cooperative Agreement DTFH61-92-X-00024) and the New Jersey Institute of Technology.

The author also would like to especially thank Kamal Azar and Thomas Grayson, graduate students from Massachussettes Institute of Technology for their invaluable effort in producing research fundamental to the basis of my research.

Finally, the author would like to thank those in the research team working on the other modules in the CTSR ATIS prototype, namely Peda Babu Kandru (route guidance), Jilong Zheng (ridesharing), and Qifeng Zheng (user-interface) for their help and support throughout my research.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES
(Continued)

CHAPTER 1

INTRODUCTION

This reseach presents a Transit Advanced Traveler Information System (TATIS) based upon a Geographic Information System (GIS). The TATIS is part of a Multi-media Advanced Traveler Information System (MATIS) which includes two more components: a Route Planning Information System and a Ride-sharing Information System. The principal characteristics of the TATIS developed in this study are the following: 1) representing multi-modal transit networks in a digital form with schedule databases; 2) development of a multiple "optimal" path algorithm that takes into account walking transfers using published time schedules; 3) incorporating user preferences and penalties in the algorithm; 4) development of a user-interface with suitable output capabilities; and 5) using the prototype for sample inquiries giving performance measures.

The TATIS was developed using the Arc/Info geographic information system (GIS) software created by ESRI, Inc.

1.1 Characteristics of Transit Operators

In order to encourage passengers to use transit systems, the transit operator must be able to handle complex inquires on how to use their system and the flexibility in providing information.

The trip making process using transit versus private automobile differs in the following ways: 1) The departure and arrival time from the origin to the destination are dictated by the bus/train schedule; 2) The trip using bus or train may require significant walking paths from the origin to a bus/train stop and from the bus/train stop to the

destination; 3) The transit trip may require more than one transfer; 4) Other socioeconomic factors such as independence, fare, safety, comfort are present.

The transit operator must at least be able to answer the question on how to go from an origin to a destination leaving at a certain time, or the more complex question on how to arrive at a destination at a given time. The term "transit operator" used in this discussion can refer to a person, a computer-based information system, or a combination of both, located in a customer service center.

Another level of complexity transpires when more than one path exists from an origin to a destination. Often the most direct or simplest path is given to the user. This is often misleading in that there may be several routes which are in the vicinity of the passenger's origin, or a variety of routes with different travel speeds at a nearby transit stop. Additionally, better routes may be possible by making one or two transfers, saving travel time. Moreover, the best route may not be valid for certain parts of the day due to schedule variability and transfer availability. Therefore, the best transit route may not be static but dynamic due to the time of travel. A well-trained transit operator would provide the user with several alternatives to choose from and allow the user to adjust his schedule and departure time accordingly.

In addition, transit operators often focus on traversing transit network within the confines of the network. For example, a user transferring from one bus to another at an intersection, terminal or station. In many situations more possibilities for arriving at a destination occur when this condition is relaxed. If the assumption is made that people are willing to walk under a certain time to a transfer station, overall travel time could be reduced and locations that were once inaccessible, or accessed by a series of roundabout routes, could now be reached and in more direct manner.

Providing user flexibility is key to any decision making process and a transit operator would be ineffective if it did not provide for it. An operator may provide the

best route, or multiple optimal routes, but must be able to provide for user preferences. These preferences include choice of mode, number of transfers, walking and waiting time to the transit stop. In addition, it must be able to handle simple questions as finding the next three arrival times of the transit vehicle.

In addition, the information provided by the transit operator must be in a form that is easily understandable both to the user and the operator. The output should be in the context of a user-friendly menu based interface providing a graphical image of the route along with a complete textual summary of the available user-preferred routes. Both forms of output should be as user-friendly as possible.

Finally, information must be given in a timely fashion. Users get impatient when they have to wait more than several seconds for answers to their questions. In this time of rapid technological advances, most people would question the effectiveness of a system that took several minutes to answer their inquiry.

1.2 General Background

1.2.1 IVHS and Components

The Intelligent Vehicle Highway System (IVHS) concept is based on the use of computer, electronics and communication technologies to increase the effectiveness of the overall surface transportation system. Many of these technologies have already advanced other modes of transportation, such as aviation, and are now beginning to be used in surface transportation. IVHS technologies will be applied to all types of vehicles, to terminals (computers, kiosks, and hand-held devices), and to all parts of the surface transportation system (freeways, urban arterial roadways, city streets, rural roads and intermodal connections) in order to improve safety, energy efficiency and environmental quality; to enhance economic productivity, to reduce congestion and improve accessibility.

The whole of IVHS is greater than the sum of its parts. Ideally every point on the system will be able to work with every other point. The scope of developing and deploying such a tightly woven, technology-driven transportation network demanded that experts divide the field into a number of interrelated categories. These categories are: Advanced Traffic Management Systems (ATMS), Advanced Traveler Information Systems (ATIS), Advanced Vehicle Control Systems (AVCS), Advanced Public Transportation Systems (APTS), Commercial Vehicle Operations (CVO), and Advanced Rural Transportation Systems (ARTS) Arlook and Jones (1993) present a summary of IVHS and its components.

1.2.2 ATIS and APTS

Advanced Traveler Information Systems, a field of IVHS, provide assistance to travelers in making trips both using private vehicles and public transportation. They support pre-trip planning with the aid of microcomputers in kiosks, homes, offices, or portable (notebook and handheld) applications (i.e., using wireless technologies), and they can include address finding, business listings, tourist information, intermodal public transit information (routes, schedules, fares, transfers) and route planning. Route planning may be augmented with average congestion models or with real-time and projected congestion information. Sweeney et al. (1993, 98) provide an excellent description of ATIS and its components.

Of special interest are pre-trip travel information and enroute transit advisories. Pre-trip travel information services will allow travelers access to a complete range of multimodal transportation information at home, work, and other major destinations. This information will be used to choose the mode of travel and departure time that aids the traveler in avoiding congestion. Enroute transit advisories will provide travelers

with real-time accurate, transit information while enroute to their destinations. This information will increase the attractiveness of transit to the traveler.

Advanced Public Transportation Systems, or APTS, are advanced navigation and communication technologies applied to all aspects of public transportation system operations. APTS provides the technology for transportation agencies to make timely transit information available to the passenger and to improve the convenience, relliability and safety of public transportation service (USDOT 1993).

1.2.3 Application of GIS to ATIS/APTS

Sweeney et al. (1993) discuss the suitability of using a geographic database system to represent transportation information, especially in regard to ATIS and APTS. Since most land transportation information is directly related to and conveniently associated with roads, the most efficient method of digitally storing and representing road map data is to use topology. This is where features of the map are stored simply as points representing road intersections (nodes), with various attributes associated with the roads (links) that interconnect nodes, in addition to various additional attributes associated with the areas (polygons) enclosed by links. Most importantly, each map feature has stored with it pointers to all related features.

1.2.4 Geographical Information Systems

Geographical Information Systems have been defined in many ways by several individuals. A more complete definition is:

> A GIS is a tool that provides data base management capabilities (including capture, selection, storage, editing, querying, retrieval, and reporting functions) for and display of spatial data, and provides the ability to perform analysis of geographic features (points, lines, and polygons) based on their explicit relationship to each other (Schweiger 1992).

An important concept that makes GIS different from other CADD (i.e., computed-aided design and drafting) systems is topology. Topology was discussed previously. GIS's contain four capabilities that are useful and necessary for building an ATIS model. First, is the capability of the input of standard street network files provided by other vendors or agencies such as the TIGER/Line files or creating similar files. Second, is the ability to do address-matching or geocoding. This allows addresses to be assigned geographic coordinates. Third, is the ability to do pathfinding, whereby an optimal path based on time or length is found from an origin to a destination through any desired stops. Fourth, is the ability to perform dynamic segmentation. This capability allows the modeling of linear spatial features using arcs or portions of arcs without disrupting the integrity of the database. Antenucci, et al. (1991) and Huxold (1991) are excellent introductory texts which provide a general perspective of the technology and its applications.

1.2.5 GIS Applications to Transportation

Dueker et al. (1991), presents that the literature on GIS applications in transportation is growing rapidly. Much of the state-level DOT work emanates from Fletcher's (1987 and 1990) pioneering applications, the Nyerges and Dueker (1988) review for FHWA, the GIS-T short courses by Fletcher and Lewis (1989, 1990a, 1990b), and the AASHTO sponsored symposia on GIS-T (Bacon and Moyer, 1989; Bacon and Moyer, 1990; Moyer and Larson, 1991; Moyer, 1992; Moyer and Ries, 1993). In addition, the NCHRP has sponsored a major research project on GIS-T (Travis, et al., 1990). There has also been an increase in the application of GIS in urban transportation planning (Kiel, 1989; Lewis, 1990; O'Neill, 1992; Prastacos 1992; Anderson, 1992; Schweiger,

1992; Javid et al., 1994; O'Neill, 1994). Antonisse (1991) examines the development of GIS-T systems in Seattle and Boston.

1.2 Motivation

In a survey conducted in 1983 by the Marketing office of the Washington Metropolitan Area Transit Authority, it was reported that 82% of callers do make the transit trip for which they had requested information and that 66% of these callers would not have taken the transit trip without the information provided by the telephone information service (Ross and Soberman, 1987). This fact demonstrates the need of transit agencies to provide information to the passenger that adequately meets their needs. This thesis address the complex issues described in Section 1.1 and explores how an off-the-shelf geographical information system can be developed and implemented to solve them.

Several of the motivations of this thesis are the following: 1) To provide a friendlier environment for transit users (i.e., provide information in a user-friendly way and in a timely fashion). This environment needs to be flexible and complete; 2) The need to develop transit route planning algorithms which can capture different users criteria in selecting a route from an origin to a destination.

1.3 Objectives

Specific objectives of this research include the following:

1. The ability to provide paths that include walking distances from and to the transit path as well as between transfer points.
2. The algorithm have the ability to handle multiple modes of transit (i.e., train, bus, subway, ferry).
3. Multiple optimal paths are to be provided to allow the user flexibility in choosing a path.

4. The inclusion of travel time along each path, providing times of departure and arrival of each route in a path.

5. The ability to answer questions about time schedules when leaving at a certain time or arriving at a certain time.

6. To allow the user to specify the criteria for path selection, whether it is based on mode choice, number of transfers, or waiting and walking time at the transit stop.

7. To provide clear and user-friendly output forms showing the graphical image of the path as well as a tabular summary of the itinerary using menus to interact with the output.

8. The ability to perform items 1 through 7 using a prepackaged GIS.

1.4 Thesis Structure

This thesis contains seven chapters. The first chapter contains the characteristics of the thesis, and characteristics of transit operators. General background is provided by an introduction to IVHS, Advanced Traveler Information Systems and Advanced Public Transportation Systems. Because the prototype relies heavily on a specialized software technology, geographic information systems, some useful references are given. Next, some previous applications of geographic information systems in transportation are examined. Motivations for the thesis, as well as specific objectives are presented. The chapter ends with an overview of the full document.

The second chapter summarizes important precursors to this thesis and provides references for further reading on selected topics. Transit Advanced Traveler Information Systems are discussed and its interrelationship with geographic information systems. Current network representations and algorithms are delineated. Finally, this second chapter looks at some previous efforts at depicting and querying transit networks and highlights the differences between the prototype of this thesis and previous research.

The third chapter begins with a non-technical overview of the application. The basic processes involved in performing a query of the system are described. The design decisions behind the prototype are explained without diving into the technical details of the implementation.

The fourth chapter explores the structure and organization of the data. The structure and relationships among the various transit files and objects stored in Arc/Info are described. A discussion of the use of dynamic segmentation is presented.

The fifth chapter discusses the prototype's algorithms and implementation choices in more detail. The software development tool and the hardware/software environment of the implementation are discussed. The data collection and preparation process is outlined. The methods for performing a network analysis, incorporating time schedules and displaying the results are given. Development of the user-interface is presented as well as the integration of the programs composing the module.

Chapter six details the procedure used in running the prototype and provides sample test runs. A summary of results is presented with execution times as well as an analysis of results.

Finally, the last chapter concludes with an evaluation of the overall project implementation and proposes further enhancements as well as directions for future research.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter begins by reviewing Transit Advanced Traveler Information systems with an emphasis on their input requirements and output capabilities. Network analysis algorithms are examined and a summary of selected previous related research on TATIS systems is given. Capabilities of the prototype of this thesis conclude this chapter.

2.2 TATIS Systems

Transit Advanced Traveler Information Systems provide assistance to travelers in making trips using public transportation via various technologies, such as by fax, modem, pagers, kiosks, or voice-conversant systems. TATIS have been referred to by other names, such as Transit Information Systems (TIS), or Passenger Information Systems (PIS). The main difference between a TIS and a TATIS is that TATIS systems allow the user to interact directly with the information system instead of having a dedicated person to act as an intermediary between the information available and the user. However, the above distinction is not necessarily followed by everyone and TIS's and ATIS's are often used interchangeably.

Nearly all transit authorities in the United States provide telephone information service to customers to help them use transit effectively. To what degree this service is computerized varies with the size and complexity of the transit system and with their public marketing strategy. Some agencies are still using manual telephone transit information systems, where the operator is responsible for determining the response to

inquires based on lists of route maps and schedule timetables. Other agencies have developed transit information systems to answer user questions. The user calls the agency and asks how to get from point A to point B at a certain time of day. The customer service operator types the origin and destination addresses and the desired arrival time into a computer terminal and the TIS package does all the calculations needed to plan the trip. The results are provided in textual form; written instructions are provided about the nearest train or bus stop, the route, time schedule, any transfers required, where to get off, how long the trip will take and how much will it cost (Azar, 1992, 10).

As referenced by Azar (1992, 10), a TIS is composed of several databases divided into two areas. The first is geographic based such as street networks, transit routes and stop locations, and landmark files. The second set of databases are attribute based such as schedules, fares and fare zones. When users give their origin and destination addresses, the addresses are matched against the street address files to extract the coordinates of the two addresses and determine the nearest bus (or train) stops. The optimal paths between the two stops are then calculated based on desired time of travel and vehicle schedules. The query results are textually displayed and a list of information required for a trip is shown. In several systems, the path, as well as the origin and destination are graphically displayed. Azar (1992) presents an excellent review of telephone-based transit information systems and provides a history of their use from the early 1970's to today (Azar, 1992, 18-28). Two examples of telephone-based transit information services are discussed next with special emphasis given to their capabilities.

2.2.1 New Jersey Transit Telephone Transit Information System

New Jersey Transit provides transit service for all of New Jersey, a population of approximately 7.8 million, using bus, train, subway and ferry modes. Currently, all transit information is provided by an information center which is accessed by a 800 number.

New Jersey Transit uses manual methods for providing transit information, namely route maps, road maps, and schedules. Their customer support operators use their knowledge of the system plus the printed information to provide directions on how to go from an origin to a destination. The information provided includes the routes, the mode, the fare and the schedule. They are able to answer inquires on how to get to a destination at a certain time. If the user is not satisfied with the provided route, only one alternate is reluctantly given since as an operator stated "our time is valuable."

In the determination of the best path, no landmarks are given nor are they capable of being given. The origin and destination must be given in the form of an intersection, house numbers are not accepted. In addition, exact bus locations are not given, just approximations. Bus stops are assumed to be at every major intersection, or quarter-mile of highway. Walking time to the bus stop is given by approximation, and transfers are assumed to be direct or separated by one to two blocks.

User preferences such as minimal transfers, and walking time, are reluctantly handled, while minimal waiting time is not addressable. The operators are quoted as "overworked and underpaid" and have job turnover problems. Currently, an Advanced Transit Information System is being developed for NJ Transit and will be in place by 1996. This menu-based TATIS will be able to answer any user inquiry and provide information on the topography of the walking trip (e.g., walk up a hill) as well as a display of the route chosen with a six second response time.

2.2.2 New York City Transit Authority TATIS

New York City, with a population of 7.3 million has an Transit Advanced Traveler Information System (TATIS) in place. This TATIS called On-Line Travel Information System (OTIS), has been in place since 1991, and is provided by the New York City Transit Authority.

As described by their Senior Systems Manager of Customer Services, Alan Ramlal (1994), their customer support operators undergo an eight-week training course on the transit system in New York City and then are trained three hours on how to use OTIS. Therefore, the operator is knowledgeable if there is a discrepancy between the output of OTIS and what is actually possible.

The operator enters origin and destination addresses and desired time of travel and the system responds with the best possible path using all potential modes (bus and subway). Addresses can be entered in the form of landmarks, intersections, or house numbers. If there is a misspelling or an ambiguity, a list of related feasible selections is presented. The time of travel can be based on departure or arrival.

The paths chosen can be based on minimal transfers or travel time. OTIS can handle up to six transfers. In order to obtain the next possible path, the operator only has to press one key and page up or down the list of paths. Transfers can be either with no distance or up to a distance of two to three blocks.

Walking time to the transit stop is calculated in a straight line distance and given in terms of bearing and miles, but barriers, natural and man-made are taken into account in the determination. Exact transit stop locations or subway entrances are given including which side of the street the location or entrance is on. The database representation of the street and transit network is provided by the New York City Department of Planning. Modifications and diversions are easily handled by the system.

The input and output screens are textual-based forms developed using DEMO2, a software package, and are designed to be as simple and easy to use. The output can be in the form of a list of directions, or a listing of the schedule of a route. This list or itinerary can be mailed or faxed to the user. A graphic display of the origin or destination with streets in a half-mile radius can be provided (using MAPINFO), but the overall route is not displayed. It is felt (by Alan Ramlal) that only a display of the origin and destination is needed, and that a textual list of directions is sufficient for the operator. If the system were to be used in kiosks, the interface as well as the output would have to be graphically based.

This ATIS has a response rate of six seconds using a PC LAN, called STAR-LAN developed by AT&T. The system runs on UNIX AT&T System 5 version 4 and has 16 dedicated servers and 1 backup master server. Each server is a 386 machine and contains the same database set. Four to six operators are connected to each server which allows for the fast response time. The database system software manager used by OTIS is CTREE, which was written in C programming language for 386 machines. CTREE is not a true relational database, but contains components of a network database where physical pointers have to be set. The route-finding algorithm was developed by Tidewater Consultants, and was also written in the C programming language. It is expected that this system will be connected with America On-line and allow all potential users to access the system. In addition, it is expected that OTIS will also incorporate the train routes provided by the Long Island Railroad (LIRR) and Metro North.

2.2.3 Other TATIS Systems

There are several regional transit agencies throughout the country that are using GIS-based TATIS. Virtually all major metropolitan regions have a telephone-based Transit

Information System. Usually these TIS's use text screens with no menus and limited graphical capabilities, such as Boston's MBTA's system. Transit agencies such as Houston Metro and Portland's Tri-Met are currently using TATIS systems based upon Arc/Info. In Dallas, DART uses GDS, which is produced by EDS, as its GIS base. Currently Seattle Metro is converting its TATIS from its own proprietary system TransGeo to an Arc/Info based TATIS. Several other regions are designing TATIS under the guidance and direction of the U.S. Department of Transportation. Some of them are the Bellevue Smart Traveler in Washington, the California Smart Traveler, and the Minnesota Guidestar Travlink. Schweiger (1992) surveys 67 transit organizations on their use of GIS and its applications. This article is dated and may not reflect current conditions.

2.3 Review of Network Algorithms and Representations

2.3.1 Introduction

In solving shortest path problems, static labeling methods have been the foremost approach. These methods have been divided into two categories: label setting and label correcting. The way they choose the next eligible node in a network to scan, account for the main difference. Promising results have been reported with new hybrid approaches that combine features of both techniques.

The practicality of shortest path algorithms depends on the computers they run on in addition to their data structure. The advancement in computer technology has enhanced the running time of these algorithms tremendously. Dial et al. (Dial, Glover, Karney and Klingman 1979) reports that the "evolution of efficient methods for network flow and shortest path problems uniquely demonstrates the power of computer implementation technology, properly applied, to yield gains that were not previously suspected. For example, a minimum cost network flow problem that

required several minutes to solve in 1968 can now be solved in 20 seconds, using the same general algorithm, computer and compiler" (Dial, Glover, Karney and Klingman 1979, 216). In addition Gilsinn and Witzgall (1973) found that improved implementation technology caused solution times for shortest path problems to drop from one minute to slightly more than one second, using the same general shortest path algorithm, computer and compiler.

2.3.2 Static Label Setting Techniques

The most popular label-setting algorithm is that developed by Dijkstra (1959). This algorithm finds the shortest path from the origin to another node at each iteration, labeling that node permanently with the cost of the path. This algorithm is readily understood, easy to code, and efficient for most tasks. Due to these facts, the Dijkstra algorithm is used by most GIS packages. Label-setting algorithms assume that all lengths are nonnegative and guarantee that the shortest path will be found, scanning each node only once. The computational requirements associated with finding the node with the minimum label in every step is one of their disadvantages. In addition, they are inapplicable to networks with negative arc costs. Ahuja, Mehlhorn, Orlin and Tarjan (1990) provide improvements to Dijkstra's algorithm to increase its efficiency and handle special situations.

2.3.3 Static Label Correcting Techniques

Label-correcting methods are another approach to finding the shortest path in which no label for a node becomes permanent until the algorithm terminates. These approaches, based on Bellman's (1958) and Moore's algorithms (1957) choose the next node to scan in some computationally 'inexpensive' manner without any condition associated with the value of the chosen node's label. They usually keep the list of scan eligible

(SE) nodes in a data structure that has small costs for inserting or deleting elements (Pallotino 1984). The disadvantage of these methods is that they can possibly repeatedly scan the same node, correcting its label (Ziliaskopoulos 1992).

D'Esopo and then later Pape (1980) refined the Bellman-Moore algorithm by using a double-ended queue to prioritize the order in which nodes are processed. A lucid explanation of the Bellman-Moore-d'Esopo-Pape algorithm is provided by Syslo, et al. (Syslo, Deo and Kowalik 1983, 235-241). Syslo reports that this algorithm is faster than Dijkstra's algorithm for sparse networks such as transportation networks (Syslo, Deo and Kowalik 1983, 246).

2.3.4 Static Hybrid Networks

Glover et al. (Glover, Klingman, Phillips and Schneider 1985) (Glover, Glover and Kingman 1984) combined the advantages of both methods into the threshold or Partitioning Shortest Path Algorithm. In this algorithm, the SE list is partitioned into two parts: an active part and a passive one (NEXT) and an active one (NOW). While only nodes from the NOW list are scanned, the nodes are inserted in the NEXT list. Whenever the NOW list is empty, a number is computed (threshold value), and all the nodes in the NEXT list with labels less than the threshold value are transferred to the NOW list. This algorithm is a hybrid model, with both label setting and label correcting characteristics. By setting the threshold value equal to the minimum label in the NEXT list, only one node will be transferred to the NOW list after every iteration, thus degenerating the algorithm to a label setting one. On the other hand, if the threshold value is set to infinity, all nodes will be transferred, and the algorithm will function as label-correcting (Kandru 1993).

Methods that do not use labeling have also been proposed based on linear algebra methods. These methods use a matrix representation of the arc lengths to find

the shortest paths between all origins and destinations simultaneously and are modifications of the work of Dantzig (Dantzig 1957) (Dantzig 1960) (Dantzig 1967) and Floyd (Floyd 1962). Syslo, et. al (Syslo, Deo and Kowalik 1983, 242-246) discuss the algorithms and their implementation.

2.3.5 Stochastic (Probabilistic) Networks

The algorithms discussed previously are deterministic since they assume that the cost of progressing along each arc remains constant and is predetermined. However, on transit networks there is uncertainty as to travel and arrival times due to factors such as traffic congestion, demand en route and malfunctions. Therefore a true representation of transit networks should be probabilistic or stochastic. Examples of stochastic shortest path algorithms are presented by Frank (1969), Eiger, Mirchandani and Soroush (1985), and Laporte, Louveaux and Mercure (1992).

2.3.6 Time Dependent Stochastic Networks

In addition, travel time on transit networks is constrained by a schedule. Transit vehicles are dispatched by a schedule and the travel time of a passenger depends on the time of his arrival at the bus stop in relation to the time schedule. Therefore travel time on transit networks is time-dependent in addition to being stochastic. The shortest path problem with time-dependent travel times was first considered by Cooke and Halsey (1966). Hall (1986) presents a method for determining the least expected travel time between two nodes in a network with travel times that are both random and time-dependent. This method uses travel time probability functions rather than expected travel times alone and shows that standard shortest path algorithms cannot be used to solve this type of problem. He notes that "the optimal 'route choice' is not a simple path but an adaptive decision rule. The best route from any given node to the final

destination depends on the arrival time at that node. Because the arrival time is not known before departing the origin, a better route can be selected by deferring the final choice until later nodes are reached" (Hall 1986, 182). This method suffers from long computational time compared to the standard shortest path algorithms but is the most realistic. Kaufman and Smith (1993) present conditions under which shortest path algorithms for deterministic networks can be used to solve time-dependent stochastic problems and provide modifications of Hall's algorithm.

2.3.7 Multiple Paths (K-paths)

In many situations, we may be interested in not only the shortest path, but the k shortest path between an origin and a destination. Two versions have been developed to solve this problem. The first version allows no path to contain repeated nodes, or no loops are allowed. Algorithms for this type of problem have been developed by Yen (1971) and improved by Lawler (1972) and Perko (1986). This approach, as presented by Lawler (Lawler 1972, 404-405) first finds the shortest path without loops and then deletes from the network all nodes and arcs along this path. The next shortest path is then found using the remaining network. This procedure is repeated until no paths are possible.

The second version of finding k paths from the origin to the destination allows loops or repeated nodes in a path. As described by Dreyfus (1969), Hoffman and Pavley (1959) proposed the earliest algorithm for this version. First, they determine the shortest path from all initial nodes to a specified destination. Then they define a "deviation from the shortest path to be a path that coincides with the shortest path from its origin up to some node j on the path, then deviates directly to some node k to the destination node via the shortest path through k. All deviations from the shortest path

between the specified origin and destination are determined evaluated and compared, and the best noted" (Dreyfus 1969, 404).

Another algorithm was developed by Pollack (1961) and can be used when k is small. In this approach the shortest path is found, each link length is in this route is set to infinity one at a time and every time the shortest path for the resultant network is found. The best of these paths is the second best path.

2.3.8 Passenger choice model

In addition to determining the shortest path, one must consider how passengers respond to route-choice decisions. Chriqui and Robillard (1975) based their study on the assumption that passengers arrive at a bus stop at random and select a bus by trying to minimize their expected total travel time. Their algorithms, as observed by Marguier and Ceder (1984) are "valid only for the case of exponential headways" (Maguier and Ceder, 1984, 2). Marguier and Cedar (1984) also state that an optimal route-choice strategy might be to disregard an arriving slow bus and wait for a fast bus versus the intuitive rule in which passengers board the routes in proportion to their frequencies. In addition "for large headways (more than 12 minutes) passengers tend to time their arrivals at the bus stops to the arrival times of the required buses" (Maguier and Ceder, 1984, 2).

2.4 Previous Research on GIS based TATIS

There are several notable previous works of research which have contributed greatly to the author's current research. Here, those applications are examined and some of their shortcommings are pointed out.

Azar (1991) explores the feasibility of using a GIS to duplicate functions of a TIS or in his words a "Passenger Information System (PIS)". He examines the

differences and commonalties between a GIS and a TIS, and defines the technological improvements, database developments, and system design strategies required for integrating TIS and GIS technologies. The steps he used in the development of a prototype TIS using GIS were:

1) Encoding the road network

2) Encoding the transit routes

3) Encoding connectivity, transfers and time restrictions

4) Pre-assigning streets to bus stops

5) Matching addresses of the origin and destination

6) Finding the shortest path (using Dijkstra's algorithm)

7) Packaging the steps into an application

This model assumed "that people always begin their transit trip at the nearest bus stop to their origin and then get off at the nearest stop to their destination" (Azar, 1991, 58). Some of the criticisms of the model were that it assigned links to the nearest stop (every address had only one stop close to it), and it did not address the issue of time schedules, only the enroute travel time. In addition, since the model was based on an older version of GIS software (Arc/Info version 5), which did not incorporate dynamic segmentation, it had a problem of addressing shared arcs between multiple routes. This thesis also pointed out the lack of a permanent node attribute table and the amount of time required to enter and exit the various modules of Arc/Info. The time required by the model to determine one route was almost 130 seconds on a PC and 40 seconds on a workstation. Of that 40 second time, 7.6 seconds were actually used to find the optimal path. Most of the remaining time was spent on input/output and entering/exiting modules. This thesis also points out the problems in using Arc/Info's macro language (AML) in that the software interprets one line of code at a time, versus compiled programs.

Hancock and Abkowitz (1992) present the development and implementation of a geographic information system for customer service for the Metropolitan Transit Authority in Nashville, Tennessee. The use of Arc/Info was ruled out based upon the work of Azar (1991). Instead, customization of a GIS package developed by Vanderbuilt University called GRAPHNET was implemented. GRAPHNET contained many of the necessary routing algorithms and geographic features and was renamed TRANSYS/DACO. The software was composed of three modules: scheduling, locating, and routing.

The scheduling module provided answers to inquires into the printed timetable. The locating module provided options for locating routes, stops, streets or landmarks. The routing option provided complete answers to questions about going from an origin to a destination at a given time. A complete interary is given and is similar to that provided by the NYCTA, with a complete graphical representation of the path being described. The roadway database was developed from TIGER/Line files, bus route files were stored as LOTUS 123 files and schedule information were maintained as WordPerfect documents. Landmarks were also incorporated into the system. Dijkstra's algorithm is used to identify the optimal path to and from a bus stop. In addition, TRANSYS/DACO determines and saves all solutions that have bus stops within 1/2-mile of the origin and destination. The solution with the minimum overall time is presented initially on the screen. Other solutions can be toggled through.

This system appears very impressive and comprehensive. Missing from the report, though, are indications as to performance measures and how the basic algorithms of GRAPHNET work especially as regard to computation of multiple paths and transfers. This fact appears to be a major shortcoming. According to a conversation with Ms. Hancock, her additions to the already prepared GRAPHNET

software were called "extremely minor." Also missing is incorporation of user preferences into the system, though this could easily be accomplished.

Grayson (1993) presents a thorough and complete method for analyzing and displaying the travel accessibility provided by a public transit system. His thesis highlights the advantages of using a linked GIS and a Relational Database Management System (RDBMS), Oracle, for transportation network analysis and a disaggregate, multipath analysis of transit accessibility. Accessibility answers the question, "How accessible is one point to another by bus provided a particular set of routes is available at a certain time of day?". His implementation relied on Arc/Info GIS, Oracle RDBMS, and FINDPATH, a custom special-purpose program written in C with embedded SQL (Structured Query Language) statements written to find multiple paths in a transit network. Grayson uses three levels of networks: geographic (roads), transit (routes and stops) and the "supernetwork" (an abstraction of the transit route that facilitates network analysis). His data scheme uses sixteen tables which store nine types of data. Also implemented is a menu-based user-interface.

Grayson uses the Bellman-Moore-d'Esopo-Pape (1980) label correcting algorithm to find the shortest path and Yen's Algorithm (1971) for finding k multiple paths. He explains the method for choosing the k shortest path, "The program identifies significantly different paths between an origin and destination by first finding the shortest path between them. Then, all the arcs that lie on this path are temporarily excluded form the network, and the shortest path is found again for this reduced subnetwork, until either it becomes impossible to go from the origin to the destination because too many arcs have been eliminated or the program reaches a specified limit on the maximum number of paths to be found (5 by default)" (Grayson, 1993, 111). This method of obtaining multiple paths is a drawback and is not feasible in the analysis of a TATIS. In a TATIS several paths can be obtained by using the same route. The

prototype in this thesis uses the same route for multiple paths and has no limit on the maximum number of paths found.

His system assumed that the transit service is frequent and highly variable and therefore, make schedules irrelevant (Grayson, 1993, 59). In addition, his prototype used buses only and used headway times based upon the time of day (Early AM, AM Peak, midday, PM peak, evening) and type of day (weekday, Saturday, or Sunday). Also assumed is that a bus has a constant speed for its entire run (Grayson, 1993, 66). Provided for, though not incorporated in this prototype, are the costs or negative benefits from certain operations, such as transfers. A critical problem with the prototype as noted by Grayson is that "the prototype does not address the fact that many transit trips involve walking as well as riding the transit vehicle. The network model does not allow passengers to transfer routes by walking between nearby bus stops" (Grayson, 1993, 187). A recommendation is to provide for "walking links" that connect nearby stops. Missing from this thesis are performance measures as to query of the prototype. It is assumed that the time involved is significant since one of the functions is that the transit network must be copied to the database package for analysis which would require some time. It is understood, though, that for purposes of displaying accessibility, computational time is not a critical factor, as compared to hand computations.

2.5 Capabilities of the TATIS Prototype

The TATIS prototype produced was designed to incorporate certain key features not present in previous implementations. The first feature is its design. The prototype was created entirely using the capabilities of a known off-the-shelf GIS, Arc/Info (an improvement over Hancock and Abkowitz (1992) and Grayson (1993)). This was done for two reasons, the first was to show the capabilities and limitations of using a GIS

alone; the second was that certain functions were present in the GIS, such as path-finding, dynamic segmentation, spatial analysis, and input/output functions

The second key feature is the incorporation of walking distances between transfers (an enhancement over Azar (1991), Hancock and Abkowitz (1992), and Grayson (1993)) . Often transit paths are given or computed along routes that intersect or have a transfer link. In many situations a shorter route, (lengthwise as well as timewise) as well as additional routes, could be determined if the radius around the transfer point was increased up to a 15 minute walking distance. Instead of creating transfer links between all possible transfer points, and creating an algorithm to find the shortest path along the transit links and the transfer links, a different method was used. This method did not use links in the determination of the shortest path, as most other algorithms, but uses transit stops as the determining factor in the creation of transit paths. At the present, the maximum number of transfers is two, but could increase with more development. It is assumed that most people are not willing to make more than two transfers, especially in adverse weather conditions. In addition, the transit paths found were those that were within 15 minutes walking distance of the origin and destination point, not using the closest bus stop to the origin/destination. This factor allowed for more flexibility and more choices to be determined.

The third key feature of this prototype is the incorporation of transit time schedules into the determination of the shortest transit path using time as the criteria (a shortcomming of Azar (1991)and Grayson (1993)). The method used is unique. First, all possible paths are found using the procedure described later. Then their overall travel time is computed taking all parts of the transit trip into account (even walking across the street for a 0 distance transfer). The paths are then sorted based on this overall travel time. Those paths which were indirect or followed illogical paths had longer travel times and would end up being at the bottom of the list of possible paths.

The fourth key feature of this prototype is the ability to provide multiple transit paths from an origin to a destination. This is possible due to the structure of the algorithm described previously. The first four paths are displayed in the output menus. One of the assumptions made is that the number of paths should be limited to four to prevent information overload on the user.

An additional feature of this prototype is the ability to allow for user preferences which find all possible paths that meet the user's criteria. Providing this flexibility is necessary for the user to see what options are available and what possibilities are not (e.g., that all potential paths involve two transfers). None of the previous research had incorporated this feature.

Another key feature of this prototype is the output provisions and their flexibility. A graphical display with the roadway network, transit routes and highlighted transit paths is provided. In addition, a tabular itinerary is given for each option or path. Both items are in a windows form. Additional options such as zooming are provided in the form of graphical menus.

Performance measures are provided for this prototype using units of computational time. Several sample runs are performed creating a variety of paths. The results are displayed, discussed and conclusions are derived. Hancock and Abkowitz (1992) as well as Grayson (1993) do not provide performance measures, though it would be beneficial if they did.

CHAPTER 3

TATIS PROTOTYPE

3.1 Introduction

This chapter describes in general terms, the steps involved in performing an origin-destination trip query using the prototype developed in this research. After this overview, the design decisions and assumptions that shaped this methodology are examined. A detailed description of these methods and their implementation is given in Chapter 5.

3.2 Description of Prototype

This research was conducted in several stages. The first task was data preparation. This task involved obtaining a suitable representation of the roadway network for the study area. Because no suitable, ready-made representation of the transit network was available, methods were developed and implemented for converting text-based representations of transit networks to a digital form suitable for analysis. In addition, printed time schedules were converted into an appropriate database structure.

The second and crucial step was the development of the network analysis algorithm and its implementation. This algorithm was developed using only the tools provided by the GIS software and did not use any of the standard network analysis algorithms.

The third step was the determination of travel time along each path and aggregating them based on user preferences. Fourth was the development of a graphical display and appropriate schedule of each path. The fifth step was the design and implementation of a menu-driven user interface that allowed for the entry of data,

the display and inquiry of the data, and its output. Finally, this tool was used for sample query analyses that highlight the advantages of using a GIS in a Transit Advanced Traveler Information System (TATIS). Chapter 5 presents a full description of the algorithms used.

The prototype is designed to perform calculations and manage the data for a variety of origin-destination trip queries. These queries allow the user to answer the question, "How do I go from an origin A to a destination B and get there by time T or leaving at time T, with my preferences?". Each analysis may be very diverse, covering different geographic areas, transit systems, times of the day and varying user preferences.

The methodology used here for finding transit paths requires a substantial amount of data to be constructed and processed. This data includes: 1) a base map of streets complete with addresses; 2) geographic descriptions of the transit routes that indicate what streets they follow; 3) geographic descriptions of the transit stops of the transit routes; 4) time table schedules of each transit route; 5) a connectivity matrix of the transit routes. Steps 1 through 3 are similar to those used by Grayson (1993, 39-40); steps 4 and 5 are further enhancements. Steps 1 through 5 are presented below.

Step One: Geographic Base Map Preparation

The first step is preparing a suitable geographic base map. This map must be geocoded by street addresses, allowing a location in the street network to be identified by giving a street address or intersection. For this prototype, the street arcs in the U.S. Bureau of the Census TIGER/Line file for Union County, New Jersey were used as the raw base map. This base map was not augmented to provide for missing zip codes, arcs and address ranges. It was felt that the necessity for augmentation of this base map was small in comparison for the need to have a proper algorithm that used the information provided by the base map. In other words, instead of "wasting time" on perfecting a

base map that could easily be replaced by an enhanced TIGER/Line file (many are available in the marketplace), the methodology and algorithm itself would be concentrated on.

Steps Two and Three: Building Transit Routes and Transit Stops

A description of the transit routes serving Union County in New Jersey was built. Since transit data was unavailable from New Jersey Transit a representative transit network prototype was developed. The development of the transit network used the research of Grayson (1993) as a guide. A set of route description files for each route included was built. The route-description files are ASCII text files that describe the path a transit route takes. For each route a stop description file was generated. This stop file is also an ASCII text file that contains the address locations of each transit stop. Using a set of macros written specifically for this application, these two types of files are converted into an internal representation of the routes, with a database structure, in the Arc/Info GIS.

Step Four: Transit Route Time Tables

Using the internal representation of the transit routes, the position of each stop in relation to each other was obtained. This information was used in generating a schedule file for each route. The schedule file contains anticipated times of arrival at each stop in the route during the schedule day. This file was created in Microsoft Excel, converted into an ASCII text file format and then converted into a suitable Arc/Info database file.

Step Five: Transit Route Connectivity Matrix

The final step in the data preparation stage was the creation of a transit connectivity matrix. This matrix compares each transit route with every other route in the system to determine if they cross or come together within a distance of a quarter-mile.

facing 30

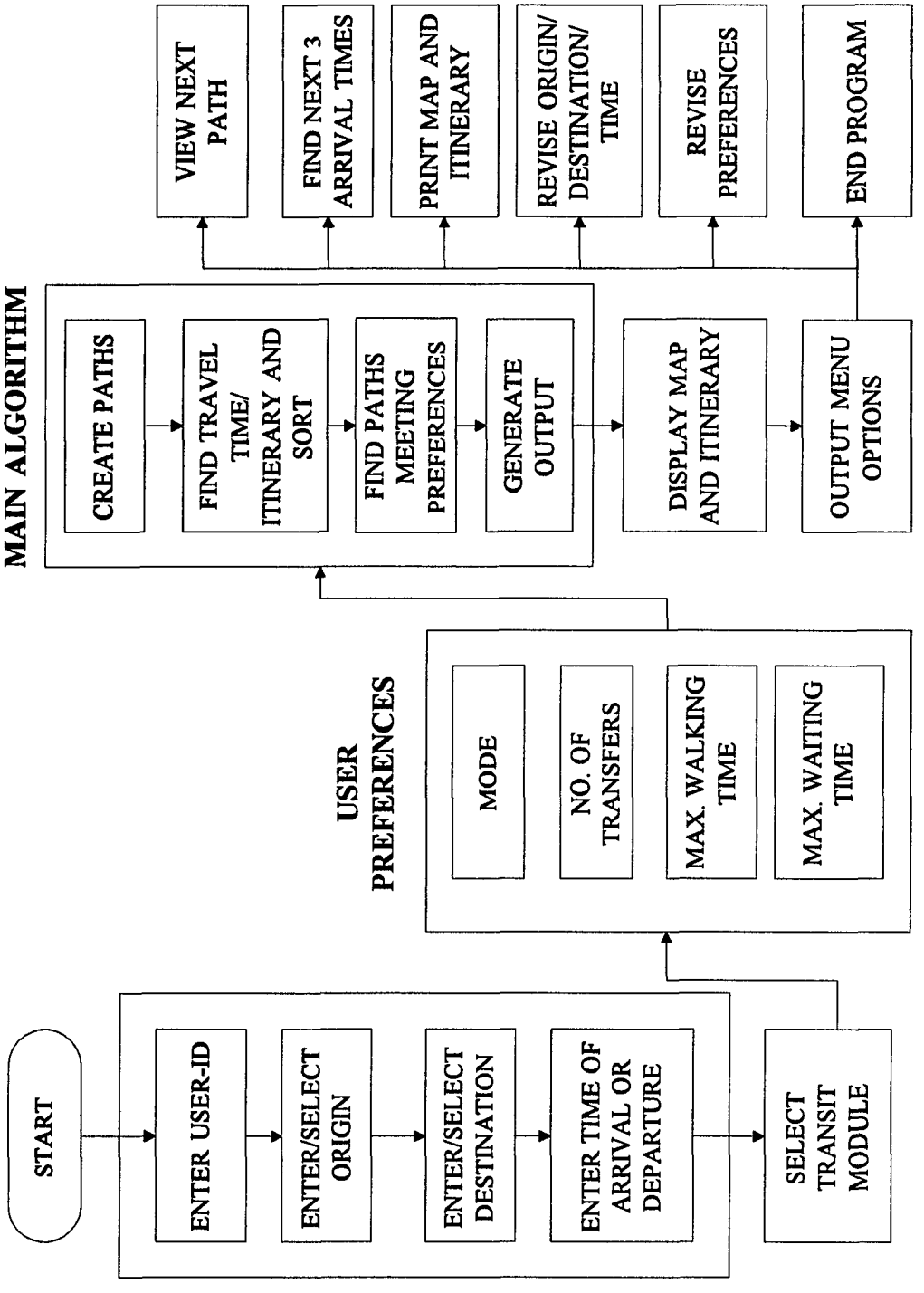


Figure 3.1 TATIS Module Flowchart

Now that the data preparation is complete, all the necessary information to begin the analysis can be entered. The information needed is both general to an ATIS, such as the origin address, the destination address, the time of departure or arrival, and that specific to the transit module of an ATIS, such as mode and transfer preference, maximum walking time, and maximum waiting time.

The analysis is divided into four steps, and are further covered in Chapter 5. The first step is to determine all the possible transit paths from the origin to the destination. In this process, we use the fundamental capabilities of GIS to solve this problem. This step relies on the transit route stops and the transit connectivity matrix. Next, the travel time for each path is found, along with the schedule values. Incorporated into this step is the computation of walking, waiting and transfer times as well as travel time. The way these values are computed varies based on the criteria of "arrival" or "departure" time. The paths are then sorted based on total travel time. The third step is to select those paths that meet the user's preferences. Finally, all the output files are generated based on the analysis. These files include the schedule itinerary, the path graphical representation, and the menus that interface with the display and the itinerary.

After the analysis or the main algorithm is completed, a graphic representation of the paths is presented along with a display of the itinerary for all the available paths. Menus are provided for displaying other routes, for zooming options, for output options, and for revisions. Chapter 5 contains an in-depth account of each menu. Chapter 6 provides an example of the prototype in action using figures of the entire analysis process. Figure 3.1 presents an overview of the processes involved in answering an origin-destination query.

3.3 Design Decisions

Many design decisions shaped the architecture of the prototype. In this section the motivations behind the choices for the prototype's tools, data structures and algorithms are explored. The following core design decisions will be discussed:

1) The architecture for the application
2) Network Structure
3) Properties of Transit Network and Riders
4) Finding Multiple Paths
5) Schedule Attributes
6) Other Simplifications

3.3.1 Application Architecture

One of the predominate philosophies behind this thesis is that an application architecture should rely on a platform of ready-made tools in order to minimize development time. The tasks conducted for this thesis were well suited to these tools. This project required functionality for:

1) storing and retrieving geographic and non-geographic data
2) graphic display of geographic data
3) analysis of transit networks.

Due to the large amounts of data and computation in this project, the prototype was designed to run on a workstation-class computer instead of a personal computer. The data management and graphic display tasks call for using a geographic information system. Arc/Info by ESRI, Inc. was selected since it is the unofficial "standard" in GIS software. Arc/Info contained most of the necessary functions in order to proceed with the tasks. As with the case with standard software, situations arise where a custom tool is needed. The user is then left to search for the tool or create it himself. This was the

case with the network analysis of this project. While Arc/Info possesses a network analysis package, that package proved inadequate for this prototype. Since a transit network that is an abstraction of a roadway network is being dealt with, a tool is needed that could analyze that abstraction. Additionally, multiple paths needed to be found and stored. Arc/Info can only find the shortest path. This required a custom-built path-finding tool. This tool was created using the macro language provided by Arc/Info, known as AML, and is called CREATE_PATHS. It is understood that a more efficient approach would be to use a programming language such as C, but it is felt that once the methodology is complete, converting it to C would not be a problem. A more complete explanation of the algorithms in the CREATE_PATHS program follows in Chapter 5.

3.3.2 Network Structure

In this paper, two types of networks are referred; the geographic network (a geographic representation of roads and intersections) and the transit route network (a representation of the transit routes and stops operating over the street network).

The geographic network describes where a street is located in a geographic coordinate system, how long it is, and where it intersects other streets. A particular physical location can only be occupied by one arc, and is bidirectional. Nodes occur wherever two or more arcs meet. The specific streets on which buses travel can be identified using the geographic network. In the transit network multiple arcs may coexist in the same physical space and the arcs are unidirectional. These factors allow us to represent multiple transit routes that use the same streets, either in the same or opposite directions. In the transit network, transit stop locations are critical since they are the points where users enter and exit the system. A train may pass by a user's house, but the user has to traverse two miles to access the train. Therefore, the multiple path

algorithm uses transit stops in determining where to arrive, transfer and depart the system.

3.3.3 Properties of the Transit Network and Riders

This prototype incorporates several simplifying assumptions about the transit system. One class of assumptions regards the behavior of the transit system and its riders. The prototype assumes that passengers will time their arrival times according to a schedule, and that transit service is not frequent. In addition, it is assumed that the prototype does not minimize expected waiting time for transfers. Under these conditions, the path-choice decision is deterministic and time-dependent. Since travel in the prototype's transit network is constrained to a schedule, many of the standard algorithms do not apply. Since the transit schedules vary during the day, the optimal route varies according to the time of entry into the system. Therefore our decision to first find all possible paths, then find their overall travel times and then rank them by this amount is sufficient.

3.3.4 Finding Multiple Paths

In determining multiple paths, this prototype finds all possible paths and then ranks them according to overall travel time. A new path is one that can be a deviation from a previous path. Finding significantly different paths between an origin-destination pair was not a concern. Those paths with minor or indirect deviations would be at the bottom of the list of paths when travel time is computed for each path and therefore be excluded. Grayson's (1993) approach to finding multiple paths limits the possibilities of traversing the transit network. Especially of concern is the exclusion of all arcs of a path from the transit network whereby a new path is to be found. There are situations

where part of a transit route could be used for a one or a two transfer combination and possibly reduce travel time.

However, even though this prototype finds all variations of a path going from an origin to a destination, a restriction is placed on the algorithm. This restriction states that for a one transfer route-pair, no two transfer combinations are possible that are further away than the transfer point of the one transfer route-pair. For example, if a path is determined containing two routes (i.e., origin, and destination route, a one transfer route-pair), a path containing an intermediate route between the origin and destination route would have to intersect these two routes before the transfer point of the one transfer route-pair. This restriction also prevents two-transfer combinations that loop along itself. A full description of this algorithm is given in chapter 5.

3.3.5 Transit Schedule Attributes

The determination of travel time in this prototype relies on published timetables. Since data was unavailable for the transit routes in Union County, New Jersey, assumed schedules were generated. These time tables were based on typical transit service in which frequency is greatest during the peak periods and is reduced during off peak times. Each route schedule was varied in departure time to prevent all buses leaving at the same time. Since the transit routes are represented in Arc/Info format, the distance of each transit stop along the route referenced from the origin can be obtained. Once these distances (measures or mileposts) are known and the time schedules for selected transit stops are retrieved, the time schedule for each transit stop can be obtained by linearly interpolating between the known transit stops. Microsoft Excel was used for this process. Since Excel converts time into decimal days, all time computations are in terms of decimal days. This linear interpolation process assumes that travel between

time points is at a constant speed which is often not the case. This process and provisions for travel when there is a change in day are discussed in Chapter 4.

3.3.6 Other Simplifying Assumptions

This prototype is adapted to be non-mode specific. Both street and rail networks are provided in the relevant TIGER files and are therefore applicable. The transit network data structure and representation allow for many modes to be represented since they operate on a fixed schedule and have fixed points of ingress and egress.

A simplification to this prototype is the nonincorporation of fare policies. The incorporation of a fare system into this prototype would encompass some slight modifications to the transit stop database and to the algorithms as well. Chapter 7 addresses this area of future research.

Another simplification to the prototype is the restriction of the number of transfers to two. If the number of transfers required was increased, the computational effort to solve this problem would be monumental (i.e. the number of possible combinations would be very large). For example if the number of transfers was increased to three (an increase of one), the computational time would approximately increase by n^2 times, where "n" is the number of possible intermediate paths found between an origin and destination.

CHAPTER 4

DATA ORGANIZATION

4.1 Introduction

This chapter explores the structure and organization of the data used by the prototype. The next chapter discusses the programs and processes that build and analyze this data. The prototype creates and uses data stored in three different ways: text files stored in Unix directories, geographic data sets packaged as Arc/Info "coverages", and tabular data stored in the Info database. The coverages, Info tables and most of the text files are used to prepare and display a geographical representation of the transit network. These three forms are also used for network analysis. Each of these different types of data will be described.

4.2 Transit Route Descriptions

This prototype uses a specialized method for developing transit route descriptions within the Arc/Info GIS. This method is described in detail in the next chapter. Two types of text files describe a single route: a path file and a stop file. Each direction a route travels is treated as a separate route (e.g. Route 88 Inbound and Route 88 Outbound), as are any variations of the service that require travel on different streets.

The internal naming convention used to describe routes was modified from Grayson (1993, 72-75) since it allowed for a comprehensive and adaptable description. The routes are categorized by a mode number, base route number, a variation number, a direction, and time of operation. The mode character and number are additional features not present in Grayson's research. The routes are identified in the form **zMRRRVVDT**, where

z is a literal character indicating transit mode (e.g. b = Bus, t = Train)
M is a one-digit transit mode number (e.g. 1 = Bus, 2 = Train)
RRR is a three-digit route number, padded with leading zeroes,
VV is a two-digit variation number, padded with a leading zero,
D is a one-digit number indicating direction (e.g. 1 = Inbound, 2 = Outbound)
T is a one-digit number indicating type of day (e.g. 1 = Weekday, 2 = Weekend)

For example, Bus Route 20 Inbound, variation 8, operating during weekdays has the identification b10200811. This representation provides for nearly all of New Jersey Transit's bus and rail operations. Modifications in the length of each item can be made easily, but must be consistent throughout the entire transit network. This identifier is required for numbering routes in the Arc/Info route system data model, described below. The duplication in mode identifiers is needed both when a numeric value of the route is required and when routes need to be selected based upon mode.

The path file is an ASCII text file describing the path a transit route takes between its first and last stops. The description takes the form of a sequence of point locations. These locations are listed as either street intersections (e.g. Central Ave / North Ave") or street addresses (e.g. "201 Main St") through which a bus route must pass. This path file is in the form of an AML file (AML is the macro script language that Arc/Info uses) and an example is provided in Appendix A and is discussed further in Chapter 5. The route creation procedure finds all the street links on the shortest path between the listed locations and includes them in the route in the proper sequence.

In addition to the path the route follows, the prototype requires the location of the transit stops. These are listed in the stop file. The locations use the same address format as the path files, but is structured in a different method. The stop files are ASCII text files which contain the addresses, plus other information relative to the bus stop, such as an unique stop-id, whether a shelter is present, etc. This stop file is then converted to an Info table which is then used within the geocoding module to obtain a point coverage for the route. Additional attributes specific to the stop such as the x and

facing 38

Table 4.1
Typical Timetable Worksheet

STOP-ID	10050111001	10050111002	10050111003	1.005E+10	1.005E+10	1.005E+10	1.005E+10	1.005E+10	1.005E+10	1.005E+10	1.005E+10	1.005E+10	1.005E+10
MEASURE	0	0.24	0.42	0.84	1.08	1.92	2.28	2.82	3.42	4.02	4.56	4.8	4.98
RUN 1	5:00 AM	5:01 AM	5:02 AM	5:05 AM	5:06 AM	5:11 AM	5:13 AM	5:16 AM	5:20 AM	5:24 AM	5:27 AM	5:28 AM	5:29 AM
2	5:30 AM	5:31 AM	5:32 AM	5:35 AM	5:36 AM	5:41 AM	5:43 AM	5:46 AM	5:50 AM	5:54 AM	5:57 AM	5:58 AM	5:59 AM
3	6:00 AM	6:01 AM	6:02 AM	6:05 AM	6:06 AM	6:11 AM	6:13 AM	6:16 AM	6:20 AM	6:24 AM	6:27 AM	6:28 AM	6:29 AM
4	6:20 AM	6:21 AM	6:22 AM	6:25 AM	6:26 AM	6:31 AM	6:33 AM	6:36 AM	6:40 AM	6:44 AM	6:47 AM	6:48 AM	6:49 AM
5	6:40 AM	6:41 AM	6:42 AM	6:45 AM	6:46 AM	6:51 AM	6:53 AM	6:56 AM	7:00 AM	7:04 AM	7:07 AM	7:08 AM	7:09 AM
6	7:00 AM	7:01 AM	7:02 AM	7:05 AM	7:06 AM	7:11 AM	7:13 AM	7:16 AM	7:20 AM	7:24 AM	7:27 AM	7:28 AM	7:29 AM
7	7:15 AM	7:16 AM	7:17 AM	7:20 AM	7:21 AM	7:26 AM	7:28 AM	7:31 AM	7:35 AM	7:39 AM	7:42 AM	7:43 AM	7:44 AM
8	7:30 AM	7:31 AM	7:32 AM	7:35 AM	7:36 AM	7:41 AM	7:43 AM	7:46 AM	7:50 AM	7:54 AM	7:57 AM	7:58 AM	7:59 AM
9	7:45 AM	7:46 AM	7:47 AM	7:50 AM	7:51 AM	7:56 AM	7:58 AM	8:01 AM	8:05 AM	8:09 AM	8:12 AM	8:13 AM	8:14 AM
10	8:00 AM	8:01 AM	8:02 AM	8:05 AM	8:06 AM	8:11 AM	8:13 AM	8:16 AM	8:20 AM	8:24 AM	8:27 AM	8:28 AM	8:29 AM
11	8:15 AM	8:16 AM	8:17 AM	8:20 AM	8:21 AM	8:26 AM	8:28 AM	8:31 AM	8:35 AM	8:39 AM	8:42 AM	8:43 AM	8:44 AM
12	8:30 AM	8:31 AM	8:32 AM	8:35 AM	8:36 AM	8:41 AM	8:43 AM	8:46 AM	8:50 AM	8:54 AM	8:57 AM	8:58 AM	8:59 AM
13	8:45 AM	8:46 AM	8:47 AM	8:50 AM	8:51 AM	8:56 AM	8:58 AM	9:01 AM	9:05 AM	9:09 AM	9:12 AM	9:13 AM	9:14 AM
14	9:00 AM	9:01 AM	9:02 AM	9:05 AM	9:06 AM	9:11 AM	9:13 AM	9:16 AM	9:20 AM	9:24 AM	9:27 AM	9:28 AM	9:29 AM
15	9:30 AM	9:31 AM	9:32 AM	9:35 AM	9:36 AM	9:41 AM	9:43 AM	9:46 AM	9:50 AM	9:54 AM	9:57 AM	9:58 AM	9:59 AM
16	10:00 AM	10:01 AM	10:02 AM	10:05 AM	10:06 AM	10:11 AM	10:13 AM	10:16 AM	10:20 AM	10:24 AM	10:27 AM	10:28 AM	10:29 AM
17	10:45 AM	10:46 AM	10:47 AM	10:50 AM	10:51 AM	10:56 AM	10:58 AM	11:01 AM	11:05 AM	11:09 AM	11:12 AM	11:13 AM	11:14 AM
18	11:30 AM	11:31 AM	11:32 AM	11:35 AM	11:36 AM	11:41 AM	11:43 AM	11:46 AM	11:50 AM	11:54 AM	11:57 AM	11:58 AM	11:59 AM
19	12:00 PM	12:01 PM	12:02 PM	12:05 PM	12:06 PM	12:11 PM	12:13 PM	12:16 PM	12:20 PM	12:24 PM	12:27 PM	12:28 PM	12:29 PM
20	12:30 PM	12:31 PM	12:32 PM	12:35 PM	12:36 PM	12:41 PM	12:43 PM	12:46 PM	12:50 PM	12:54 PM	12:57 PM	12:58 PM	12:59 PM
21	1:00 PM	1:01 PM	1:02 PM	1:05 PM	1:06 PM	1:11 PM	1:13 PM	1:16 PM	1:20 PM	1:24 PM	1:27 PM	1:28 PM	1:29 PM
22	1:45 PM	1:46 PM	1:47 PM	1:50 PM	1:51 PM	1:56 PM	1:58 PM	2:01 PM	2:05 PM	2:09 PM	2:12 PM	2:13 PM	2:14 PM
23	2:30 PM	2:31 PM	2:32 PM	2:35 PM	2:36 PM	2:41 PM	2:43 PM	2:46 PM	2:50 PM	2:54 PM	2:57 PM	2:58 PM	2:59 PM
24	3:00 PM	3:01 PM	3:02 PM	3:05 PM	3:06 PM	3:11 PM	3:13 PM	3:16 PM	3:20 PM	3:24 PM	3:27 PM	3:28 PM	3:29 PM
25	3:30 PM	3:31 PM	3:32 PM	3:35 PM	3:36 PM	3:41 PM	3:43 PM	3:46 PM	3:50 PM	3:54 PM	3:57 PM	3:58 PM	3:59 PM
26	4:00 PM	4:01 PM	4:02 PM	4:05 PM	4:06 PM	4:11 PM	4:13 PM	4:16 PM	4:20 PM	4:24 PM	4:27 PM	4:28 PM	4:29 PM
27	4:30 PM	4:31 PM	4:32 PM	4:35 PM	4:36 PM	4:41 PM	4:43 PM	4:46 PM	4:50 PM	4:54 PM	4:57 PM	4:58 PM	4:59 PM
28	5:00 PM	5:01 PM	5:02 PM	5:05 PM	5:06 PM	5:11 PM	5:13 PM	5:16 PM	5:20 PM	5:24 PM	5:27 PM	5:28 PM	5:29 PM
29	5:20 PM	5:21 PM	5:22 PM	5:25 PM	5:26 PM	5:31 PM	5:33 PM	5:36 PM	5:40 PM	5:44 PM	5:47 PM	5:48 PM	5:49 PM
30	5:40 PM	5:41 PM	5:42 PM	5:45 PM	5:46 PM	5:51 PM	5:53 PM	5:56 PM	6:00 PM	6:04 PM	6:07 PM	6:08 PM	6:09 PM
31	6:00 PM	6:01 PM	6:02 PM	6:05 PM	6:06 PM	6:11 PM	6:13 PM	6:16 PM	6:20 PM	6:24 PM	6:27 PM	6:28 PM	6:29 PM
32	6:15 PM	6:16 PM	6:17 PM	6:20 PM	6:21 PM	6:26 PM	6:28 PM	6:31 PM	6:35 PM	6:39 PM	6:42 PM	6:43 PM	6:44 PM
33	6:30 PM	6:31 PM	6:32 PM	6:35 PM	6:36 PM	6:41 PM	6:43 PM	6:46 PM	6:50 PM	6:54 PM	6:57 PM	6:58 PM	6:59 PM
34	6:45 PM	6:46 PM	6:47 PM	6:50 PM	6:51 PM	6:56 PM	6:58 PM	7:01 PM	7:05 PM	7:09 PM	7:12 PM	7:13 PM	7:14 PM
35	7:00 PM	7:01 PM	7:02 PM	7:05 PM	7:06 PM	7:11 PM	7:13 PM	7:16 PM	7:20 PM	7:24 PM	7:27 PM	7:28 PM	7:29 PM
36	7:30 PM	7:31 PM	7:32 PM	7:35 PM	7:36 PM	7:41 PM	7:43 PM	7:46 PM	7:50 PM	7:54 PM	7:57 PM	7:58 PM	7:59 PM
37	8:00 PM	8:01 PM	8:02 PM	8:05 PM	8:06 PM	8:11 PM	8:13 PM	8:16 PM	8:20 PM	8:24 PM	8:27 PM	8:28 PM	8:29 PM
38	8:30 PM	8:31 PM	8:32 PM	8:35 PM	8:36 PM	8:41 PM	8:43 PM	8:46 PM	8:50 PM	8:54 PM	8:57 PM	8:58 PM	8:59 PM
39	9:00 PM	9:01 PM	9:02 PM	9:05 PM	9:06 PM	9:11 PM	9:13 PM	9:16 PM	9:20 PM	9:24 PM	9:27 PM	9:28 PM	9:29 PM
40	9:45 PM	9:46 PM	9:47 PM	9:50 PM	9:51 PM	9:56 PM	9:58 PM	10:01 PM	10:05 PM	10:09 PM	10:12 PM	10:13 PM	10:14 PM
41	10:30 PM	10:31 PM	10:32 PM	10:35 PM	10:36 PM	10:41 PM	10:43 PM	10:46 PM	10:50 PM	10:54 PM	10:57 PM	10:58 PM	10:59 PM
42	11:15 PM	11:16 PM	11:17 PM	11:20 PM	11:21 PM	11:26 PM	11:28 PM	11:31 PM	11:35 PM	11:39 PM	11:42 PM	11:43 PM	11:44 PM

y coordinate and measure are added later. An example of the stop file is provided in Appendix B and is discussed further in Chapter 5.

This prototype uses the route identifier and suffixes three digits to represent a unique stop-id. This id is unique to the transit network and can be used as a reference to the time schedule. Due to the fact that the point coverage created for each of the transit routes is ordered form (e.g. the first address is created first with a node id of 1, and so on), much of the program of this module relied on this fact. In other words, node numbers were used as a referencing system instead of stop-id's. If a bus stop coverage already containing points were provided whereby the node referencing system was not consistent, stop-id's would be used. In this prototype, using node numbers helped to simplify the coding and representation process.

4.3 Route Service Attributes

In addition to providing representation of transit routes and stops, transit time schedules must be in a suitable form. In this prototype, they are represented as Info tables with the same route identifier used to describe the route and stops but with an extension of ".sc1". These files are created using the route system data model to provide the linear referencing system of each route. The time schedules for specific points are entered in a Microsoft Excel worksheet. An example of a time schedule worksheet is given in Table 4.1. Using the referencing system, interpolated arrival times for each point can be computed. This worksheet is saved in a suitable ASCII data format with the extension "scx". This schedule file is then converted to a representative Info table. As noted previously, the time representation used in this prototype was decimal days (e.g. 0.5 represents 12:00 pm). Special macros had to be written to convert decimal days to hours-minutes-time of day format and back for input and output functions.

Other characteristics, such as the name of the destination, the name of the variation and handicap accessibility are closely related to the route definition. These characteristics are added as attributes to the route system data model, namely the route attribute file (RAT) file, as will be discussed later.

4.4 Arc/Info Data Organization

This prototype uses the Arc/Info GIS to maintain digital geographic maps built from vector features: arcs connecting nodes, standalone points, and polygons. A map is often comprised of several layers, each representing a class of geographic data such as roads, rivers, or county boundaries. In Arc/Info each layer of a map is called a coverage. A coverage contains data representing the geographic and non-geographic attributes of spatial features. Arc/Info stores the geographic description in specially formatted files that cannot be accessed directly by the user. The non-spatial data is stored in attribute tables in Info, a quasi-relational database closely associated with Arc/Info. The user can easily manipulate the contents and structure of these tables. A coverage can have one or more types of topology, or spatial feature types, associated with it. Coverages contain points, arcs (or lines), or polygons. Representation of each feature is in the form of an Info database file. Polygon coverages contain arcs and are represented by PAT files, and arcs are defined by nodes at each end. Arcs are represented by AAT files. Points are similar to nodes, but are dissociated for any lines and are stored independently. Points are also stored as PAT files while nodes are stored as NAT files.

This prototype uses a number of Arc/Info coverages. The most important of these is the street network base map, contained in a line coverage called MASTER. The version of this map was constructed by converting the Union County, NJ TIGER/Line files into a temporary coverage and then extracting the road features into a final coverage.

The route construction process, described in the next chapter, creates a number of point coverages for each route and a coverage containing all of the routes point coverages. These point coverages are primarily transit stop coverages and contain the route identification associated with the route they lie on. For example, for route b10200811, there would be a point coverage with the name b10200811.

4.5 The Route System Data Model

The transit network is uniquely different from that of the roadway network in its representation graphically and datawise. In a transit network, multiple routes operate over the same set of streets or railroad tracks. Arc/Info's standard arc-node topology is inadequate for this representation. Arc/Info does not allow two arcs to overlay each other. Separate representations are needed for each route in each variation and direction. In addition, though transit routes may use portions of arcs, their route is linear in nature and in sequence. Each route has its own referencing system based on its origin and destination.

In Arc/Info version 6.0 , ESRI, Inc. introduced an alternative means of representing a network of any kind, especially transit networks, called the route system data model or dynamic segmentation. As defined by Grayson (1993, 78), a route system is a logical representation of a group of routes, implemented using tables that maintain a direct relationship to the geometry of an underlying line coverage without replicating that geometry. This data model addresses and overcomes the route representation problems presented in the previous paragraph. This prototype uses only one route system, named BUS, to represent a typical transit network.

Route systems in Arc/Info are implemented using two linked Info tables, the route attribute table (RAT) and the section table (SEC). The RAT contains one record for every route. The prototype stores the route identifier in addition to internal and

user-id's for each route in this table. For the MASTER coverage and the route system BUS, this table has the name MASTER.RATBUS. The section table (SEC) contains the representation of the specific street links that comprise a route and the order in which those links appear. For the MASTER coverage and the route system BUS, this table has the name MASTER.SECBUS.

Route systems are directly related to a specific line coverage. The internal-ID and user-ID columns use the route system name as their prefix because there may be more than one route system per coverage. For example, for the coverage MASTER and the route system BUS, the internal-ID's and user-ID's in the RAT and SEC files are called BUS# and BUS-ID, respectively.

A route is represented by an ordered set of sections. The SEC contains one row for each section and is stored in the order by which they occur in the route. A section refers to exactly one arc or a piece of an arc in the underlying line coverage.

The RAT and SEC files are linked to each other. Each section in the SEC file belongs to exactly one route. This route is identified by an item in the SEC file, called ROUTELINK#, that corresponds to an internal-ID in the RAT file. The SEC file has a similar relationship with the AAT file, since each section in the SEC file refers to exactly one arc in the underlying line coverage. The arc corresponding to a section is identified in the SEC file by the item ARCLINK#, which refers to an internal-ID in the AAT.

A section may refer to an entire arc or a piece of it as mentioned previously, and can be oriented in the same or opposite direction as that arc. Arc/Info represents these relationships between a section and its underlying arc by defining a section in terms of its starting and ending positions along the arc. These positions, stored in the items F-POS and T-POS, represent these positions as percentages of the arc's length

relative to its starting point. The arc's from-node is at the 0% position and the to-node at the 100% position.

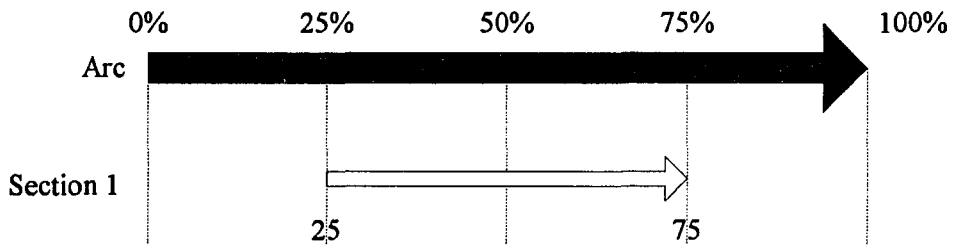


Figure 4.1 Relationship Between One Section and the Underlying Arc

Section 1 represents part of an arc and has the same orientation as the arc. Its value for F-POS is 25 and for T-POS is 75. These values mean that the section begins at 25% of the arc's length from the from-node and extends to the 75% position, comprising 50% (75 - 25 = 50) of the arc's length overall.

In order to support dynamic segmentation, Arc/Info creates a unique linear coordinate system for each route. The units of this coordinate system can be selected by the user; by default, it is the same as used for the arc lengths. The street network MASTER uses the geographic coordinate system, using latitude and longitude, so that lengths are decimal arc units. These lengths must be converted into appropriate values of a linear measurement. The chosen measurement was miles. The linear coordinate system permits the GIS to find a location along a route using a one-dimensional "measure" in that system (e.g. a milepost number or the number of feet from a reference point). The coordinate system measures that correspond to the beginning and end of a section are contained in the SEC file under the columns F-MEAS and T-MEAS. The expression (T-MEAS - F-MEAS) is the "length" of the section using the current coordinate system. For further details see Grayson (1993, 82-87).

A complete discussion on the topic of dynamic segmentation is presented by Dueker and Vrana (1993) where they discuss the systematic analysis of the requirements for dynamic segmentation applications and provide clarification of concepts, terminology and data models. Examples from vendor approaches are used to illustrate the problems with existing terminology and differences in approach and implementation.

CHAPTER 5

IMPLEMENTATION

5.1 Introduction

This chapter describes in detail the methodology of the prototype in Chapter 3. Chapter 4 focused on the structure of the data, while this chapter explores the programs that manipulate that data. The software development tools, the operating environment, the algorithms used, and the specific programs that were implemented will be examined. Three phases of implementation are discussed: data collection and preparation, network analysis and production of output. Brief descriptions of the program files can be found in the Appendix.

5.2 Software Development Environment

The geographic information system underlying this application is Arc/Info Version 6.1.1 created by ESRI, Inc. It provides a graphical tool, spatial analysis tools, and a rudimentary database management system. Arc/Info contains many modules; those used by the prototype include:

1. ARC, the interface to most of Arc/Info's spatial analysis functions;
2. ARCPLOT, a graphical query and display module, also containing path-finding algorithms;
3. TABLES, a simplified front-end to the Info database system;

Arc/Info can be programmed to automate various tasks by writing macros in AML (for ARC Macro Language). In addition, AML allows for creation of graphical menus and forms. These macros execute slowly since AML is an interpretor, not a compiler.

Arc/Info is capable of storing large amounts of spatial data, including the street network base map and the attributes that accompany its arcs and nodes. The GIS can

additionally perform selected spatial operations on this data. These operations include determining the "buffer" around a set of arcs or points and identifying all the arcs or points that fall inside this "buffer". In addition, Arc/Info stores a group of related routes as a "route system". A route is composed of one or more "sections", each relating directly to an arc in the underlying line coverage.

5.3 Computing Environment

Development of this prototype was performed on a Sun SPARCstation 10 workstation running SunOS Release 5.2 using Open Windows Version 3.2. This Unix-based workstation is equipped with 32 megabytes of memory and approximately 1.8 gigabytes of disk storage. This workstation (called PEGASUS) is connected into the New Jersey Institute of Technology computing network via Ethernet and provides for four licenses of Arc/Info to be opened at the same time.

5.4 Data Collection and Preparation

One of the major phases of the research for this thesis was devoted to collecting data, writing routines to process it, and preparing it for use. The procedures involved in the data collection and preparation process that builds the necessary representation are discussed.

5.4.1 Road Network Base Map

Since buses operate over the same street network as automobiles, a detailed and accurate representation of the roadway network is needed to build a bus route network. In addition, a railroad network needs to be present and a part of the transit route network. The United States Bureau of the Census has collected and distributed topographic data including the needed street centerlines and rail lines in the form of

TIGER/Line files, one set for each county. TIGER stands for Topologically Integrated Geographic Encoding and Referencing system. These files are widely available on compact discs (CD's) and provide a starting point for assembling data based on transportation networks.

5.4.1.1 Converting TIGER file to an Arc/Info Coverage

The original source for the base map was an Arc/Info coverage of Union County, New Jersey created from raw TIGER files. The geographic extent of this map was limited to this county because the project which this research is based focused on potential users within this county. In addition to street centerlines and rail lines, the TIGER files contain other features such as political boundaries and hydrography that are not needed for our analysis. TIGER data contains a Census Class Feature Code (CFCC) for every arc which uses a general classification (e.g. road, railroad) with a specific classification (e.g. local road). A new coverage containing only road and railroad features extracted from the original TIGER coverage was created.

In using the TIGER files, several limiting factors which were very frustrating were noticed. The first item recognized was that the TIGER files were missing ranges of data values of attributes. The attributes most needed for this implementation were those containing address ranges and zip codes. This project's method of creating transit routes as well as performing network analysis relies heavily on address matching to identify locations in the street network. Many arcs were missing address ranges and several had incorrect spelling of the street names. Street names are usually not abbreviated (e.g. "Martin Luther King Blvd" vs "M.L. King Blvd."). As a result, a "hit" rate of about 70 percent occurred (i.e., 30 percent of the time an addressmatch was not found). In addition, on several occasions, multiple locations for addresses were found. Arc/Info's address matching algorithm can optionally use a "zone" identifier to assist in

finding a match. The logical choice for the zone is the five-digit postal ZIP code since they are widely used and are included in the TIGER data. Through use of this data set, ZIP codes were found missing from about fifteen percent of the county. Correcting this problem would involve creating ZIP code polygons and overlaying them on the roadway network. This action would result in splitting many of the arcs. In addition, there were inaccuracies in arc length and position.

These two problems, missing or incorrect addresses plus missing zip codes dictated one of the following actions: revise the existing TIGER database, purchase an enhanced TIGER database, or make due with the existing TIGER database. Due to time, personnel, and budgetary constraints, the third option was chosen. It was felt that our algorithms should be independent of the base topographic database used. An enhanced TIGER database could always be purchased in the future. As a result, many of the addresses chosen to be used were unique to the county (only one location was found). Grayson (1993, 116-123, 132-136) provides solutions to revising the existing TIGER database to deal with these plus other problems, such as missing arcs or zigzag intersections.

5.4.2 Building the Transit Route Network

With our address-geocoded base map of streets and rail lines in place, the process of encoding the transit route network that use these features can begin. As mentioned previously, no ready-made digital network of the bus and rail routes was available from New Jersey Transit. New Jersey Transit provided a digital network of the bus and rail routes several weeks after a prototype transit network was in place and that format was in another GIS, namely TRANSCAD by Caliper Corp. This format was not easily compatible with Arc/Info. Therefore, a representative transit network using major collector and arterial roads as defined by Hagstrom maps was constructed. A total of

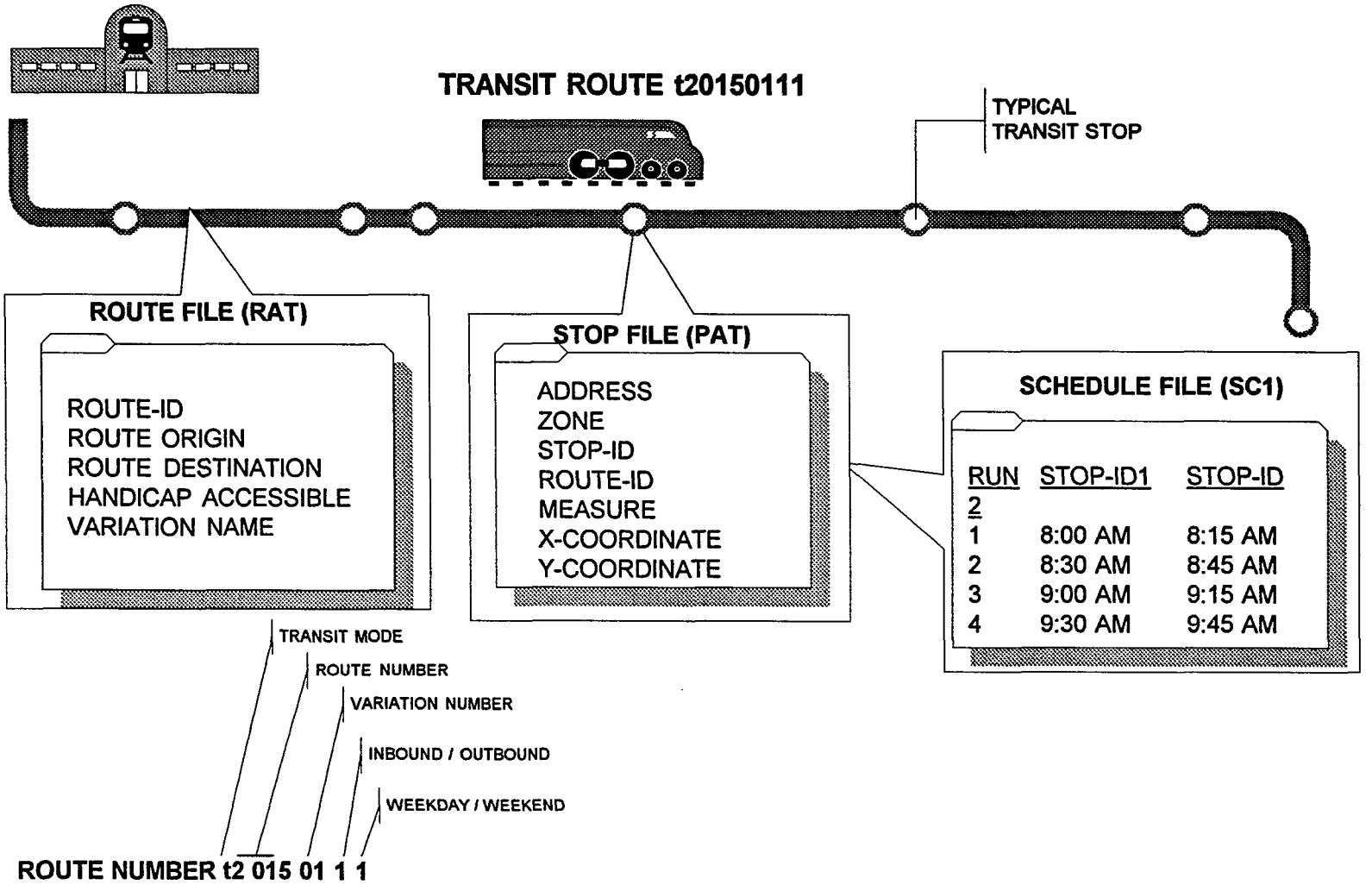


Figure 5.1 Schematic Database of Prototype

sixteen routes were created, or eight inbound-outbound. Of those eight routes, seven were designated as local routes and one designated as an express route. Figure 5.1 presents a schematic of the data structure used in this prototype.

5.4.2.1 Encoding the Path of a Route

The main challenge in building routes is selecting the arcs from the street network to incorporate into a route. As Grayson (1993) points out the following: "The obvious approach, picking the arcs by hand using a mouse pointing device is tedious, error-prone and slow. Arc/Info has limited accuracy when selecting objects using the mouse. If features are too close together in the display window, a mouse click directly on top of the desired arc often selects a nearby arc. Remedying this problem requires either removing the incorrectly selected arc from further consideration, which requires typing a verbose command, or zooming in the window for a more detailed view. It is easy to overlook small arcs and accidentally select incorrect arcs using this manual method. Making a change to a route requires repeating this drill over and over."

Rather than adopting this approach, it was decided to take advantage of Arc/Info's address-matching and path-finding capabilities to simplify the task of mapping a transit route's path and its stops. This method allows the description of a route using two specially formatted text files as described in Chapter 4 which are the following: a path AML file that describes a path a route takes, and the stop file that list the location of stops as they occur along a route. The route's path representation is built first and then supplemented with the location of stops. The requirements for this approach is the following: first, a digital map of the road network with address geocoding, second, an addressmatching to match street addresses and intersections to point locations in the network, and third, a tool for finding the shortest path between several points in the network.

The first step is to create the path AML text file that describes the route. This AML macro contains the needed NETWORK commands to create the route plus locations where the route passes through. The file has one line for each location and can be in the form of an intersection (e.g. "Chestnut St / North Ave") or an address (e.g. "201 Morris Ave") and are in the order that they are passed along the trip. Generally, the locations represent intersections where the bus turns, but other locations along the route are added for clarity and to ensure that the shortest-path routine will choose the desired path.

The route system representation of a route's path is constructed from the path AML file. A path AML file is written for each route and contains the route identification number (e.g. b10050111.aml). The macro matches the addresses in its file to geographic locations along a transit route's path and then finds the shortest path between the points identifying the arcs that connect these locations. The route description is stored in a RAT and SEC file.

5.4.2.2 Encoding Transit Stop Locations

Adding the location of stops to the route representation is a separate process from the representation of the route itself. In this process, address matching is integral to our approach. The approach used is similar to that of Grayson (1993) where each route requires a stop file containing the address of each stop and optional related data. The stop file is an ASCII text file using a route identifier (e.g. b10050111.txt) and contains the location, stop-id, and route-id, separated by commas. Each location is on a separate line. A macro called CREATE_STOPS.AML reads these files, converts them into Info tables and then matches the addresses against the street network using the Arc/Info ADDRESSMATCH command. The result of this macro is a point coverage of all the transit stops of the specified route ordered in same manner as in the stop file.

Additional attributes for each of the stop coverages are needed. These are the measure and the x and y coordinate. As discussed previously, the "measure" of a point is defined as its relationship to the linear referencing system used in the route system data model. In our prototype, this measure is in decimal arc format using the geographic coordinate system. The measure for each point in each route's coverage is performed using the Arc/Info command ADDROUTEMEASURE. The x and y coordinates represent the longitude and latitude respectively and are obtained by using the Arc/Info command ADDXY. These values are then stored in the attributes LAT, and LONG for each point coverage.

In this prototype, stops can be located anywhere along the route's path; at intersections or at building addresses. This is due to the fact that the prototype relies on the location of the stops in relation to the whole route and does not consider nodes belonging to the route as does Grayson's (1993) prototype. The route itself is used to give measurement to the transit stops and to provide a graphical image of the route, but is not used in the network analysis.

5.4.2.3 Encoding Transit Schedules

Once all of the route's transit stops are converted into representative point coverages and measures are assigned to each point in each coverage, the generation of transit schedules for each route can proceed. First, the stop-id's and measures for each point is output to a text file. That text file is then imported into Microsoft Excel and is transposed. Next, time values for selected checkpoints are entered using the suitable time formula provided. Then interpolated time values are computed for each point between the known checkpoints. As a side note, Microsoft Excel represents time in terms of decimal days and therefore, the resulting time values are generally between 0 and 1. The completed worksheet is then output to a suitably formatted ASCII text file,

which is then converted to a representative Info table with the name of the route identifier and the extension ".sc1".

Since timetables are represented in a different format than a standard day (e.g. they progress from early morning 5:00 am to early morning 3:00 am, instead of from 12:00 am to 12:00 am). Special provisions had to be made to the formatting of the schedule files. Duplication of schedules was accomplished before and after the 12:00 am to 12:00 am time period. For example, after 12:00 am (value of 1.0) several runs were added until 3:00 am (1.125). This provision would allow for questions involving departure time (e.g. "I am leaving at 11:30 pm"). The schedule was also modified to account for the changeover in time periods prior to 12:00 am (0.0 value). A value of -999 was placed for all stops prior to the 12:00 am (0.0 value) time period. For example, if the first run had a changeover to 12:00 am during its run, the values of time of the stops prior to that period were assigned a value of -999. To provide for questions involving arrival time (e.g. "I need to be at a location by 1:00 am) the algorithm was modified to handle such questions and is described later.

5.5 Generation of the Connection Matrix

The prototype used in this application relies on the connectivity of the transit stops in the transit network. This connectivity is handled by a "connection matrix" where all transit routes are compared with each other. If the transit route intersects directly or less than a quarter-mile then the value is set to 1, else the value is set to 0. This connection matrix gives the state of the current transit network and tells which transfers are possible and which are not. The AML macro generating this matrix is called CONNECT.AML. This macro first obtains the list of transit stop point coverages generated previously, and creates an Info table in which the first attribute contains the list of all the coverages and each transit route identifier becomes an

attribute. Each column is then assigned a default value of zero. Each transit stop point coverage is compared with each of the other transit stop point coverages using Arc/Info's NEAR command. The near command appends the distance from the point of the near coverage to the point of the comparison coverage. Cursors, an Arc/Info concept that will be explained later, are used to find if there are any occurrences of 0 distance or a distance of 0.25 miles. A conversion factor of 60 is used to convert degree arcs into decimal miles. If a value of 0 or 0.25 miles is found, a value of 1 is assigned to the attribute (comparison transit route) of that record (actual transit route). An example of this table is shown below. Columns represent attributes, while rows represent records.

Table 5.1 Sample Connection Matrix

Route\Route	10	20	30	40
10	0	1	0	0
20	1	0	1	0
30	0	1	0	1
40	0	0	1	0

As noted from Table 5.1, Route 10 intersects with Route 20, while Route 20 intersects with Routes 10 and 30. From this table it can be seen that Routes 10 and 30 both intersect with Route 20. This fact will allow the creation of possible two-transfer scenarios, as will be discussed later.

Cursors, as mentioned previously, are unique functions which allow the user to progress down a database one record at a time and allow individual attributes to be extracted and written to. The use of cursors is key to all of the algorithms used in this prototype.

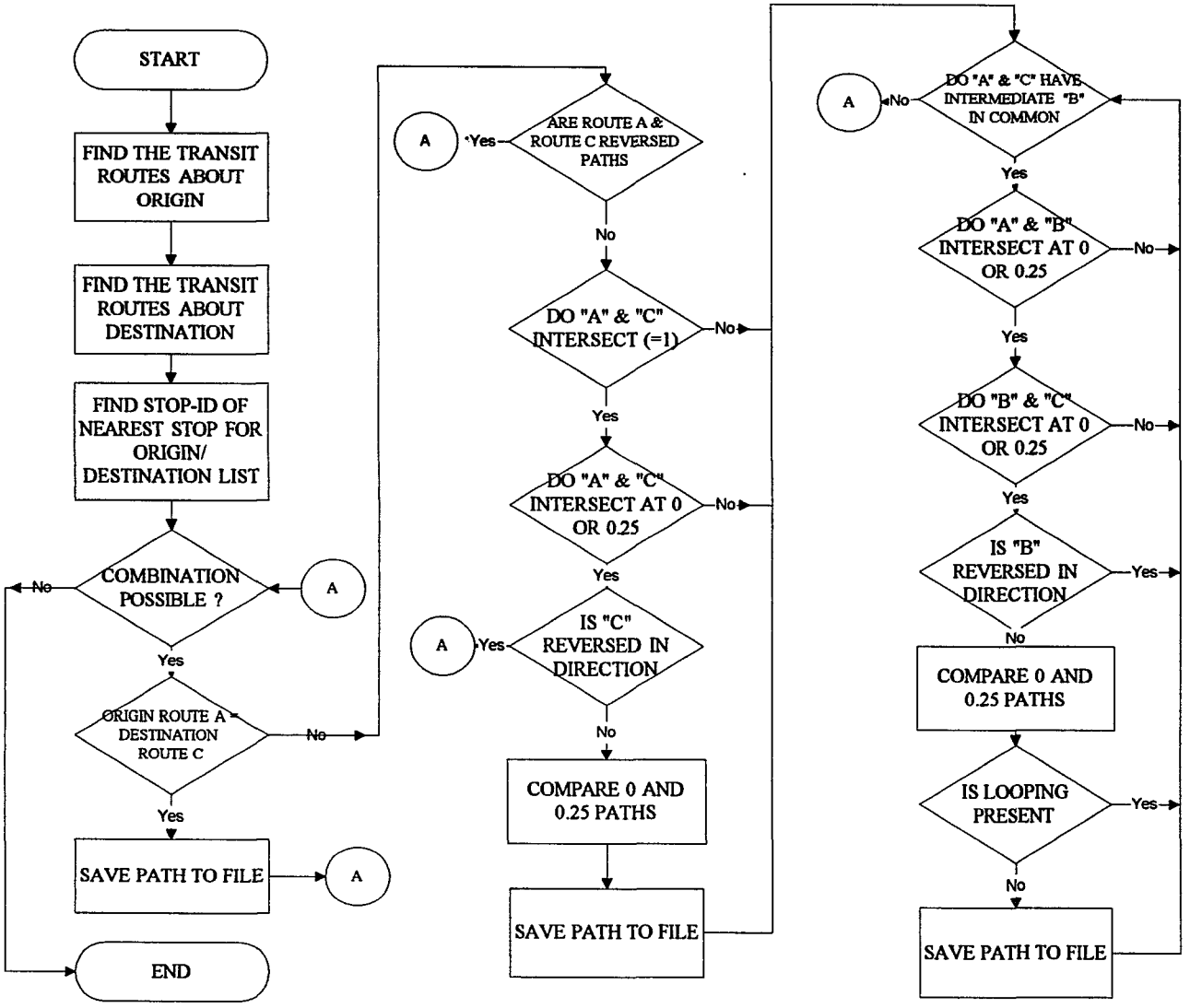


Figure 5.2 Schematic of CREATE_PATH.AML

5.6 Finding Paths

The algorithm for finding paths in a transit network is called CREATE_PATHS.AML. This algorithm is unique from other approaches in that transit stops are used exclusively for the analysis versus the traditional shortest path algorithms which use the cost of traversing an arc in the determination of the shortest path. The algorithm contained in the CREATE_PATHS.AML file is based on a simple heuristic approach. The methodology for this algorithm is summed below.

Find the transit routes about the origin and destination. Compare each origin route to each destination route for an intersection of 0.25 miles or 0 distance. If any intersections are found, store information on the combination. Find all possible routes that intersect with each of the origin and destination route combinations using the connection matrix. Compare the intermediate route with the origin and destination routes for intersections of 0.25 miles or 0 distance. If any intersections are found, store the information on the combination.

This summary is very brief and does not provide the many additional checks and balances needed to accomplish this task. This algorithm relies on nested loops and cursors to meet its purpose. A full description of the algorithm is given next and then specific elements are addressed later. An example of the process of this algorithm completes this section.

5.6.1 Overview of Path Finding Algorithm

The CREATE_PATHS algorithm uses several data sets which must be developed previously. These data sets include the transit stop point coverages, discussed previously, an entire transit stop network point coverage (created using the Arc/Info APPEND command), and Info tables containing the locations of the origin and destination points. Figure 5.2 presents a simplified schematic of the flow of this

program. The sequence of processing is summarized in the pseudocode description below:

```
Check for the existence of temporary origin coverages
Addressmatch the origin location against the MASTER street coverage
Buffer around the resulting location for a distance of 0.25 miles
Intersect the buffer with the transit stop network
Obtain a list of origin transit routes using the frequency command
Using Cursors set the values of the transit routes equal to index variables
Count the number of origin transit routes
Check for the existence of temporary destination coverages
Addressmatch the destination location against the MASTER street coverage
Buffer around the resulting location for a distance of 0.25 miles
Intersect the buffer with the transit stop network
Obtain a list of destination transit routes using the frequency command
Using Cursors set the values of the transit routes equal to index variables
Count the number of destination transit routes

Find the closest point in each origin transit route to the origin point
        Store the node number, x and y coordinate of this point
        Store the bearing (direction) and distance from the origin point

Find the closest point in each destination transit route to the destination point
        Store the node number, x and y coordinate of this point
        Store the bearing (direction) and distance to the destination point

For each origin-destination pair
        Check to see if the pair are equal. If they are equal save variables and go to
        next pair.
Else
        Check to see if pair is the same route but in different directions. If yes go to the
        next pair.
        Obtain the value of the pair from the connection matrix.
        If the value equals one then
                Using the Arc/Info Near command, compare the origin route with the
                destination route.
                Using Cursors, progress through the origin route from the node closest
                to the origin.
                If the value of the distance is 0.25 miles or less first, hold that point,
                store it, find the
                distance and direction to that point and continue progressing until a 0
                distance is found.
```

If the value of the distance is 0 first, hold that point, store it and discontinue progressing.
Check for comparisons where the 0.25 mile and 0 distance are compared to the same point in the other coverage.
Find the measures in the destination coverage of the transfer point and the departure point.
For situations where there is a 0.25 and 0 transfer distances, Find the time required to traverse each situation and compare.
If the departure point node of the destination route is less than the transfer point node then go to two-transfer comparison.
Save all required information to indexed variables

If the value does not equal one (two-transfer route)

Search the connection matrix for routes that intersect both the origin and destination
Save the intermediate routes and a total count.
Using the Arc/Info Near command, compare the origin route with the intermediate route.
Using Cursors, progress through the origin route from the node closest to the origin.
If the value of the distance is 0.25 miles or less first, hold that point, store it, find the distance and direction to that point and continue progressing until a 0 distance is found.
If the value of the distance is 0 first, hold that point, store it and discontinue progressing.
Using the Arc/Info Near command, compare the destination route with the intermediate route.
Using Cursors, progress through the destination route from the beginning node to the node closest to the destination.
If the value of the distance is 0.25 miles or less first, hold that point, store it, find the distance and direction to that point and continue progressing until a 0 distance is found.
If the value of the distance is 0 first, hold that point, store it and discontinue progressing.
For the first transfer point:
Check for comparisons where the 0.25 mile and 0 distance are compared to the same point in the intermediate coverage.
Find the measures in the intermediate coverage of the transfer point.
For situations where there is a 0.25 and 0 transfer distances, Find the time required to traverse each situation and compare.
For the second transfer point:
Check for comparisons where the 0.25 mile and 0 distance are compared to the same point in the intermediate coverage.
Find the measures in the intermediate coverage of the transfer point.

For situations where there is a 0.25 and 0 transfer distances, Find the time required to traverse each situation and compare.
Check intermediate route if it is reversed in direction
Check for looping of the intermediate route
Save all required information to indexed variables
Save all indexed variables to a temporary ASCII text file.

5.6.2 Specific Elements

CREATE_PATHS.AML primary computation is the comparison of the transit stops of two transit routes. The Arc/Info command NEAR is used for this comparison. The NEAR command appends the distance to and name of the nearest node in the comparison file. Cursors can be used to progress down the compared file and look for all occurrences of 0 and 0.25 mile distances. The program looks for a 0 or 0.25 mile distance, if it finds a 0.25 mile distance, it holds that point and continues to the end of the file or until a 0 distance point is found. If a 0 distance point is found first, it holds that point and exits the loop.

There may be situations where there is a 0.25 mile transfer point before a 0 distance transfer point, for example, a parallel route which eventually meets. In this case, the time it would take to walk the distance to the transfer point must be computed and compared to the distance taking a 0 distance transfer. This computation uses distances and average speeds. The assumed waiting time is not included for simplification purposes. If there is a situation where both options produce the same time, it is assumed that the user will travel on a 0 distance transfer route. The average walking speed was assumed to be 2 mph (or 3 ft/s) and the average transit speed was assumed to be 18 mph.

This one transfer computation is illustrated below. Progressing from the origin transit stop, a 0.25 mile distance transfer is found at A. This point is held and the algorithm continues until a 0 distance transfer point is reached, C1. The corresponding

nodes of the comparison route can then be obtained, namely B and C2, respectively. Next the travel time from A to B to D is obtained (where D is the location of the transit stop closest to the destination). The distance from A to B is based on a straight line distance, while the distance from B to D is based on measure distances. Alternatively, the travel time from A to C1 to C2 to D is computed based on measure distances. A feature that could be added is the user's negative benefit costs (or weight factor) in walking some distance for a transfer. One should note that this procedure works if the destination is in a loop or outside a loop, as shown below.

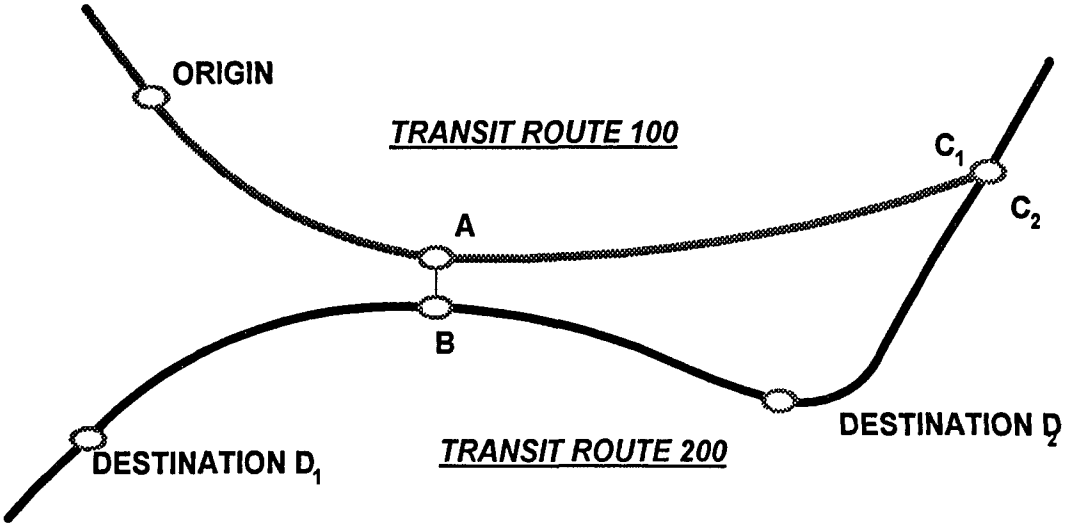


Figure 5.3 One Transfer Computation

Another type of comparison that should be avoided and checked is that when a specific route is compared to another route and the comparison node points are the same for the 0.25 distance and 0 distance transfer. This is illustrated below in Figure 5.4. In this figure, transit route 100 is compared with transit route 200. At location C there is a 0.25 transfer distance between itself and node 2. At location D there is a transfer distance of 0 between itself and node 2. Both locations (C and D) are

compared with the same point (2). This algorithm handles the above situation by storing the node of the 0.25 mile transfer distance comparison point (2) and compares with the comparison point of the 0 distance transfer. If these two points are found to be the same, the 0.25 mile transfer point is disregarded.

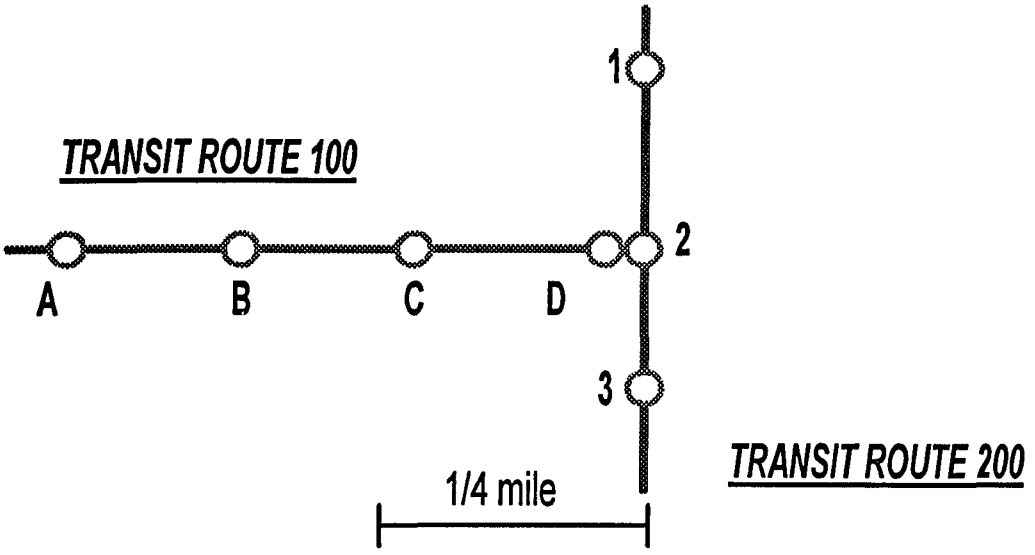


Figure 5.4 Same transfer point comparison

In addition to having to consider loops with one transfer, two transfer loops must also be considered. Basically, each loop is treated independently and then compared. Figure 5.5 illustrates a typical problem where there are two loops (and shortcuts) present. First the travel time along link A-B is compared with the travel time on links A-C1, plus C2-B. Then the travel time along link D-E is compared with the travel time on links D-F1 plus F2-E. The paths with the shortest travel time are used. Once this procedure is performed, the directionality of the intermediate route can be checked.

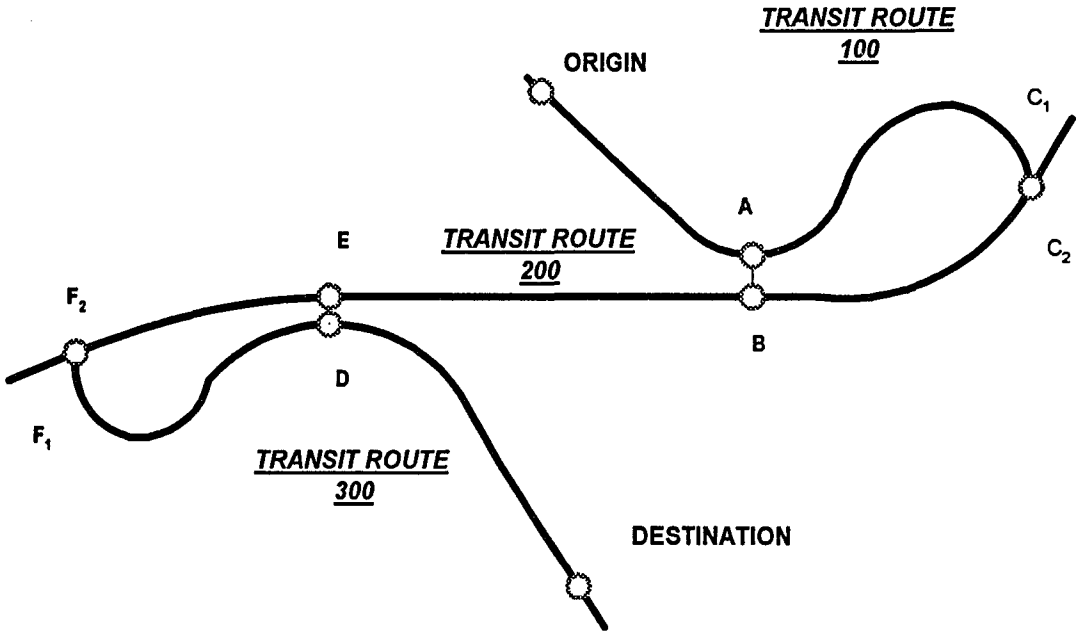


Figure 5.5 Two Transfer Comparison

When comparing a two transfer path with three transit routes, there exists the possibility of one of the routes doubling over another route. This type of a path, though logical to the computer, may be illogical to the user. The directed path may guide the user to take a local bus to a terminus and then an express bus to the destination instead of taking a local bus to a destination. This type of doubling (or looping) is shown below in Figure 5.6. Transit route 200 inbound and outbound operate on the same roadway. To go from an origin to a destination, one can take transit route 100 outbound at node 5, leave at node 6, transfer to transit route 200 inbound at node 3 and depart at node 5. Another possible path from the origin to the destination would be to take transit route 100 outbound at node 5, depart at node 6, transfer to transit route 200 outbound at node 7. Then take this route to the end (node 9) transfer to transit route 200 inbound at node 1 and depart at node 5. This possible path, though illogical to a user and would require much travel time is acceptable to the GIS database since the inbound and

outbound routes are separate definitions. Even though the routes operate over the same roadway, the prototype would look at this situation as separate and distinct routes.

The solution to this problem uses the fact that the node numbers increase as one progresses along a route's travel path and that a worst case solution would allow the travel along nodes less than those occurring in a one-transfer path. For example, in the situation below, the network algorithm would produce a one-transfer path (Route 100 Outbound to Route 200 Inbound transferring from node 6 to node 3). The transfer node of the second route would be stored (node 3). The network algorithm would then produce a two transfer path (as described previously), where the third route would be the same as the second route (in a one transfer path) and its transfer point node would be 1. Since one is less than three, this would mean that the new path doubled upon itself and would be a worst case, and therefore not be considered. This provision prevents paths not only from doubling upon themselves, but prevent two-transfer paths which are further away than one-transfer paths operating on the same routes.

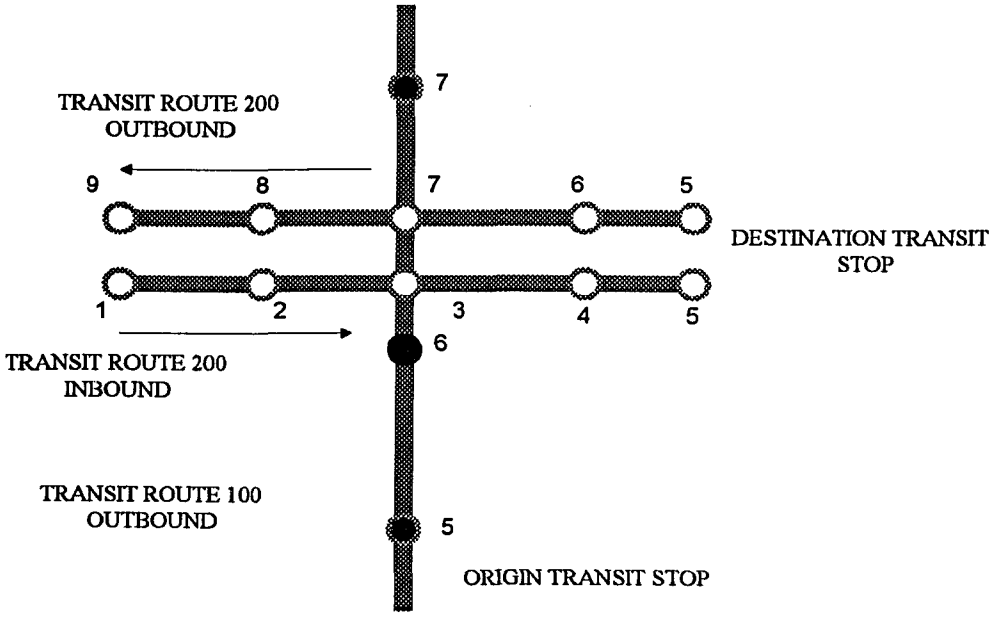
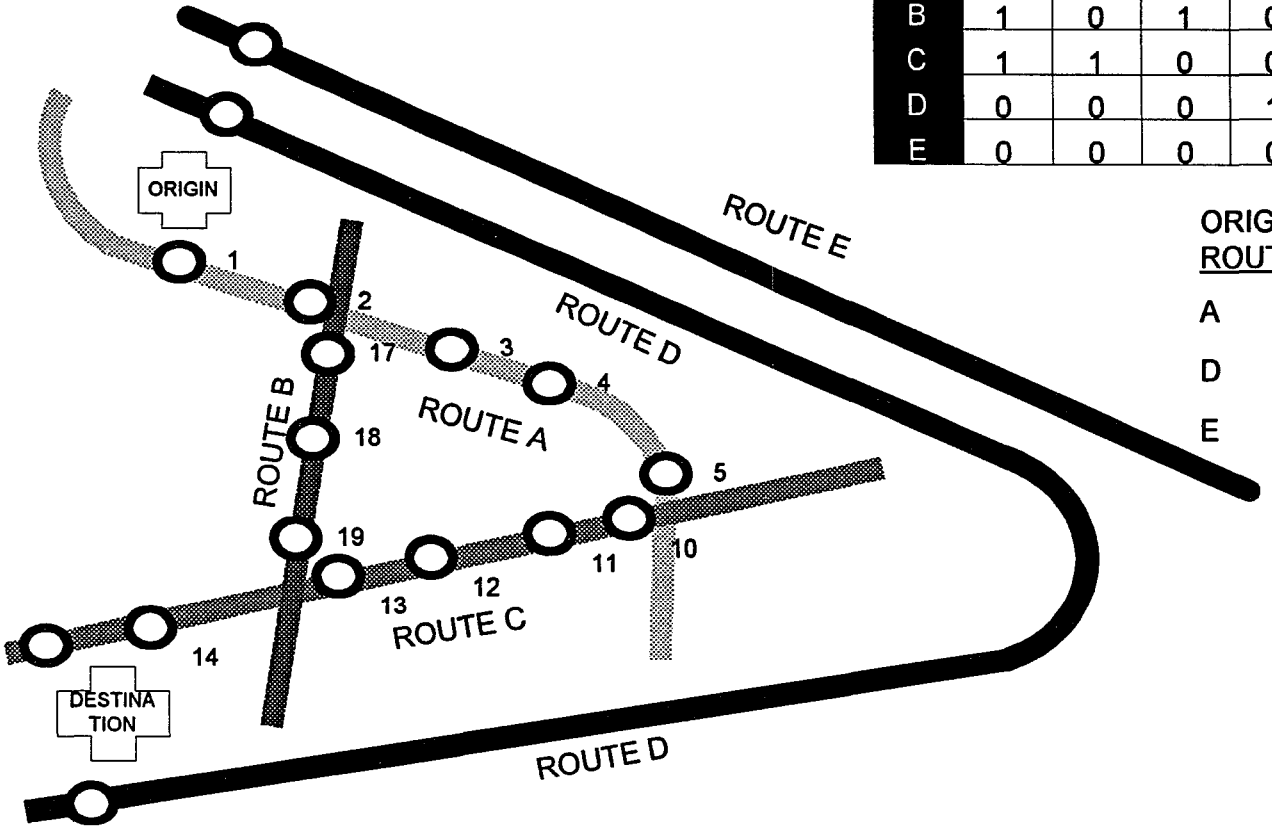


Figure 5.6 Schematic of Doubling Condition

CONNECTING ROUTE MATRIX

	A	B	C	D	E
A	0	1	1	0	0
B	1	0	1	0	0
C	1	1	0	0	0
D	0	0	0	1	0
E	0	0	0	0	1



ORIGIN ROUTES	DESTINATION ROUTES
A	C
D	D
E	

Figure 5.7 Example Network Schematic

The critical output of the CREATE_PATHS.AML is an ASCII text file containing all of the particular elements needed for each path. The name of the file uses an user-ID number with the extension ".pth". The user-ID is a value that is required to be entered and is described in Chapter 6. Each line contains one variable and the sequence is the same for each individual path. The sequence is as follows:

Number of paths
Route Identification Number of First Route
Route Identification Number of Second Route
Route Identification Number of Third Route
Node number at the end of the First Route
Node number at the beginning of the Second Route
Node number at the end of the Second Route
Node number at the beginning of the Third Route
Transfer distance between First and Second Route
Transfer distance between Second and Third Route
Direction from the Transfer Point of the First Route to the Second Route
Direction from the Transfer Point of the Second Route to the Third Route
Node number of the closest origin transit stop
Distance from the origin to the closest origin transit stop
Direction from the origin to the closest origin transit stop
Node number of the closest destination transit stop
Distance from the destination to the closest destination transit stop
Direction from the destination to the closest destination transit stop

5.6.3 Example of Network Analysis

This section presents an example of how the CREATE_PATHS algorithm works on a typical transit network. Figure 5.7 presents an example transit network schematic. In this network it can be seen that more transfers are required when the route length is shortened (e.g. Route D is longer than Route A and Route C). Included is a representation of the connection matrix that describes this network.

temporary ASCII text file which would be used in determining the overall travel time for each path.

5.7 Finding Travel Time and Itinerary of Each Path

The next step in the main algorithm's process is the determination of the travel time and itinerary of each path found using the CREATE_PATHS.AML. The determination of travel time depends on the "type" of travel time provided. Travel time can be designated as either "arrival" or "departure". Specifying "arrival" indicates that the person desires to arrive at a destination at the given time, while the term "departure" indicates that a person desires to leave an origin at the given time. Since these two terms have different meanings and different methods of computation, they have been divided into two separate macros: TRAVEL_ARR.AML and TRAVEL_DEP.AML. Both of these macros have provisions for user costs or negative benefits for waiting and walking travel times, but are not used in this prototype.

5.7.1 TRAVEL_DEP.AML

The TRAVEL_DEP.AML macro determines the travel time and itinerary of each path and sorts the paths based on overall travel time. The criteria for this macro is that the user desires to depart an origin point at a certain time. The data required for this routine includes the text file produced by the CREATE_PATHS.AML plus the time of departure. The TRAVEL_DEP.AML macro first reads the ASCII path file and converts each line into an indexed variable. Then for each path the travel time and schedule are found. Once all the travel times for all paths are determined, an ASCII text file is created containing all of the data from the previously generated text file (under CREATE_PATHS.AML) plus the schedule and travel times generated by the TRAVEL_DEP.AML macro. This newly generated text file is represented using the

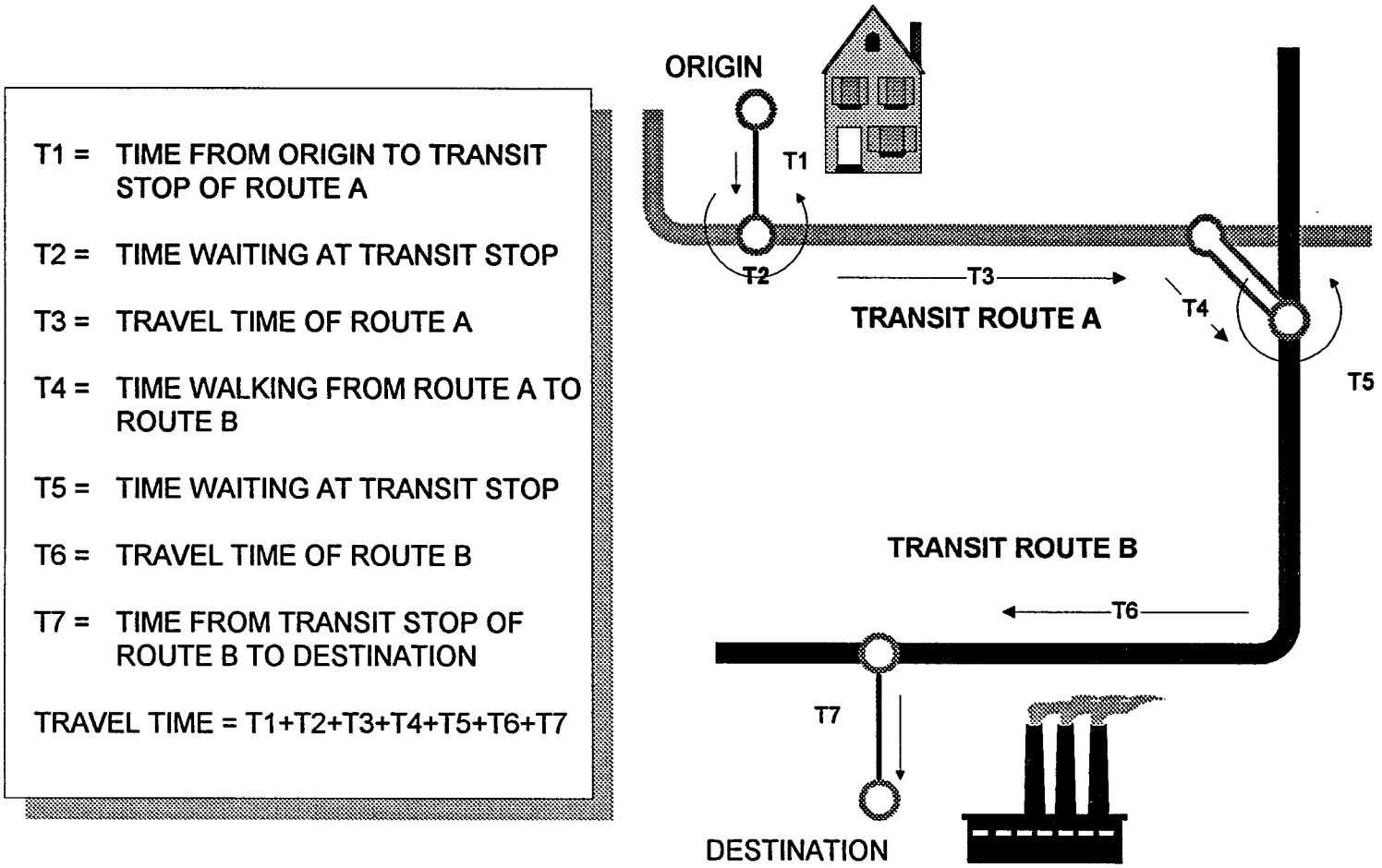


Figure 5.8 Schematic of Computation of Travel Time Based on Departure Criteria

user-ID with the extension ".s1" and is then converted into a representative Info table using the user-ID with the extension ".pth". This Info Table is then sorted based on travel time in descending order (the shortest travel time is first). The overall travel time is the difference between the time of arrival at the destination point minus the desired time of departure at the origin point. This definition may present some unusual situations. For example, one path may have a shorter initial wait time and a longer enroute travel time while another path may have a long initial wait period and a short enroute travel time, both routes taking the same overall travel time. In this situation, both paths would be ranked equally.

Figure 5.8 presents a schematic of the factors involved in the computation of travel time when based on departure time. This example is based on a two-path (one transfer) path. The pseudocode for this path is as follows:

Use the specified departure time as the start time, called running time
Calculate the time to travel from the origin point to the first route's arrival transit stop (T2).
Add that time to the running time
Go to the Info schedule file for that route and find the first occurrence of a run with a time greater than that of the running time. Save that time, run number and time of arrival at the first route's departure point.
Calculate the waiting time at the first transit stop (T2) by subtracting the time of arrival of the first transit route with the running time.
Calculate the time to walk from the departure point of the first transit route to the arrival point of the second transit route (T4). If there is no distance between the two points, use a value of three minutes.
Calculate the running time as the time of arrival of the departure point of the first transit route plus the walking transfer time, T4.
Go to the Info schedule file for the second route and find the first occurrence of a run with a time greater than that of the running time. Save that time and time of arrival at the second route's departure point.
Calculate the waiting time at the second transit stop (T5) by subtracting the time of arrival of the second transit route with the running time.
Calculate the time to travel from the departure stop of the second transit route to the departure point (T7).
Calculate the running time as the time of arrival of the departure point of the second transit route plus the walking time to the destination point, T7.

Calculate the change in time subtracting the running time from the specified departure time.

Note that this computation of travel time takes all factors into account when making a transit trip, not just en-route travel time alone. The average walking speed is assumed to be 2 mph or 3 ft/s which is consistent with standard averages and is somewhat below several of them. This speed could be reduced to 2.5 or 2 ft/s to account for senior citizens, handicapped, or small children. This reduction could provide for the allowed time in reaching the transit stops, since travel time is based on a straight line distance from the origin point to the transit stop. The computations for a three-path (two-transfer) path are similar to the above computations. As noted previously, the overall travel time is used as the basis for determining the order of priority for the paths. It is anticipated that the user will reduce his overall travel time by timing his departure to the arrival of the shortest en-route travel path once the complete information is made available.

5.7.2 TRAVEL_ARR.AML

The TRAVEL_ARR.AML macro also determines the travel time and itinerary of each path and sorts the paths based on overall travel time. The criteria for this macro is that the user desires to arrive at a destination point at a specific time. Finding the travel time of each path is a little difficult to comprehend since the algorithm works backward in time. The data required for this routine includes the text file produced by the CREATE_PATHS.AML plus the time of arrival. The TRAVEL_ARR.AML macro first reads the ASCII path file and converts each line into an indexed variable. Then for each path the travel time and schedule are found. Once all the travel times for all paths are determined, an ASCII text file is created containing all of the data from the previously generated text file (under CREATE_PATHS.AML) plus the schedule and travel times generated by the TRAVEL_ARR.AML macro. This newly generated text

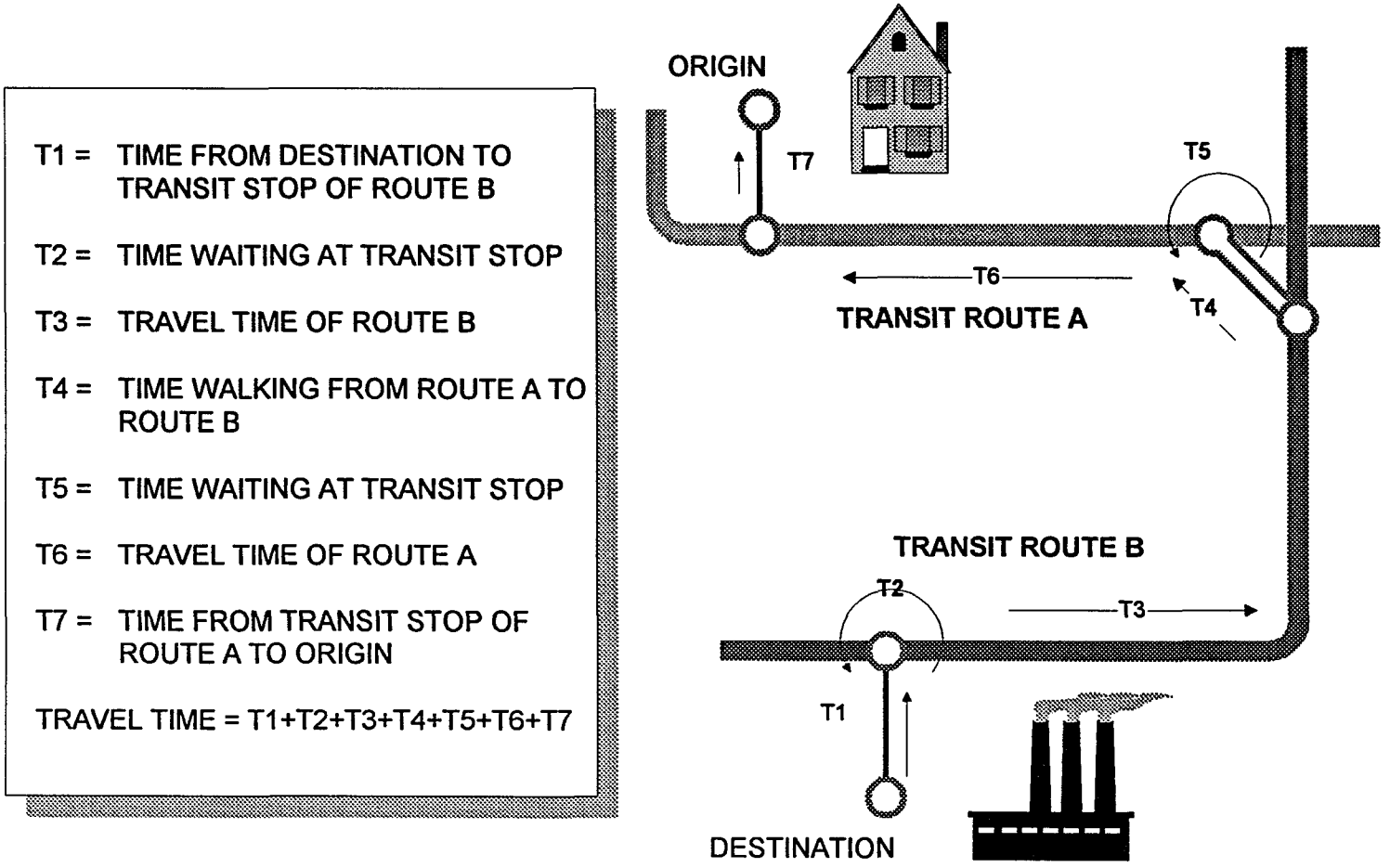


Figure 5.9 Schematic of Computation of Travel Time Based on Arrival Criteria

file is represented using the user-ID with the extension ".s1" and is then converted into a representative Info table using the user-ID with the extension ".pth". This Info Table is then sorted based on travel time in descending order (the shortest travel time is first). The overall travel time is the difference between the desired time of arrival at the destination point minus the calculated time of departure at the origin point. As in the case with the TRAVEL_DEP.AML macro this definition may present some unusual situations. For example, one path may have the user arrive at the destination point earlier than another path, but both occurring before the desired arrival time and with the same overall travel time. In this situation, both paths would be ranked equally.

Figure 5.9 presents a schematic of the factors involved in the computation of travel time when based on arrival time. This example is based on a two-path (one transfer) path. The pseudocode for this path is as follows:

Use the specified arrival time as the start time, called running time.
Calculate the time to travel from the destination point to the second route's departure transit stop (T1).
Subtract that time from the running time.
Go to the Info schedule file for that route and find the first occurrence of a run with a time less than that of the running time. Save that time and time of arrival at the second route's origin stop.
Calculate the actual arrival time by adding the time of arrival of the second route's departure stop with the time needed to walk to the destination point, T1.
Calculate the difference in time from the desired time of arrival at the destination point versus the actual time of arrival (T2).
Calculate the time to walk from the arrival point of the second transit route to the departure stop of the first transit route (T4). If there is no distance between the two points, use a value of three minutes.
Calculate the running time as the time of arrival of the origin stop of the second transit route minus the walking transfer time, T4.
Go to the Info schedule file for that route and find the first occurrence of a run with a time less than that of the running time. Save that time, run number and time of arrival at the first route's origin stop.
Calculate the difference in time between the running time and the time of the departure point of the first transit route. This is the wait time at the transfer point (T5).
Calculate the time to walk from the origin stop of the first transit route to the origin point (T7).

Subtract this time (T7) from the time of the arrival of the first transit route to obtain the final anticipated departure time from the origin, or the new running time.
Calculate the change in time by subtracting the running time from the specified arrival time.

Note that this computation of travel time again takes all factors into account when making a transit trip. The computations for a three-path (two-transfer) path are similar to those for a one-transfer trip. As noted previously, this computation works backward and is complicated. Figure 5.9 helps in the understanding of this computation though not completely. The overall travel time is used as the basis for determining the order of priority for the paths. It is assumed that the time spent waiting at the end of the trip, prior to the desired arrival time is part of the overall travel time. It is also anticipated that the user will choose the path which allows him to leave as late as possible and have enough "cushion" time to arrive at the destination within a suitable time. This "cushion" time may vary with the individual and with the reliability of the transit service.

5.8 Incorporation of User Preferences

As mentioned previously, the incorporation of user preferences in the creation and display of transit paths is critical in attracting users to use transit. The best path for one individual may not be the case for another. This prototype takes four factors into consideration in selecting which of the paths to display. These factors are as follows:

- mode choice
- number of transfers
- walking time to the transit stop
- waiting time at the transit stop

The user makes choices from a menu as displayed in Figure 6.5 in Appendix D. Several of these factors, especially mode choice and number of transfers are reflective of this prototype and could be expanded with added functionality. The questions are

structured to provoke a response of the maximum willingness of a person in make a transit preference, not a response based upon the desire of the individual. For example, the question is stated "How long are you willing to wait at a transit stop?" instead of "Maximum waiting time". It is anticipated that after a series of selections, the user will realize that more choices are possible when the search criteria is relaxed.

The macro that incorporates user preference is called PREF.AML and basically uses the Arc/Info RESELECT command to search for paths in the user's path Info file (as defined previously, with the extension ".pth") that meet the user's criteria. A menu is displayed showing how many paths meet the user's criteria and if the user wishes to revise them (Figure 6.6 in Appendix D). Finally, the selected set is then saved as a separate Info file with the extension ".pre". Since the paths in the "pth" Info file are already sorted based on travel time, the paths in the "pre" Info file are ordered based on travel time also. Next each user preference will be discussed and its implementation given.

For mode choice, provisions for bus, train or both are given. Users have several perceptions about each mode based upon their individual experiences with them. A full discussion of this topic is out of the scope of this thesis. Since the route identifier incorporates a character designation for mode, the Arc/Info command RESELECT can select those paths that contain "b" for bus, or "t" for train. In this prototype there was only one mode designated and denoted by the letter "z" (refer to Chapter 4 for a description of the route identification method). Therefore, though the prototype provided for multiple modes, it used one mode.

The number of transfers a person is willing to make is also dependent on the user and environmental conditions (weather, neighborhoods, fare). The maximum transfers that this prototype uses is two. This is deemed representative of the willingness of most individuals in a suburban area. The determination of paths which

meet the number of transfers criteria is based on the presence of additional transit routes in the path. For example, if the maximum number of transfers a user is willing to make is one, all routes with an intermediate route value of "0" would be selected, if the number of transfers is zero, all routes with a destination route value of "0" would be selected.

The next question asked is "How long are you willing to walk to the transit stop?". The length of the trip to the transit stop was developed using time versus distance since it is easier for a person to understand and visualize a 10 minute walk versus 4 city blocks or a quarter-mile walk. The length in time is given in increments of 5 minutes with a maximum of 15 minutes. It takes approximately 12 minutes for a person to walk a quarter-mile at 2 mph or 3 ft/s, so these values are reasonable. The time selected is converted into an appropriate value of distance (using 2 mph) and then the Arc/Info command RESELECT is used to obtain those paths that are less than this value in each of their walking distance components (e.g. walking distance from origin point to arrival point of first bus, first transfer distance, second transfer distance, and distance from departure point of last bus to destination point).

The final user preference asks the user how long he is willing to wait at a transit stop. Note that this question is based on a person's willingness versus desire, since the user will always desire the minimum waiting time. The waiting time is given in 5 minute increments since the user can easily understand the difference between a 10 and a 15 minute wait. In certain time periods the headway, or time between successive transit vehicles is greater than 20 minutes (off-peak), therefore a selection can be made for paths with waiting times greater than 20 minutes. The time selected is converted into decimal days and then the Arc/Info command RESELECT is used to obtain those path that are less than this value in each of their waiting time components (e.g. waiting time

at the arrival of the first transit route, the first transfer point and the second transfer point).

5.9 Generation of Output

Once the paths have been found, sorted and extracted based on user preferences, they must be presented to the user. This presentation should be as simple and non-ambiguous as possible and yet be complete. In the case of travel along transit paths, the user would like to see the transit path and an itinerary or schedule of that path. The graphical representation is important for clarification and visualization of the itinerary though not necessary exclusively. In a personal communication, Alan Ramlal (1994) of the NYCTA stated that, "trained operators do not need to see a graphical display of the transit route, the itinerary would be adequate". Since this TATIS is geared for an average user, a graphical display is also presented. This graphical display should display the currently selected path with all other paths shown. The paths are color coded and the selected path is the path that is on top of the other paths. The roadway network is also shown so that the user can relate the transit path to the roads traversed as well as the neighborhoods they pass through. Zooming capabilities are an important feature to allow the user to obtain a closer view of portions of the transit path, especially those area involving walking transfers.

The itinerary was created to be in a tabular format since most people can relate to a transit timetable. A complete verbose description of the itinerary is possible, but people tend to get easily distracted by the additional wording and therefore a schedule with only essential information is necessary.

Two macro files were created to produce output for this prototype: OUTPUT_R.AML and OUTPUT_R2.AML. OUTPUT_R.AML produces the

coverages, menus and itinerary files necessary, while OUTPUT_R2.AML displays the output files.

5.9.1 Discussion about OUTPUT_R.AML

The first output macro OUTPUT_R.AML is extensive and as mentioned previously, generates the needed output files, coverages, and menus. The pseudocode for this macro is given below.

First retrieve all information from the user's ".pre" Info file and convert them to indexed variables.
Remove old output coverages if present.
Retrieve the locations and zones from the origin and destination Info tables.
For each path, retrieve the route number, destination, variation, handicap accessibility of each route from the route allocation table (RAT file).
For each path, retrieve the measure, location and zone of each arrival and destination stop of each route from the route point coverage (PAT file).
Open a schedule text file for writing in the format "user-ID".path
For each path, use text strings and write to the text file the schedule or itinerary using the values of the variables obtained.
For each path, reselect the routes defined by the measures of each arrival and destination stop. Write the selection to a file and use the Arc/Info command RESELECT to create a coverage for each of the paths.
Create a menu file for display of output based on the number of paths from the user's ".pre" Info file called "draw.menu".

The creation of the text files, namely the schedule file (".path") and the "draw.menu" file rely on the use of the Arc/Info command FORMAT, which allows for values of variables to be placed in a string using tab settings. In our prototype, there were some difficulties with the use of this command and it was used when absolutely necessary. Instead, character strings were set equal to variables which were then written to text files.

As noted in the pseudocode listing, the graphical output of the paths selected are in the form of coverages. These coverages contain the roadway arcs as well as the route and section definitions for each route (i.e. the extracted route system was copied

with the coverage). In addition, the address definition file was copied with the coverage. Since this file was not needed in the display of the path, it was deleted. It is our opinion that the creation of a separate coverage for each path was the best overall choice out of several available. Choosing to extract and create new route systems for each path doubled the computational time. In addition, using the RESELECT command when displaying the routes also increased overall computational time. It was felt that once the path coverages were created, they could easily be viewed and labeled with minimal effort.

The "draw.menu" file is created each time the prototype is run. This is due to the fact that the number of selected paths may vary each time and therefore the menu must reflect these changes. The original form menu was created using Arc/Info's FORMEDIT command which produced a representative menu text file. This file was then duplicated and modified to be dynamic using the AML macro language.

5.9.2 Discussion about OUTPUT_2R.AML

The second output macro, OUTPUT_2R.AML is very small and displays the results of OUTPUT_R.AML. The graphical display is performed within the Arc/Info ARCPLOT module and is contained within a window. This display contains the roadway network shaded in gray, the transit network, shaded in olive and the individual paths color coded and displayed on top of each other in the order of the menu selection. The viewing area of the paths is composed of the limits of the available paths. Each path is color coded based on a certain scheme; red for the first, green for the second, blue for the third and yellow for the fourth. In the prototype, each path is called by the term "Option" in the draw menu. The option selected is then shown over the other paths and highlighted by a thicker line with the route numbers shown.

Refer to Figure 6.7 in Appendix D for a typical example of the output provided by OUTPUT_2R.AML In this figure, several menus and output forms are seen. The following is a discussion on each item in this Figure. The graphical display was described previously. The "Options Tool" is the "draw.menu" discussed previously. In this menu, the legend describes the color of each path or option and allows the user to select which option to view. A macro is then activated modifying the view to reflect the selected option. This figure displays option number "1".

Below the "Options Tool" is the "Pan/Zoom Tool" which contains icons representing the different zooming functions available. They are from right to left: zoom window, zoom in, zoom out, pan and zoom extents. The zooming functions rely on a macro called ADI_ZOOM.AML provided by the GIS software in the tutorial directory. This macro was modified for use in this prototype. The icons are also taken from this directory.

The itinerary is presented in an xedit window in which its size, position, and font are specified. The Arc/Info POPUP command had no capabilities for modification and therefore was not used (The POPUP command displays text in a scrollable window). Even though the xedit window allows the user to modify the text file, while the POPUP window does not, it was felt that the flexibility in the size and position of the window was a reasonable tradeoff. The schedule is in tabular format indicating for each path the anticipated time of arrival/departure, the origin address, time of departure, first transit stop address, walking distance, and direction to the transit stop.

The main menu for output is found above the graphical display and is titled "Output Menu for Transit". This menu provides many capabilities and provisions for future enhancement. The "Zoom" and "View Options" buttons activates the "Pan/Zoom Tool" and the "Options Tool" respectively. Though these two tools are automatically activated when OUTPUT_2R.AML is run, if the user accidentally closes one of these

two tools, he can re-issue the two menus by selecting the respective buttons. The "Output" button produces a small submenu which allows for printing of the schedule file and map. These two output function are executed by small macros. The "Other" button produces a submenu which allows for modifications and further information. This submenu allows for modification of user preferences, modification of origin, destination, and/or time, and a list of the next three departure times of the first route of the path selected. The macro that allows for modification of user preferences uses the R_PREFERENCE.AML macro and uses the menu originally used in obtaining user preferences (PREFERENCE.MENU). This macro basically repeats the process described in this prototype from the point of determination of preferences through generation and display of output. The last item, finding the next three departure times of the first route, produces a small display below the graphical display in the left corner and is for display purposes (i.e. cannot be modifiable). This option uses the run number obtained in the determination of travel time and finds the next three successive departure times. This macro is contained in the THREE_O_DEPART.AML file. The last button on the main menu is the "Quit" button which deletes all threads and variables and exits the user from the TATIS. Figure 5.11 presents a schematic of the OUTPUT_R.AML file and is discussed further in section 5.10.

In order to run the several menus and output options at the same time, Arc/Info threads were used. A thread carries input from a AML program, keyboard, or an AML menu to the AML processor. Input from these sources stack on the thread with the most current at the top of the stack. With a single thread, the user can interact with only one input source at a time. Multiple threads can be created to allow for multiple input sources (ESRI, 1993, 13.14).

facing 75

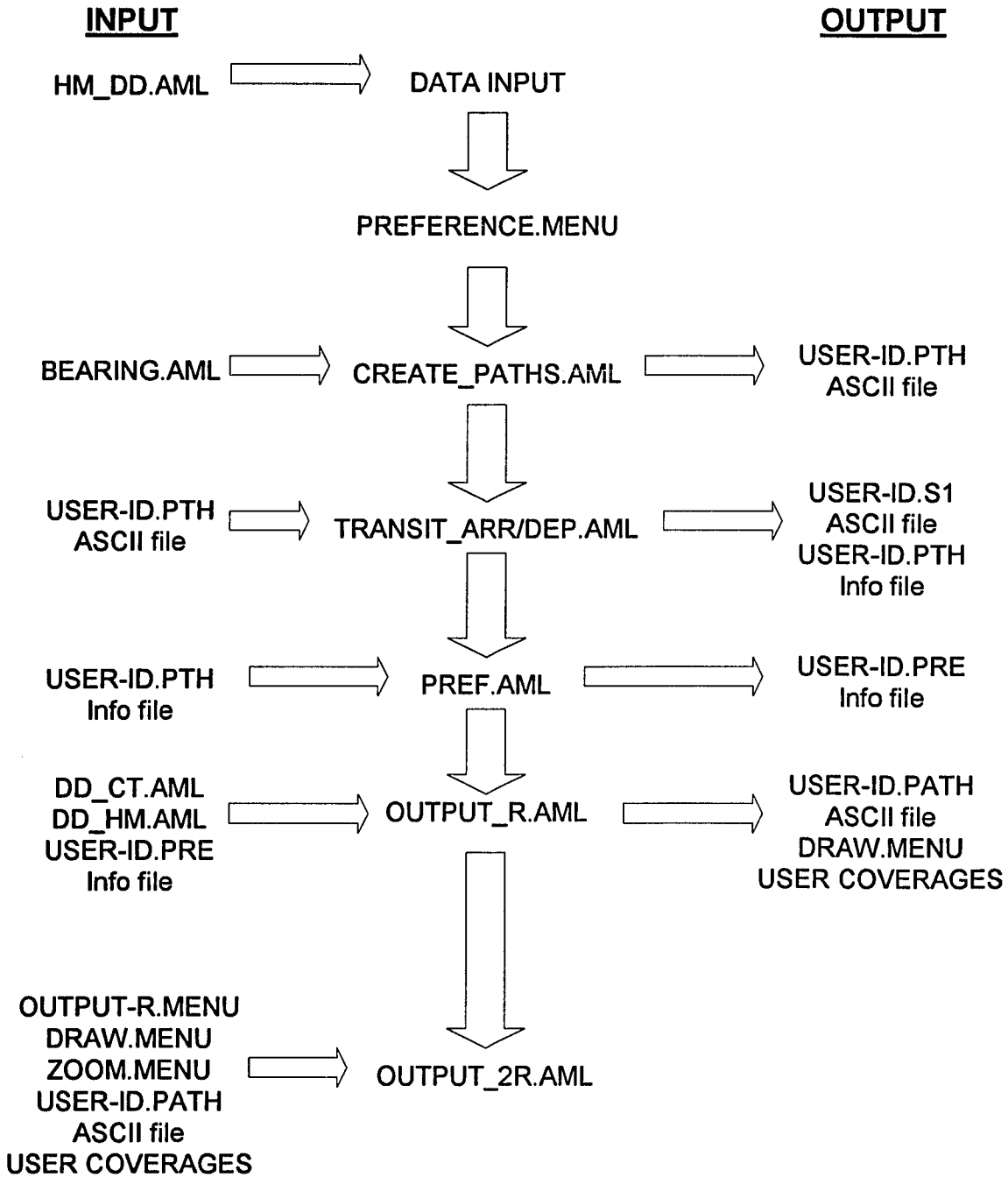


Figure 5.10 Schematic of TRANSIT-A.AML

5.10 Integration of Prototype Algorithms

Previously, each individual macro was discussed independently of each other. In actuality, they form a sequence that rely on each other. The macro that combines all of the individual components of this prototype's algorithm are contained in the TRANSIT-A.AML. A schematic of this macro is shown in Figure 5.10. The items in the center column are the sequence of programs run in this macro. To the left of this sequence are the files, macros, menus and coverages used by these programs. To the right of the sequence are the files, menus and coverages created by each macro in the sequence. All of the output for each individual user is stored in a separate subdirectory with the users-ID as the name of the subdirectory.

This algorithm starts with data input. This input can be from any source but contains the user-ID, origin and destination address, and time of arrival/departure. It is assumed that these values are error-checked prior to running the algorithms. The HM_DD.AML macro converts the time into decimal days. Next the user preference menu is run where the user preference items are set equal to variables. The CREATE_PATHS.AML macro is run to determine possible paths between an origin and destination and uses the BEARING.AML macro to determine the direction between two points. The output generated by the CREATE_PATHS.AML is the temporary user-ID.pth file. This file is used by the TRANSIT_ARR.AML and TRANSIT_DEP.AML macros which determine the travel times and schedules for each path. The output of these macros is a temporary user-ID.s1 file in which each row contains all the variable values of a path separated by commas, and the user-ID.pth Info file which is produced from the user-ID.s1 file. The user-ID.pth file is used by the PREF.AML to determine the paths which meet the user's preferences and those paths are output to the user-ID.pre Info file. The OUTPUT_R.AML macro takes this user-ID.pre file and with the help of DD_CT.AML and DD_HM.AML macros (which

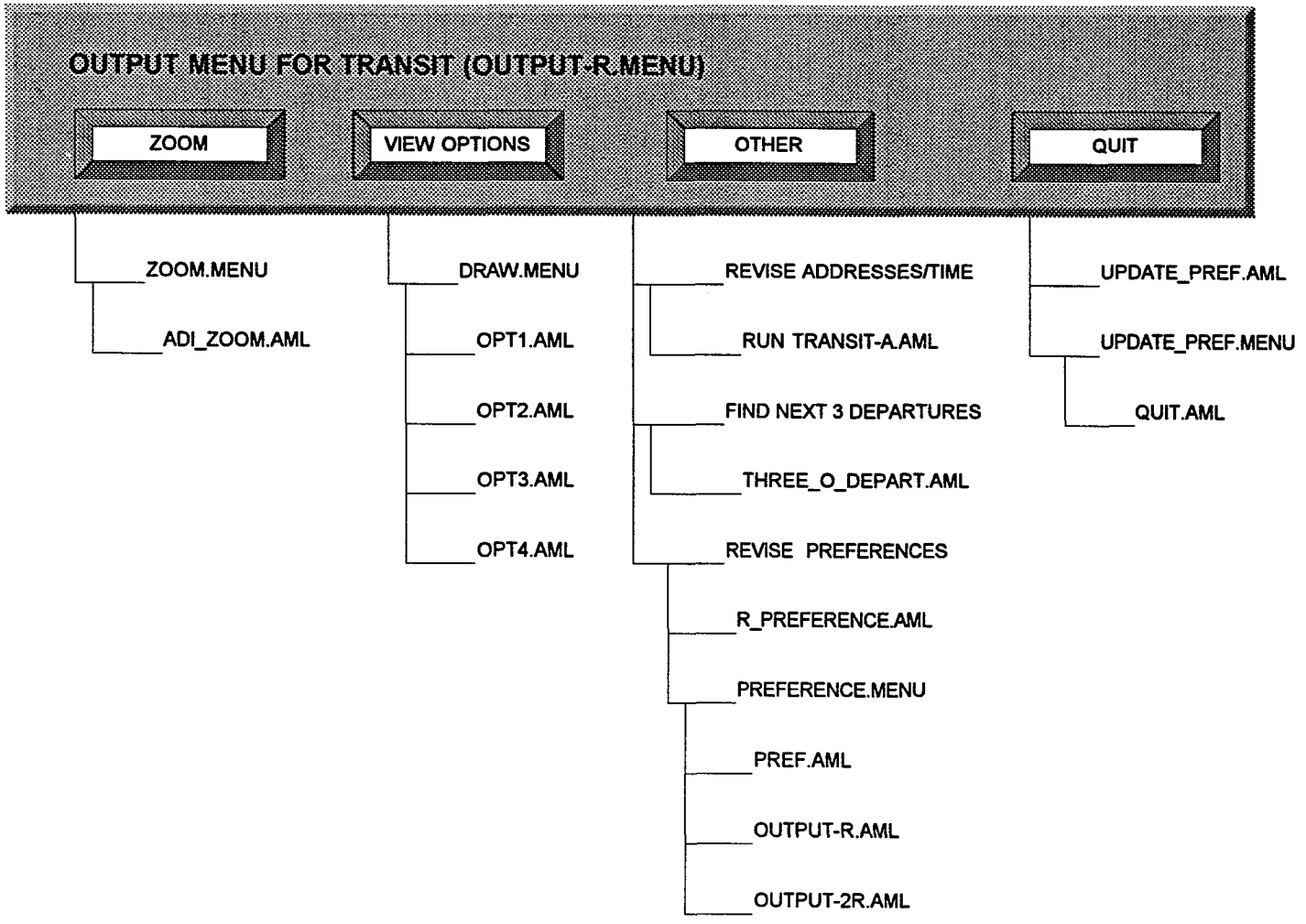


Figure 5.11 Schematic of Main Menu

convert decimal days into hours and minutes) produces the user-ID.path file which contains the itinerary, the draw.menu, and the path coverages. The OUTPUT-R.MENU, DRAW.MENU, ZOOM.MENU, user-ID.path file, and the path coverages are used by the OUTPUT_2R.AML to produce the final display as shown in Figure 6.7. in Appendix D.

The schematic presented is a simplified version of TRANSIT-A.AML and the major input used. In actuality, many additional databases are used such as the point attribute table (PAT), route attribute table (RAT), and schedule files. In addition, the TRANSIT-A.AML macro provides for situations where no or minimal changes are made to the data input. In these cases, not all of the algorithms are run, only a few are activated. The order of activation depends on which data is modified. For example, if the origin or destination is modified, the macro starts from CREATE_PATH.AML and progresses downward. If the preferences are changed, the macro starts from PREF.AML and progresses downward. Furthermore, the TRANSIT-A.AML provides for precalculation of origin-destination pairs, notably home-work, and work-home trips. This provision allows for easy retrieval of known and relatively unchangeable origin-destination pairs. In order to allow for this provision, all output and input are provided a prefix which is reflective of the origin-destination pair. For example "-hw" for home-work trips, "-wh" for work-home trips, and "-p" for one other trip.

Once the OUTPUT_2R.AML macro is run in the TRANSIT-A.AML macro. The OUTPUT-R.MENU is presented above the graphical display. This menu is the main menu for the output options. A schematic of this menu and the programs used is presented in Figure 5.11. The "Zoom" button brings up the ZOOM.MENU which runs the ADI_ZOOM.AML macro. The "View Options" button activates DRAW.MENU which allows each path to be viewed. By selecting a pushbutton, one of four macros is run (OPT1.AML, OPT2.AML, OPT3.AML, and OPT4.AML).

The "Other" button brings up a submenu which allows three functions to be performed. To "Revise Addresses/Time" the TRANSIT-A.AML is run over again asking questions regarding new input. To "Find the Next 3 Departures" the THREE_O_DEPART.AML macro is run. To "REVISE PREFERENCES" the R_PREFERENCE.AML is run which activates the PREFERENCE.MENU. Once selected, the PREF.AML, OUTPUT-R.AML and OUTPUT-2R.AML are then run. The "Quit" button activates the UPDATE_PREF.AML macro. This macro uses the UPDATE_PREF.MENU and asks the user whether he wishes to keep the changes made to the user preferences, if any. The QUIT.AML macro is then run which deletes all threads and variables and exits the TATIS program.

CHAPTER 6

SAMPLE ANALYSIS

6.1 Introduction

The preceding chapters have provided an overview of the prototype's purpose, data, algorithms, and components. This chapter turns to an example of the prototype in action. A sample origin and destination is given and the resulting interface menus and output forms are reviewed. Several sample origin-destination pairs are computed and the performance of the prototype is measured, discussed and summarized.

6.2 Sample Test Case

The following is a brief discussion of the prototype and a typical query. Figures are presented to aid in the understanding.

Figure 6.1 and Figure 6.2 show the prototype transit network developed for Union County, New Jersey. There are a total of eight routes (16 looking at inbound and outbound as separate routes), seven of which are local routes, one is an express route. The routes were chosen use major collector and arterial roads. Chapter 5 discusses their generation more specifically.

Figures 6.3 through Figure 6.16 in Appendix D progress through a typical query using this prototype. First the user enters an ID number (Figure 6.3). Then the user specifies his origin and destination addresses and time (Figure 6.4). Time is given as "arrival" ("I want to be at a location by 8:00 am), or "departure" (I am leaving my house at 6:30 am). The menu in Figure 6.4 is not a finished form but allows for entry of data into the prototype. Intersections are used since they can easily be located and verified with a map.

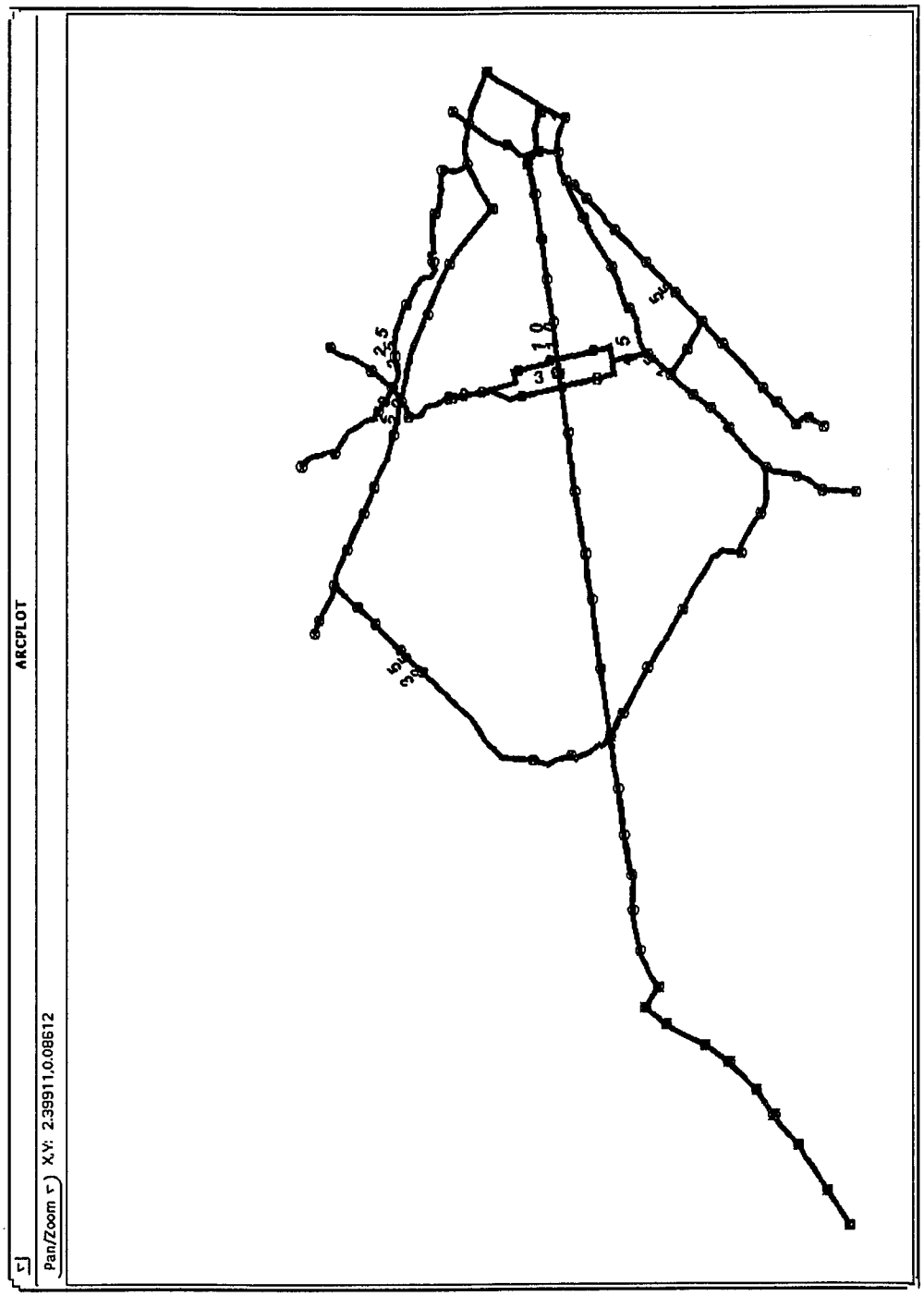


Figure 6.1 Transit Route Network

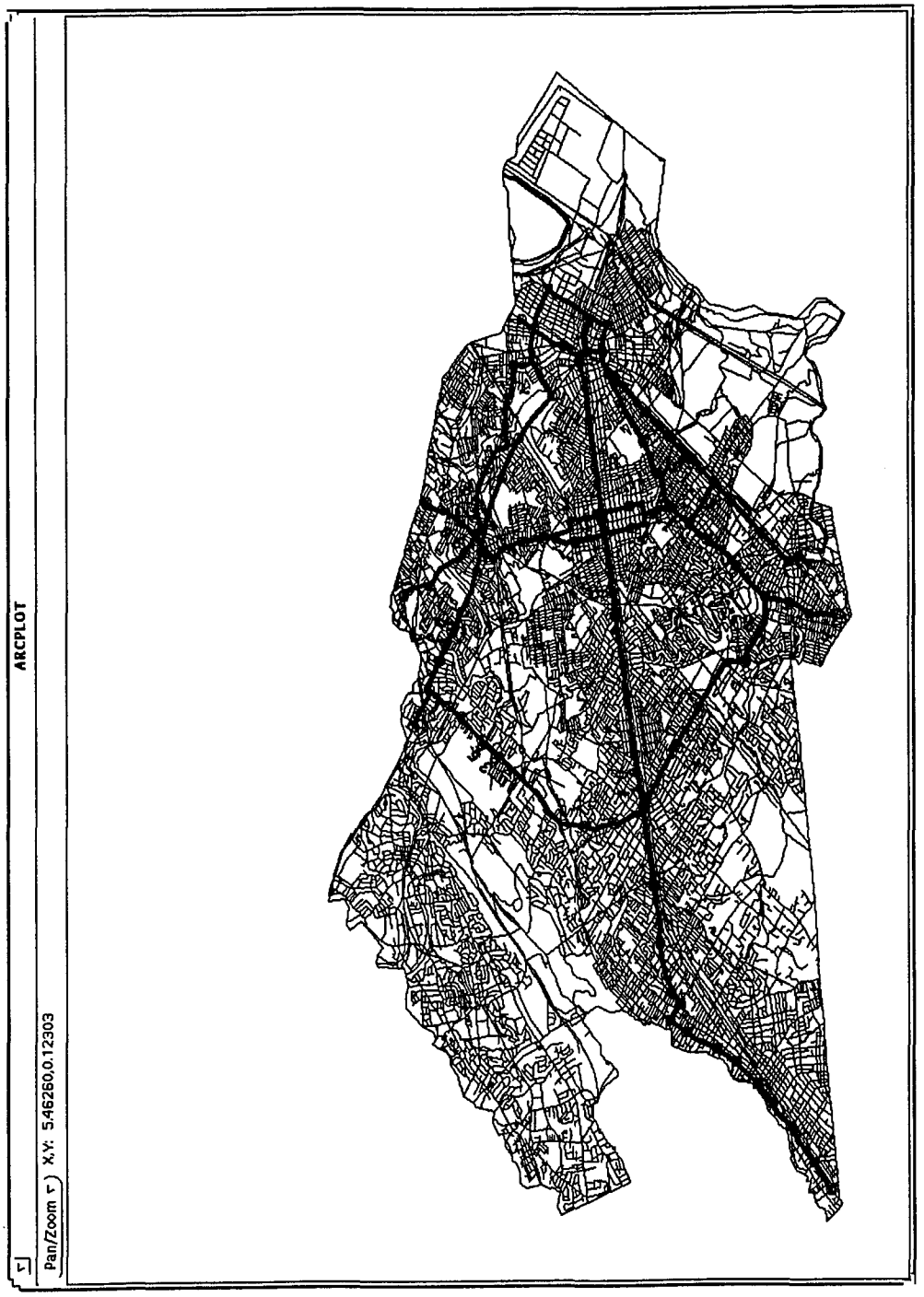


Figure 6.2 Transit Route Network with Road Network

Figure 6.5 displays the user preference menu, where the user enters his preferences as regards to mode, transfers, maximum walking time and maximum waiting time. The program then progresses, finds all available paths, finds their travel times and sorts them, and then selects those that meet the user's criteria. Figure 6.6 shows the message given to the user and allows them to refine their criteria. Once done, the output files are generated and displayed. Figures 6.7 to 6.10 display the view the user sees when the respective option is chosen. Note that a map of the path and itinerary are displayed. The itinerary is in a tabular format that is easy to follow. The output of the itinerary is given in Appendix E. The next three departures are given as well. The figures previously mentioned also demonstrate that the path closer to the origin and destination usually is the path with the shortest overall travel time (though this may not be the case always).

The menus are simple and easy to understand. A brief description of each menu follows; Chapter 5 contains an in-depth account. A draw menu is presented which displays how many routes were found and allows the user to select the push-button to view the path. These paths are color coded and can be viewed in reference to each other. This is provided so that the user can see the difference between each route.

A zoom menu is provided and allows the user to perform the following functions by selecting the appropriate icon: zoom in providing a window, zoom in providing a center, zoom out providing a center, pan, and zoom to the extents of the selected routes.

A general menu (called "Output Menu for Transit") is provided above the display and in addition to allowing the user to review the zoom and draw menus, an output menu is provided which allows the itinerary and map to be printed. An "Other" menu is provided and allows the user to revise his origin/destination/time or user preferences. If the user chooses to revise his preferences, an appropriate menu displays

for selection (Figure 6.11). Once the choices are made, the module will inform him, via menus, how many paths meet that criteria and whether he would like to make more revisions (Figure 6.12). If the answer is "No", the module will present him with those paths that meet his revised criteria (Figures 6.13 to 6.15). Included in the general menu is a "Quit" item. When the user selects this item, a menu is displayed informing him of the changes made to his preferences and asks whether he would like to keep these changes (Figure 6.16). After answering this question, the user exits the system.

6.3 Performance Measures

The previous section described the interface between the user and the computer in using the prototype for a specific origin-destination pair. Measures of the prototype's performance are necessary to demonstrate its real-world applicability. In addition to demonstrating that the prototype works in the manner desired, it may be helpful to analyze the performance as a function of time providing indications of how much time are needed to perform the various algorithms. From these indications or measures of time, comments can be formed, which will be discussed further in Chapter 7.

6.3.1 Description of Test Scenarios

Six different origin-destination pairs were selected for analysis. Each individual origin-destination pair was run twice; once in the forward direction, and once in reverse. This produced a total of twelve different runs. The six origin-destination pairs are described as follows:

Pair 1 (Producing runs 1-A and 1-B)
Origin: Intersection of North Avenue and Maple Street, Garwood, NJ
Destination: Intersection of Burlington Avenue and Morris Avenue, Union, NJ

Pair 2 (Producing runs 2-A and 2-B)
Origin: Intersection of Ross Street and Saint George's Avenue, Linden, NJ
Destination: Intersection of Chestnut Street and North Avenue, Garwood, NJ

Pair 3 (Producing runs 3-A and 3-B)
Origin: Intersection of Sheridan Avenue and Westfield Avenue, Roselle Park, NJ
Destination: Intersection of Morris Avenue and Potter Avenue, Union, NJ

Pair 4 (Producing runs 4-A and 4-B)
Origin: Intersection of Liberty Avenue and Morris Avenue, Union, NJ
Destination: Intersection of Burnet Avenue and Vaux Hall Road, Union, NJ

Pair 5 (Producing runs 5-A and 5-B)
Origin: Intersection of Springfield Avenue and Vaux Hall Road, Union, NJ
Destination: Intersection of Gordon Street and Westfield Avenue, Roselle Park, NJ

Pair 6 (Producing runs 6-A and 6-B)
Origin: Intersection of Emerson Avenue and Main Street, Rahway, NJ
Destination: Intersection of Chilton Street and Rahway Avenue, Elizabeth, NJ

The final output of each pair (runs 1-A to 6-A) is displayed from Figure 6.17 to Figure 6.22 in Appendix F. On all of the previously mentioned figures, the first option is highlighted. The origin-destination pairs were chosen to produce significantly different paths from each other and to produce a variety in the number of paths produced.

The definition of the performance measure used in describing this prototype is the computational time (in minutes) to perform a specific task. The tasks include performance of individual sections of a particular algorithm, the total specified algorithm, and the total prototype (which is a compilation of all the algorithms). The major algorithms used in this prototype are CREATE_PATHS.AML (creation of paths), TRAVEL_ARR.AML or TRAVEL_DEP.AML (determination of travel time of each path), PREF.AML (restriction of paths meeting user criteria), OUTPUT-R.AML (creation of tangible forms of output, e.g. files, menus, coverages), and OUTPUT-2R.AML (display of output) (See Chapter 5).

Table 6.1 Summary of Performance Intervals

RUN	1-A (Min.)	1-B (Min.)	Average (Min.)	Percent Of Total	2-A (Min.)	2-B (Min.)	Average (Min.)	Percent Of Total	3-A (Min.)	3-B (Min.)	Average (Min.)	Percent Of Total
Task 1	0.20	0.20	0.20	0.9%	0.23	0.18	0.21	0.8%	0.18	0.25	0.22	0.7%
Task 2	2.12	2.33	2.23	9.5%	2.70	2.03	2.37	9.5%	2.08	2.38	2.23	7.2%
Task 3	1.23	1.22	1.23	5.3%	2.10	1.18	1.64	6.6%	1.78	1.83	1.81	5.8%
Task 4	14.45	13.52	13.99	60.0%	18.56	12.12	15.34	61.4%	20.75	21.90	21.33	68.5%
CREATE_PATHS TOTAL	18.00	17.27	17.64	75.6%	23.59	15.51	19.55	78.3%	24.79	26.36	25.58	82.2%
Task 5	0.70	0.57	0.64	2.7%	0.37	0.40	0.39	1.5%	0.68	0.58	0.63	2.0%
Task 6	0.27	0.47	0.37	1.6%	0.22	0.22	0.22	0.9%	0.32	0.40	0.36	1.2%
TRAVEL-ARR/DEP TOTAL	0.97	1.04	1.01	4.3%	0.59	0.62	0.61	2.4%	1.00	0.98	0.99	3.2%
PREF TOTAL	0.40	0.40	0.40	1.7%	0.40	0.35	0.38	1.5%	0.63	0.53	0.58	1.9%
Task 7	0.20	0.25	0.23	1.0%	0.18	0.17	0.18	0.7%	0.22	0.27	0.25	0.8%
Task 8	0.02	0.02	0.02	0.1%	0.03	0.02	0.03	0.1%	0.02	0.02	0.02	0.1%
Task 9	0.45	0.55	0.50	2.1%	0.50	0.30	0.40	1.6%	0.35	0.43	0.39	1.3%
Task 10	0.37	0.40	0.39	1.7%	0.18	0.15	0.17	0.7%	0.17	0.15	0.16	0.5%
Task 11	1.42	1.47	1.45	6.2%	2.68	1.43	2.06	8.2%	1.47	1.42	1.45	4.6%
Task 12	0.38	0.38	0.38	1.6%	0.28	0.32	0.30	1.2%	0.35	0.42	0.39	1.2%
OUTPUT-R TOTAL	2.84	3.07	2.96	12.7%	3.85	2.39	3.12	12.5%	2.58	2.71	2.65	8.5%
OUTPUT-2R TOTAL	1.33	1.32	1.33	5.7%	1.40	1.23	1.32	5.3%	1.28	1.40	1.34	4.3%
TOTAL TIME	23.54	23.10	23.32	100%	29.83	20.10	24.97	100.0%	30.28	31.98	31.13	100.0%
Time Excluding Task 4			9.34				9.63				9.81	
Number of Paths	5	5			5	5			7	7		
Number of 1 Route Paths	0	0			0	0			0	0		
Number of 2 Route Paths	1	1			1	1			1	1		
Number of 3 Route Paths	4	4			4	4			6	6		

RUN	4-A (Min.)	4-B (Min.)	Average (Min.)	Percent Of Total	5-A (Min.)	5-B (Min.)	Average (Min.)	Percent Of Total	6-A (Min.)	6-B (Min.)	Average (Min.)	Percent Of Total
Task 1	0.33	0.17	0.25	2.1%	0.18	0.20	0.19	1.3%	0.27	0.35	0.31	1.5%
Task 2	2.83	2.10	2.47	20.8%	2.05	2.10	2.08	14.4%	2.32	2.77	2.55	12.4%
Task 3	1.20	1.15	1.18	9.9%	1.17	1.20	1.19	8.2%	1.77	1.77	1.77	8.6%
Task 4	5.78	3.63	4.71	39.7%	7.10	7.83	7.47	51.8%	12.27	10.45	11.36	55.4%
CREATE_PATHS TOTAL	10.14	7.05	8.60	72.5%	10.50	11.33	10.92	75.7%	16.63	15.34	15.99	77.9%
Task 5	0.10	0.08	0.09	0.8%	0.18	0.18	0.18	1.2%	0.17	0.18	0.18	0.9%
Task 6	0.52	0.15	0.34	2.8%	0.22	0.22	0.22	1.5%	0.37	0.48	0.43	2.1%
TRAVEL-ARR/DEP TOTAL	0.62	0.23	0.43	3.6%	0.40	0.40	0.40	2.8%	0.54	0.66	0.60	2.9%
PREF TOTAL	0.63	0.22	0.43	3.6%	0.28	0.42	0.35	2.4%	0.50	0.63	0.57	2.8%
Task 7	0.20	0.07	0.14	1.1%	0.08	0.12	0.10	0.7%	0.18	0.17	0.18	0.9%
Task 8	0.02	0.02	0.02	0.2%	0.02	0.03	0.03	0.2%	0.02	0.02	0.02	0.1%
Task 9	0.37	0.13	0.25	2.1%	0.22	0.25	0.24	1.6%	0.38	0.40	0.39	1.9%
Task 10	0.07	0.05	0.06	0.5%	0.08	0.10	0.09	0.6%	0.08	0.13	0.11	0.5%
Task 11	0.53	0.45	0.49	4.1%	0.82	0.85	0.84	5.8%	0.82	1.13	0.98	4.8%
Task 12	0.45	0.15	0.30	2.5%	0.25	0.30	0.28	1.9%	0.38	0.45	0.42	2.0%
OUTPUT-R TOTAL	1.64	0.87	1.26	10.6%	1.47	1.65	1.56	10.8%	1.86	2.30	2.08	10.1%
OUTPUT-2R TOTAL	1.30	1.02	1.16	9.8%	1.17	1.20	1.19	8.2%	1.20	1.37	1.29	6.3%
TOTAL TIME	14.33	9.39	11.86	100.0%	13.82	15.00	14.41	100.0%	20.73	20.30	20.52	100.0%
Time Excluding Task 4			7.16				6.95				9.16	
Number of Paths	1	1			2	2			3	3		
Number of 1 Route Paths	0	0			0	0			1	1		
Number of 2 Route Paths	1	1			0	0			1	1		
Number of 3 Route Paths	0	0			2	2			1	1		

facing 84

The sections of each algorithm are broken down into smaller definitions or tasks numbers for the sake of demonstrating the time needed to implement these tasks. Their breakdown into specific task numbers are based upon the modules of AML code used to create the algorithms. They indicate major break points in the definition of the algorithms. The tasks are defined as follows:

CREATE_PATHS.AML
Task 1: Creation of origin-destination Info files.
Task 2: Finding the transit routes in the origin and destination buffer zones.
Task 3: Locating the nearest transit stop of each of the transit routes in Task 2.
Task 4: Creation of all possible paths, writing results to a temporary text file.

TRAVEL-ARR/DEP.AML
Task 5: Reading temporary text file from Task 4, saving to indexed variables and finding schedules.
Task 6: Writing results to a temporary text file, creation of an Info file, importing the text file into the Info file, and sorting.

OUTPUT-R.AML
Task 7: Assigning Info attributes to indexed variables, remove old output coverages.
Task 8: Retrieve route information about each path.
Task 9: Obtain linear distances or "measures" of each route in each path.
Task 10: Create itinerary ASCII text file.
Task 11: Reselect and produce output coverages of each path.
Task 12: Generate DRAW.MENU

6.3.2 Discussion of Results

A summary of the performance times of the twelve runs discussed previously is presented in Table 6.1. This table includes time intervals for each of the tasks mentioned in the prior sections, total time for each algorithm and total time to run the prototype. Included are the number of paths generated for each run and their breakdown in terms of the number of routes that compose each path. Averages for each pair of runs (i.e. 1-A and 1-B) as well as percentages of each time with respect to the total time are presented additionally.

From Table 6.1 several key points and trends can be pointed out; they are as follows:

- The total time to run the prototype is significant. Even the lowest time (11.9 minutes) is beyond the expectations and limitations of a normal user to the system; though the total time values are within reason in the performance of a computer application in a typical engineering office. For example, the calculation of a mathematical model of a surface using CADD applications may take the same amount of time.

- The total time increases as the number of paths increase. This fact is demonstrated in Figure 6.23 where a linearly increase in time is demonstrated as the number of paths found increase.

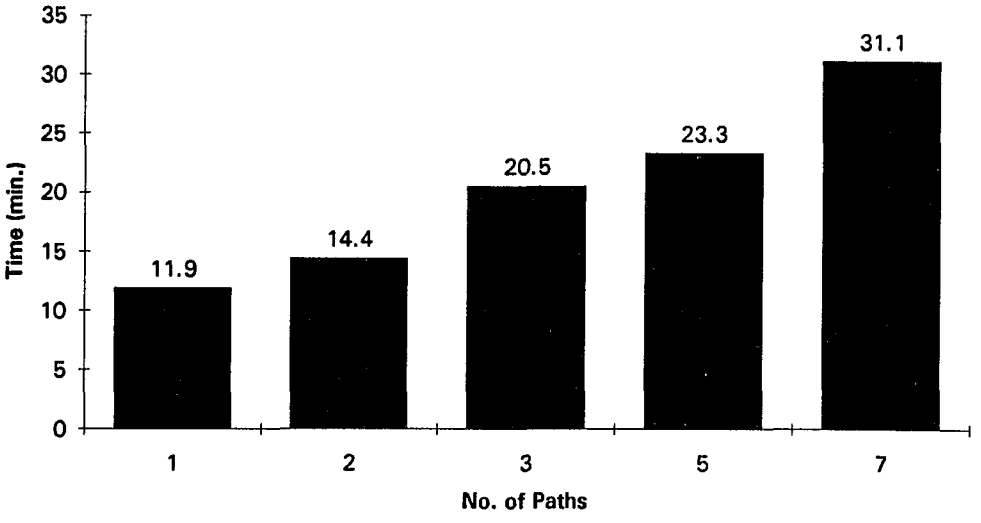


Figure 6.23 Total Time vs Number of Paths

- The majority of the total time is comprised from the CREATE_PATHS.AML algorithm. The percentage of time used by this algorithm increases from 72.5% for 1 path to 82.2% for 7 paths.

- Of the CREATE_PATHS.AML algorithm, task 4 (creation of all possible paths) takes the majority of the total time and increases from 39.7% for 1 path to 68.5% for 7 paths. More paths require more computational time.

- Task 2 (finding transit routes in the origin and destination buffer zones) takes the next largest amount of time, but remains constant at about 2 to 3 minutes irregardless of the number of routes. Task 2 is independent of the number of paths created.

- Excluding task 4 from the total time computation results in virtually the same amount of time required to perform the analysis. The remaining time is about 7 minutes for 1 to 2 paths and then jumps to 9-10 minutes for 3 to 7 paths. This increase in time can be accounted for by the increase in the number of paths.

- The implementation of the OUTPUT-R.AML algorithm requires from 1.3 minutes to 3.1 minutes generally depending on the number of paths generated. The task that requires the most time of this algorithm is task 11 (the reselection and creation of the output coverages).

- For each pair of runs (1-A and 1-B), the total time is generally the same. Exceptions are found between run 2-A and 2-B and between run 4-A and 4-B. The differences between the two sets of runs are present throughout each analysis, even in tasks that are independent of the number of paths created (e.g. task 2). Since no other person was accessing the file server that the prototype resides on, it is difficult to find an explanation for these differences.

6.3.3 Summary of Results

Based upon Table 6.1 and the previous discussion, the CREATE_PATHS.AML algorithm and especially task 4 takes a majority of the computational time of the prototype. In addition, even if the computational time of task 4 was reduced to several seconds, the remainder of the prototype would require at most 10 minutes. Ten minutes is an inadequate response time for an ATIS when the attention span of an average user is far less. In order to encourage users of the prototype, it must be tailored to be responsive to the time that most people are willing to wait for information. Based upon the author's experiences, this time is judged to be under 30 seconds. The computational time additionally increases when the number of paths increase. Since this prototype used only 16 routes, it is also anticipated that the computational time would increase when the number of routes increased to represent actual conditions.

While the computational time of the algorithms appears to be a negative attribute, it must be taken into the context of the success of the algorithms. The large computational time indicates a weakness of Arc/Info, not the methodology of the algorithms which are robust and stable. This weakness demonstrates the need for a different approach to the implementation of the algorithms. Chapter 7 presents conclusions based upon the results of the sample analyses and provides further possible improvements which include using a compiled programming language such as "C" for a substantial portion of the prototype.

CHAPTER 7

CONCLUSIONS

7.1 Introduction

The preceding chapters described:

1) the motivations for this thesis;

2) the literature that serves as a foundation for this work;

3) the methodology and design decisions;

4) the structure of the data;

5) the details of the implementation;

6) a sample analysis using the prototype

In this concluding chapter, the success of the prototype is evaluated, its deficiencies discussed, and improvements suggested. This chapter begins with general conclusions. The next section provides recommendations for future work giving specific improvements to the prototype.

7.2 General Conclusions

This thesis developed a method for the design and use of a Transit Advanced Traveler Information System (TATIS) using an off-the-shelf Geographic Information System (GIS). This research had several goals which are the following: 1) representing multi-modal transit networks in a digital form with schedule databases; 2) development of a multiple "optimal" path algorithm that takes into account walking transfers using published time schedules; 3) incorporating user preferences and penalties in the algorithm; 4) development of a user-interface with suitable output capabilities; and

5) using the prototype for sample inquiries giving performance measures. This thesis has demonstrated that the above five goals are possible using a prepackaged GIS alone.

7.2.1 GIS Packages

There are several advantages in using an "off-the-shelf" fully-functional GIS package alone. These type of packages provide many of the functions needed in the development of a TATIS such as path finding and dynamic segmentation. In addition, they provide the tools needed to develop user-interfaces. GIS packages provide the ability to create maps and complex visualizations. Furthermore, GIS software provide macro languages by which system developers can create capabilities not provided by either automatically, improving on them or substituting them as necessary. Developers can add functionality to meet new needs rather than waiting for the vendor of a closed, single-purpose system to add a necessary feature in a new release. Several GIS packages provide macro languages, and therefore if the methodology is complete, substituting one fully functional GIS package with another compatible one is possible.

One of the disadvantages of using a prepackaged GIS is that many of its intensive functions rely on the speed of the equipment they reside on. Using a slow hard disk, limited size of RAM or a slow CPU will slow the performance time of the packages. The stability of the platforms or versions is another issue. The vendors may alter key parts of their data schemes, jeopardizing the existing data built on the old structure. Similarly, new software versions may cripple, remove or change the behavior of some old features. Therefore, developers who use these GIS packages must anticipate that some development time will routinely have to be devoted to supporting new versions of the underlying software as it is released. Nevertheless, if the methodology behind the algorithm is correct and has been suitably tested, these migration efforts would appear small in comparison to duplicating the functionality of

these packages alone. The choice of how much one is willing to rely on the package is also dependent on other factors such as response time and programming effort. Next, each of the five goals will be examined providing advantages and disadvantages.

7.2.2 Representation of Transit Networks

The first goal was the representation of multi-modal transit networks with schedule databases. This goal was accomplished by relying on the address-matching and dynamic segmentation capabilities of Arc/Info. Text files describing the route and transit stops were converted into a transit route system and point coverages respectively. By using text files, there is a level of independence from the geography of the base map, easing the task of recreating the transit network for a new base map. Travel time was incorporated by converting text files containing the time schedule of each route into a database file. Since only critical time points are given in a printed time table, time values of intermediate points were interpolated.

The transit network relies on the Union county TIGER/Line files. Difficulties such as missing zip code ranges, address ranges, street names, and arcs hindered the success of this task. In addition, since transit data was unavailable at the development stage of this research, a transit prototype network was developed. The transit network was designed to be representative of a typical transit network with variations in time schedules. This fact easily led to the use of text files and attribute representation described in the previous paragraph. If actual transit data from NJ Transit was used in the development of this prototype, the manner and method of representing the transit network may have been different. Additionally, the scope of the prototype was restricted to one county in Northern New Jersey, Union County. Ideally, the entire Northern New Jersey region should be represented including transit routes by different carriers. Modifying the route identifier number (described in Chapter 4) and using the

implementation methods described in Chapter 5 would allow this possibility to become reality. Another drawback in the representation of the transit network is that since it relies upon the TIGER/Line database, the network's coordinate system is in terms of Latitude and Longitude. This system was used to maintain integrity with the other modules of the ATIS development team. As a result, all measures of distance used a conversion factor to obtain distances in terms of miles. An easy solution would have been to use the Arc/ Info PROJECT command to convert the roadway network into state plane coordinates.

A shortcomming in the representation of the transit network is the reliance upon timetables in determining travel time and the linear interpolation used in obtaining intermediate stops. This assumes that transit speeds are constant between listing time-points and that no delays occur. In actuality, delays do occur and the speed of the transit vehicle is not constant. A solution to this difficulty uses Global Positioning Systems (GPS) with Automatic Vehicle Location (AVL) to obtain real-time positions and times of transit vehicles. Knowing actual positions and times can more accurately predict itineraries.

7.2.3 Development of Multiple "Optimal" Paths

The second goal was the development of a multiple "optimal" path algorithm that takes into account walking transfers using published time schedules. Since Arc/Info uses the arc coverage, rather than the route systems built upon it, as the basis for network analysis, another method of finding optimal paths was determined. Therefore, Arc/Info's network modeling network was not useful in this prototype except as a tool for building the transit network representation. This goal was achieved by using a connectivity matrix which indicates if a transit route intersects with another route. Each transit route about an origin is compared with the transit route about a destination

using the connectivity matrix and comparison of its transit stops. Other measures are used to prevent illogical or looping paths. The travel time for each path is obtained from the schedule database, and then the list is sorted based on lowest travel time. An itinerary for each path is obtained for situations where time of departure or time of arrival is given. This ability, especially working backward from a time of arrival, is difficult for a person to perform but is easy for a computer. The ability to handle walking transfers, especially without digitizing the countless number of transfer links possible is a major accomplishment of this research.

Currently, the prototype is designed to handle two possible transfers. Modification to the algorithm is possible to handle more transfers; it is assumed that the computational effort to perform this task would increase significantly. An additional future improvement would, in the computation of transfer times, provide optimization to minimize transfer time or coordinate departure times. The method of determining paths is heuristic in nature and special situations must be accounted for (i.e., transferring first at a 0.25 mile distance when close by is a 0 distance transfer). If transfer links were easily available or possible, a revision to this goal would be to use a standard network algorithm.

7.2.4 User Preferences

The third goal was incorporating user preferences and penalties in the algorithm. This goal was accomplished by providing the user a menu with a choice of preferences. The user would choose his preferences and those paths that meet his criteria would be extracted from the list of possible paths. The user is told how many paths out of the total meet his criteria and allows for revision of preferences. This goal primarily uses the Arc/Info RESELECT command. This ability to provide for user preferences is an unique addition that sets this research apart from others.

7.2.5 Development of User-interface and Output

The fourth goal is the development of an user-interface with suitable output capabilities. By using the graphical display and menu interface function, this goal was accomplished. The user only has to point-and-click to see the next route, zoom in for a closer look, print or plot output and even find the next three departure times of the first route of the selected path. The output forms and menus strive to be clear and user-friendly; no typing is required.

Due to time constraints, there were several shortcomings to the output provided. The first is the lack of roadway annotation. Annotation is necessary for the user to clearly understand the path the route takes and what critical roads it passes. It also provides the user with a sense of bearing, where the user may see a familiar road and then be able to judge relationships to it.

There is a single graphical display window. This window displays the first four routes possible in relation to each other with the selected route highlighted by a thicker lineweight. In order for the user to clearly see the relationship of each route to all other routes, several graphical displays could be more useful. In this situation, the path selected would be in a main window and the other routes in separate smaller windows. This possibility would be easily to accomplish by using the map composition functions of the ARCPLOT module in Arc/Info.

7.2.6 Performance Measures

The fifth goal was to use the prototype for sample inquires giving performance measures. Chapter 6 demonstrated that the prototype was capable of handling a variety of inquiries successfully. The computational time required though, ranged from 11.9

minutes to 31.1 minutes as displayed in Figure 6.23. This response time demonstrates some of the weaknesses of using Arc/Info alone.

Arc/Info's AML macro language is interpretive instead of being compiled. This means that each line of AML code is executed once at a time. This fact slows the speed of execution. In addition, many of the commands, such as the NEAR command which the algorithm uses extensively, take several seconds to run. These two factors demonstrate that using a GIS package alone though will perform the analysis, has an inadequate response time. These weaknesses of Arc/Info do not negate the methodology of the algorithm, but necessitate a different approach to the implementation. A possible approach is described in the next section.

7.3 Directions for Future Research

In the preceding section the goals of this research, the shortcommings of their implementation and possible improvements were detailed. Here areas of future research or major improvement to this prototype are presented.

The first and major improvement to this prototype is the use of a programming language to implement a majority of the functions of the algorithms. The optimal language would be "C". This popular language interfaces well with Arc/Info, and runs quickly and efficiently when compiled. It is assumed by translating many of the functions of this prototype to the C programming language, the response time would be under one minute.

The second improvement would be the incorporation of real-time data into the algorithm by use of AVL technologies. This improvement would provide accurate schedule and transfer times. It is anticipated that transit agencies would release this data even if the data indicates a low "on-time" rate.

Another area of future research is the incorporation of a fare system into the algorithm. Incorporating a fare structure into the prototype would not be difficult. Each stop along a route could be assigned a zone number; by knowing the origin and destination stop of each route the difference in zones could be found and a fare value assigned. The cost of transfers could also be incorporated into the prototype.

A fourth area of future research is to develop user penalties or negative weighting factors for different parts of the transit trip. Based upon an user's response to several questions, one could derive a model which would indicate how much an additional transfer or 5 minutes of waiting time translates into a representative value of travel time. Once these models are derived, the true measure of the optimal path for the user can be obtained.

A final area of research could be to include paratransit into the TATIS prototype. The current prototype allows for fixed route transit systems, such as bus, rail, subway, and ferry. Paratransit or "dial-a-ride" services do not operate over the same paths each day or each run but vary according to location and demand. Inclusion of paratransit into the prototype would necessitate an interface with a route-planning ATIS using the Arc/Info PATH command.

APPENDIX A
SAMPLE PATH FILE

Appendix A: Sample path file (z10100111.aml)

This macro contains address locations along Route 10, variation 1, Inbound, weekdays. The locations are generally corners where the transit vehicle turns. Additional locations insure a greater degree of accuracy in the determination of the route.

```
mapextent master
arcs master
netcover master bus
addresscover master
coordinate keyboard address
path * 10100111
Rock Ave / Front St
Somerset St / Front St
Front St / Terrill Rd
Terrill Rd / Midway Ave
Morse Ave / North Ave
North Ave / Maple St
North Ave / John St
Chestnut St / Westfield Ave
Morris Ave / Westfield Ave
Broad St / E Grand St
Grand St / Oak St
Spring St / North Ave
[unquote ' ']
linecolor 2
routes master bus
&pause
linecolor 1
coordinate mouse
```

APPENDIX B

SAMPLE STOP FILE

Appendix B: Sample stop file (z10100111.txt)

This file contains address locations of transit stops along Route 10, variation 1, Inbound, weekdays. The locations are intersections where the transit vehicle stops for passengers. The file is converted into an representative Info database file which is then used to create a point coverage for the route using address-matching commands.

Fields: Location, Stop-ID, Route-ID

"Rock Ave / Front St",10100111001,10100111
"Clinton Ave / Front St",10100111002,10100111
"West End Ave / Front St",10100111003,10100111
"Elmwood Pl / Front St",10100111004,10100111
"Somerset St / Front St",10100111005,10100111
"Richmond St / Front St",10100111006,10100111
"Farragut Rd / Front St",10100111007,10100111
"Raymond Ave / Front St",10100111008,10100111
"Terrill Rd / Front St",10100111009,10100111
"Terrill Rd / Midway Ave",10100111010,10100111
"Russell Rd / Midway Ave",10100111011,10100111
"Tillotson Rd / Midway Ave",10100111012,10100111
"Morse Ave / North Ave",10100111013,10100111
"Whittier Ave / North Ave",10100111014,10100111
"Tuttle Pkwy / North Ave",10100111015,10100111
"Central Ave / North Ave",10100111016,10100111
"Chestnut St / North Ave",10100111017,10100111
"Winslow Pl / North Ave",10100111018,10100111
"N Union Ave / North Ave",10100111019,10100111
"Elizabeth Ave / North Ave",10100111020,10100111
"Gordon St / Westfield Ave",10100111021,10100111
"Chestnut St / Westfield Ave",10100111022,10100111
"Pershing Ave / Westfield Ave",10100111023,10100111
"Palisade Rd / Westfield Ave",10100111024,10100111
"Elmora Ave / Westfield Ave",10100111025,10100111
"Chilton St / Westfield Ave",10100111026,10100111
"Morris Ave / Westfield Ave",10100111027,10100111
"Broad St / E Grand St",10100111028,10100111
"Grand St / Oak St",10100111029,10100111
"Spring St / North Ave",10100111030,10100111

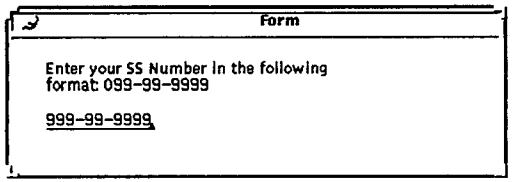
APPENDIX C
SAMPLE FOUND PATHS FILE

Appendix C: Sample found paths file (u111-11-1111.p1)

This file is generated from the CREATE_PATHS.AML and is used by the TRAVEL_ARR.AML and TRAVEL_DEP.AML macros.

1	Number of paths
z10250111	Identifier of first route (origin route)
z10350121	Identifier of second route (intermediate route)
z10200121	Identifier of third route (destination route)
4	Node number at the end of origin route
7	Node number at the beginning of intermediate route
12	Node number at the end of intermediate route
1	Node number at the beginning of destination route
0.2133	Transfer distance between origin and intermediate routes
0	Transfer distance between intermediate and destination routes
Southwest	Direction from the end of the origin route to the the intermediate route
beginning of	
0	Direction from the end of the intermediate route to the beginning of the destination route
2	Node number of the beginning of the origin route
0.0628	Distance from origin point to the beginning of the origin route
Northeast	Direction from the origin point to the beginning of the origin
5	Node number of the end of the destination route
0.1563	Distance from destination point to the end of the destination route
Northwest	Direction from the end of the destination route to the destination point
END	End of path description

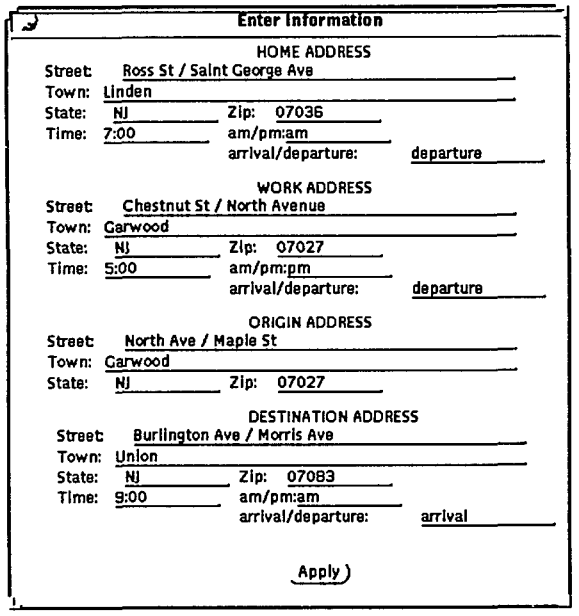
APPENDIX D
SAMPLE ANALYSIS




Console


nk

Figure 6.3 User-ID Menu



Console

Figure 6.4 Origin/Destination/Time Menu

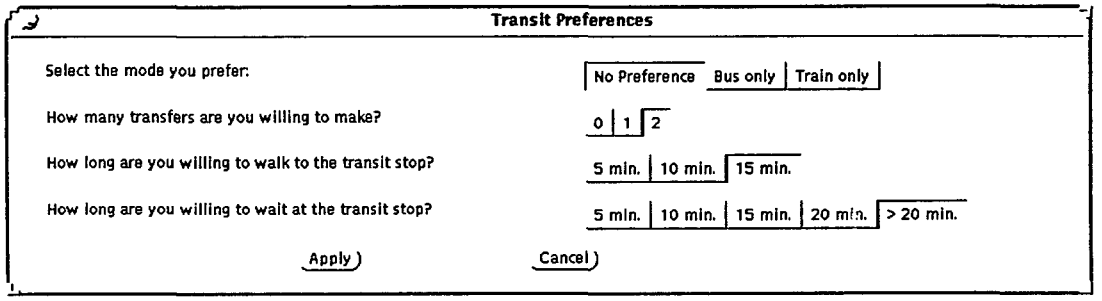


Figure 6.5 Preference Menu

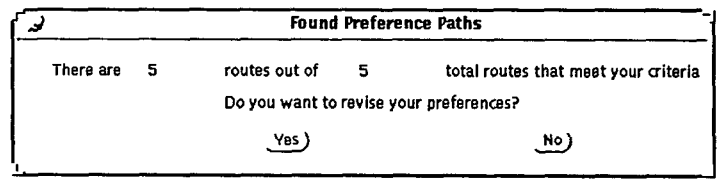


Figure 6.6 Results of Preference Search

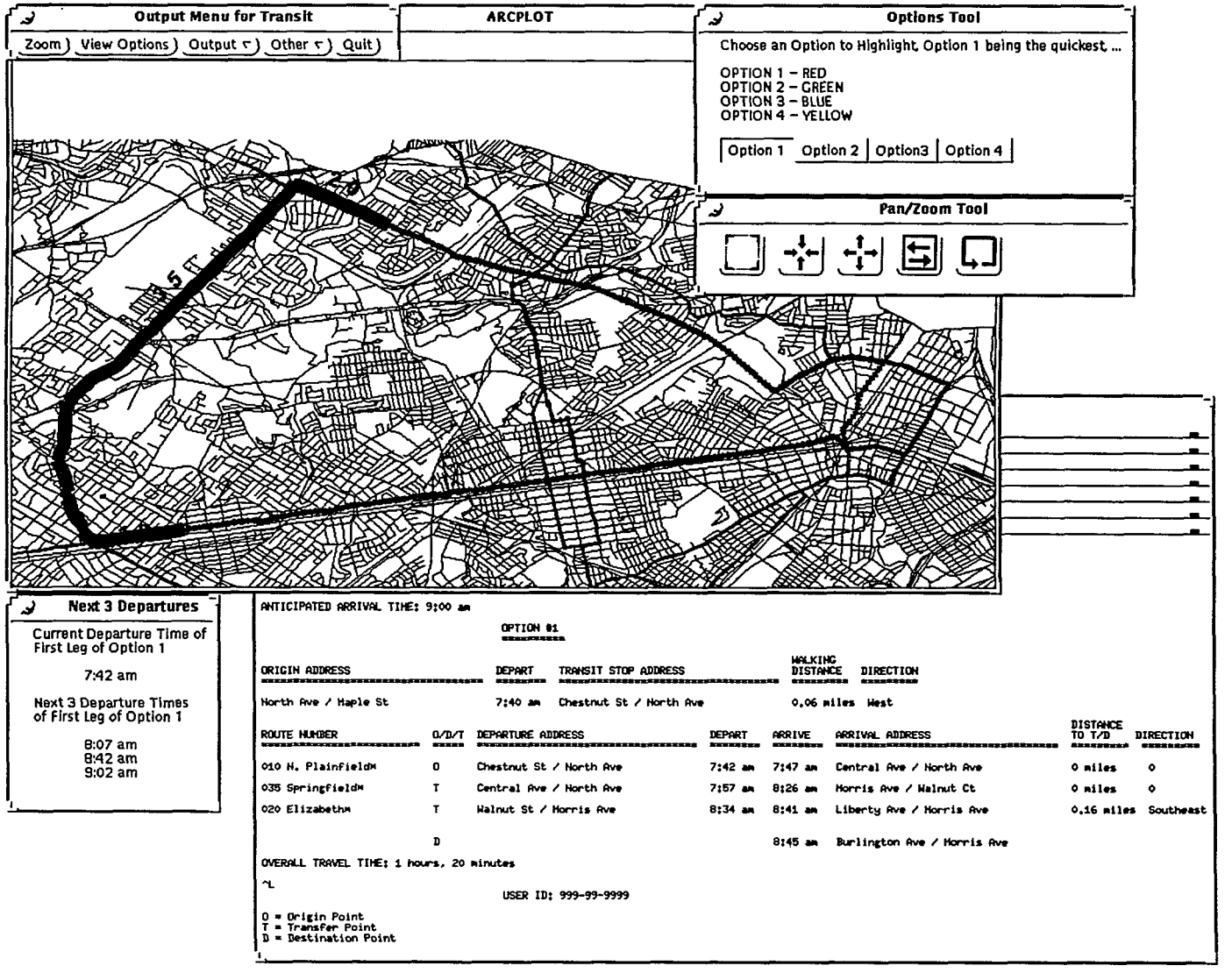


Figure 6.7 Output for Option 1

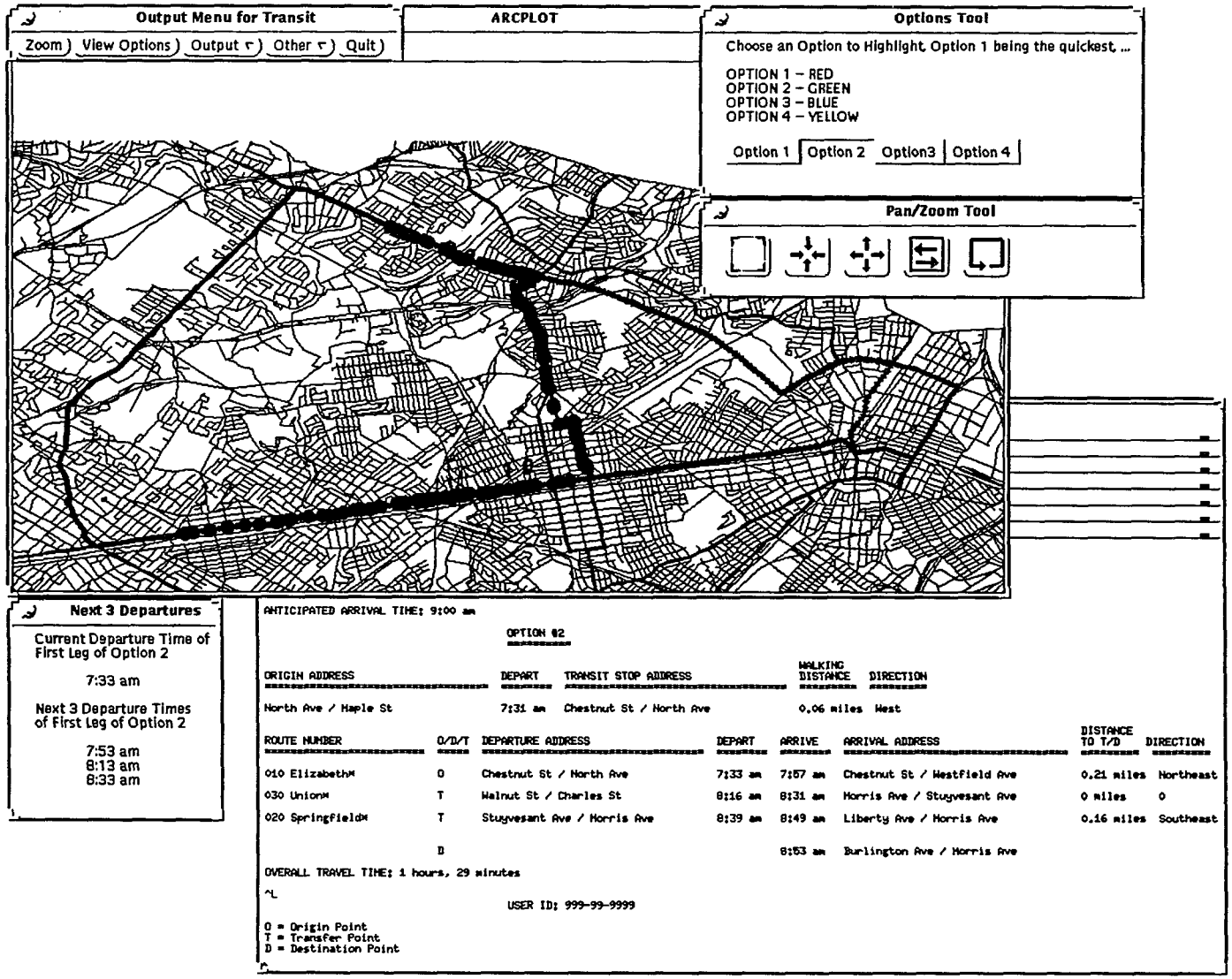


Figure 6.8 Output for Option 2

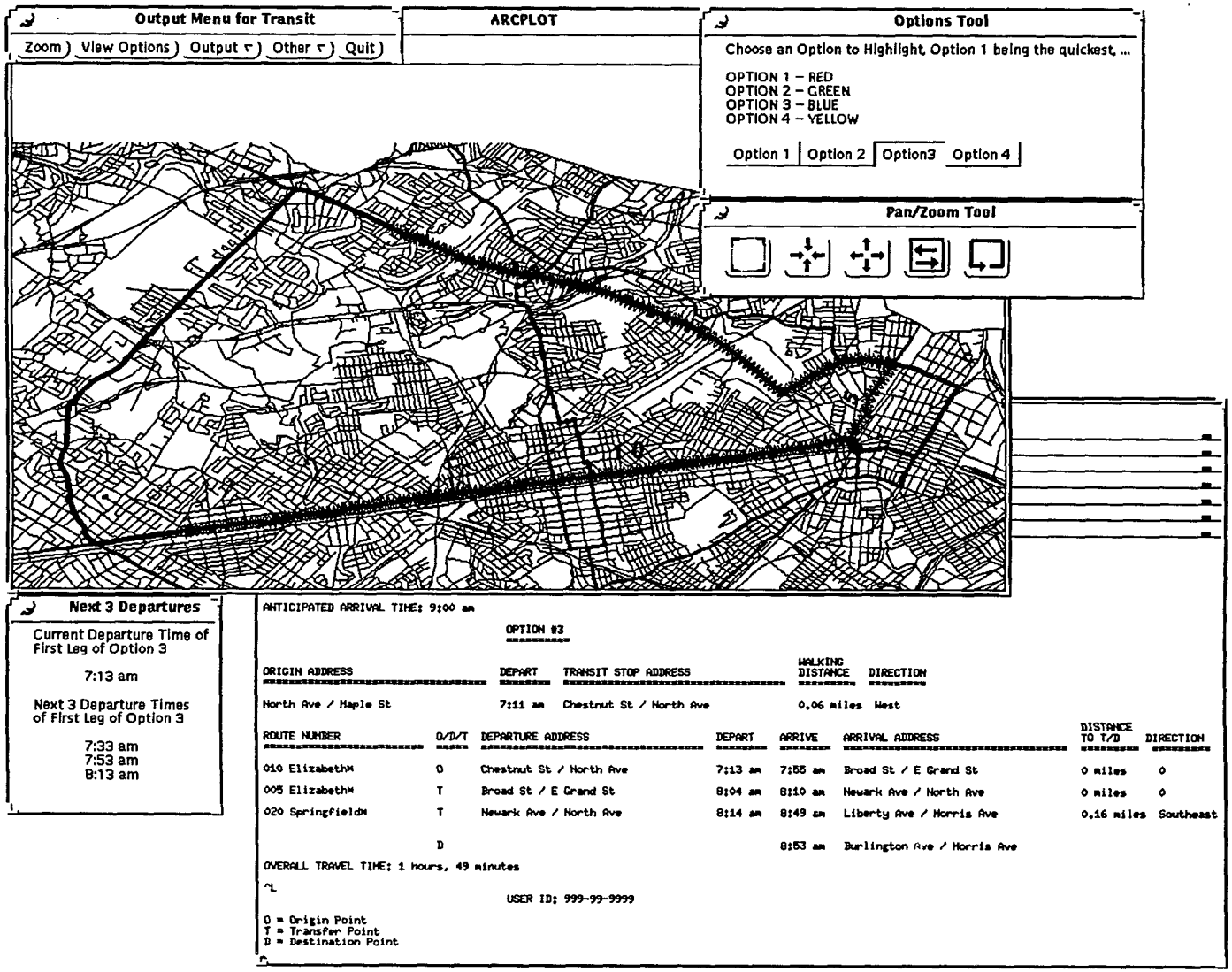


Figure 6.9 Output for Option 3

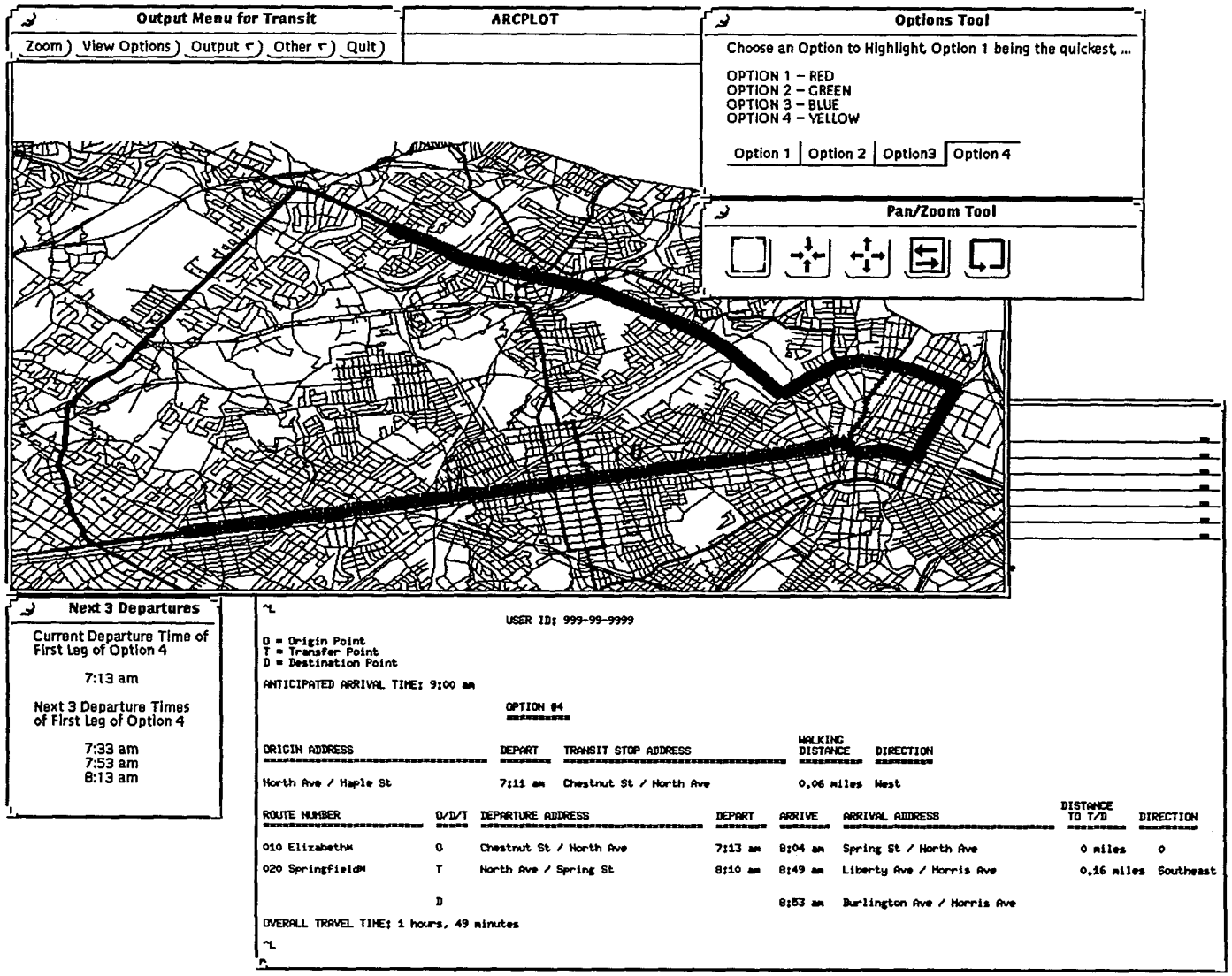
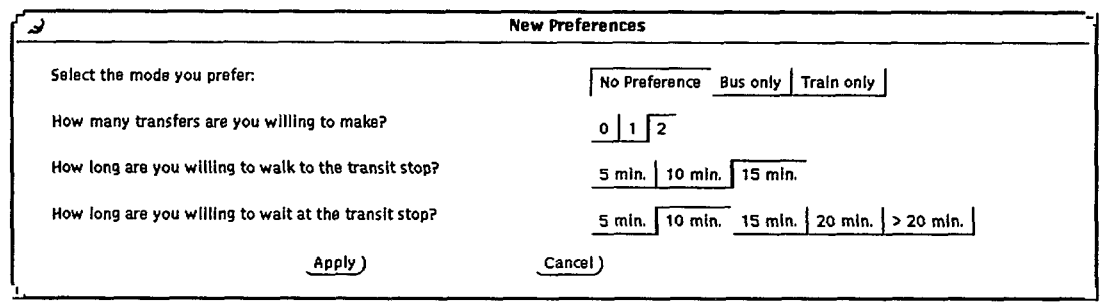


Figure 6.10 Output for Option 4



Console

Figure 6.11 Revised Preferences Menu

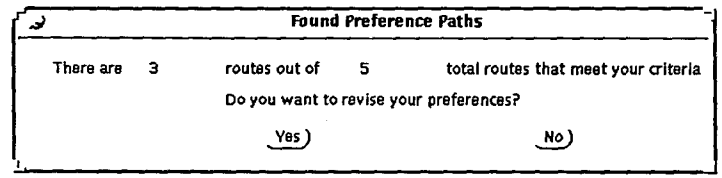


Figure 6.12 Found Preferences Menu

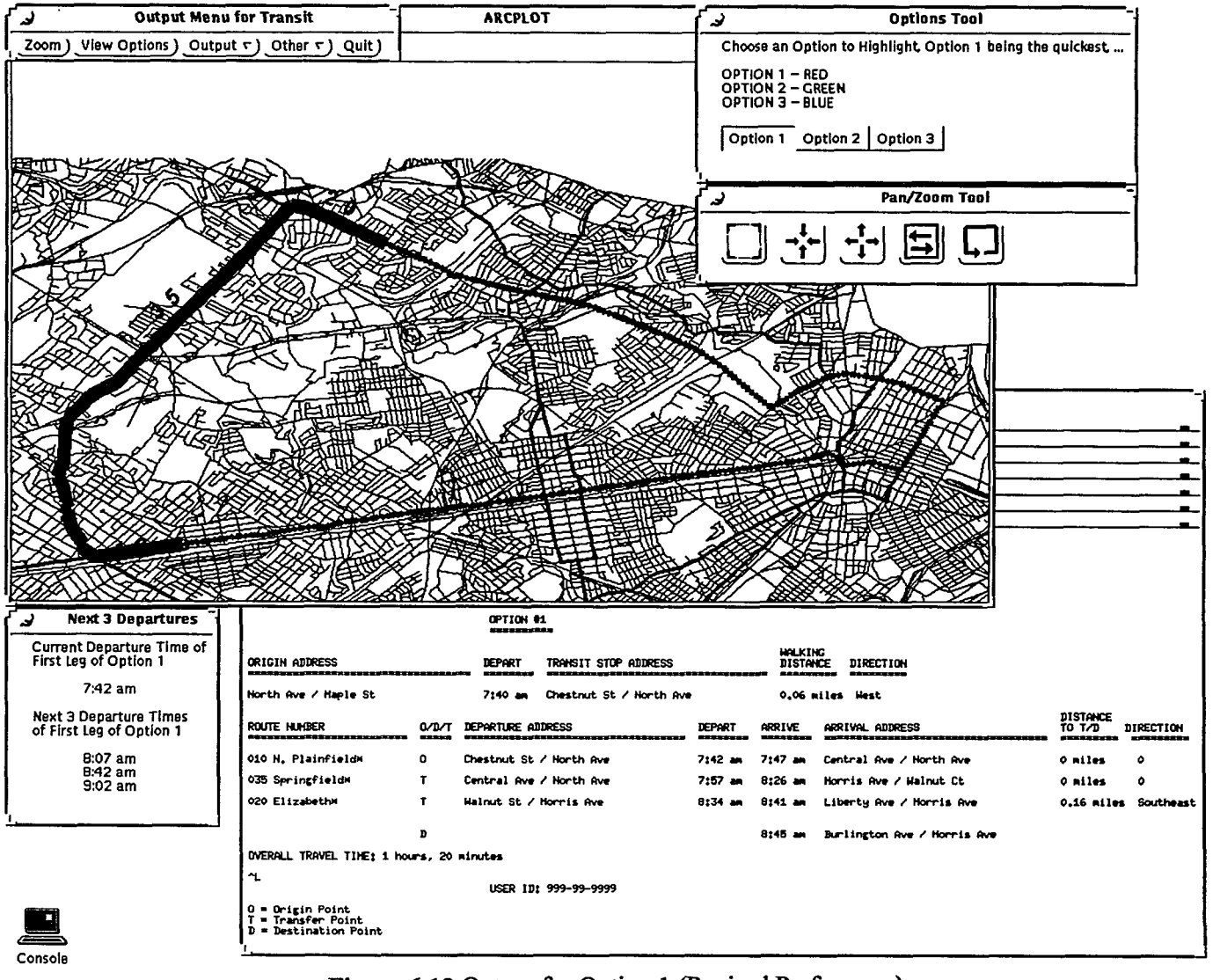


Figure 6.13 Output for Option 1 (Revised Preferences)

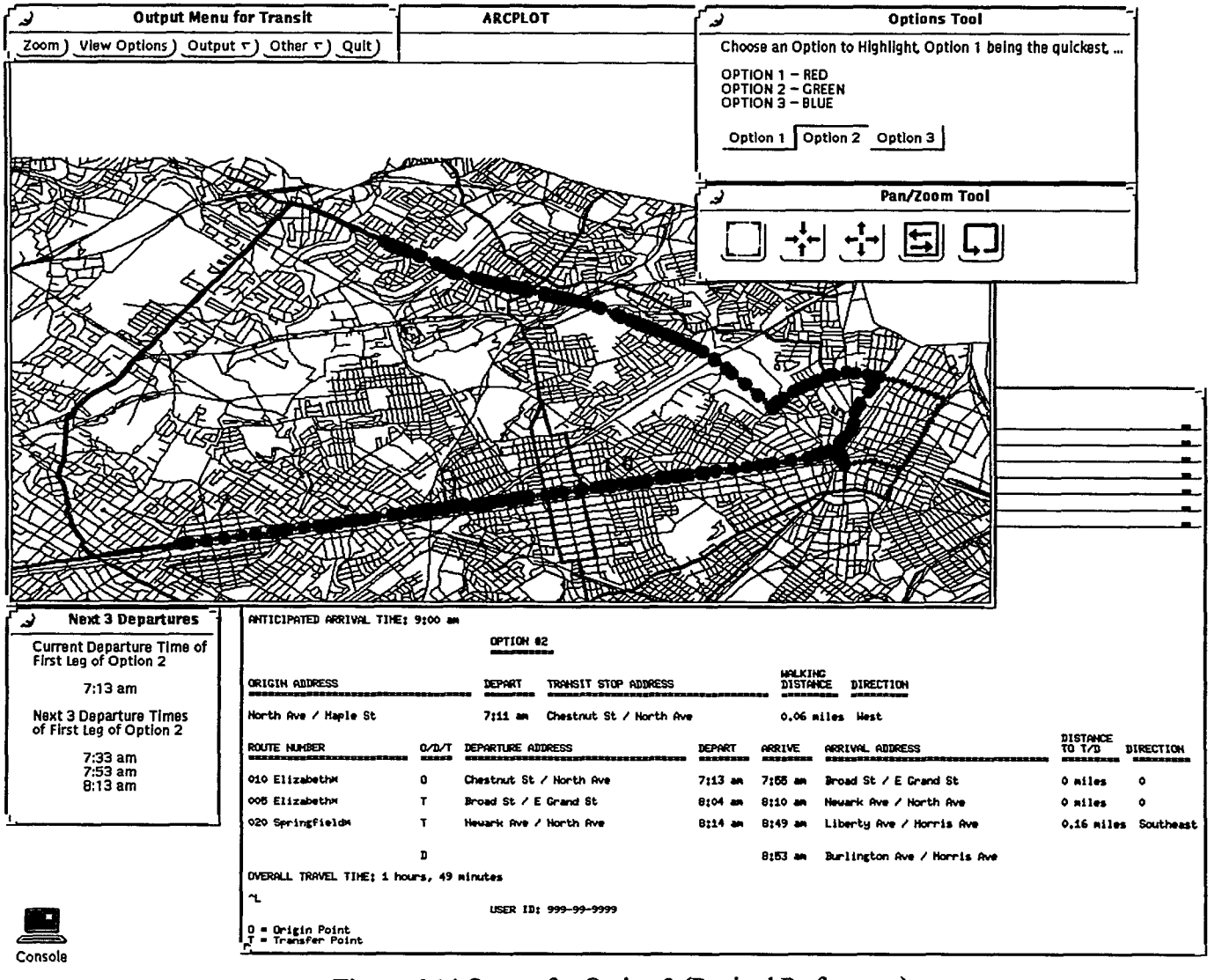


Figure 6.14 Output for Option 2 (Revised Preferences)

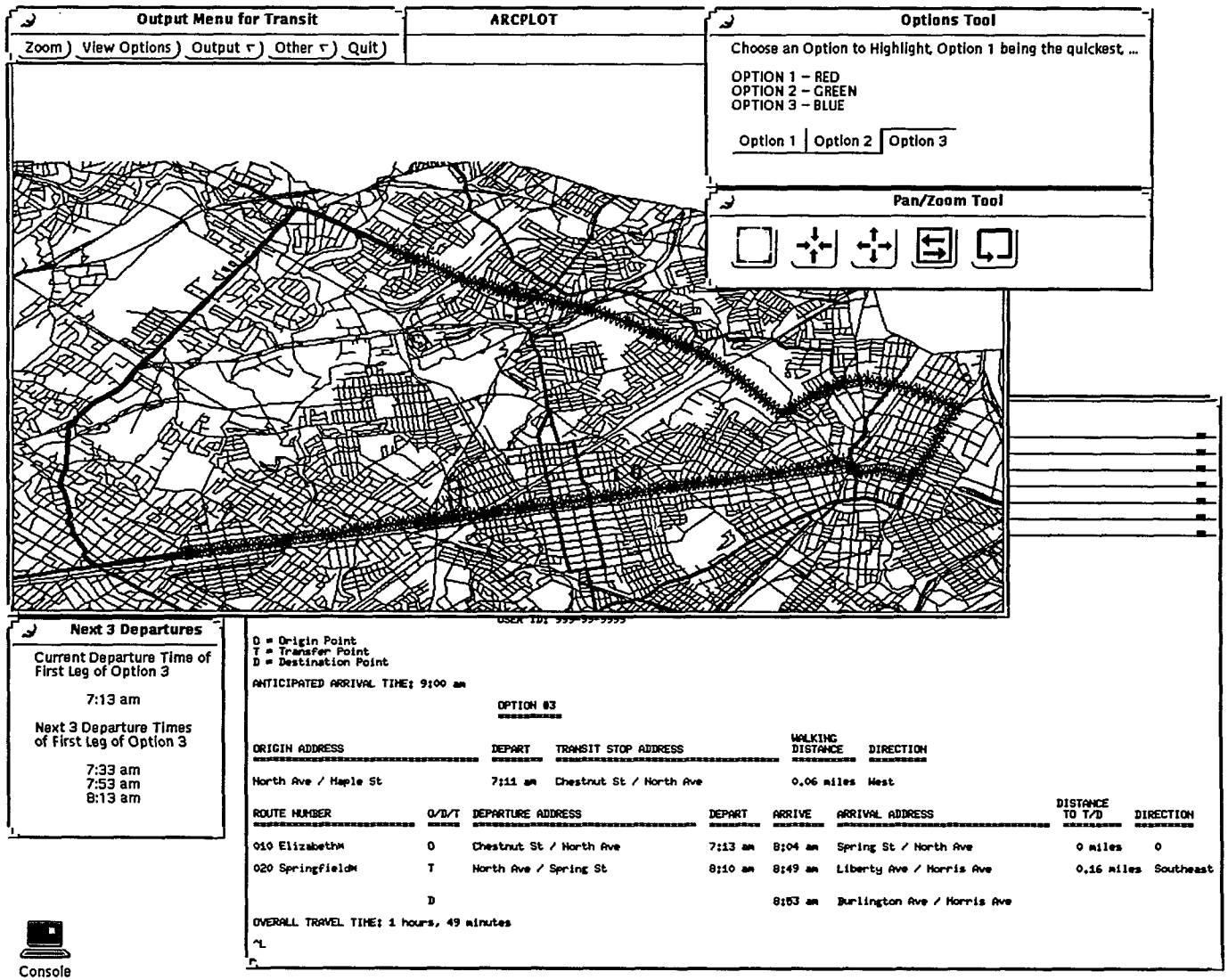


Figure 6.15 Output for Option 3 (Revised Preferences)

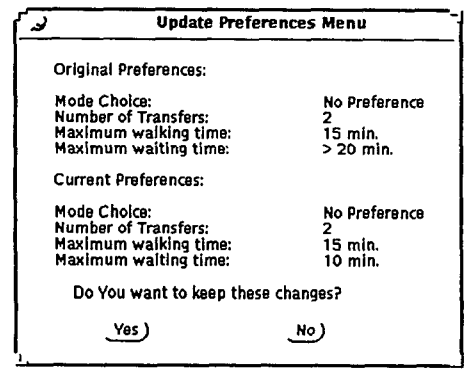


Figure 6.16 Final Update Menu

APPENDIX E
SAMPLE OUTPUT FILE

USER ID: 999-99-9999

O = Origin Point
T = Transfer Point
D = Destination Point

ANTICIPATED ARRIVAL TIME: 9:00 am

OPTION #1

ORIGIN ADDRESS	DEPART	TRANSIT STOP ADDRESS	WALKING DISTANCE	DIRECTION
North Ave / Maple St	7:40 am	Chestnut St / North Ave	0.06 miles	West

ROUTE NUMBER	O/D/T	DEPARTURE ADDRESS	DEPART	ARRIVE	ARRIVAL ADDRESS	DISTANCE TO T/D	DIRECTION
010 N. Plainfield*	O	Chestnut St / North Ave	7:42 am	7:47 am	Central Ave / North Ave	0 miles	0
035 Springfield*	T	Central Ave / North Ave	7:57 am	8:26 am	Morris Ave / Walnut Ct	0 miles	0
020 Elizabeth*	T	Walnut St / Morris Ave	8:34 am	8:41 am	Liberty Ave / Morris Ave	0.16 miles	Southeast
	D			8:45 am	Burlington Ave / Morris Ave		

OVERALL TRAVEL TIME: 1 hours, 20 minutes

USER ID: 999-99-9999

O = Origin Point
T = Transfer Point
D = Destination Point

ANTICIPATED ARRIVAL TIME: 9:00 am

OPTION #2

ORIGIN ADDRESS	DEPART	TRANSIT STOP ADDRESS	WALKING DISTANCE	DIRECTION
North Ave / Maple St	7:31 am	Chestnut St / North Ave	0.06 miles	West

ROUTE NUMBER	O/D/T	DEPARTURE ADDRESS	DEPART	ARRIVE	ARRIVAL ADDRESS	DISTANCE TO T/D	DIRECTION
010 Elizabeth*	O	Chestnut St / North Ave	7:33 am	7:57 am	Chestnut St / Westfield Ave	0.21 miles	Northeast
030 Union*	T	Walnut St / Charles St	8:16 am	8:31 am	Morris Ave / Stuyvesant Ave	0 miles	0
020 Springfield*	T	Stuyvesant Ave / Morris Ave	8:39 am	8:49 am	Liberty Ave / Morris Ave	0.16 miles	Southeast
	D			8:53 am	Burlington Ave / Morris Ave		

OVERALL TRAVEL TIME: 1 hours, 29 minutes

USER ID: 999-99-9999

O = Origin Point
T = Transfer Point
D = Destination Point

ANTICIPATED ARRIVAL TIME: 9:00 am

OPTION #3

ORIGIN ADDRESS	DEPART	TRANSIT STOP ADDRESS	WALKING DISTANCE	DIRECTION
North Ave / Maple St	7:11 am	Chestnut St / North Ave	0.06 miles	West

ROUTE NUMBER	O/D/T	DEPARTURE ADDRESS	DEPART	ARRIVE	ARRIVAL ADDRESS	DISTANCE TO T/D	DIRECTION
010 Elizabeth*	O	Chestnut St / North Ave	7:13 am	7:55 am	Broad St / E Grand St	0 miles	0
005 Elizabeth*	T	Broad St / E Grand St	8:04 am	8:10 am	Newark Ave / North Ave	0 miles	0
020 Springfield*	T	Newark Ave / North Ave	8:14 am	8:49 am	Liberty Ave / Morris Ave	0.16 miles	Southeast
	D			8:53 am	Burlington Ave / Morris Ave		

OVERALL TRAVEL TIME: 1 hours, 49 minutes

USER ID: 999-99-9999

O = Origin Point
T = Transfer Point
D = Destination Point

ANTICIPATED ARRIVAL TIME: 9:00 am

OPTION #4

ORIGIN ADDRESS	DEPART	TRANSIT STOP ADDRESS	WALKING DISTANCE	DIRECTION
North Ave / Maple St	7:11 am	Chestnut St / North Ave	0.06 miles	West

ROUTE NUMBER	O/D/T	DEPARTURE ADDRESS	DEPART	ARRIVE	ARRIVAL ADDRESS	DISTANCE TO T/D	DIRECTION
010 Elizabeth*	O	Chestnut St / North Ave	7:13 am	8:04 am	Spring St / North Ave	0 miles	0
020 Springfield*	T	North Ave / Spring St	8:10 am	8:49 am	Liberty Ave / Morris Ave	0.16 miles	Southeast
	D			8:53 am	Burlington Ave / Morris Ave		

OVERALL TRAVEL TIME: 1 hours, 49 minutes

APPENDIX F
OUTPUT OF PERFORMANCE RUNS

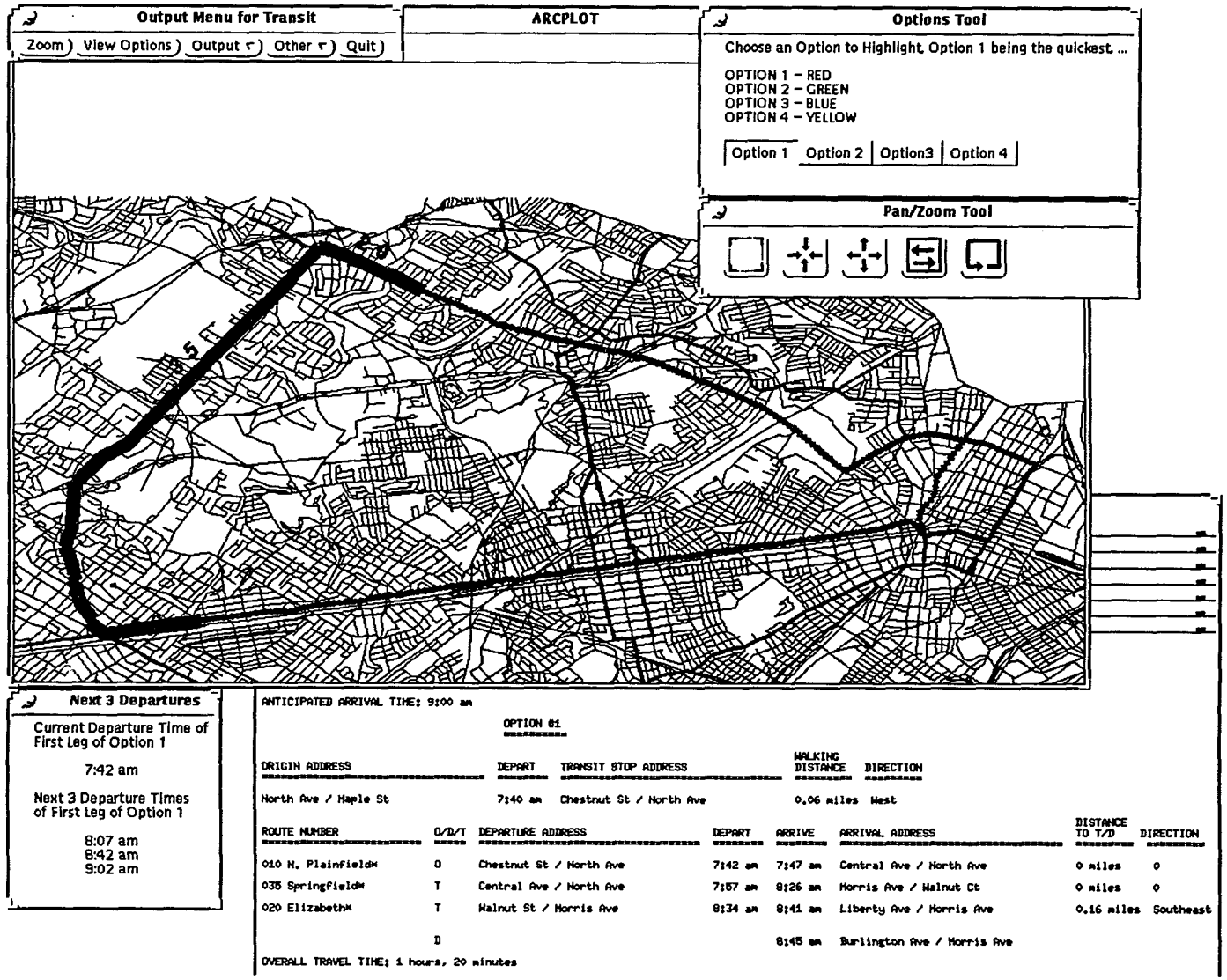


Figure 6.17 Output of Run 1-A

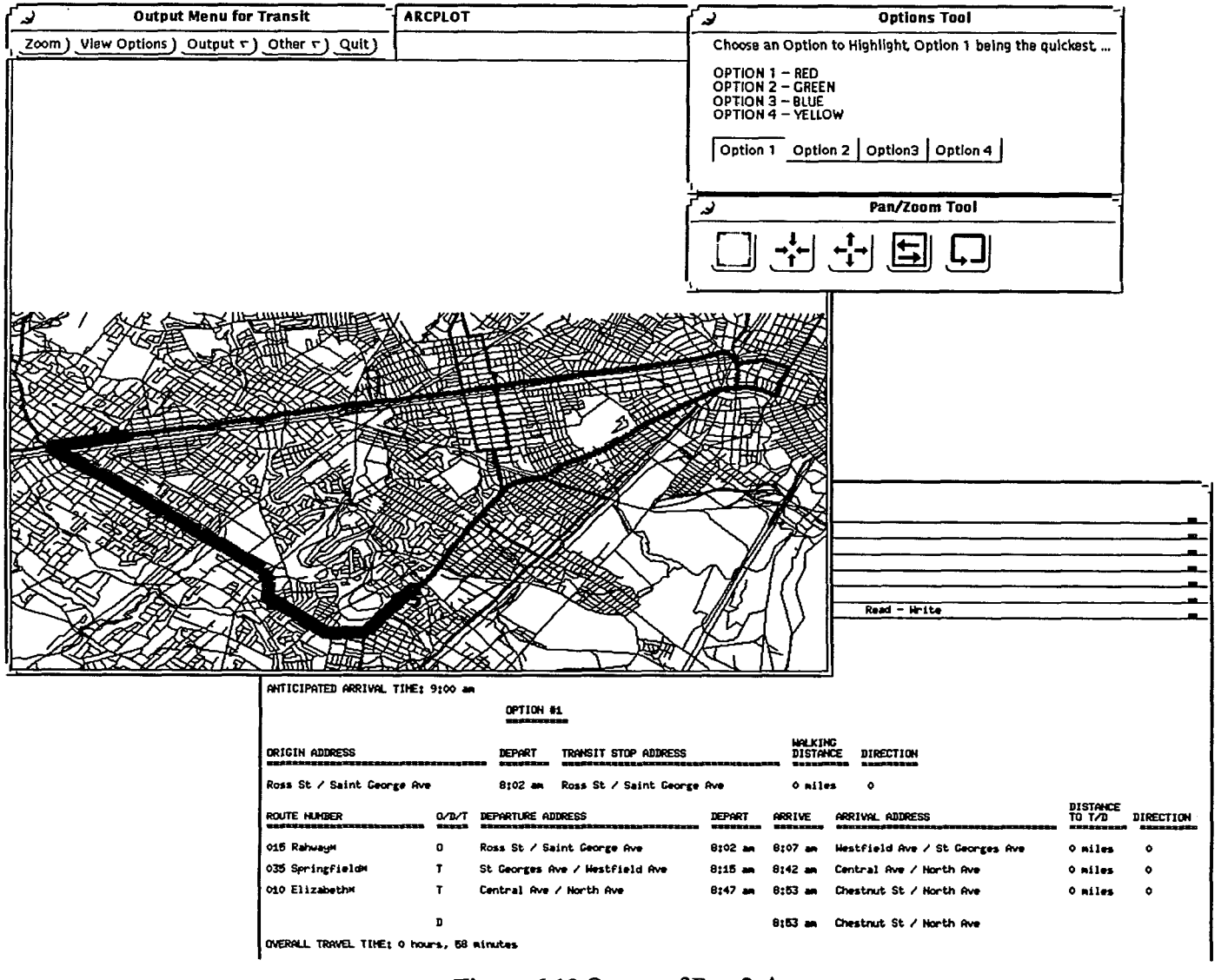


Figure 6.18 Output of Run 2-A

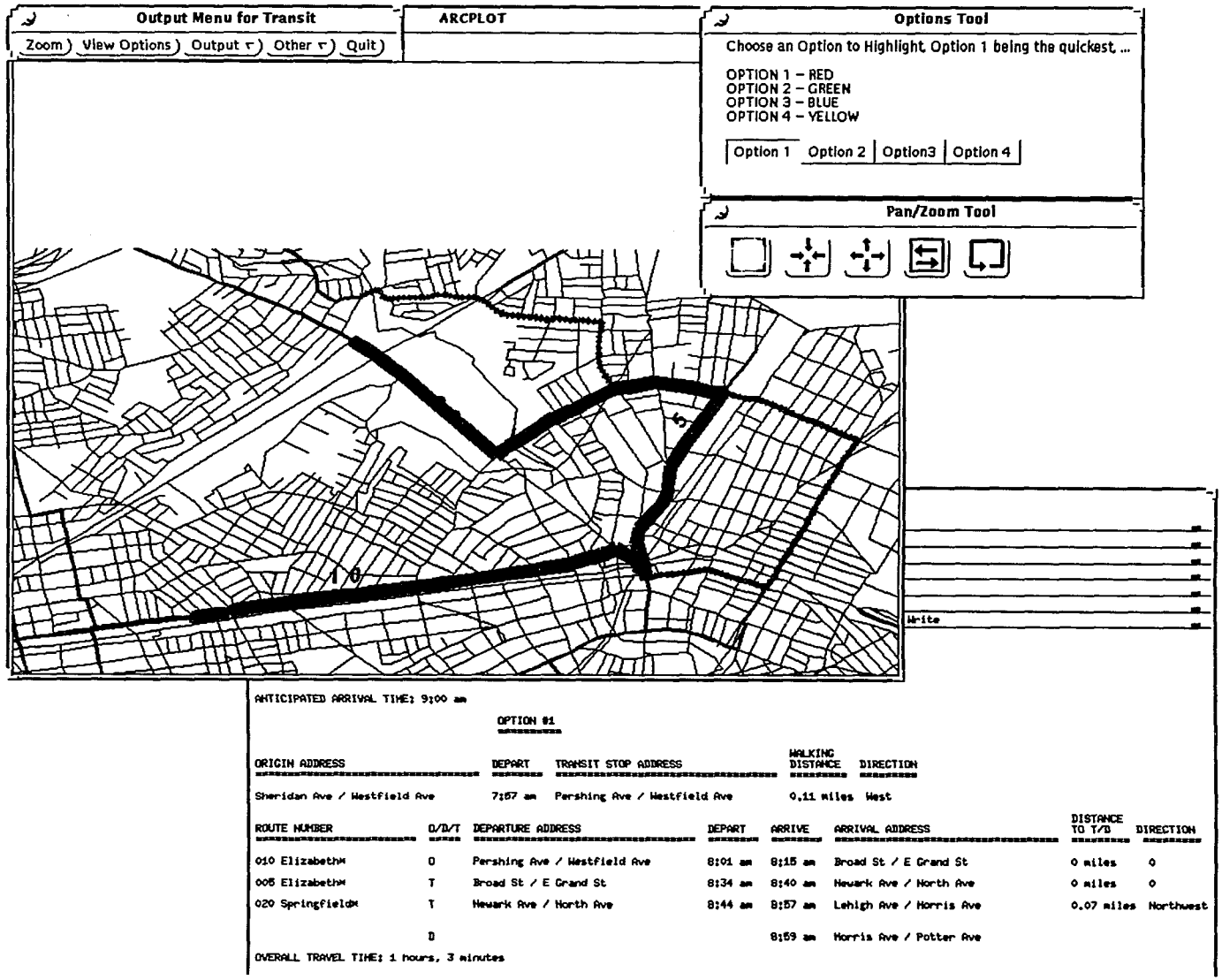


Figure 6.19 Output of Run 3-A

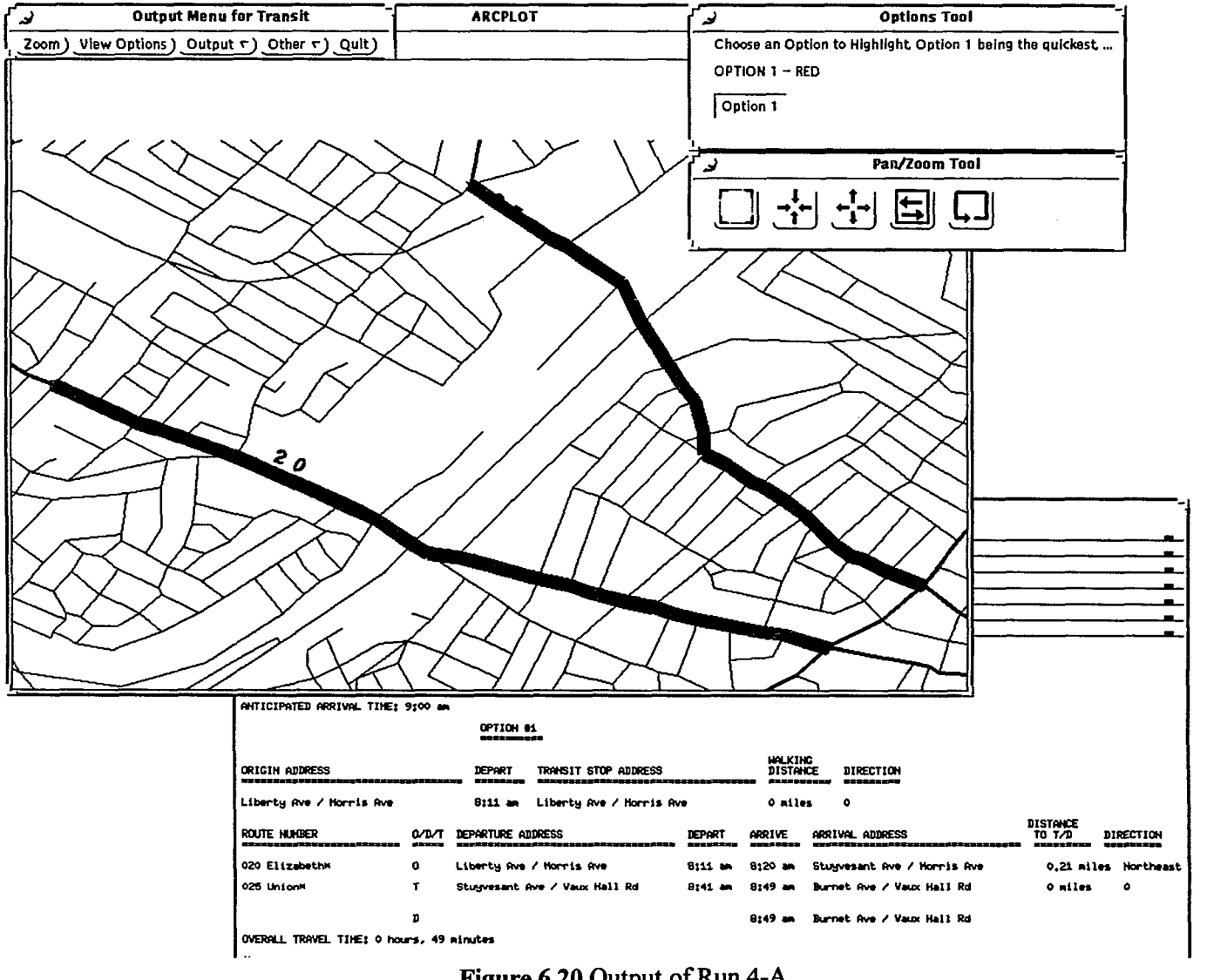


Figure 6.20 Output of Run 4-A

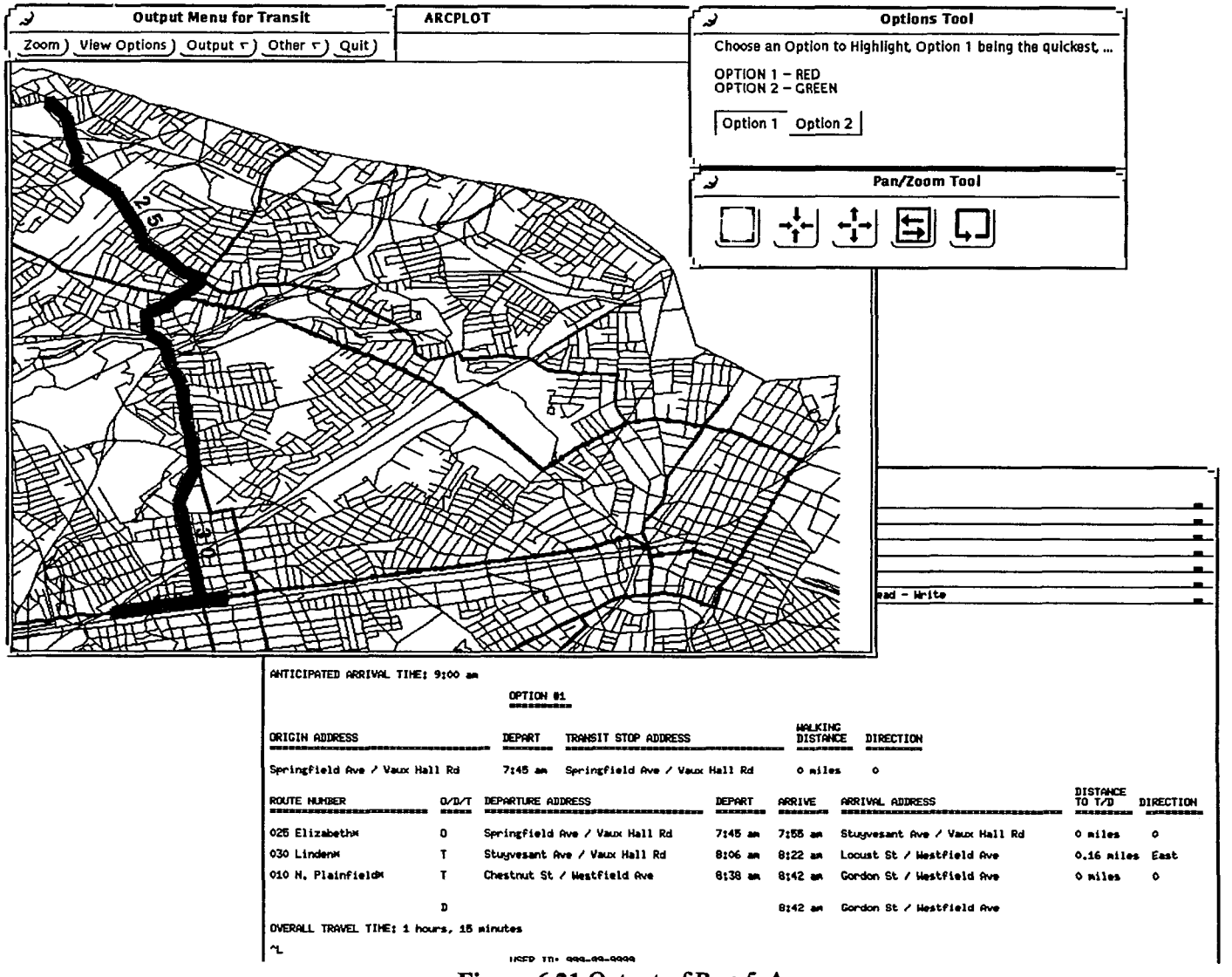


Figure 6.21 Output of Run 5-A

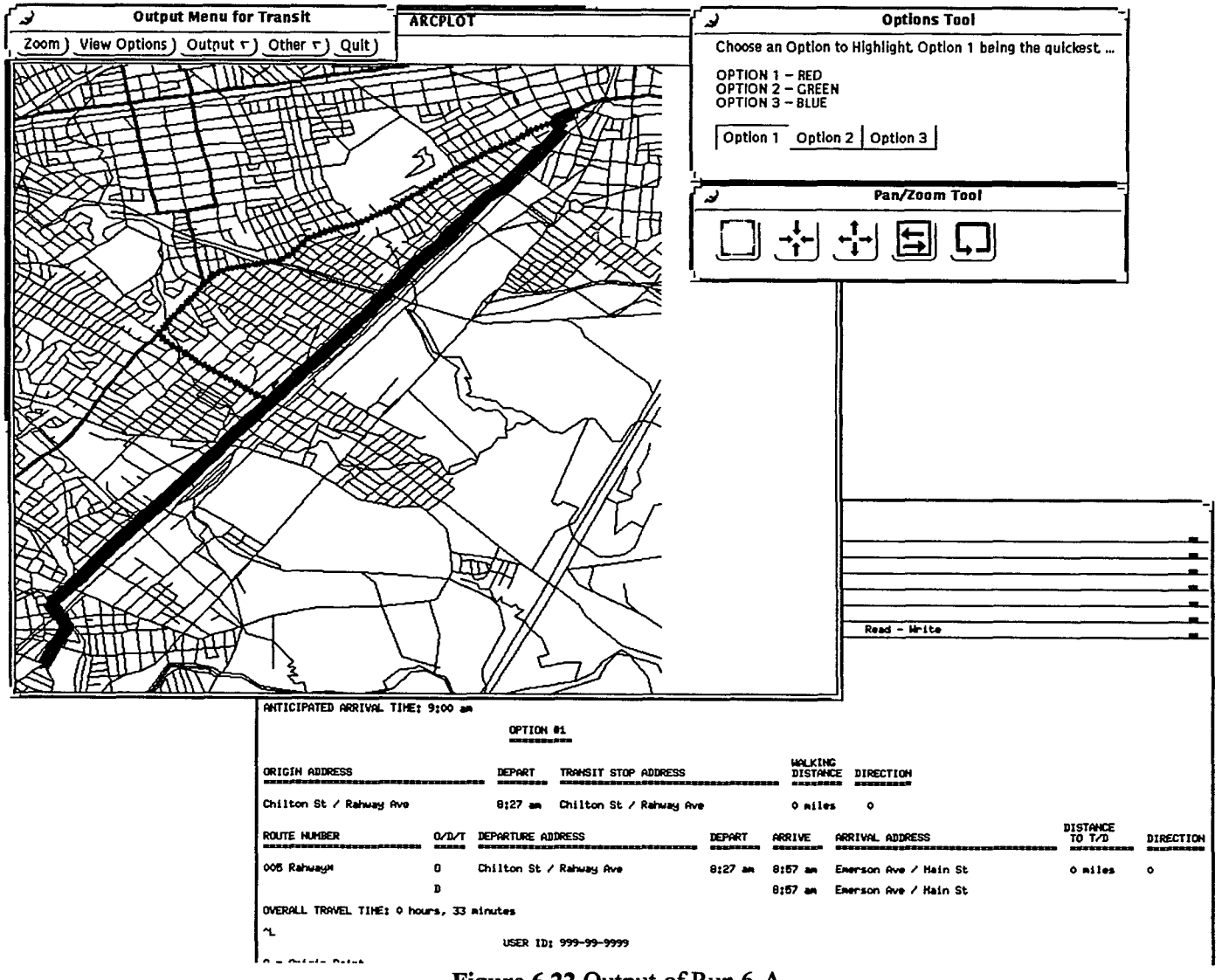


Figure 6.22 Output of Run 6-A

APPENDIX G
DESCRIPTION OF ARC/INFO TABLES

Appendix G: Description of Arc/Info Tables

Table G-1: Structure of route attribute table for coverage MASTER, route system BUS (MASTER.RATBUS)

```
Arc: items master.ratbus
```

COLUMN	ITEM NAME	WIDTH	OUTPUT	TYPE	N.DEC	ALTERNATE NAME	INDEXED?
1	BUS#	4	5	B	-		-
5	BUS-ID	4	12	B	0		-
9	DESTINATION	20	20	C	-		-
29	VARIATION	30	30	C	-		-
59	HANDICAP	2	2	C	-		-
61	ROUTE	10	10	I	-		-

Table G-2: Structure of section table for coverage MASTER, route system BUS (MASTER.SECBUS)

```
Arc: items master.secbus
```

COLUMN	ITEM NAME	WIDTH	OUTPUT	TYPE	N.DEC	ALTERNATE NAME	INDEXED?
1	ROUTELINK#	4	5	B	-		-
5	ARCLINK#	4	5	B	-		-
9	F-MEAS	4	12	F	3		-
13	T-MEAS	4	12	F	3		-
17	F-POS	4	12	F	3		-
21	T-POS	4	12	F	3		-
25	BUS#	4	5	B	-		-
29	BUS-ID	4	5	B	-		-

Table G-3: Structure of route point attribute table (z10050111.pat)

```
Arc: items z10050111.pat
```

COLUMN	ITEM NAME	WIDTH	OUTPUT	TYPE	N.DEC	ALTERNATE NAME	INDEXED?
1	AREA	8	18	F	5		-
9	PERIMETER	8	18	F	5		-
17	Z10050111#	4	5	B	-		-
21	Z10050111-ID	4	5	B	-		-
25	LOCATION	35	35	C	-		-
60	STOP-ID	16	16	N	0		-
76	ROUTE-ID	10	10	I	-		-
86	MASTER#	4	5	B	-		-
90	MASTER-ID	4	5	B	-		-
94	MASTER-SIDE	1	1	C	-		-
95	SCORE	4	5	B	-		-
99	BUS#	4	5	B	-		-
103	MEASURE	4	12	F	3		-
107	X-COORD	8	18	F	5		-
115	Y-COORD	8	18	F	5		-
123	LAT	8	18	F	5		-
131	LONG	8	18	F	5		-

Table G-4: Structure of connectivity matrix table (CONNECTING.DAT)

```
Arc: items connecting.dat
```

COLUMN	ITEM NAME	WIDTH	OUTPUT	TYPE	N.DEC	ALTERNATE NAME	INDEXED?
1	BUSTRAIN-ID	15	15	C	-		-
16	Z10050111	2	2	B	-		-
18	Z10050121	2	2	B	-		-
20	Z10100111	2	2	B	-		-
22	Z10100121	2	2	B	-		-
24	Z10100211	2	2	B	-		-
26	Z10100221	2	2	B	-		-
28	Z10150111	2	2	B	-		-
30	Z10150121	2	2	B	-		-
32	Z10200111	2	2	B	-		-
34	Z10200121	2	2	B	-		-
36	Z10250111	2	2	B	-		-
38	Z10250121	2	2	B	-		-
40	Z10300111	2	2	B	-		-
42	Z10300121	2	2	B	-		-
44	Z10350111	2	2	B	-		-
46	Z10350121	2	2	B	-		-

Table G-5: Structure of typical schedule table (z10050111.sc1)

```
Arc: items z10050111.sc1
```

COLUMN	ITEM NAME	WIDTH	OUTPUT	TYPE	N.DEC	ALTERNATE NAME	INDEXED?
1	S10050111001	10	10	N	7		-
11	S10050111002	10	10	N	7		-
21	S10050111003	10	10	N	7		-
31	S10050111004	10	10	N	7		-
41	S10050111005	10	10	N	7		-
51	S10050111006	10	10	N	7		-
61	S10050111007	10	10	N	7		-
71	S10050111008	10	10	N	7		-
81	S10050111009	10	10	N	7		-
91	S10050111010	10	10	N	7		-
101	S10050111011	10	10	N	7		-
111	S10050111012	10	10	N	7		-
121	S10050111013	10	10	N	7		-
131	S10050111014	10	10	N	7		-
141	S10050111015	10	10	N	7		-
151	S10050111016	10	10	N	7		-
161	S10050111017	10	10	N	7		-
171	S10050111018	10	10	N	7		-

Table G-6: Structure of found paths table (u111-11-1111.pth)

```
Arc: items u111-11-1111.pth
COLUMN   ITEM NAME          WIDTH OUTPUT  TYPE N.DEC  ALTERNATE NAME     INDEXED?
   1  WT-TIME                 10     10     N     8                        -
  11  ORIGIN-RT               10     10     C     -                        -
  21  DESTINATION-RT          10     10     C     -                        -
  31  INTERMEDIATE-RT         10     10     C     -                        -
  41  TRANSFER-1PT             4      4     I     -                        -
  45  TRANSFER-2PT             4      4     I     -                        -
  49  TRANSFER-3PT             4      4     I     -                        -
  53  TRANSFER-4PT             4      4     I     -                        -
  57  WALKDIST1               10     10     N     8                        -
  67  WALKDIST2               10     10     N     8                        -
  77  DIRECTION1              10     10     C     -                        -
  87  DIRECTION2              10     10     C     -                        -
  97  ORIGIN-PT                4      4     I     -                        -
 101  ORIGIN-DIST             10     10     N     8                        -
 111  ORIGIN-BEAR             10     10     C     -                        -
 121  DESTINAT-PT              4      4     I     -                        -
 125  DESTINAT-DIST           10     10     N     8                        -
 135  DESTINAT-BEAR           10     10     C     -                        -
 145  T-TIME                  10     10     N     8                        -
 155  DEPARTURE_TIME          10     10     N     8                        -
 165  TIME-A-ORIGIN           10     10     N     8                        -
 175  RUN-ORIGIN               4      4     I     -                        -
 179  TIME-D-ORIGIN           10     10     N     8                        -
 189  TIME-A-INTERMEDI        10     10     N     8                        -
 199  TIME-D-INTERMEDI        10     10     N     8                        -
 209  TIME-A-DESTINATI        10     10     N     8                        -
 219  TIME-D-DESTINATI        10     10     N     8                        -
 229  ARRIVAL_TIME            10     10     N     8                        -
 239  CHANGE-TIME             10     10     N     8                        -
```

Table G-7: Structure of user prefered paths table (u111-11-1111.pre)

```
Arc: items u111-11-1111.pre
COLUMN   ITEM NAME          WIDTH OUTPUT  TYPE N.DEC  ALTERNATE NAME     INDEXED?
   1  WT-TIME                 10     10     N     8                        -
  11  ORIGIN-RT               10     10     C     -                        -
  21  DESTINATION-RT          10     10     C     -                        -
  31  INTERMEDIATE-RT         10     10     C     -                        -
  41  TRANSFER-1PT             4      4     I     -                        -
  45  TRANSFER-2PT             4      4     I     -                        -
  49  TRANSFER-3PT             4      4     I     -                        -
  53  TRANSFER-4PT             4      4     I     -                        -
  57  WALKDIST1               10     10     N     8                        -
  67  WALKDIST2               10     10     N     8                        -
  77  DIRECTION1              10     10     C     -                        -
  87  DIRECTION2              10     10     C     -                        -
  97  ORIGIN-PT                4      4     I     -                        -
 101  ORIGIN-DIST             10     10     N     8                        -
 111  ORIGIN-BEAR             10     10     C     -                        -
 121  DESTINAT-PT              4      4     I     -                        -
 125  DESTINAT-DIST           10     10     N     8                        -
 135  DESTINAT-BEAR           10     10     C     -                        -
 145  T-TIME                  10     10     N     8                        -
 155  DEPARTURE_TIME          10     10     N     8                        -
 165  TIME-A-ORIGIN           10     10     N     8                        -
 175  RUN-ORIGIN               4      4     I     -                        -
 179  TIME-D-ORIGIN           10     10     N     8                        -
 189  TIME-A-INTERMEDI        10     10     N     8                        -
 199  TIME-D-INTERMEDI        10     10     N     8                        -
 209  TIME-A-DESTINATI        10     10     N     8                        -
 219  TIME-D-DESTINATI        10     10     N     8                        -
 229  ARRIVAL_TIME            10     10     N     8                        -
 239  CHANGE-TIME             10     10     N     8                        -
```

APPENDIX H
LIST OF IMPLEMENTED AML MACROS

Appendix H: List of implemented AML macros

Name	Function
adi_zoom.aml	Performs the desired zooming functions on the display
bearing.aml	Determines the directional bearing given two points
connect.aml	Generates the connectivity matrix
create_measure.aml	Finds the measures of each transit point coverage
create_paths_r.aml	Finds all possible paths from an origin to a destination
create_stops.aml	Converts stop ASCII file to Info file and performs addressmatching
dd_ct.aml	Converts change in decimal days to number of hours and minutes
dd_hm.aml	Converts decimal days to hours:minutes time of day
hm_dd.aml	Converts hours:minutes time of day to decimal days
msinform.aml	Generates dynamic menu indicating error conditions
msworking.aml	Generates dynamic menu indicating status of programs
opt1-r.aml	Displays all found paths, transit network, roadway network, and highlights and labels the first path
opt2-r.aml	Displays all found paths, transit network, roadway network, and highlights and labels the second path
opt3-r.aml	Displays all found paths, transit network, roadway network, and highlights and labels the third path
opt4-r.aml	Displays all found paths, transit network, roadway network, and highlights and labels the fourth path
output_r.aml	Generates all concrete forms of output
output_2r.aml	Displays all forms of output

Name	Function
pref.aml	Selects all paths that meet user's preferences
quit.aml	Runs update_pref.aml
r_address.aml	Allows for revision of addresses and runs the prototype over again
r_pref.aml	Allows for revision of user preferences and runs the output algorithms over again
r_travel_time.aml	Allows for revision of travel time and runs the prototype over again
redraw.aml	Refreshs the graphical image of the paths found
route-1.aml	Typical AML that converts an ASCII schedule file into an Info database file
ssn-check.aml	Checks the user-id for validity
text-r.aml	Generates xedit window for schedule itinerary
three_o_depart_r.aml	Finds the next three departure times of the first route of the selected path and creates menu displaying the information
transit-a.aml	Controls the flow of the programs that compose the prototype
travel_arr.aml	Determines the travel times of each path based on a time of arrival
travel_dep.aml	Determines the travel times of each path based on a time of departure
update_pref.aml	Asks the user whether the current preferences should be saved and then exits the prototype
z10050111.aml	Typical AML that converts a series of locations into a route as part of a route system using dynamic segmentation

APPENDIX I

LIST OF IMPLEMENTED ARC/INFO MENUS

Appendix I: List of implemented Arc/Info menus

Name	Function
draw.menu	Displays the "View Options" menu
msinform.menu	Informs user of an error condition
msworking.menu	Displays informational status of progress of program
output-r.menu	Displays the "Output Menu for Transit" menu
pref2.menu	Displays the number of paths that meet user's preferences
preference.menu	User preference menu
temp_address.menu	Temporary data entry form containing addresses and time of travel
u-id.menu	User-ID menu
update_pref.menu	Indicates prior and current preferences and whether to update preferences
zoom.menu	Displays "Pan/Zoom Tool" menu

BIBLIOGRAPHY

Ahuja, R. et al., "Faster Algorithms for the Shortest Path Problem," *Journal of the Association for Computing Machinery,* vol. 37, no. 2, pp. 213-223, 1990.

Anderson, L., "Applying Geographic Information Systems to Transportation Planning," *Transportation Research Record,* no. 1305, pp. 113-116, 1992.

Antennucci, J. et al., *Geographic Information Systems: A Guide to the Technology.* New York: Van Nostrand Reinhold, 1991.

Antonisse, R., "GIS-T Applications in Transit: Recent Experience in Seattle and Boston," *Proceedings of 1991 Geographical Information Systems (GIS) for Transportation Symposium,* 1991.

Arlook, J. and Jones, R., "Tracking IVHS," *Geo Info Systems,* vol. 6, pp. 39-43, 1993.

Azar, K., "Integrating Geographic Information Systems Into Transit Passenger Information Systems," Master's Thesis, Massachusetts Institute of Technology, 1991.

Bacon, M. and Moyer, D., "Proceedings of 1989 Geographical Information Systems (GIS) for Transportation Symposium." American Association of State Highway and Transportation Officials, 1989.

Bacon, M. and Moyer, D., "Proceedings of 1990 Geographical Information Systems (GIS) for Transportation Symposium." American Association of State Highway and Transportation Officials, 1990.

Bellman, R., "On a Routing Problem," *Quarterly Journal on Applied Mathematics,* vol. 16, no. 1, pp. 87-90, 1958.

Chriqui, C. and Robillard, P., "Common Bus Lines," *Transportation Science,* vol. 9, no. 1, pp. 115-121, 1974.

Cooke, K. and Halsey, E., "The Shortest Route Through a Network with Time-Dependent Internodal Transit Times," *Journal on Mathematical Analysis and Applications,* vol. 14, pp. 493-398, 1966.

Dantzig, G., "Discrete-Variable Extremum Problems," *Operations Research,* vol. 5, pp. 266-277, 1957.

Dantzig, G., "On The Shortest Route Through a Network," *Management Science,* vol. 6, pp. 187-190, 1960.

Dantzig, G., "All Shortest Routes in a Graph," *Theory of Graphs.* New York: Gordon and Breach, pp. 91-93, 1967.

Dial, R, et al., "A Computational Analysis of Alternative Algorithms and Labeling Techniques for Finding Shortest Path Trees," *Networks,* vol. 9, pp. 215-248, 1979.

Dijkstra, E., "A Note on Two Problems in Connexion with Graphs," *Numerische Mathemalik,* vol. 1, pp. 269-271, 1959.

Dreyfus, S., "An Appraisal of Some Shortest-Path Algorithms," *Operations Research,* vol. 17, no. 3, pp. 395-412, 1969.

Dueker, K., Vrana, R., and Bishop, G., "GIS Applications in Urban Public Transportation: Pilot Projects and Implementation Strategies for Tri-Met, Portland, Oregon." Center for Urban Studies, Portland State University, 1991.

Dueker, K., and Vrana, R. 1993. "Dynamic Segmentation Revisited: A Milepoint Linear Data Model," *URISA Journal,* vol. 5, no. 1, pp. 94-105, 1993

Eiger, A., Mirchandani, P. and Soroush, H., "Path Preferences and Optimal Paths in Probabilistic Networks," *Transportation Science,* vol. 19, no. 1, pp. 75-84, 1985.

Environmental Systems Research Institute, Inc. "Arc/Info Geographic Information System, Version 6.1" (computer program), Sun Sparcstation 10 Solaris 3.2, Environmental Systems Research Institute, Inc. Redlands, CA., 1992.

Environmental Systems Research Institute, Inc., *Dynamic Segmentation Arc/Info User's Guide, Version 6.0.*. Redlands, CA: Environmental Systems Research Institute, Inc., 1991.

Environmental Systems Research Institute, Inc., *Network Analysis; Modeling Network Systems, Arc/Info User's Guide, Version 6.1.*, Redlands, CA: Environmental Systems Research Institute, Inc.,1992.

Environmental Systems Research Institute, Inc. *Understanding GIS: The Arc/Info Method.* Redlands, CA: Environmental Systems Research Institute, Inc., 1992.

Fletcher, D., "Pavement Management Decision Support Using a Geographic Information System." Wisconsin DOT., 1987.

Fletcher, D., "Integrating Photolog Data Into a Geographic Information System," *URISA Journal,* vol. 2, no. 1, 1990.

Fletcher, D. and S. Lewis., *GIS and Transportation: The Workbook.* Boston: Urban and Regional Information Systems Association, 1989.

____, "Introductory GIS-T Workshop" Geographic Information Systems for Transportation Symposium, Urban and Regional Information Systems Association, San Antonio, 1990.

____, *GIS and Transportation: The Workbook.* Edmonton, Canada: Urban and Regional Information Systems Association, 1990.

Floyd, R., "Algorithm 97: Shortest Path," *Comm. ACM,* vol. 5, pp. 345,1962.

Frank, H., "Shortest Paths in Probabilistic Graphs," *Operations Research,* vol. 17: pp. 583-599, 1969.

Gilsinn, J. and Witzgall, C.,"A Performance Comparison of Labeling Algorithms for Calculating Shortest Path Trees," NBS Technical Note 772, U.S. Department of Commerce, 1973.

Glover, F., Glover, R., and Klingman, D., "Computational Study of an Improved Shortest Path Algorithm," *Networks,* vol. 14, pp. 25-36, 1984.

Glover, F. et al., "New Polynomial Shortest Path Algorithms and Their Computational Attributes," *Management Science,* vol. 31, no. 9, pp. 1106-1127, 1985.

Grayson, T., "Modeling and Visualizing Transit Accessibility: An Approach Based on Geographic Information Systems and Relational Database Technology." Master's Thesis, Massachusetts Institute of Technology 1993.

Hall, R., "The Fastest Path Through a Network with Random Time-Dependent Travel Times," *Transportation Science,* vol. 20, no. 3, pp. 182-188, 1986.

Hancock, K, and Abkowitz, M., "The Use of Geographic Information Systems for Customer Service in Urban Public Transportation," Transportation Research Board Annual Meeting. Report no. 920896, pp. 1-35, 1992.

Hoffman, W. and Pavley, R., "A Method for the Solution of the n-th best path problem," *Journal of the Association for Computing Machinery,* vol. 6, pp. 506-514, 1959.

Huxhold, W., *An Introduction to Urban Geographic Information System*. New York: Oxford University Press, 1991.

Javid, M. et al., "Application of GIS in Planning Transit Services for People with Disabilities," Transportation Research Board Annual Meeting Report no. 940266, 1994.

Kandru, P., "Route Planning Using Geographic Information Systems." Miscellaneous Paper MP:93-004, New Jersey Institute of Technology, Newark, N.J., pp. 5-6, 1993.

Kaufman, D., and Smith, R., "Fastest Paths in Time-Dependent Networks For Intelligent Vehicle-Highway Systems Application," *IVHS Journal*, vol. 1, no. 1, pp. 1-11, 1993.

Kiel, D. E., "Integrating TIGER with GIS and Travel Demand Models," Presented at the Applied Geography Conference Annual Meeting. Binghamton, N.Y., 1989.

Laporte, G., Louveaux, F., and Mercure, H., "The Vehicle Routing Problem with Stochastic Travel Times," *Transportation Science*, vol. 26, no. 3,pp. 161-169, 1992.

Lawler, E., "A Procedure for Computing the K Best Solutions to Discrete Optimization Problems and its Application to the Shortest Path Problem," *Management Science*, vol. 18, pp. 401-405, 1972.

Lawler, E., *Combinatorial Optimization: Networks and Matroids*. New York: Holt, Rinehart and Winston, 1976.

Lewis, S., "Use of Geographical Information Systems in Transportation Modeling," *ITE Journal*, 1990.

Marguier, P. and Ceder, A., "Passenger Waiting Strategies for Overlapping Bus Routes," *Transportation Science*, vol. 18, no. 3, pp. 207-226,1984.

Moore, E., "The Shortest Path Through a Maze," Proceedings of an International Symposium on the Theory of Switching, Part II, Cambridge, MA, 1957.

Moyer, D. and Larson., B., "Proceedings of 1991 Geographical Information Systems (GIS) for Transportation Symposium." American Association of State Highway and Transportation Officials, 1991.

Moyer, D., "Proceedings of 1992 Geographical Information Systems (GIS) for Transportation Symposium." American Association of State Highway and Transportation Officials, 1992.

Moyer, D. and Ries, T., "Proceedings of 1993 Geographical Information Systems (GIS) for Transportation Symposium." American Association of State Highway and Transportation Officials, 1993.

Neenan, B. and Huang, G., "Information Requirements For an Integrated Transit/Traffic Management and Traveler Information System," *IVHS Journal*, vol. 1, no. 2, pp. 167-180, 1993.

Nyerges, T, and Dueker, K.J., "Geographic Information Systems in Transportation," Regional Computer-Assisted Cartography Conference: Summary, Published by the Pennsylvania Dept. of Transportation, 1988.

O'Neill, W., Ramsey, R., and Chou, J., "Planning and Analysis of Transit Routes and Stops Using Geographic Information Systems," Transportation Research Board Annual Meeting Report no. 920453, 1992.

O'Neill, W., and Bennion, M., "Building Transportation Analysis Zones Using GIS," Transportation Research Board Annual Meeting Report no. 940476, 1994.

Pallotino, S., "Shortest Path Methods: Complexity, Interrelations and New Propositions," *Networks*, vol. 14, pp. 257-267, 1984.

Pape, U., "Algorithm 562: Shortest Path Lengths (H)," *ACM Transactions on Mathematical Software*, vol. 6, no. 3, pp. 450-455, 1980.

Perko, A., "Implementation of Algorithms for K Shortest Loopless Paths," *Networks*, vol. 16, pp. 149-160, 1986.

Pollack, M., "The kth Best Route Through a Network," *Operations Research*, vol. 9, no. 4, p. 578, 1961.

Prastacos, P., "Integrating GIS Technology in Urban Transportation Planning and Modeling," *Transportation Research Record*, no. 1305, pp. 123-130, 1992.

Ramlal, A., Personal Communication, 1994

Ross, D. and Soberman, R., "Evaluation of Public Transit Automated Telephone Information Systems," *Transport Canada* TP 8234 E, 1987.

Schweiger, C., "Current Use of Geographic Information Systems in Transit Planning," *Transportation Research Record,* no. 1349, pp. 93-106, 1992.

Sweeney, L., Sheldrick, M., Zavoli, W., and Buxton, J., "APTS Benefits of Geographic Databases," Proceedings of the IVHS America 1993 Annual Meeting, pp. 94-99, 1993.

Syslo, M., Narsingh, D., and Kowalik, J., *Discrete Optimization Algorithms with Pascal Programs.* Englewood Cliffs, N.J.: Prentice-Hall, Inc.,1983.

TIGER/Line Census Files. (TIGER 1991), Machine readable data files. Prepared by the U.S. Bureau of the Census. Washington D.C.: U.S. Bureau of the Census, producer and distributor.

Travis, L., A. Vonderohe, R. Smith, and W. Berg,. "Research on Adaption of GIS for Transportation: The Server-Net Model," Proceedings of 1990 Geographical Information Systems (GIS) for Transportation Symposium, 1990.

U.S. Department of Transportation, "What is APTS?," Technical Assistance Brief 1, pp. 1-2, 1993.

Yen, J., "Finding the K Shortest Loopless Paths in a Network," *Management Science,* vol. 17, no. 11, pp. 712-716, 1971.

Zilaskopoulos, A., "Design and Implementation of Some K Shortest Path Algorithms With Application to Intelligent Vehicle Highway Systems," Master's Thesis, University of Texas at Austin, 1992.