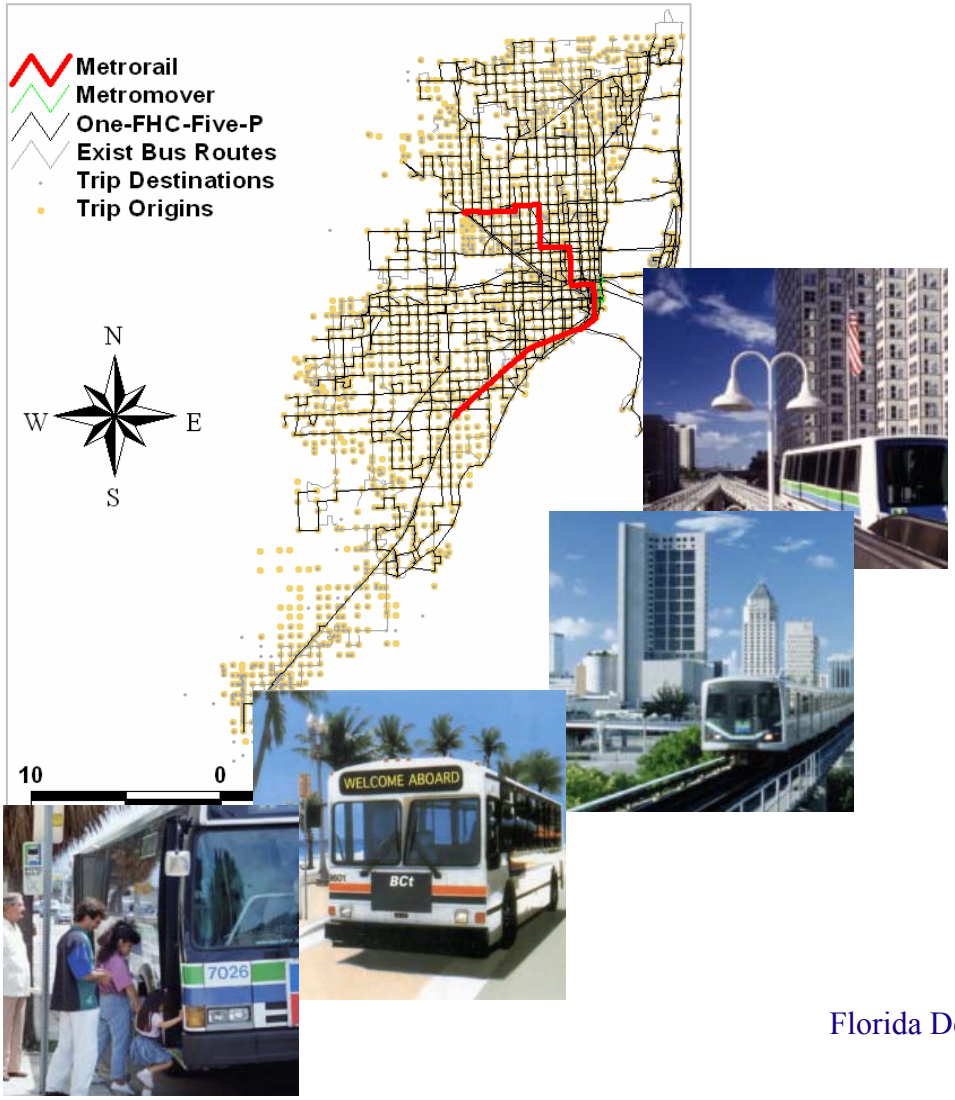
OPTIMIZATION OF TRANSIT NETWORK TO MINIMIZE TRANSFERS

Final Report

Contract No. BD-015-02

December 2003



Prepared for
Research Center
Florida Department of Transportation

Prepared by
Lehman Center for Transportation Research
Florida International University

OPTIMIZATION OF TRANSIT NETWORK TO MINIMIZE TRANSFERS

Final Report

Contract No. BD015-02

Prepared for

Research Office
Florida Department of Transportation
605 Suwannee Street, MS 30
Tallahassee FL 32399-0450
Tel: (850) 414-4615
Fax: (850) 414-4696

Prepared by

Fang Zhao, Ph.D., P.E.
Lehman Center for Transportation Research
Department of Civil & Environmental Engineering
Florida International University
Miami, FL 33199
Phone: (305) 348-3821
Fax: (305) 348-2802
Email: zhaof@fiu.edu

and

Albert Gan, Ph.D.
Assistant Professor
Lehman Center for Transportation Research
Florida International University
10555 W Flagler Street
Miami, Florida 33174
Phone: (305) 348-3116
Fax: (305) 348-2802
E-mail: gan@eng.fiu.edu

December 2003

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
Final Report for BD015-02		

4. Title and Subtitle	5. Report Date
OPTIMIZATION OF TRANSIT NETWORK TO MINIMIZE TRANSFERS	December 2003
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Fang Zhao and Albert Gan	

9. Performing Organization Name and Address	10. Work Unit No. (TRAIS)
Lehman Center for Transportation Research, Department of Civil and Environmental Engineering, Florida International University, Miami, Florida 33199	11. Contract or Grant No. BD015-02

12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
Research Office Florida Department of Transportation 650 Suwannee Street, MS 30 Tallahassee, Florida 32399-0450	Final Report December 2001 – December 2003
	14. Sponsoring Agency Code

15. Supplementary Notes
FDOT Project Manager
Ike Ubaka

16. Abstract
This report presents a mathematical methodology for transit route network optimization. The goal was to provide an effective computational tool for the optimization of a large-scale transit route network. The objectives were to minimize transfers and optimize route directness while maximizing service coverage. The formulation of the method consisted of three parts: representation of transit route network solution search spaces, representation of transit route and network constraints, and a search schemes capable of finding the expected global optimal result. The search methods included greedy search, fast hill climbing, and an integrated simulated annealing, tabu and greedy search algorithm. The methodology was implemented in a computer program, and was tested with several benchmark problems. It was also applied to a large-scale realistic network optimization problem in Miami-Dade County, Florida. The test results showed that the methodology developed in this study was capable of improving a given network solution in terms of average transfers and transit service coverage. A prototype application for transit network optimization was implemented with TransCAD as the user interface.

17. Key Word	18. Distribution Statement
Transit network, network optimization, search space, greedy search, fast hill climbing, simulated annealing, tabu list, TransCAD	

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

ACKNOWLEDGEMENTS

This study was sponsored by a research grant from the Research Office of the Florida Department of Transportation (FDOT). The author would like to thank the Project Manager, Mr. Ike Ubaka of the FDOT Public Transit Office, for his guidance and support throughout the research. The authors also thank Dr. Lee-Fang Chow for her assistance in preparing the transit network data.

EXECUTIVE SUMMARY

Introduction

One of the goals of the Florida Department of Transportation's 2020 Transit Strategic Plan is to help Florida Transit Systems improve their performance. Reducing the number of transfers involved in a transit trip is an important step towards that end. According to a transit onboard survey conducted as part of the recent South Florida Travel Characteristics Study (Carr Smith Corradino 2000), about one half of the transit trips involved at least one transfer. Approximately 15% involved two or more transfers. A survey (Stern 1996) of various transit agencies in U.S. indicated that about 58% of the respondents believed that transit riders were only willing to transfer once per trip. It is widely accepted in the U.S. transit industry that the willingness of riders to transfer has a significant impact on the total ridership of a transit system (Newman et al. 1983). This perception is based on empirical transit market studies or past experiences, and if it is valid, the ridership of a transit system may increase significantly by merely reducing transfers through the optimization of the existing transit route network configuration without expanding services or demanding more resources. In addition to ridership, transit route design or configuration may also have a significant impact on transit budgets and route coverage, and thus contributing to the transit system effectiveness and efficiency and the overall quality of life in the urban areas being served.

Factors that affect the passengers' willingness to transfer vary, and may include waiting time and walking distance required during transfers, reliability of bus schedules, bus headways, loading factor, and so on. One of the main concerns is the security of passengers and criminal activities that sometimes occur at transfer locations, especially in the evening or at night on low frequency bus routes, where missing one connecting bus may result in a long wait and cause safety concerns (Stern 1996). Transfers increase riders' travel difficulty, which, in turn, has a negative impact on the overall ridership of a transit system. Avoiding unnecessary transfers and/or reducing transfer difficulty have become an important transit design or service improvement issue. Various remedies or improvements have been suggested or implemented, including coordinating bus schedules, increasing bus frequencies, increasing the number of sheltered and well lit bus transfer stops, and reconfiguring existing transit network to reduce transfers. Since this study is concerned with issues related to transit route network (TRN) reconfiguration or optimization to minimize riders' transfers, the literature review and description presented in this report will mainly deal with issues on transit route network optimization and related topics.

Transfers are a result of a network configuration that is unable to provide direct services between all pairs of origins and destinations. Clearly, the main reason is the constraint on resources, which is faced by all transit properties. It is difficult to eliminate or noticeably reduce bus transfers without significantly increasing bus fleet size and total bus route miles, both of which will result in the increase of operating costs. Moreover, the "inertia" in transit services, i.e., the inability to immediately or fully respond to changes in population, housing, employment, and demographics (such instant response may be neither desirable nor practical), also contributes to transfers that may be potentially eliminated.

For transit systems with small bus route networks, a seasoned planner may obtain reasonable bus route network results by utilizing his or her knowledge and experience, carrying out simple ridership data collection and analysis, and following certain simple guidelines. For large urban areas where the number of bus routes may be close to or over a hundred, and the number of bus stops in thousands, intuition, past experiences, and simple guidelines may not be enough to produce even near-optimal transit route network configurations due to various complicated factors involved in transit route network design. Systematic methodologies are therefore needed to obtain better transit route network configurations.

The goal of this study was to develop a methodology and a software tool for optimizing bus transit services to reduce transfers. The research objectives included:

(1) Provide an understanding of the current state-of-the-art and the state-of-the-practice in transit network optimization.
(2) Develop a methodology for optimizing transit service configuration based on a synthesis/refinement of the state-of-the-art and the state-of-the-practice. The methodology would have the ability to deal with a larger transit network than had been reported in the literature. Based on the methodology a more robust network optimization tool would be produced, which could be used for practical planning purposes.
(3) Develop a user-friendly computer tool for transit agencies to optimize their bus services with optimal transfers.

Literature Review

A TRN optimization problem may be stated as the determination of a set of transit routes given a transit demand distribution in a transit service area, subject to a set of feasibility constraints, to achieve objective(s) that optimize the overall quality of a TRN. Mathematically, a TRN optimization problem may be formulated as a special form of integer optimization problems called a *combinatorial* optimization problem where the solution search space is a set of various subsets (or combinations) of an integer set. The search space of a combinatorial optimization problem may be extremely large even for a small integer set due to the combinatorial characteristics.

Solving a TRN optimization problem involves the search for an optimal set of feasible transit routes with unknown topology/geometry. It is difficult to solve problems with a large number of integer variables since the associated solution procedure involves discrete optimization, which usually requires the search for optimal solutions from an intractable search space (Garey and Johnson 1979).

A great deal of research has been conducted in the area of transit network optimization, including TRN optimization and TNS optimization, or a combination of the two. The methods in the literature may be roughly grouped into two categories: mathematical approaches, and heuristic approaches. However, there are no clear boundaries between these approaches. In this study, we consider an approach a mathematical approach if the problem is formulated as an optimization problem over a relatively complete solution search space, and generic solution search methods are then employed to obtain the solutions. Examples of such algorithms include various greedy

type algorithms, hill climbing algorithms, tabu search algorithms, and stochastic based search algorithms such as genetic algorithms (GAs) and simulated annealing search schemes. References and descriptions of various mathematical search algorithms may be found, for example, in (Aarts and Lenstra 1997, Bertsekas 1998). We consider an approach a heuristic approach if domain specific heuristics, guidelines, or criteria are first introduced to establish a solution strategy framework, and mathematical programming or any other search techniques are then employed to obtain the best results. The main difference between a mathematical approach and a heuristic approach lies in the fact that the mathematical approach formulates the problem on a solution space with certain completeness that, theoretically, should include global optimal solutions. In contrast, the heuristic approach formulates a problem directly on possible solution sub-spaces defined based on domain specified heuristic guidelines.

The advantage of heuristic approaches is that they are always able to provide feasible solutions to problems of any size. The main disadvantage of heuristic approaches is that their results are almost certainly not global or even local optimal solutions. This may be attributed to the fact that search schemes in heuristic approaches are usually ad hoc procedures based on computer simulations of human transit design processes guided by heuristic rules. The corresponding search spaces are usually not clearly defined and the search results are likely to be biased toward existing systems or any systems on which the set of design heuristics are based.

Compared with other methods in transit network design, mathematical optimization approaches usually have more rigorous problem statements and solid theoretical ground. Under certain conditions, such as the case of convex problem defined domain and objective functions, a global optimal solution can be guaranteed. Disadvantages of mathematical optimization approaches in TRN design are the following:

(1) In practice, the optimization problem derived from a realistic transit network system may be either non-convex or, in most cases, with unknown convexity, and in such cases, results produced from a mathematical optimization process may be at most a local optimal.

(2) The resultant mathematical optimization systems derived from realistic combinatorial TRN problems are usually at least NP-hard, which refers to problems for which the number of elementary numerical operations is not likely to be expressed or bounded by a function of polynomial form (Garey and Johnson 1979). The NP-hard intractability is due to the need to search for optimal solutions from a large search space made up by all possible solutions. For this reason, existing mathematical optimization solution approaches to the TRN problems are usually applied to relatively small and idealized networks for small urban areas or medium-sized urban areas with coarse networks. The route network structures may also be limited to certain particular configurations. As a cautionary note, the computational intractability that is used to describe a problem or an algorithm is only an asymptotic estimate to the solvability of a problem, and is usually based on the worst-case scenarios. It may only be true when the size of the problem becomes very large, which may not be the case for a particular practical problem at hand.

While applications of mathematical optimization to large realistic transit network design problems are limited thus far, it has been quite successful in other related fields due to the development of powerful mathematical tools in combinatorial search fields, e.g., operations research, electric power scheduling and distribution, etc.

Solution Methodology

The methodology was developed based on the following considerations:

- The method should be generally applicable to the design and optimization of a wide range of TRN problems in practice.
- The solution method should be as generic as possible and should not favor any particular transit network configurations.
- Solutions obtained from this method should give reasonably good results in a reasonable amount of time as permitted by the current computer power affordable to most transit agencies. The results should improve as the computer resource or power increases, and should approach the global optimum when there is no computer resource limitation.

The solution methodology consisted of the following components:

(1) *Representation of Street Nodes, Transit Service Area*, *Transit Routes*, *and Route Network*. Street nodes are represented by an integer set called the street node set where each integer element in this set is associated uniquely with a street node. The transit service area is represented by an integer vector space called the street network set where each integer vector has two integer components corresponding to the starting and ending node identifiers (IDs) of a street segment. A transit route is represented by an integer vector (route vector) with a sequence of ordered integer components, and each of the integer components is associated with a street node on the route. A route network is represented by a set of route vectors. The goal of a TRN optimization is to find a set of route vectors or a route network that corresponds to an optimal goal function or functions.

(2) *Representation of Search Spaces for Transit Routes and Transit Network*. All the solution search spaces are flexibly, iteratively, and locally defined since a complete global search space for a combinatorial optimization problem will evidently result in an intractable solution search space even for a small street network. The formulation of various local solution search spaces is aimed at obtaining search spaces that are computationally tractable for existing computational resources to perform local searches.

(3) *Constraints for Transit Route Network*. Integer constraints in this study include the following: (a) Fixed route constraints prescribing fixed guideway lines or bus routes that are specified by transit planners to meet certain planning goals, which will remain unchanged during the optimization process; (b) Constraints prescribing starting, ending, or in-between areas through which transit routes must pass. These areas may be major activity centers or transfer points; (c) Route length constraints for individual transit lines or for the entire system; and (d) Constraints on the number of transit stops on individual

routes. The above constraints are simple constraints since they do not require function evaluations. Identifying more simple constraints in an integer optimization problem may reduce the sizes of corresponding solution search spaces significantly. In this study several function constraints are also introduced to avoid selecting routes with unreasonable shapes or directness. Such constraints include various route directness constraints and OOD (Out-of-Direction) constraints. The function constraints confine the solution selecting process to solution subspaces consisting of routes that satisfy certain users specified route shape or directness criteria.

(4) *Formulation of Various Optimization Objective Functions*. Objective functions considered in this study are various trip coverage functions or their combinations. The goal is to obtain a TRN structure with minimum transfers while optimizing service coverage. If a trip between an OD pair requires no transfers, the trip is called a zero-transfer trip, while a trip between an OD pair that requires k or fewer transfers will be called a k-or-less transfer trip. A k-or-less transfer trip coverage function, or simply a k-or-less transfer function, is defined as the total number of OD trips that can be accomplished with k or fewer transfers in a transit network service area given a service configuration.

The use of any of the transfer functions alone as the objective function may result in the optimization of one TRN parameter at the cost of others. This study also introduced two objective functions, the average number of vehicle boarding functions, that combine several trip coverage functions thus may give more balanced results. The value of a boarding function are always no smaller than 1, with the optimal value is 1, indicating that all trips are zero-transfer trips.

(5) *Solution Search Algorithms*. There are five solution search algorithms developed in this study. The basic assumption in formulation of these algorithms is that the demand distribution in a TRN service area has certain continuity. In other words, nodes with certain transit demands are probably close to nodes with similar demands. In such cases, it will be more effective in searching for better solution by evaluating paths that are near nodes or areas with higher trip distributions. The first two search algorithms are greedy type search schemes. The basic search strategy of a greedy type search method is to follow the first solution candidate that gives a better goal function value in a search process. Advantages of these two methods are that in general they converge relatively fast and need less memory spaces. However, these two greedy type methods may be trap into poor local optima. The other two search algorithms are based on the hill climbing methods. Conceptually, the hill climbing method is similar to the deepest decent method in continuous research fields. In stead of following the first encountered better result as in greedy type search methods, a hill climbing search process follows the best result in a local search space.

In the description of the two greedy type search methods and the two hill climbing search methods, issues regarding the global-ness of the search results from these methods have not been addressed. In fact, all these methods are local search methods, although the two hill climbing methods may produce better results due to larger search spaces. For non-

convex optimization system, a local search scheme may suffer from premature convergence to or being trapped into one of the local optimal results near initial guess route network. The last search algorithm is also based on a greedy type method. However, to avoid premature convergence to poor local optima, an escape schemes based on the integrated simulated annealing and tabu search methods is included in the greedy search process. This resultant method, called the integrated simulated annealing, tabu, and greedy search method, is capable to escape from any local optima, and theoretically, will visit a global optima eventually with a probability of 1.0.

Numerical Experiments

The methodologies for transit route network design optimization developed from this study has been tested through several benchmark problems studied or developed by Mandl (1979, 1979a), Shih and Mahmassani (1994), and Baaj and Mahmassani (1991). The selection of these benchmark problems is based on the following considerations. Firstly, these problems have been well documented with detailed descriptions of street network input data, OD matrix input data and, most importantly, transit route network results and the corresponding objective function values, which can be used for comparison with results from this study. Secondly, methodologies developed by these authors and the corresponding numerical results seem to be well acknowledged in the transit planning research community. Finally, these benchmark problems are real, practical problems, although with some simplification assumptions, making them appropriate for use as benchmark problems to evaluate results from this study. It needs to be emphasized that the comparison between the results from this study and those from the benchmark problems was merely to validate the methodology developed in this study. In fact, the design objectives and design variables involved in this study are different from those used in studies by Mandl (1979, 1979a), Shih and Mahmassani (1994), and Baaj and Mahmassani (1991). In this study, the objective functions are those which reflect a transit system's network directness and transfer directness, and the design variables are integer vectors representing transit routes. The benchmark problems are TN optimization problems, which include both TRN design optimization and certain parts of TNS (transit network scheduling) design optimization problems, and the design objectives are, in addition to various network directness functions, user or operator costs, such as transfer times, waiting times, etc. A more comprehensive comparison of the methods developed in this study with those benchmark problems may be appropriate after work on transit scheduling optimization is completed. Presently, the comparison involves only those design variables and results related to or obtained from TRN optimization. The numerical tests are performed based the Mandl's problem (1979, 1979a) and the Miami-Dade County transit system. The results showed that the methodology developed in this study was able to improve results from the benchmark problems and improve the system performance in Miami-Dade County compared to the existing system, assuming that the transit demand estimated from the Miami-Dade County 1999 FSUTMS model was accurate.

Software Development

The methodology developed in this research has been implemented in a prototype GIS based program called OPTNet (OPTimization Package for transit Network). The program was implemented with TransCAD as the front end user interface. It allows the user to specify input

data including street network, transit demand (OD matrix), optional route configurations, initial guess routes, fixed routes, pre-specified service areas, optimization parameters, etc. The program can display the input such as streets, initial transit network, and OD matrix graphically as well as the output. The attributes and performance statistics of the optimized transit network are also provided. The user may choose to modify the program output to arrive at a final network configuration by adjusting selected routes and stop locations, or, after making modifications, choose to use the current solution as the initial input into OPTNet for further optimization.

Summary and Conclusions

The methodology developed from this work has a systematic mathematical statement of the TRN problems including the definition of various objective functions, solution search spaces and constraint conditions commonly used in transit planning fields, and a systematic scheme that flexibly defines solution search spaces based on available computing resources and/or optimization problem sizes. Two local search schemes have been developed to obtain results for practical problems of a large-scale in a reasonable amount of time.

The feasibility of the proposed method has been tested through practical TRN optimization problems of realistic sizes. Numerical results showed that the methodology developed in this work was capable of tackling large-scale transit network design optimization problems. Further improvements may include development of TRN optimization methods that consider dynamic transit demand, demand and travel time in different time periods of a day, and waiting and transfer penalties.

Recommendations

As mentioned before, a complete transit network design optimization process should include two design optimization components, i.e., transit route network design optimization and transit network scheduling design optimization. The present work dealt with the optimization of transit route network structure in an attempt to find the optimal route network layouts in terms of network directness, transfer directness, and ridership coverage. However, to realize those optimal characteristics allowed by the resultant route network obtained from the route network design stage, the optimization of transit network schedule design should be implemented. Design variables in network scheduling optimization may include vehicle headways and timetables, and the optimization objective functions may be the user cost, operator cost, or a combination of the two. Constraints may include minimum/maximum vehicle headways, passenger waiting times, vehicle load factors, fleet size, and so on. Although traditional heuristic methods may produce a workable transit schedule by following certain guidelines or criteria, it is important to develop effective mathematical optimization methodologies for transit scheduling design since the differences in cost benefits between a workable result and an optimal or even a good result may be significant, especially for large scale transit network system.

Another important improvement would be to allow the use of travel time instead of travel distance in the optimization process. This is because a shortest path measured by distance may not be the shortest path measured by travel time. For transit users, they are more sensitive to

travel time and less to travel distance. More importantly, unless travel time is used in optimization, travel time savings provided by rapid transit services such as Miami-Dade County's Metrorail and Busway and transit users' preference for these rapid transit services cannot be properly considered and results will not be accurate.

In this study, the street networks were assumed or approximated as undirected networks. This assumption is valid for street networks involving one way streets if any two corresponding one way segments of a bus route are close to each other and the lengths (or travel times) of the two segments are more or less the same. This assumption may not be true in some practical situations when the two one way segments are far apart or when travel time on these two segments are noticeably different. Thus it is desirable to extend the current methodology to support optimization problems based on directed street networks.

Because network travel time varies by route and by time of day, to model a transit network accurately with travel time as the cost measure, it is necessary to consider time-of-day models and network optimization for different periods of time. This means that transit demand for desirable time period during a day is needed. Currently, the accurate estimation of transit OD matrix remains a challenging task. Although several methods based on limited survey data and statistic technologies have been reported in the literature (Tsygainitzky 1979, Simon and Furth 1985, Furth and Navick 1992), they are usually limited to one transit route or one transit corridor, and their validity for use for transit route networks remains unclear. The Automatic Passenger Counters (APC) technology, especially the smart card technology, is a promising ridership and OD data collection means. However, before such technologies become available, additional study will be needed to estimate transit demand and OD matrix. Improvements in OD matrix accuracy may be possibly achieved through modeling and the use of existing ridership data. Sample data collected for selected routes may provide good estimates of the spatial distribution and the temporal patterns of transit demand. These estimates may also be extrapolated to areas not currently being served by transit. One difficulty in using existing ridership data such as boarding and alighting data is the lack of historical data or data in electronic format as most transit properties do not systematically preserve or utilize such data. It is recommended that tools be provided to transit properties to help them preserve such data electronically and allow them to retrieve and analyze the data, and that the potential of utilizing these data for the purpose of helping estimate transit demand be studied.

To improve the speed of the OPTNet program, different strategies and techniques will need to be investigated and applied. The full potential of mathematical optimization approaches to find global or near global optimal results for large-scale transit network analysis seems to lie in parallel computing techniques. This is because the power of a single processor computer is limited by our current knowledge of physics. The computing power of a parallel computer relies on both individual processor's speed and the number of processors in the computers. Although the power of a single computer process is limited, such limitation may be removed if multiple processors are connected to form a parallel computer. Therefore, implementing OPTNet on a parallel platform is naturally the next development stage for any promising mathematical optimization methodologies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1 Problem Statement

One of the goals of the Florida Department of Transportation's 2020 Transit Strategic Plan is to help Florida Transit Systems improve their performance. Reducing the number of transfers involved in a transit trip is an important step towards that end. According to a transit onboard survey conducted as part of the recent South Florida Travel Characteristics Study (Carr Smith Corradino 2000), about one half of the transit trips involved at least one transfer. Approximately 15% involved two or more transfers. A survey (Stern 1996) of various transit agencies in U.S. indicated that about 58% of the respondents believed that transit riders were only willing to transfer once per trip. It is widely accepted in the transit industry in the U.S. that the willingness of riders to transfer has significant impact on the total ridership of a transit system (Newman et al. 1983). This perception is based on empirical transit market studies or past experiences, and if it is valid, the ridership of a transit system may increase significantly by merely reducing transfers through the optimization of the existing transit route network configuration without expanding services or demanding more resources. In addition to ridership, transit route design or configuration may also have a significant impact on transit budgets and route coverage, and thus contributing to the transit system effectiveness and efficiency and the overall quality of life in the urban areas being served.

Factors that affect the passengers' willingness to transfer vary and may include waiting time and walking distance required during transfers, reliability of bus schedules, bus headways, loading factor, and so on. One of the main concerns is the security of passengers and criminal activities that sometimes occur at transfer locations, especially in the evening or at night on low frequency bus routes, where missing one connecting bus may result in a long wait and cause safety concerns (Stern 1996). Transfers increase riders' travel difficulty, which, in turn, has a negative impact on the overall ridership of a transit system. Avoiding unnecessary transfers and/or reducing transfer difficulty have become an important transit design or service improvement issue. Various remedies or improvements have been suggested or implemented, including coordinating bus schedules, increasing bus frequencies, increasing the number of sheltered and well lit bus transfer stops, and reconfiguring existing transit network to reduce transfers. Since this study is concerned with issues related to transit route network reconfiguration or optimization to minimize riders' transfers, the literature review and description presented in this report will mainly deal with issues on transit route network optimization and related topics.

Transfers are a result of a network configuration that is unable to provide direct services between all pairs of origins and destinations. Clearly, the main reason is the constraint on resources, which is faced by all transit properties. It is difficult to eliminate or noticeably reduce bus transfers without significantly increasing bus fleet size and total bus route miles, both of which will result in the increase of operating costs. Moreover, the "inertia" in transit services, i.e., the inability to immediately or fully respond to changes in population, housing, employment, and demographics (such instant response may be neither desirable nor practical), also contributes to transfers that may be potentially eliminated.

For transit systems with small bus route networks, a seasoned planner may obtain reasonable bus route network results by utilizing his or her knowledge and experience, carrying out simple ridership data collection and analysis, and following certain simple guidelines. For large urban areas where the number of bus routes may be close to or over a hundred and the number of bus stops in thousands, intuition, past experiences, and simple guidelines may not be enough to produce even near-optimal transit route network configurations due to various complicated factors involved in transit route network design. Systematic methodologies are therefore needed to obtain better transit route network configurations.

1.2 Goals and Objectives

The goal of this study was to develop a methodology and a software tool for optimizing bus transit services to reduce transfers. The research objectives included:

1. Provide an understanding of the current state-of-the-art and the state-of-the-practice in transit network optimization.

(4) Develop a methodology for optimizing transit service configuration based on a synthesis/refinement of the state-of-the-art and the state-of-the-practice. The methodology would have the ability to deal with a larger transit network than had been reported in the literature. Based on the methodology a more robust network optimization tool would be produced, which could be used for practical planning purposes.

2. Develop a user-friendly computer tool for transit agencies to optimize their bus services with optimal transfers.

1.3 Organization

This report is divided into six chapters. In the remaining chapters, a review of literature in the field of transit network and schedule optimization is presented in Chapter 2. Chapter 3 describes the methodology developed in this study, including the problem and solution representation and solution search methods. Validity tests of this methodology are described in Chapter 4, where the results from this study are compared to some well known benchmark problems. The methodology was also applied to a large-scale transit network configuration problem based on the Miami-Dade County transit system, and the results are presented. Chapter 5 discusses the development of a computer program that implements the methodology from this study. Finally, conclusions, recommendations, and future direction of further research are provided in Chapter 6.

2. LITERATURE REVIEW

2.1 Background

In general, a transit network (TN) design process has two design stages: transit route network (TRN) design and transit network scheduling (TNS) design. A TRN design problem deals with issues related to transit route layout or coverage, while a TNS design problem involves transit vehicle headway or frequency design, and vehicle assignment timetables or bus schedule tables. Mathematically, a TN design problem may be stated as the determination of a set of transit routes and associated frequencies, subject to a set of feasibility constraints, to achieve the desired objectives that minimize the overall cost, usually a combination of user and operator costs. User costs are often measured by the total travel time incurred to the users, which consists of access time, waiting time, in-vehicle travel time, transfer time, etc. Operator costs depend on the fleet size, transit vehicle size, transit vehicle miles, and vehicle operation hours required for a particular route configuration. Feasibility constraints may include, but are not limited to, maximum allowable bus headway, vehicle load factors, and available resources including capital and operating costs.

In practice, a more realistic TN design problem statement may involve the minimization of multiple, often conflicting objectives under complex multiple constraints. For instance, minimization of operator costs, maximization of coverage of transit service area and service hours, and minimization of the number of transfers are objectives that conflict with each other since increasing the transit service coverage area or reducing riders' transfers will increase the operator costs. In such cases, compromises must be made to balance different objectives to obtain solutions. Different constraints may also be explicitly or implicitly in contradiction to each other or to certain optimization objectives. For example, a transit planner may specify a constraint as the maximum number of bus routes allowed in a transit system. Limits placed on the total number of bus routes are usually a result of practical consideration based on available transit budget, since an increase in the number of bus routes usually results in a larger bus fleet size and more bus drivers if the existing levels of service are to be maintained. The transit planner may also set a constraint on the maximum length of individual transit routes based on consideration of issues such as difficulty in maintaining bus schedule adherence, bus driver fatigue, and so on. Inappropriate setting of these two constraints may result in difficulty in obtaining a solution because to cover a given transit coverage area, a solution procedure may have to increase either the total number of bus routes in the system, or to increase the coverage of individual bus route, i.e., the length of certain bus routes or their service hours.

Mathematically, a typical optimization process may be stated as bellow.

Maximize/minimize: $f(\boldsymbol{x}, \boldsymbol{n})$ for all $\boldsymbol{x} \in \boldsymbol{R}$ (read as $\boldsymbol{x}$ in $\boldsymbol{R}$) and $\boldsymbol{n} \in \boldsymbol{I}$ [2.1a]

Subject to:

(a) Equality constraints:

$$g_i(\boldsymbol{x}, \boldsymbol{n}) = 0,\ i = 1, 2, \ldots, k \quad [2.1b]$$

(b) Inequality constraints:

$$h_j(\boldsymbol{x}, \boldsymbol{n}) \leq 0,\ j = 1, 2, \ldots, m \quad [2.1c]$$

where $f(\boldsymbol{x}, \boldsymbol{n})$ is an objective function, $\boldsymbol{x} = \boldsymbol{x}(x_1, x_2,\ldots, x_r)$ is a vector (or, more generally, a set of vectors) with continuous or real components, and $\boldsymbol{n} = \boldsymbol{n}(i_1, i_2,\ldots, i_s)$ is a vector (or, more generally, vectors) with discrete or integer components. $\boldsymbol{R}$ in [2.1a] is a space of vectors with continuous variable components, which usually has the following form,

$$\boldsymbol{R} = \boldsymbol{R}\{\boldsymbol{x}(x_1, x_2, \ldots, x_r) \mid a_i \le x_i \le b_i, i = 1, 2, \ldots, r\}, \quad [2.2a]$$

where real numbers a_i and b_i define the range of vector $\boldsymbol{x}$'s i^{th} component. $\boldsymbol{I}$ in [2.1a] is a set or space[1] of vectors with integer variable components,

$$\boldsymbol{I} = \boldsymbol{I}\{\boldsymbol{n}(i_1, i_2, \ldots, i_s) \mid n_j \le i_j \le m_j, j = 1, 2, \ldots, s\}, \quad [2.2b]$$

where integers n_j and m_j define the range of vector $\boldsymbol{n}$'s j^{th} component. For simplicity, a vector with continuous components will be referred to as a continuous or real vector and a vector with integer components will be referred to as an integer vector. If the optimization problem defined in [2.1] does not involve integer vectors, i.e., $\boldsymbol{n} = \boldsymbol{0}$, then the problem is considered a traditional or continuous optimization problem. However, if the optimization problem in [2.1] involves only integer vectors, i.e., $\boldsymbol{x} = \boldsymbol{0}$, it is called an integer optimization problem. A mixed integer optimization problem is one that contains both integer and continuous vectors, i.e., $\boldsymbol{x} \neq \boldsymbol{0}$ and $\boldsymbol{n} \neq \boldsymbol{0}$.

A combinatorial optimization problem is a special case of integer optimization problems. It refers to an integer optimization problem where the integer vector's component set $(i_1, i_2,\ldots, i_s)$ in vector $\boldsymbol{n} = \boldsymbol{n}(i_1, i_2,\ldots, i_s)$ is an ordered subset of a larger integer set, $N\{i_1, i_2, \ldots, i_s\}$ $(s \le n)$. For example, the set that includes all ordered two-integer subsets of the integer set $\boldsymbol{N}\{1, 2, 3\}$ is {(1, 2), (1, 3), (2, 3), (2, 1), (3, 1), (3, 2)}, and the integer vector set defined on such a set is called a combinatorial integer set or space based on $\boldsymbol{N}\{1, 2, 3\}$,

$$\mathbf{C}_{\mathbf{N}}^{(2)}\{\boldsymbol{n}\} = \mathbf{C}_{\mathbf{N}}^{(2)}\{\boldsymbol{n}(1,2), \boldsymbol{n}(1,3), \boldsymbol{n}(2,3), \boldsymbol{n}(2,1), \boldsymbol{n}(3,1), \boldsymbol{n}(3,2)\}.$$

For clarity, in this report, a set (integer or real) enclosed in parentheses, e.g., $(a_1, a_2,\ldots, a_k)$, represents an ordered set, while a set (integer of real) enclosed in braces, e.g., $\{a_1, a_2,\ldots, a_k\}$, represents an unordered set. In general, a combinatorial space based on all s-integer subsets of an integer space $\boldsymbol{N}\{i_1, i_2,\ldots, i_t\}$ may be expressed as

$$\mathbf{C}_{\mathbf{N}}^{(s)}\{\boldsymbol{n}\} = \{\boldsymbol{n}(i_1, i_2, \ldots, i_s) \mid \text{for all } (i_1, i_2, \ldots, i_s) \subset \boldsymbol{N}\{i_1, i_2, \ldots, i_n\} \quad [2.3a]$$

It may be seen that as n and s, i.e., the size of space $\boldsymbol{N}\{i_1, i_2,\ldots, i_n\}$ and the number of components of vector $\boldsymbol{n}(i_1, i_2,\ldots, i_s)$, increase, the corresponding combinatorial space $\mathbf{C}_{\mathbf{N}}^{(s)}\{\boldsymbol{n}\}$ can be very large. In expression [2.3a], all the integer vectors have the same number of component s. if s can vary within certain range, e.g., $(s_{min} \le s \le s_{max})$ where $s_{\min}$ and $s_{\max}$ are two given integers,

[1] In general, a set is referred to a group of individuals that have certain common features such as a vector set, a solution set, an integer set etc. A space is a set with certain geometric or other measurements such as length, distance etc. For example, a 2D vector space is a set, and one of the geometric measurements is the length of individual vectors.

then the combinatorial space based on all s-integer subsets ($s_{min} \le s \le s_{max}$) of an integer space $\boldsymbol{N}\{i_1, i_2,\ldots, i_t\}$ may be expressed as

$$\mathbf{C}_{\mathbf{N}}^{(s_{\min}, s_{\max})}\{\boldsymbol{n}\} = \{\boldsymbol{n}(i_1, i_2, \ldots, i_s) \mid \text{for all } (i_1, i_2, \ldots, i_s) \subset \boldsymbol{N}\{i_1, i_2, \ldots, i_n\}\ (s_{min} \le s \le s_{max}). \quad [2.3b]$$

Space $\boldsymbol{C}_{\boldsymbol{N}}^{(s_{\min}, s_{\max})}$ will be referred to as a combinatorial space based on integer space $\boldsymbol{N}$, and space $\boldsymbol{N}$ will be referred to as the base integer space, or simply the base space, of combinatorial space $\boldsymbol{C}_{\boldsymbol{N}}^{(s_{\min}, s_{\max})}$. For $\boldsymbol{N}\{1, 2, 3\}$, $s_{\min} = 1$ and $s_{\max} = 2$, the combinatorial vector space defined in [2.3b] becomes

$$\mathbf{C}_{\mathbf{N}}^{(1,3)}\{\boldsymbol{n}\} = \{\boldsymbol{n}(1), \boldsymbol{n}(2), \boldsymbol{n}(3), \boldsymbol{n}(1,2), \boldsymbol{n}(1,3), \boldsymbol{n}(2,3), \boldsymbol{n}(2,1), \boldsymbol{n}(3,1), \boldsymbol{n}(3,2)\}.$$

A mixed combinatorial problem is similar to a combinatorial optimization problem but it also includes real vectors, i.e., $\boldsymbol{x} \ne \mathbf{0}$, and $\boldsymbol{n}$ is defined on a combinatorial vector space. TN design is a typical combinatorial optimization problem, where the base integer space $\boldsymbol{N}\{i_1, i_2,\ldots, i_n\}$ is the set of all potential transit stops (or street nodes that are suitable for use as transit stops) in the transit service area. An integer vector $\boldsymbol{n} = \boldsymbol{n}(i_1, i_2,\ldots, i_s)$ may represent a transit route if the vector's component set ($i_1, i_2,\ldots, i_s$), an ordered s-integer subset of the base space $\boldsymbol{N}\{i_1, i_2,\ldots, i_n\}$, represents the sequence of the transit route's stops/nodes. A TRN may be represented by a set of integer vectors,

$$\boldsymbol{T}^{(l)} = \boldsymbol{T}^{(l)}\{\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_l\}, \quad [2.4a]$$

where l is the number of transit routes in the route network system, and

$$\boldsymbol{n}_j = \boldsymbol{n}\left(i_1, i_2, \Lambda\ , i_{s_j}\right), (j = 1, 2, \ldots, l) \quad [2.4b]$$

are the l transit route vectors, s_j is the number of transit stops on transit route $\boldsymbol{n}_j$. It may be seen that a transit route vector is a member of the combinatorial space $\boldsymbol{C}_{\boldsymbol{N}}^{(s_{\min}, s_{\max})}$ defined in [2.3b], i.e., $\boldsymbol{n}_j \in \mathbf{C}_{\mathbf{N}}^{(s_{\min}, s_{\max})}\{\boldsymbol{n}\}$ ($j = 1, 2, \ldots, l$) and a transit route network is a subset/subspace of $\boldsymbol{C}_{\boldsymbol{N}}^{(s_{\min}, s_{\max})}$, i.e., $\boldsymbol{T}^{(l)}\{\boldsymbol{n}\} \subset \mathbf{C}_{\mathbf{N}}^{(s_{\min}, s_{\max})}\{\boldsymbol{n}\}$. Based on the above description, a TRN design optimization problem may be stated as follows:

Maximize/minimize: $f(\boldsymbol{x}, \boldsymbol{T}^{(l)})$ for all $\boldsymbol{x} \in \boldsymbol{R}$ and $\boldsymbol{T}^{(l)} \subset \boldsymbol{C}_{\boldsymbol{N}}^{(s_{\min}, s_{\max})}$ [2.5a]

Subject to:

(a) Inequality constraints:

$$l_{\min} \le l \le l_{\max} \quad [2.5b]$$

$$s_{\min} \le s_i \le s_{\max},\ i = 1, 2, \ldots, l \quad [2.5c]$$

(b) Equality constraints:

$$g_i(\boldsymbol{x}, \boldsymbol{T}^{(l)}) = 0,\ i = 1, 2, \ldots, i_g \quad [2.5d]$$

(c) Other inequality constraints:

$$p_i(\boldsymbol{x}, \boldsymbol{T}^{(l)}) \le 0,\ i = 1, 2, \ldots, i_p \quad [2.5e]$$

(d) Certain topology constraints on transit routes:

$$q_i(\boldsymbol{n}),\ i = 1, 2, \ldots i_q \qquad [2.5f]$$

where $\boldsymbol{T}^{(l)}$ is a TRN with l routes defined in [2.4a] and [2.4b]; $l_{\min}$ and $l_{\max}$ are, respectively, the minimum and the maximum numbers of transit routes in the network system; and $s_{\min}$ and $s_{\max}$ are, respectively, the minimum and the maximum numbers of nodes/stops on a transit route. $\boldsymbol{C}_{\boldsymbol{N}}^{(s_{\min}, s_{\max})}$ in [2.5a] is the combinatorial space defined in [2.3b]. Expressions [2.5d] and [2.5e] represent other constraints not explicitly stated such as route directness constraints, etc. Topology constraints for transit routes include, for example, that a transit route should not have loops or intersect itself and that neighboring transit stops must be connected with street segments. The transit network optimization problem defined in [2.5] is a typical integer or mixed integer (if real vectors $\boldsymbol{x} \neq \boldsymbol{0}$) combinatorial optimization problem. The solution of such problems is difficult even with relatively small number of integer variables since the associated solution procedure involves discrete optimization, which usually requires the search for optimal solutions from a combinatorial search space with an extremely large number of possible solutions (Papadimitriou and Steiglitz 1982, Nemhauser and Wolsey 1988).

Typical mixed combinatorial optimization problems in TN design are those of which the objectives are to find both the optimal TRN structure and the optimal values of certain design parameters associated with the route network, such as bus frequencies, transfer times, etc. Optimization of TRN configuration involves the search for an optimal set of feasible transit routes of which the topology (or the structure of the route network) is unknown. For bus route scheduling optimization problems, analytical approximation or transformation methods (Newell 1973) have been used, where the original integer optimization problem is related to a gradient or sub-gradient based optimization problem involving only continuous variables, i.e., $\boldsymbol{n} = \boldsymbol{0}$. Optimization of continuous system is a well-developed research field in which various powerful solution algorithms are available. However, it is unlikely that such approximation or transformation exists for the optimization of TRN configuration problem due to the complex nature of transit network topology.

A great deal of research work has been done in the area of transit network optimization, including transit route design, scheduling optimization, or a combination of the two. The methods reported in the literature may be roughly grouped into the following categories:

(1) Mathematical optimization based approaches;
(2) Heuristic approaches;
(3) Genetic algorithm (GA) based approaches;
(4) Knowledge-based expert system (KBES) based heuristic approaches; and
(5) Other methods.

Mathematical optimization approaches include mainly two classes of optimization: traditional gradient based approaches that mainly deal with problems with continuous variables or discrete/integer variables that could be approximated by or related to continuous variables, and those that deal with integer optimization and/or combinatorial optimization. Heuristic approaches are widely used in transit planning since they utilize planners' knowledge, past experience, and physical intuition to incrementally improve existing systems. They are easy to

understand and always produce reasonable solutions, but may not be transferable to other systems. Genetic algorithm based approaches are conceptually closed to mathematical optimization. However, it may be more convenient to consider them as a separate group of approaches since these methods use solution search schemes/algorithms that are quite different from those used in the traditional mathematical approaches. KBES is a branch of Artificial Intelligence (AI) and has been widely applied in various fields with great success. KBES based heuristic approaches attempt to simulate the decision making process by formalizing the knowledge and heuristic rules used by transit planners in designing bus route network, which serve to quickly narrow down the search space and to find reasonable solutions.

The above method of categorizing various solution approaches for transit route network design in the literature is merely designed for the convenience for this study. There are other ways to group the different methodologies. The choices made to identify various approaches reflect in part personal taste/opinion, or expertise, and in part a preference depending on researchers/planners' emphases, goals, or sometimes convenience to describe and compare different approaches. For example, Chua and Silock (1982) identified the following approaches to transit network restructuring and optimization:

(1) Manual approach using service standards and guidelines;
(2) Systems analysis using standard travel demand and trip assignment models;
(3) Market analysis using manual trip assignment for corridors or small service areas;
(4) Systems analysis with interactive graphics;
(5) Heuristic procedures; and
(6) Mathematical optimization.

This categorization also reflects the historical evolution of transit network design methods in the past.

Baaj (1990) characterized different approaches based on the formulation or methods used in the following design components/stages: objective function, demand, constraints, passenger behavior, and solution techniques. Shih and Mahmassani (1994) included two more features: the decision variables and service types. Another classification used by Ceder and Israeli (1997) was as follows:

(1) Simulation models that simulate passenger flow;
(2) Ideal network methods; and
(3) Mathematical programming models.

Due to the complexity in transit network design, there are no clear boundaries between different approaches. A method based on mathematical optimization formulation may incorporate heuristic guidelines to limit its search space to obtain solutions with limited computing power. A method based on certain heuristic guidelines or criteria may also employ mathematical programming techniques in one or more of its design stages or components. An approach is considered as a mathematical optimization approach if the problem is formulated as a mathematical optimization problem on a relatively complete solution search space. Mathematical (or generic) heuristic search algorithms may then be introduced in order to make

the resultant system solvable with available computer resources. Here *mathematical* heuristic search algorithms refer to those search methods that are less domain knowledge specific and have been used in a wide range of network applications. Examples of such algorithms include greedy type algorithms, hill climbing algorithms, nearest-neighbor algorithms, simulated annealing approaches, Tabu search algorithms, and so on. Descriptions of various mathematical heuristic algorithms may be found, for example, in (Aarts and Lenstra 1997, Bertsekas 1998, Gould 1988, Hillier and Lieberman 1986, Nemhauser and Wolsey 1988).

An approach is considered a heuristic approach if domain specific heuristic guidelines or criteria, mostly based on transit planners' past experiences, are first introduced to establish a solution strategy framework, usually consisting of several solution stages, and at each or some of the stages, mathematical optimization or other solution techniques are employed to obtain the best results. The main difference between these two types of approaches lies in the fact that the mathematical optimization approach formulates the problem on a solution space that, theoretically, should include all possible solutions. Heuristic guidelines or algorithms are then applied to reduce the search space size. In contrast, the heuristic approach formulates a problem directly on possible solution sub-spaces defined based on heuristic guidelines. Mathematical programming techniques or other solution means may be then used to find the best solution from the solution sub-spaces. It is inevitably that some approaches in the literature may qualify as both mathematical programming and heuristic approaches, while some other methods may not be considered as either.

Table 2.1 provides a summary of the main features of some of the approaches reported in the literature where *Math Opt.* stands for mathematical optimization, *GA* for genetic algorithm, *NN* for neural networks, *AI* for artificial intelligence, and *Gen*. for generalized. The fifth column classifies the solution approaches. H&M (Heuristic and Mathematical) indicates that the solution was first established based on a heuristic framework but certain mathematical optimization methods were employed at some design stages to reduce solution spaces. It is not surprising that most of the approaches in the table fall into this class (as are in the literature). In column 7, titled *Constraints*, HG means heuristic guidelines.

The next three sections describe the various methods used in transit route network design and related research fields, especially those that are relevant to the optimization of transit network configurations that minimize transfers. The review focuses on various approaches that have been used in TN design processes and those with good potentials to improve TN design procedures and results. The purpose is to investigate the applicability, maturity, and the potential for further improvements of the various methodologies currently used in transit planning field. Detailed description and comparison on various individual authors' work on transit planning fields may be found in (Baaj 1990, Shih and Mahmassani 1994, Axhausen 1984), among others.

Table 2.1 Main Features of Some Approaches Used in Transit Network Design

Year	Author	Optimization Objectives	Design Variables	Solution Approaches	Service Types	Constraints	Others
1967	Lampkin & Saalmans	Gen. user time	Route & frequency	H&M	Fixed route & schedule	Operating cost	Transfer considered
1974	Silman et al.	Gen. user cost	Route & frequency	H&M	Fixed route & schedule	Operating cost	Transfer considered
1976	Mandl	Gen. Time	Route	H&M	Fixed route & schedule	Route coverage, directness	Transfer considered
1979	Dubois et al.	Gen. user time	Route & frequency	H&M	Fixed route & schedule	Operating cost	Transfer considered
1988	Van Nes et al.	No. of direct trips	Route & frequency	H&M	Fixed route & schedule	Operating cost, flee size	Transfer considered (simple examples)
1981	Hasselström	Cost & user benefit	Route & frequency	H&M	Fixed route & schedule	Operating cost	Transfer considered
1990	Baaj	Multi-object approach	Route & frequency	H&M	Fixed route & schedule	Multi-constraints (HG)	–
1991	Israel & Ceder	Gen. time & fleet size	Route & frequency	H&M	Fixed route & schedule	–	–
1991	Bander	Optimal path	Route itinerary	Math dynamic programming	Fixed route & schedule	–	Formulation and discussion
1992	Baaj & Mahmassani	Route network AI-based representation	Transit network design	AI		Multi-constraints (HG)	Formulation and discussion
1992	Bookbinder & Désilets	Disutility function reflect transfer inconvenience	Timetable & headway (offset time variables)	Math Opt.	Fixed route & schedule (random bus travel times)	HG	Formulation & simple examples
1994	Shih & Mahmassani	Multi-object approach	Route & frequency	Heuristic	Fixed route & variable schedule	Multi-constraints (HG)	–
1995	Baaj & Mahmassani	Multi-object approach	Route & frequency	Hybrid AI heuristic	–	Multi-constraint	Transfer considered
1995	Hill & Fu	Total expected waiting time	Headway, transfer time, & schedule	Math Opt. stochastic	Fixed route & schedule	–	Theory with ideal examples
1998	Bruno et al.	Gen. cost to access rapid transit line	Location of a rapid transit line	Math Opt. dynamic programming	Fixed route & schedule	Route connectivity, coverage, etc.	Multi-transit models
1998	Pattnaik et al.	Gen. time (user & operator)	Route network	GA	Fixed route	Frequency, load factor	–

Year	Author	Optimization Objectives	Design Variables	Solution Approaches	Service Types	Constraints	Others
1998	Janarthanan & Schneider	Increase productivity of TNWD	Knowledge based design parameters	AI-KBES	Various	Various	Applied to prototype problem
1999	Soehodo & Koshi	Gen. social cost	Route & frequency	H&M	–	–	Transfer considered
2001	Caramia	Travel time & management cost	Route network & vehicle No.	GA & NN	–	–	Applied to real case
2002	Bielli et al.	Total operator and user cost	Route shape & headway	MATH	–	–	–
2003	Chien et al.	Transfer directness	Route network	MATH	–	Route length, waiting time, load factors, etc.	–

2.2 Mathematical Optimization Techniques

Optimization is a process or a technique to search for the best solution for a problem or a procedure as effective, perfect, or useful as possible. In mathematical terms, optimization is a technique for finding a maximum/minimum value of an objective function of several variables subjecting to a set of constraints. For transit network design problems, mathematical optimization is usually formulated as constrained mixed integer optimization problems, which are usually combinatorial problems. The resultant systems are then solved with the appropriate linear/nonlinear programming or other techniques. For combinatorial TN design problems with a large number of integer variables, heuristic algorithms and certain relaxation or approximation schemes are usually applied to either reduce the size of the search space or transform the problem at hand to another solvable problem.

The TN design optimization problem may be stated as the determination of a set of transit routes and/or associated frequencies, subject to a set of constraints, to achieve the desired objective(s) that minimizes the overall cost and other objectives that have impacts on the overall ridership of the transit system. The overall cost is generally a combination of user and operator costs. Other objectives may include minimizing the number of transfers, waiting time, and so on. Constraints are usually expressed as inequality conditions, such as the maximum allowable bus headway, vehicle load factor, bus fleet size, maximum route length, number of routes, and the integrality restrictions on some or all the variables, as well as other requirements related to transit polices.

Compared with other methods used in transit network design, mathematical optimization based solution approaches usually have a more complete solution search space and rigorous problem statements. Results from a properly defined mathematical model are usually not biased toward any existing route networks. The theory behind the solution algorithms, such as various linear or nonlinear programming techniques and mathematical heuristic algorithms developed based on graph theory and combinatorial optimization research, is relatively mature and has been applied successfully in various fields such as air traffic control, operations research, telecommunication

network design, electric power system scheduling and distribution, and so on for a long time. There are a variety of available solution algorithms and/or program code in the literature and market (Mor and Wright 1993, Bertsekas 1998, Bertsekas 1991, Gould 1988, Nemhauser and Wolsey 1988). Under appropriate conditions such as continuity, once or twice differentiable conditions of the objective functions, unknowns or design parameters, convex problem defined domains, etc., the mathematical optimization based methods usually can give more accurate results than other methods. It is well accepted that the mathematical programming approaches have the potential to become powerful and systematic tools for complicated TN design problems.

The existing mathematical optimization methods, however, have a number of disadvantages. Firstly, conditions to guarantee a stable, rapid convergence to a unique global optimal solution are usually quite strict. Such conditions include, for example, the existence of the first and the second order derivatives of the objective functions with sufficient smoothness, convex feasible search regions usually defined by various inequality constraints, Kuhn-Tucker conditions, and solution search space sizes computationally feasible for available computer resources (CPU time and storages) (Sheffi 1985), and so on. The necessary condition for an optimal solution to exist is related to the characteristics of the first order derivative (or the gradient matrix for multi-variable problems) of the objective functions. This condition is sometimes called the first order condition. The sufficient condition of an optimal solution is related to the characteristics of the second order derivatives (or the Hessian matrix for multi-variable problems). These two conditions are sometimes referred to as the first and the second order conditions for an optimization system. For practical problems, to meet the above conditions or their approximations may require rigorous mathematical derivation and cumbersome regularity schemes.

Secondly, the overall (solvable) condition of a mathematical optimization system derived from a TN design problem may be sensitive to the setting of certain design parameters. For example, Newell (1979) illustrated nonconvex optimization example by studying the effect of the rider's waiting time to the resultant system (for a description of non-convexity please refer to Appendix A). Other factors such as various constraints may also result in a nonconvex system. Existing literature does not provide information on rigorous mathematical analyses or methodologies to avoid or to deal with factors that result in a nonconvex system. For a nonconvex optimization system, an optimal solution from the minimization process is not guaranteed. A solution may only be a local optimal solution (stationary or saddle point solution, to be more accurate) near the initial guess network provided by the user. A global optimal solution must be chosen from all the possible local optimal results. For complex problems, there seem to be no effective ways to systematically locate all the possible regions where local optimal solutions may occur. Consequently, in engineering practices a reasonable near- or sub-optimal solution may have to be accepted.

Finally, due to the need to search for optimal solutions from a large search space made up by all possible solutions, the resultant mathematical optimization systems derived from realistic mixed combinatorial TN design problems (e.g., problems involving both transit network configuration and other parameters such as bus frequency and bus feeder assignments) are usually NP-hard (Bertsekas 1998, Nemhauser and Wolsey 1988, Magnanti and Wong 1984). The term “NP-hard” refers to problems for which the number of elementary numerical operations (comparison,

addition, subtraction, multiplication, etc.) or computer CPU time needed to solve the problems is not likely to be expressed or bounded by a polynomial function of form $p(l, n, m, f_i)$, where l, n and m may represent, in a TN design optimization problem for example, the number of bus routes, the number of possible transit stops, and the number of street segments of the street network system, and f_i $(i = 1, 2, \ldots k)$ are unknown parameters such as bus frequencies defined on the transit route network. Problems or algorithms for which the worst-case measure of the number of elementary operations or CPU time needed to obtain solutions can be expressed in polynomial forms are called problems or algorithms with *polynomial time complexity* (PTC). Any problems for which no polynomial algorithms exist are called intractable problems, or problems with intractable time complexity (ITC). An example of the ITC class of problems is the exponential time complexity (ETC) problem, where the number of required elementary operations is an exponential function, such as $a^{p(l, n, m)}$ $(a > 1)$. There are other kinds of problems that sit between the PTC and ITC classes. The *non-deterministic polynomial* (or NP time complexity) problems/algorithms, including NP hard and NP complete problems[2], belong to these classes of problems (Garey and Johnson 1979). Table 2.2, fashioned after the one from (Gould 1988), illustrates the time required for solving problems of various time complexity classes with respect to problem sizes n. For a particular problem, n usually represents the number of unknowns, nodes, degrees of freedom, or the size of the data set. The time complexity order (TCO) in Table 2.2, where TCO = $O(f(n))$ and $f(n) = n, n^2, n^3, 2^n,$ *or* 3^n, represents the number of operations required to solve a problem of size n, and the symbol $O(f(n))$ (read big oh of $f(n)$) represents the order of $f(n)$, or to be more specific, $O(f(n)) = c\, f(n)$ where c is a nonzero constant. To illustrate, Table 2.2 assumes $c = 1$ and that one operation requires an average of 0.000001 second. The problems described in the first three rows are in the PTC class because the numbers of operations of the problems can be expressed in terms of polynomial forms of the associated problem size n, i.e., the linear polynomial form TCO = n, the quadratic polynomial form TCO = n^2, and the cubic polynomial form TCO = n^3. The CPU times required to solve these three PTC class problems for n = 10, 30, 50, and 100 are also given in the table. The last two rows are problems in the ITC class, or precisely, in the ETC class. It may be seen that as the problem size increases, the CPU time required to solve these two ITC class problems increases exponentially. For n = 100, which is a relative small number for a practical TN design problem, the CPU time is 2^{48} centuries for TCO = 2^n and 3^{70} centuries for TCO =3^n. Such numbers are in astronomical order, and are intractable for any existing computers.

Table 2.2 Comparisons of Solution Time for Various Time Complexity Classes and Problem Sizes

Time Complexity Order (TCO)	Problem Size			
	$n = 1$	$n = 30$	$n = 50$	$n = 100$
$O(n)$	0.00001 sec	0.00003 sec	0.00005 sec	0.0001 sec
$O(n^2)$	0.0001 sec	0.0009 sec	0.0025 sec	0.01 sec
$O(n3)$	0.1 sec	24.3 sec	5.2 min	2.7 hr
$O(2^n)$	0.001 sec	17.9 min	35.7 yrs	2^{48} cent
$O(3^n)$	0.059 sec	6.5 yrs	2×10^8 cent	3^{70} cent

Notes: sec – seconds; yrs – years; cent – century

[2] The terms *NP hard* and *NP complete* both refer to the hardest or most difficult problems in NP classes. NP hard usually refers to optimization problems while NP complete is for decision problems.

An example of the PTC class problem is the multiplication of two matrices of order n, $C = AB$. Following the typical rules for matrix multiplication, the element of the resultant matrix C is expressed as:

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj} \ .$$

It is easy to see from the above expression that the calculation of c_{ij} involves n multiplications, and n-1 additions. Since there are n^2 entries in matrix C, i.e., c_{ij} (i = 1, 2, …, n), the matrix multiplication algorithm requires a polynomial form of $p(n) = n^2(n + n - 1) = 2n^3 - n^2$ operations to complete the task. Thus the time complexity order of this problem is

$$\text{TCO} = O(p(n)) = O(2n^3 - n^2) = O(n^3).$$

As a cautionary note, the term *time complexity order* or *computational intractability* that is used to describe a problem, an approach, or an algorithm is only an asymptotic estimate to the solvability of a problem, and is usually based on the worst-case scenarios. It may only be true when the size of the problem becomes very large, which may not be the case for a particular practical problem at hand. Some simplex algorithms are probably in the exponential time complexity category (Evans and Minieka 1992). However, various simplex algorithms have been widely used to solve practical mathematical programming problems of realistic sizes.

Combinatorial TRN design optimization problems are usually at least in NP class since polynomial-time algorithms for finding optimal solutions are unlikely to exist. Some TRN algorithms are ITC because the number of elementary operations or the CPU times is usually in the order of exponential time complexity, such as $3^{p(n,m)}$. An example of ITC problem for TRN design is illustrated in Figure 2.1. The solid line and black circles represent an existing bus route and its stops/nodes, while the white circles and doted lines show the possible bus stops/nodes and the street segments near the bus route. Assume that one would like to find the bus route configuration with the minimum travel impedance from bus stop 1 to stop n by evaluating all the possible bus routes near the existing one. Note that a route with minimum travel impedance may not be the route with the shortest distance since the travel impedance usually includes the effects of roadway functional class, traffic congestion, delays at intersections, and other factors that resist or slow down traffic flow. It may be easily seen that the total number of all possible bus routes in Figure 2.1 should be a finite number. Exhaustive search schemes, such as the exhaustive enumeration (checking the feasibility of every possible solution and comparing feasible solutions' objective function values to find the minimum), should guarantee a global optimal for any optimization problems based on a finite solution space. To estimate the number of all possible bus routes based on the existing bus route $\boldsymbol{r}(1, 2, \ldots n)$, one may use the procedure bellow.

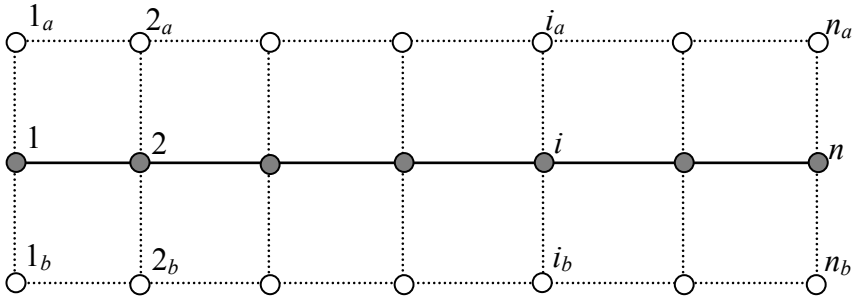


Figure 2.1 A Transit Route and Neighboring Street Network

1. Start from the existing bus route $\boldsymbol{r}(1, 2, \ldots, n)$:
   a) Keep all nodes on the existing route fixed except node 1.
   b) Adjusting node 1 to obtain the following additional routes.
      (1) Option 1: $\boldsymbol{r}(1, 1_a, 2_a, 2, \ldots, n)$
      (2) Option 2: $\boldsymbol{r}(1, 1_b, 2_b, 2, \ldots, n)$

2. For any of the three bus routes in Step 1:
   a) Keep all nodes on the existing route fixed except node 2.
   b) Adjusting node 2 to obtain the following routes, noting that one could obtain two new routes from each of the three existing routes in Step 1, thus there are six new routes resulting from this step:
      (1) Option 1: $\boldsymbol{r}(1, 2, 2_a, 3_a, 3, \ldots, n)$
      (2) Option 2: $\boldsymbol{r}(1, 2, 2_b, 3_b, 3, \ldots, n)$
      (3) Option 3: $\boldsymbol{r}(1, 1_a, 2_a, 3_a, 3, \ldots, n)$
      (4) Option 4: $\boldsymbol{r}(1, 1_a, 2_a, 2, 2_b, 3_b, 3, \ldots, n)$
      (5) Option 5: $\boldsymbol{r}(1, 1_b, 2_b, 3_b, 3, \ldots, n)$
      (6) Option 6: $\boldsymbol{r}(1, 1_b, 2_b, 2, 2_a, 3_a, 3, \ldots, n)$
3. For any one of the six bus routes obtained in Step 2 and three bus routes obtained in Step 1 (nine altogether):
   a) Keep all nodes on the existing route fixed except node 3.
   b) Adjusting node 3 to obtain 18 new routes since, as in Step 2, one could obtain two new routes from each of the nine existing routes in steps 1 and 2. Thus there are 18 new routes resulted in this step.

4. Repeat the procedures described in Step 2 and Step 3 to node 4 through node n-1.

The total number of possible bus routes based on this procedure may be calculated. Since at the k^{th} step, the total number of possible bus routes is $S_k = S_{k-1} + 2S_{k-1} = 3S_{k-1}$, where S_{k-1} and S_k are the total numbers of possible bus routes at $(k-1)^{\text{th}}$ and k^{th} steps, respectively, and from Step 1, $S_1 = 3$, the total number of possible bus routes at the last step, i.e., Step n - 1, will be

$$S_{n-1} = 3S_{n-2} = 3(3\,S_{n-3}) = 3^2 S_{n-3} = 3^3 S_{n-4} = \ldots = 3^{n-2} S_1 = 3^{n-1}.$$

Thus, the problem of finding the bus route with minimum travel impedance by evaluating all possible bus routes is exponentially intractable with the time complexity order TCO $=3^{n-1}$. For n

= 100, a reasonable number of bus stops for an urban bus route, the CPU time required to evaluate all the 3^{99} alternative paths, with the current computing technologies, may be well over the age of the universe, which is believed to be about 140 billion years old.

For a network of a realistic size, the number of bus routes or lines *l*, the number of possible bus stops *n*, and the number of street segments *m,* may be in the order of hundreds or thousands. Thus, the selection of an optimal transit route network structure through an exhaustive search scheme or any search scheme with completeness may involve computer operations or CPU time of astronomical proportions (Newell 1979, Garey and Johnson 1979). Domain specific heuristic search schemes, either mathematical or physical, may have to be introduced into a mathematical optimization process to improve the solvability of a problem.

It is worthwhile to point out that the above mentioned difficulties in the current mathematical optimization approaches are not a result of the mathematical optimization formulation, which merely simulates/represents the physical reality of the transit network system. The nonconvex thus non-uniqueness of the optimal solution due to various local optima as well as the large search spaces is inherent from the complicated and combinatorial nature of TN design physical reality. Other approaches will encounter the same difficulties as the size of their possible solution search spaces increases. Due to the significant computing time, existing mathematical optimization solution approaches to TN design problems are usually applied to relatively small and idealized networks for small urban areas or medium-sized urban areas with coarse networks. The route network structures may also be limited to certain particular configurations where solution uniqueness or convexity may be recovered, such as many-to-one, one-to-many, or spinal network structures, etc. However, while applications of mathematical optimization to large realistic transit network design problems are limited thus far, it has been quite successful in other related fields, e.g., operations research, electric power scheduling and distribution, etc. Large systems that involve millions of unknowns and constraints have also been solved successfully (Bertsekas 1998).

Previous research in TN design that involves various mathematical optimization techniques, either partially or whole, includes, among others, the work by Lampkin and Saalmans (1967), Byrne and Vuchic (1972), Silman et al. (1974), Byrne (1975), Rapp and Gehner (1976), Dubois et al. (1979), Mandl (1979), Hasselstrom (1981), Oudheusden et al. (1987), Oldfield and Bly (1988), LeBlanc (1988), Kuah and Perl (1988), Désilets (1989), Chang (1990), Norojono (1990), Israel (1990), Bookbinder and Désilets (1992), Hill and Fu (1995), Ceder and Israel (1997), and Soehodo and Koshi (1999). Most of these studies introduced some heuristic criteria or certain simplification assumptions to limit the solution search space or to reduce the optimization objectives to a particular network structure or a few design parameters, e.g., route spacing, route length, stop spacing, bus size, bus headway or frequency, and rider waiting times. Additionally, most of the studies were also based on predetermined route network structures. Additional information on and reviews of various mathematical optimization approaches may be found in (Chua 1984, Axhausen and Smith 1984, Baaj 1990, Soehodo and Koshi 1999).

2.3 Traditional Heuristic Approaches

2.3.1 Overview

Heuristic approaches, or the traditional heuristic approaches, have been used for bus route network design for a long time and may be traced back many decades to the time of the earliest bus transit systems. They are still the most widely used methods in transit planning and route network design problems. The term "traditional" here describes various heuristic approaches used in transit planning that are to be distinguished from mathematical heuristic algorithms in graph theory and in various mathematical combinatorial research fields, and from AI-based heuristic approaches. The main differences between these heuristic methods are that traditional heuristic approaches are strong domain-specific and depend heavily on personal knowledge and experiences, while mathematical heuristic algorithms are relatively less domain-specific since these methods usually apply to a variety of applications in different fields with little or no modifications in their basic search criteria and principles. Examples of various mathematical heuristic methods that have been widely used in engineering, economics, social and natural sciences include greedy algorithms, nearest neighbor algorithms, depth-first, breadth-first, best-first search schemes, rollout algorithms, Tabu search methods, and simulated annealing methods, and so on (Bertsekas 1998, Wolsey 1998). AI-based heuristic approaches are also strongly domain-specific. However, the focus of these methods is the organization of domain knowledge and automated reasoning processes, and a particular heuristic solution scheme may be only a part of a complete AI-based solution process. Ideally, an AI-based heuristic approach should include any solution algorithms appropriate to the problem at hand.

Mathematically, heuristic methods are a solution technique that improves the search efficiency or solvability while possibly sacrificing claims of completeness. It is a practical tool for problems of which the search space is too large or for which there are no appropriate ways to define a complete search space. Unlike mathematical optimization approaches where the formulation is intended to, theoretically, find the global optimal solution, heuristic approaches are usually formulated to obtain a local optimal, a sub-optimal, or any solutions that result in certain improvements over existing ones. Pearl (1984) defines heuristics methods as "criteria, methods, or principles for deciding which among several alternative courses of action promises to be the most effective in order to achieve some goal." Heuristic methods are usually problem dependent since their search criteria, principles, and guidelines are domain or problem specific.

In TN design, heuristic approaches are usually a combination of applications of guidelines and procedures for route selections and bus frequency/headway determination, based on criteria established from past experiences, ridership and demand data, cost and feasibility constraints, intuition of the transit planners, as well as some policies out of certain social and/or political considerations. The route network structures obtained from heuristic approaches tend to be of certain types that are intuitive and conceptually easy to understand or accept by planners. They are usually shaped by historical reasons or affected by existing systems that have evolved gradually with demographic changes in the urban areas they serve. Typical network types include, for example, radial networks, grid networks, trunk/feeder or transit-corridor networks, as well as transit-center networks (Toliver et al. 1989).

To date, especially with the rapid development in computer techniques, the heuristic approaches have experienced significant advances. A heuristic approach in TN design may include several design stages or components. At each stage, sub-design problems are usually formulated. Depending on the complexity of the sub-problems and the availability of the associated solution tools, solutions are obtained based on heuristic guidelines, analytical formulation, or mathematical programming techniques. Typical design goals of sub-problems may be, for instance, route network skeleton, bus route network spacing, bus stop spacing, headway, transfer times, waiting time, or transit network reconfiguration. Results from different design stages are then evaluated and modified iteratively in each design cycle and interactively between different design stages using heuristic guidelines (Shih and Mahmassani 1994, Baaj 1990).

The advantages of heuristic approaches are several. Firstly, they are always able to provide feasible solutions to TN design problems of any size. For instance, given a problem of a realistic size, the issue concerning a heuristic approach is mainly the effectiveness of its solution, or the improvement of its solution over existing ones. In contrast, the challenge for a mathematical optimization problem is often whether or not the problem can be solved with existing computer resources.

Secondly, the formulation and/or solution strategy of each design stage is relatively independent to others, thus making it easier to adopt a solution method that best fits a particular design goal or stage, such as linear or nonlinear programming methods for bus route network generation (mostly bus route network skeleton or certain predefined configurations), analytical solutions for bus route frequency design, and stochastic methods for estimating waiting time, etc.

Thirdly, it is relatively easy to incorporate various constraints into the solution procedures, since heuristic approaches usually select solutions from a possible solution space that already meet most of the design constraints. Finally, it is relatively easy to deal with issues or design requirements due to political reasons, such as concerns in social fairness regarding coverage of transit route network, traffic flow, noise level, and so on.

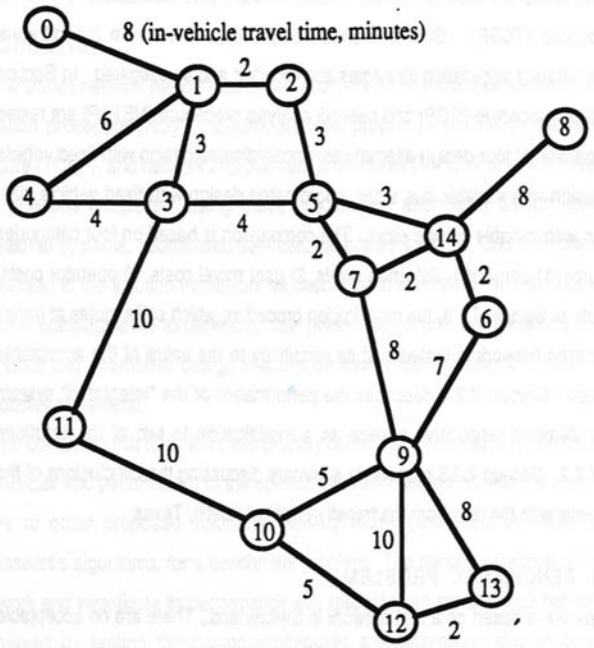
Heuristic approaches also have disadvantages. The main disadvantage of heuristic approaches is that their results are almost certainly not global or even local optimal. This may be attributed to the fact that search schemes in heuristic approaches are usually ad hoc procedures based on computer simulations of human transit design processes guided by heuristic rules. The corresponding search spaces are usually not clearly defined and the search results are likely to be biased toward existing systems or any systems on which the set of design heuristics are based. Moreover, heuristic approaches usually decompose complex TN design problems into several manageable sub-problems, and appropriate solution schemes and/or approximate assumptions are then applied to these sub-problems. However, a solution that has optimal performances separately at different design stages or based on particular route network configuration may not necessarily be the optimal result. Although solutions from various heuristic approaches have been described as local optima or sub-optima, performances of various heuristic approaches in the literature are usually evaluated or judged by improvements, sometimes in terms of percentages of changes of certain design parameters over existing systems, which may not be valid statements on the optimal-ness of the results.

Literature on various heuristic approaches may be found, among others, in (Lampkin and Saalmans 1967, Rea 1972, Silman et al. 1974, Mandl 1979, Dubois et al. 1979, Hasselstrom 1981, Ceder and Wilson 1986, Van Nes, et al. 1988, Baaj 1990, and Israeli and Ceder 1991). A thorough review of various heuristic approaches, including mathematical optimization methods, may be found in (Baaj 1990, Axhausen 1984, Chua and Silcock 1982).

The next two subsections provide a brief description of several heuristic methods that have been applied to practical transit network design problems for medium-sized and relative large transit service areas. The studies have been well documented including detailed descriptions of solution algorithms, heuristic guidelines, and, most importantly, computer code and input data sets. These methods have been used or tested for practical transit network design problems, although some are quite simplified. The work summarized here seems to be well acknowledged in the transit planning research community, thus may serve as a good starting point as well as benchmarks for this study.

2.3.2 Mandl's Method

(1) Mandl (1979, 1979a) developed a heuristic algorithm for urban transit network design. The method was applied to a coarse transit network design problem based on a real network in Switzerland. Mandl's problem was a small and dense network of 15 nodes with a total demand of 15,570 trips per day, a relatively high demand density for the small network. The travel time between the two farthest nodes in the network was 33 minutes along the shortest path. The network connectivity of Mandl's problem is shown in Figure 2.2, where each transit node is labeled by an integer, and the in-vehicle travel time in minutes between adjacent connected node pairs is indicated next to the corresponding street segment. The transit demand matrix for the 15 transit nodes is also illustrated in Figure 2.2. The matrix shows the average number of passenger trips per day between each transit node pair. Figure 2.3 illustrates Mandl's final solution network. Important measures of Mandl's solution network included, for example, 100% service coverage, 69.94% of the trips involving no transfers, 29.93% of the trips involving one transfer, and only 0.13% of the trips needing more than one transfer. Mandl's work has been widely cited by researchers. Axhausen (1984) tested Mandl's algorithm to a transit network based on data from Madison Metro to evaluate the feasibility of Mandl's algorithm for transit networks of a different size and configuration. Baaj and Mahmassani (1991) and Shih and Mahmassani (1994) applied Mandl's algorithm and testing problem as a benchmark to test their solution frameworks and results.



Transit Demand Matrix in List Form :

(0 400 200 60 80 150 75 75 30 160 30 25 35 0 0)
(400 0 50 120 20 180 90 90 15 130 20 10 10 5 0)
(200 50 0 40 60 180 90 90 15 45 20 10 10 5 0)
(60 120 40 0 50 100 50 50 15 240 40 25 10 5 0)
(80 20 60 50 0 50 25 25 10 120 20 15 5 0 0)
(150 180 180 100 50 0 100 100 30 880 60 15 15 10 0)
(75 90 90 50 25 100 0 50 15 440 35 10 10 5 0)
(75 90 90 50 25 100 50 0 15 440 35 10 10 5 0)
(30 15 15 15 10 30 15 15 0 140 20 5 0 0 0)
(160 130 45 240 120 880 440 440 140 0 600 250 500 200 0)
(30 20 20 40 20 60 35 35 20 600 0 75 95 15 0)
(25 10 10 25 15 15 10 10 5 250 75 0 70 0 0)
(35 10 10 10 5 15 10 10 0 500 95 70 0 45 0)
(0 5 5 5 0 10 5 5 0 200 15 0 45 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)

Figure 2.2 Mandl's Swiss Network and Transit Demand Matrix (Reproduced from Mandl 1979)

Mandl's transit network design approach consists of two stages. During the first stage, a feasible initial route network is generated. The emphases are service coverage and directness. The solution procedures at this stage may be summarized bellow.

(1) Find the shortest paths between all connected transit nodes to form a possible solution space. This is achieved via a mathematical optimization algorithm.

(2) For each transit node pairs, select one shortest path from all possible shortest paths, which may be more than one, following heuristic guidelines.

(3) Create initial transit routes/lines by connecting selected transit node or terminal pairs following heuristic guidelines.

(4) Check service coverage, and add unserved nodes (nodes that are not connected to a transit route) to appropriate existing routes or create new routes with the unserved nodes as route terminals.

Geometrically, the above algorithm should always produce a feasible route network although it may not be the best one. The second stage of Mandl's method tries to minimize an objective function of total travel time, including in-vehicle and waiting times, following a hierarchical, iterative procedure. The limitation of Mandl's method is the lack of consideration of the demand pattern during the route construction stage.

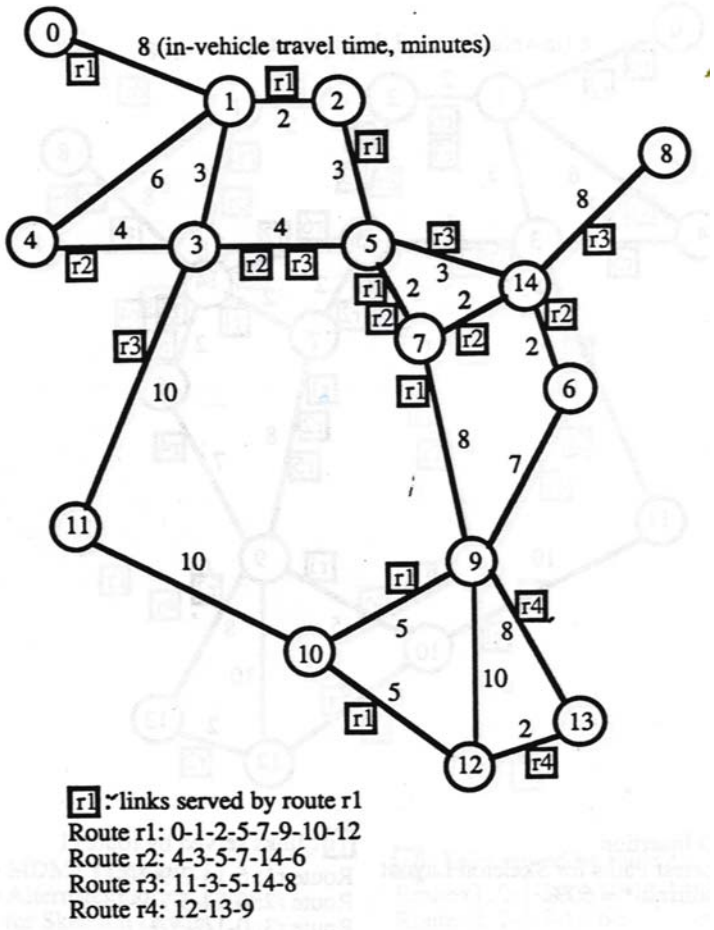


Figure 2.3 Route Network Layout by Mandl's Algorithm (Reproduced from Mandl 1979)

2.3.3 Shih and Mahmassani's Method

Shih and Mahmassani (1994) developed a heuristic framework and computer based procedures for transit network design. Their work is based on earlier work of Baaj (1990) and Baaj and

Mahmassani (1991) with the consideration of (a) the ability to identify transfer centers and incorporate the transfer center concept into the network design process; (b) provision of alternative design concept including conventional and coordinated bus schedules, as well as timed-transfer concepts; (c) allowing the option of variable vehicle size. Creation of transfer centers is one approach to minimize transfer times by selecting appropriate locations (often major employment or activity centers) in a transit network where a large number of transit lines (including trunk and feeder lines) will converge or intersect and by coordinating the transit vehicle arrival and departure times so that waiting time for passengers may be minimized. Design of a transit network with this approach involves identification of transfer center locations, modification of transit routes, and redesign of schedules.

Shih and Mahmassani's approach consists of the following design stages or components.

(1) Route Generation Procedure (RGP)

The RGP was evolved from earlier work by Baaj and Mahmassani (1991) with the incorporation of the transfer center concept. It is a typical example of heuristic route network design. First, it generates a possible route network solution space by connecting all the transit demand node pairs with the shortest paths. It provides options to include modified or alternative shortest paths to expand the solution space. Next, it begins to query the planner for service directness, transit coverage level, and other appropriate input or data that could be used to define the initial network skeletons, such as transit node pairs that can possibly serve as transit centers and node pairs with high demand values. The network skeletons are then expanded to more detailed route network iteratively via various node selection and insertion strategies guided by the transit planner's knowledge and expertise. The RGP terminates when the objectives of the transit system coverage and service directness are achieved.

(2) Network Analysis Procedure (NETP)

The NETP consists of a series of iterative procedures for trip assignment, bus vehicle sizing, frequency setting, and computation of various system performance measures. Values of various transit design parameters such as bus fleet size, route frequencies, vehicle sizes and other measures reflecting service quality, system utilization, and user and operator costs are provided by the route network results from RGP. The NETP could also be used for transit system evaluation by comparing various parameters obtained from the NETP with a set of parameters either from an existing system or from earlier iterations.

(3) Transit Center Selection Procedure (TCSP)

The TCSP selects feasible transit node pairs as transit centers according to certain heuristic criteria that reflect commonly used guidelines for transfer centers. Criteria identified by the authors include: (a) proximity to a major activity center; (b) location in a population cluster or in major community area; (c) reasonable distance from other transit

centers; (d) good transfer opportunities; and (e) feasibility considerations such as accessibility, land availability, and geographical limitations.

(4) Network Improvement Procedure (NIP)

The NIP seeks to improve a set of routes generated by the RGP that may be economically and operationally infeasible. The route modification acts either at the transit coverage level or at the route structure level. Modifications include route discontinuation, route merging, route splitting, and branch exchange of routes.

Shih and Mahmassani's approach was been tested with Mandl's small and dense network benchmark problem shown in Figure 2.2. This benchmark problem was also used by Baaj and Mahmassani (1991) to test their solution framework (from which Shih and Mahmassani's approach evolved). Figure 2.4 shows one of the route network layout from Baaj and Mahmassani's algorithm, while Figure 2.5 shows results from Shih and Mahmassani's network layout based on the transit center concept. Table 2.3 presents a comparison of the results from the above different route network design approaches. Improvements from Shih and Mahmassani's approach over the earlier work by Mandl (1979) and Baaj and Mahmassani (1991) included, for example, 82.59% 0-transfer service coverage compared with Mandl's 69.94% (Mandl, 1979) and Baaj and Mahmassani's 78.61% (Baaj and Mahmassani 1991), and a total transfer time (penalty) of 13,350 compared with Mandl's 23,500 (Mandl, 1979) and Baaj and Mahmassani's 16,650 (Baaj and Mahmassani 1991).

Table 2.3 Comparison of Approaches for Mandl's Benchmark Network Problem (Reproduced from Shih and Mahmassani, 1994)

Network Characteristics	RGP with Transit Center Concept		Baaj and Mahmassani's Solution	Mandl's Solution
	Coordinated	Uncoordinated		
% demand zero-transfer	82.59	82.59	80.99	69.94
% demand one-transfer	17.41	17.41	19.01	29.93
% demand two-transfers	0	0	0	0.13
% total demand unsatisfied	0	0	0	0
Total travel time (minutes)	225,102	203,936	217,954	219,094
Total in-vehicle travel time	191,826	170,328	180,356	177,400
In-vehicle waiting time	20,933	-	-	-
Total out-of-vehicle time	19,726	20,058	22,804	18,194
Total transfer time (penalty)	13,550	13550	14,800	23,500
Fleet size	87	84	82	99
Operation cost ($)	4,043.14	3,924.26	3,830.03	4,620.61
Fuel consumption (gallons)	346.8	336.6	328.52	396.33
Third set of input design parameters: 70% minimum total demand satisfied without transfers, MD node insertion strategy, and shortest path heuristic				

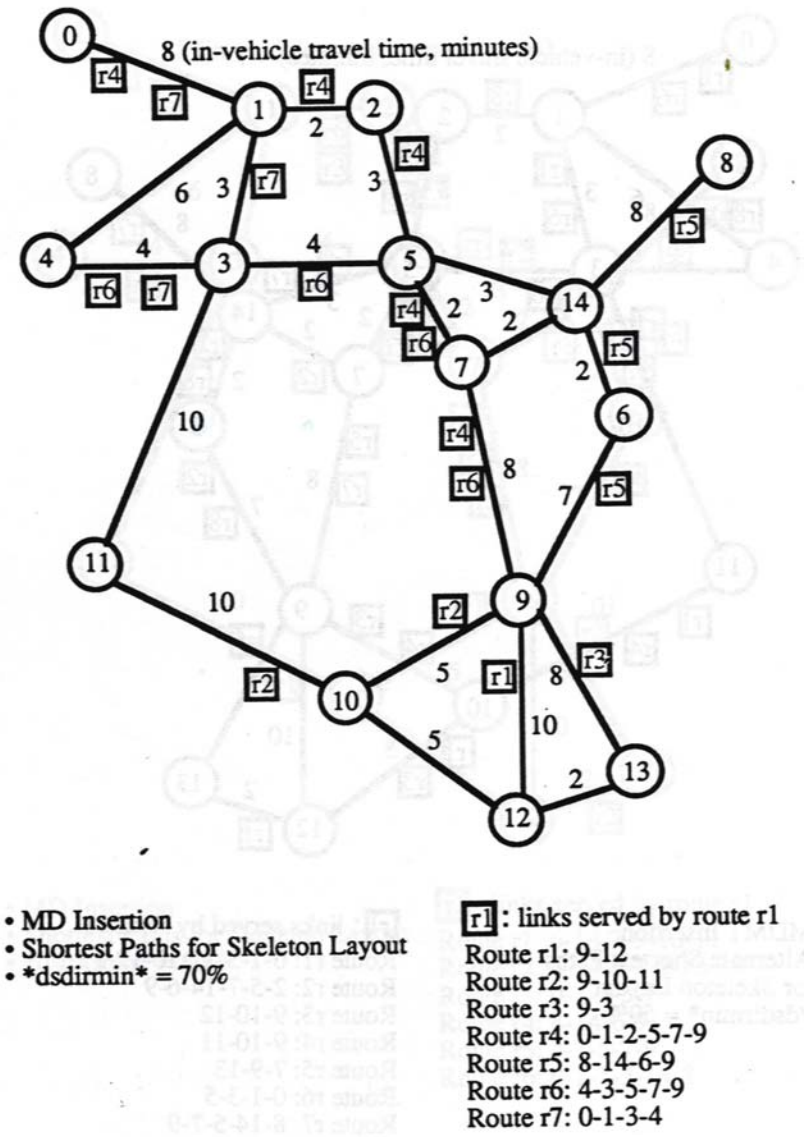


Figure 2.4 Route Network Layout Case 3 by Baaj and Mahmassani's Approach (Reproduced from Shih and Mahmassani 1994)

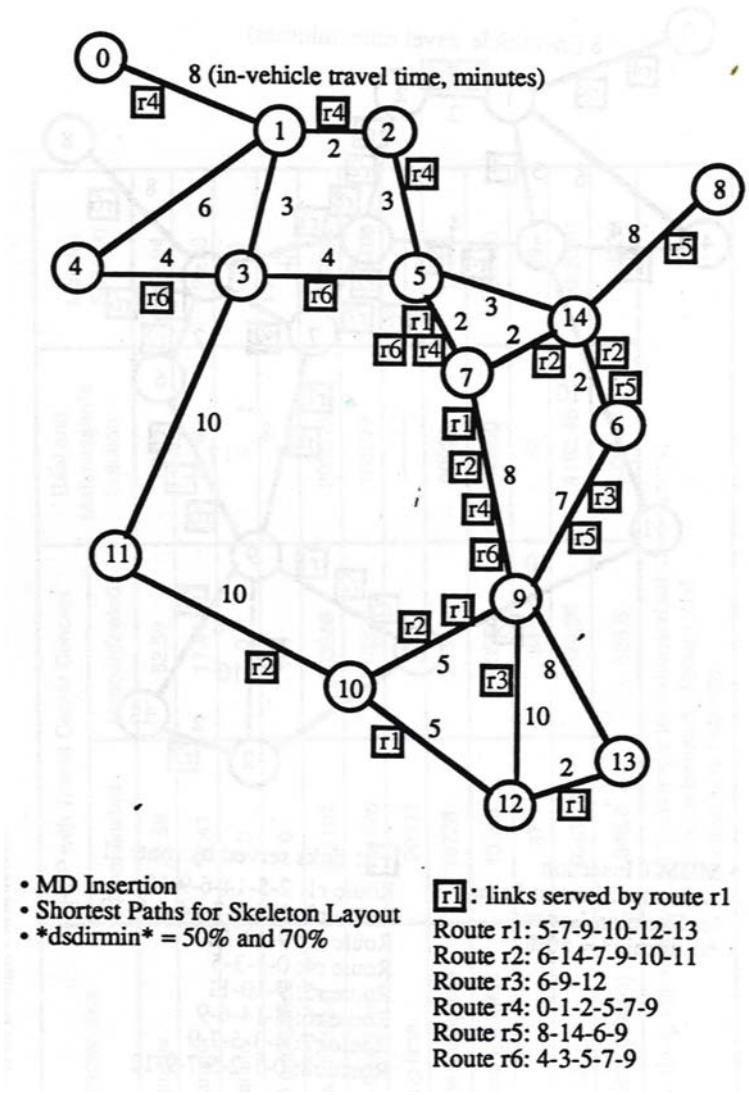


Figure 2.5 Route Network Layout Cases 1 & 3 by Shih and Mahmassani's Approach (Reproduced from Shih and Mahmassani 1994)

The performance of Shih and Mahmassani's method was further investigated with an actual transit network and data from the transit system of Austin, Texas. The service area was represented by a transit network defined based on 177 transit nodes. All the 177 nodes were selected from an existing transit network consisting of 40 fixed-schedule bus routes. To obtain the demand matrix, which is a critical input data set for any transit network design and is often estimated from on-off surveys regularly conducted by many transit agencies, Shih and Mahmassani applied Tsygainitzky's fluid analogy model (Tsygainitzky 1979). Tsygainitzky's model for generating the transit demand matrix has been successfully tested by Simon and Furth (1985) for a single transit line, and has been widely used by transit agencies due to its simplicity and relative reliability (Shih and Mahmassani 1994). However, the accuracy or validity of using this model to obtain the demand matrix for a transit network that consists of more than a single bus line is unclear. The network connectivity of Shih and Mahmassani's problem was defined by two lists: a list of various locations and their associated node numbers and a street network

list or matrix generated from street links that connected these 177 nodes and that were suitable for bus operations.

Figure 2.6 shows a portion of the network node definition list for the Austin problem. The network definition list was obtained by assigning an integer number to each of the physical locations of bus stops in the network service area. Figure 2.7 shows a partial list of the Austin transit network's nodal connectivity array. The network connectivity array lists all the connected neighboring nodes and the corresponding distances for each transit node. Complete input data of this test problem may be found in Shih and Mahmassani's original manuscript. Figure 2.8 shows a part of Austin's street network. Several transit nodes that may be used as transit centers are also indicated. These figures show an example of the basic input data needed to perform a transit route network optimization task. The same kinds of data inputs or their approximation are also required for this research. Figures 2.6 and 2.7 show the essential data for transit network design optimization task, which must come from field survey or from a transit agency's network database obtained from previous investigation to ensure an accurate simulation of the real world.

To describe the results from the application of Shih and Mahmassani's algorithm to the Austin transit network design problem will require a lengthy discussion on various effects of changes of the design parameters such as the relationships between the zero-transfer trips and the system's minimum service coverage, the total in-vehicle and total out-of-vehicle travel times versus the level of directness of the system, and so on. The best solution network in terms of overall service coverage and zero-transfer trip coverage gave 98% coverage of the total trips and 80% of zero-transfer trips, although the user and operator costs corresponding to such results are unclear. A composite indicator that may reflect the overall performance of a solution network seems to be needed to characterize the "optimal-ness" of the results corresponding to various parameters' settings.

Node	Location
0	Pflugerville Elementary
1	2nd at Congress
2	6th at Congress
3	11th at Colorado
4	Guadelupe at 15th
5	Guadelupe at 24th
6	Guadelupe at 45th
7	Lamar at Koenig
8	N. Lamar Transit Center
9	Mearns Meadow at Rutland
10	San Jacinto at 11th
11	Rosewood at Chicon
12	Oak Springs at Airport
13	Lott at Prock
14	2nd at Brazos
15	Brazos at 6th
16	Rio Grande at MLK
17	Lamar at 38th
18	Burnet at Koenig
19	Northcross at Foster
20	MLK at Chicon
21	7th at Colorado
22	7th at I-35
23	7th at Chicon
24	7th at Pleasant Valley
25	Riverside at Montopolis
26	Guadelupe at MLK
27	Speedway at 38th
28	Woodrow at Koenig
29	Anderson at Woodrow
30	12th at Chicon
31	12th at Airport
32	E. 12th at Springdale
33	U.S. 183 at Technicenter (ACC)
34	Trinity at MLK
35	45th at Duval
36	Highland Mall
37	Georgian at Rundberg
38	Braker at Bluff Bend
39	Braker at Dessau
40	Oltorf at Burton
41	E. 5th at Pleasant Valley
42	Airport at Springdale
43	MLK at Springdale
44	Manor at Rogge
45	Berkman at Briarcliff
46	Cameron at St. Johns
47	W. 12th at West Lynn
48	Enfield at Exposition
49	Rockmoor at Windsor

Node	Location
50	35th at Pecos
51	Laguna Gloria
52	S. 1st at Barton Springs
53	S. 1st at Oltorf
54	S. lst at Ben White
55	S. 1st at Emerald Wood
56	William Cannon at S. 1st
57	William Cannon at Woodhue
58	26th at San Jacinto
59	26th at Lafayette
60	38 1/2 at Cherrywood
61	Hancock Center
62	Red River at MLK
63	Colorado at 2nd
64	Lamar at Barton Springs
65	Manchaca at Lamar
66	Manchaca at Ben White
67	Cannon League at Matthews
68	Congress at Oltorf
69	Congress at Ben White
70	Sheraton at Congress
71	Fort Clark at Battle Bend
72	William Cannon at Congress
73	Bluff Springs at William Cannon
74	San Jacinto at MLK
75	Monroe at Congress
76	St. Edwards at Eastside
77	Oltorf at Parker
78	Woodward at Parker
79	Red River at 26th
80	51st at Airport
81	Barton Springs at Bouldin
82	S. 5th at Oltorf
83	Banister at Ben White
84	Westgate Mall
85	Nueces at MLK
86	E. 2nd at Chicon
87	Lyons at Springdale
88	Gardner at Lotus
89	Gardner at Levander
90	E. 6th at I-35
91	MLK at Airport
92	FM 969 at Craigwood
93	Travis St. School
94	24th at Rio Grande
95	Windsor at Harris
96	35th at Jefferson
97	Hancock at Bull Creek
98	Northland at Balcones
99	Village Center at Far West

Node	Location
100	Robert Mueller Airport
101	Manor at Loyola
102	Crystalbrook at Loyola
103	RBJ Center
104	38 1/2 at I-3 5
105	29th at Guadelupe
106	35th at Exposition

Figure 2.6 Partial List of the Austin Problem's Transit Node Definition (Reproduced from Shih and Mahmassani 1994)

```
'((0 ((37 49.0) (38 42.0) (115 58.1) (146 56.7) (147 57.05) (148 63.35) (153 30.45)))
  (1 ((14 0.35) (52 0.75) (63 0.35) (87 17.5) (108 1.25) (116 4.25) (154 2.45)(172 4.65)))
  (2 ((3 2.75) (15 0.4)(21 0.75) (22 2.95) (108 0.4) (168 2.9) (176 0.8)))
  (3 ((2 2.75)(21 1.35) (168 0.85) (171 1.25) (175 0.35)))
  (4 ((10 3.4) (16 2.3)(26 1.45) (34 3.5) (47 5.25) (74 3.5) (85 2.15)(141 3.75) (170 2.4) (171 2.5) (175 1.8)))
  (5 ((26 2.15) (34 4.9) (85 2.7) (94 1.15) (105 2.7)))
  (6 ((7 5.65) (17 4.7) (27 4.65) (28 9.05) (35 2.85) (97 9.65) (105 6.1)))
  (7 ((6 5.65) (8 9.65) (28 2.5) (29 10.5) (35 9.1) (36 5.9) (146 10.15)))
  (8 ((7 9.65) (28 10.05) (29 3.5) (36 9.0) (37 7.0) (38 21.0) (112 5.6)(113 8.1) (114 8.95) (115 13.0) (146
      7.2) (147 11.25) (148 13.9)))
  (9 ((38 12.9) (113 4.2) (114 2.45) (149 9.0) (153 13.6) (155 7.5)))
  (10 ((11 7.05) (14 3.4) (15 2.0) (22 3.5) (30 6.95) (34 3.2) (74 2.15)(86 6.9) (90 3.5) (103 6.55) (104 10.5)
      (116 7.2) (141 2.35) (168 0.4)(171 2.95)))
  (11 ((10 7.05) (12 7.35) (23 2.65) (24 6.0) (30 1.4) (42 9.15) (87 7.9)(90 6.15)))
  (12 ((11 7.35) (13 7.5) (24 7.0) (31 1.45) (32 3.4) (42 2.85) (87 4.9)))
  (13 ((12 7.5) (32 6.55) (42 7.8)))
  (14 ((1 0.35) (10 3.4) (15 1.2) (22 3.85) (34 6.65) (90 3.4) (103 7.5)(108 1.35) (172 5.05)))
  (15 ((2 0.4) (14 1.2) (22 2.85) (103 8.6) (108 0.7) (168 1.8)))
  (16 ((4 2.3) (85 0.4) (94 1.9) (95 5.7) (175 4.15)))
  (17 ((6 4.7) (18 10.7) (27 3.3) (28 9.75) (95 8.75) (96 2.95) (97 10.5)(105 4.75)))
  (18 ((17 10.7) (19 11.15) (28 2.85) (97 7.8) (98 4.6)))
  (19 ((18 11.15) (29 5.95) (97 16.85) (98 12.6) (109 4.25) (112 9.55)(164 9.4) (165 3.95)))
  (20 ((30 2.1) (59 1.75) (62 3.6) (91 5.6) (131 6.15)))
  (21 ((2 0.75) (3 1.35) (22 2.95) (63 1.8) (140 3.95) (171 0.8)))
  (22 ((10 3.5) (14 3.85) (15 2.85) (21 2.95) (23 4.0) (30 5.7) (34 5.6)(62 4.8) (90 0.4) (104 10.5) (141 3.95)
      (168 3.45)))
  (23 ((11 2.65) (24 4.7) (41 4.4) (86 1.7) (90 4.4)))
  (24 ((12 7.0) (41 1.2) (42 7.35) (87 4.55) (89 7.2) (132 11.2)))
  (25 ((11 6.0) (40 12.05) (117 12.1) (118 16.1) (130 12.15) (132 7.9)(133 6.35) (134 3.95))) (26 ((4 1.45) (5
      2.15) (85 0.75) (170 1.1)))
  (27 ((6 4.65) (17 3.3) (35 4.5) (58 4.4) (61 4.4) (104 4.8) (105 4.25)(170 7.2)))
  (28 ((6 9.05) (7 2.5) (8 10.05) (17 9.75) (18 2.85) (29 8.05) (36 7.45)(97 5.25)))
  (29 ((7 10.5) (8 3.5) (19 5.95) (28 8.05) (36 10.0) (112 6.7)(164 8.55)))
  (30 ((10 6.95) (11 1.4) (20 2.1) (22 5.7) (31 6.9)(141 4.4)))
  (31 ((12 1.45) (30 6.9) (32 4.75) (91 2.25) (100 4.9)))
  (32 ((12 3.4) (13 6.55) (31 4.75) (33 14.3) (42 4.2) (43 3.6) (92 9.2) (100 6.3)))
  (33 ((32 14.3) (43 7.9) (88 7.2) (92 4.7) (101 12.25) (102 11.05)(148 21.35)))
  (34 ((5 4.9) (14 6.65) (22 5.6) (58 3.6) (62 0.7) (74 0.35)))
  (35 ((6 2.85) (7 9.1) (27 4.5) (36 13.0) (58 6.05) (61 3.7) (80 4.2)(104 3.15)))
  (36 ((7 5.9) (8 9.0) (29 10.0) (35 13.0) (37 23.45) (45 11.2) (46 8.4)(80 2.3) (142 7.7) (146 9.8) (148 18.9)))
  (37 ((8 7.0) (36 23.45) (38 12.65) (39 14.0) (114 1.55) (115 14.65)(146 11.9) (147 11.9) (148 18.2) (0
      49.0)))
  (38 ((8 21.0) (9 12.9) (37 12.65) (39 3.65) (114 12.05) (115 19.6)(146 18.9) (147 15.25) (148 17.5) (149
      8.75) (153 14.0) (0 42.0)))
  (39 ((37 14.0) (38 3.65) (115 14.0) (147 15.6) (148 16.1) (153 20.65)))
  (40 ((25 12.05) (77 1.15) (117 5.5) (120 3.7) (130 6.7)))
  (41 ((23 4.4) (24 1.2) (86 4.8) (87 5.25) (103 7.35) (117 8.1) (120 9.45)(132 8.75)))
  (42 ((11 9.15) (12 2.85) (13 7.8) (24 7.35) (32 4.2) (87 1.75) (88 8.25)(89 7.0) (132 8.6)))
  (43 ((32 3.6) (33 7.9) (44 8.55) (91 7.35) (92 5.95) (100 6.5) (101 8.75)(102 12.95)))
  (44 ((43 8.55) (45 7.5) (100 10.35) (101 4.95)))
  (45 ((36 11.2) (44 7.5) (46 5.65) (101 10.65) (142 4.6) (148 10.5)))
  (46 ((36 8.4) (45 5.65) (101 15.4) (115 6.65) (142 5.95) (146 5.8)(147 4.5) (148 10.15)))
  (47 ((4 5.25) (48 7.0) (95 4.2) (107 7.55) (108 8.4) (126 8.75) (140 5.1)(171 6.15) (175 30.95) (176 7.6)))
  (48 ((47 7.0) (49 6.65) (95 6.15) (96 11.55) (106 9.15) (107 2.9)))
  (49 ((48 6.65) (50 7.85) (106 9.95)))
  (50 ((49 7.85) (51 10.15) (97 13.3) (98 10.5) (106 1.45)))
```

Figure 2.7 Partial List of the Austin Problem's Transit Node Connectivity (Reproduced from Shih and Mahmassani 1994)

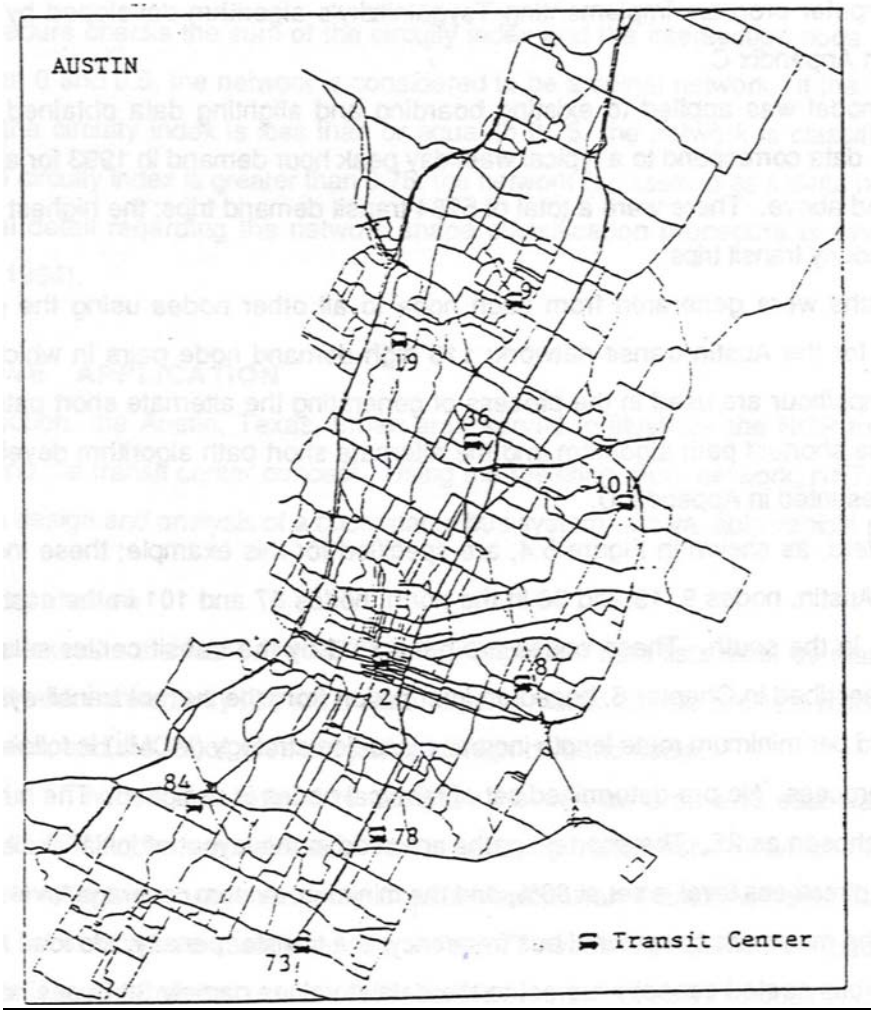


Figure 2.8 Street Network in Austin Area
(Reproduced from Shih and Mahmassani 1994)

2.4 Heuristic Approaches Using Knowledge-Based Expert Systems

Knowledge-based expert systems (KBES) are one of the AI techniques. AI is a branch of computer science partially rooted in cognitive psychology and attempts to simulate and realize human's intelligence abilities on computers. KBES approaches involve organizing and capturing the domain knowledge of one or more experts and using it to establish a knowledge base and inference rules, which may then be applied to obtain good knowledge, advice, and tools about how to solve particular problems. KBES and related AI tools are widely used in various engineering, manufacturing, businesses, and other fields, such as mechanical or structural design, diagnosis of structural failures, construction planning/scheduling, medical diagnosis, and so on (Giarratano and Riley 1993, Dym and Levitt 1991).

Heuristic methods in transit planning and route network design employ empirically derived guidelines/procedures based on criteria established from experiences and intuitions of the transit planners to search for near-optimal solutions. Apparently, a successful heuristic method depends

heavily on the knowledge and experiences of capable planners who are not only keenly aware of issues arising during transit network design processes, but also capable of formulating and/or finding the best mathematical or other solution tools to solve the sub-problems in various design stages. In practice, such experts are rare. The goal of a KBES is to systematically extract and integrate the knowledge related to a design process into a computer program, usually in an interactive graphic framework, that will provide suggestions and comparisons of whole or partial solutions, diagnosis of problems, or recommendations regarding issues encountered during a design or planning process. A KBES makes the expertise of a few experts available to many less knowledgeable and may substantially improve the productivity of the design procedure and help search for the best solutions.

While KBESs have become popular and successful in many fields, their applications to transit planning and route network design seem to be relatively limited compared with their applications in other fields. Janarthanan and Schneider (1998) investigated the applicability of KBESs to transit network design and developed an interactive KBES program to assist in the development of high-performance transit network designs.

Baaj and Mahmassani (1995) presented a framework of a hybrid AI/operations research solution approach, which combined AI search techniques with vehicle routing heuristics and transit system analysis methods. In a separate research, they developed an AI based representation and search algorithms for transit network design to reduce the search space and to obtain solutions efficiently. Other work applying expert systems or related AI tools to transit route network problems includes, among others, Baaj and Mahmassani (1990).

The following is a summary of the advantages of KBES based approaches and related AI design tools:

(1) They systematically incorporate the knowledge, expertise, experience, and various design rules and/or code into a computer "expert system" program. This allows for persons who do not have in-depth knowledge about every aspects of transit planning or route network design, or for experienced planners who could not get into details of all the issues involved in a design task due to time or resource constraints, to obtain effective solutions.

(2) They allow the users to access various powerful algorithms or procedures developed by others, especially those mathematical solution tools developed by people who have in-depth knowledge in mathematical and/or computer science fields.

(3) Since KBES programs usually operate interactively, they allow the users to quickly compare the merits of different results due to changes in one or more design parameters. This may be particularly useful to design problems with conflicting objectives.

The disadvantages of an AI based approach include:

(1) It may be expensive to develop a good KBES program since the necessary knowledge needs to be first captured from various experts in all the related fields, based on which a well designed code/program called a knowledge base must be developed. The

development of a substantial knowledge base may require a team of "knowledge engineers" to work with experts from several related fields for a possibly prolonged period of time, ranging from several weeks to several years.

(2) Sometimes different experts may provide inconsistent heuristic rules.

(3) To keep the knowledge base up-to-date, continual maintenance may be necessary, which may also be costly.

2.5 Optimization Formulations Based on Genetic Algorithms

The last three decades have witnessed an increasing interest in biologically motivated approaches like artificial neural networks and genetic algorithms for solving optimization problems (Goldberg 1989, Holland 1992, Haupt and Haupt 1998). Genetic algorithms (Gas) are search and optimization methods based on the principle of natural selection, i.e., survival of the fittest. The basic idea behind GAs is that individuals and their off-springs that best fit or adapt to the surrounding environment have the best chance to survive. In a typical GA, a population of individuals (usually potential solutions) undergoes a sequence of transformation through the application of "genetic operators" and a selection process. Those individuals that best fit the surrounding environments (usually defined by a problem's objective functions and constraints) will have a better chance to survive the selection process, and their off-springs may have a better chance to survive the transformation and selection processes of the next generation. After some number of generations, the solutions converge, and the individual with the best fitness score represents, hopefully, the optimum solution of the system. Detailed descriptions on various GA based methods and applications may be found in (Goldberg 1989, Holland 1992, and Haupt and Haupt 1998), among others.

Early work on applying GAs to transportation problems includes, for example, Vignaux and Michalewicz's work on linear and nonlinear transportation problems (Vignaux 1989, 1991). In the transit planning field, Chakroborty et al. (1995) used GAs in bus route network scheduling problems and concluded that genetic algorithm was an efficient tool for similar optimization problems in the transportation field. Pattnaik et al. (1998) applied GAs to urban bus route network design problem. The methods were then applied to a network involving seven to 20 routes. Recently, Caramia et al. (2001) presented an iterative scheme based on GA that involves neural network (NN) in function evaluation. The goal of their study is to improve the performance of existing bus networks by reducing the average travel time and management cost. Numerical results for a real world problem indicated that their GA method might be more effective in terms of computational time compared with classical assignment method.

The advantages of GA-based methods are summarized bellow.

(1) Unlike the traditional gradient matrix based mathematical solution search schemes, GA formulations do not require the calculation of the gradient matrix and any other higher order derivative matrices, or their approximation, of the objective function with respect to all the unknowns. The calculation of the gradient matrix or its approximation is a major computational burden in the traditional mathematical optimization approaches, and the

accuracy and condition of the gradient matrix usually have significant effects on the stability, rate of convergence, and accuracy of the finial results of the solution process.

(2) A GA based solution method directly carries out its search on a population of individuals (i.e., potential solutions) and the objective functions themselves, not their derivatives. Therefore, there is no need to formulate a system of governing equations that represent or simulate the relationship between various parameters and unknowns mathematically. This is particularly attractive for practical applications where it is difficult to establish mathematical formulation to accurately and effectively simulate complex situations. TN design problems are good examples of such cases.

(3) Constraints are relatively easy to incorporate into a GA solution process. They may be simply defined as a part of the environmental conditions or by imposing large penalties on individuals that violate certain constraints thus reducing their survival possibility in the selection process. This may be especially suitable to problems where constraints are complicated and cannot be properly defined.

(4) GA has been an active research field for the past several decades and results have been widely used in various application fields. There are many existing algorithms and computer codes (Haupt and Haupt 1998).

GAs also have two main disadvantages:

(1) GAs are stochastic algorithms whose search methods are based on natural evolution principle, i.e., genetic inheritance and Darwinian strife for survival process. This, as are the cases for all other algorithms based on probability theory, may not guarantee optimum solutions, although by randomly choosing a sufficiently large number of "individuals", the probability of a GA solution being close to the real optimal solution will increase.

(2) Mathematically, GAs may be categorized as weak solution search schemes that make few assumptions about the problem domains and function characteristics, such as the smoothness and consistence or compatibility of the objective functions, design parameters, unknowns, and constraints. While this makes GAs enjoy wide applicability, especially for problems of complex nature, it also causes GAs to suffer from combinatorial explosive solution costs due to huge solution search space when dealing with large-scale problems.

Although GAs have been widely employed in a variety of fields, their applications in transit network design have been relatively limited compared to other approaches in the literature. The potentials of GAs to solve transit network design problems need to be further explored.

2.6 Summary

Generally speaking, a combinatorial optimization problem with a large number of integrality variables is computationally intractable to produce an optimal solution with any existing

computer resources. This difficulty, along with the natural evolution of various approaches in transit planning design fields, results in a rich variety of heuristic solution approaches in transit network design field. Instead of finding global optima, these heuristic approaches are designed to improve existing systems or to find near optimal solutions in a reasonable amount of computational time by taking advantages of the special features inherent in the structure of individual problems. These heuristic approaches are often strongly domain- or problem-specific, and rely heavily on transit polices, design guidelines, and planners' past experiences, as well as the existing transit systems. Although achievements or good improvements have been reported, these heuristic approaches used in transit network design suffer from a lack of generality. A heuristic method that produces good solution in one transit system design case may not offer similar benefits when applied to another transit system without significant modifications. A heuristic method applicable to a wide range of transit network design problems seems unavailable currently.

The mathematical optimization approaches, while widely accepted as having the potential to become powerful and systematic design tools for TN design problems, seem still in the exploratory stage in transit planning research field. They are mainly used in small and/or idealized transit network systems, or used in the optimization of particular design parameters, such as headway, waiting time, and bus route or bus stop spacing. Thus far, there has been no report on mathematical optimization approaches that are capable of solving transit design problems of realistic sizes or handling a complete transit design cycle (from route network construction and frequency setting to scheduling and so on). The reason for this may lie in the fact that mathematical optimization approaches usually define a solution space with completeness that is computationally intractable to existing computer resources. This seems to have prevented further exploration of the mathematical optimization approaches to practical transit network design problems. The literature on transit network design indicated that it is quite common that studies often begin with derivation of the problem statements based on mathematical optimization theory including definition of objective functions and formulation of various equality and inequality constraints. However, after encountering the computational intractability of the resulting system, researchers would turn to traditional heuristic approaches.

Ironically, the research and application of various mathematical optimization approaches, especially in integer and combinatorial optimization fields, have been one of the most successful fields in mathematical science and other related disciplines such as operations research, transportation, communication, and so on. In fact, the past three decades ushered in an exciting era of research and applications of combinatorial optimization and graph theory, of which network optimization has been an important part. A variety of powerful mathematical heuristic algorithms have developed to tackle network optimization problems that are NP-hard or computationally intractable. While NP-hard and/or computationally intractable network optimization problems are usually associated with extremely large solution spaces, for most practical applications, the vast majority of the solutions in such spaces (possibly over 99% for transit network design problems) are far away from any merely reasonable/usable solutions. Such characteristics have been recognized by researchers in various research fields. Because the degree of solution difficulty for integer optimization problems is usually directly related to the number of integer variables (Hillier and Lieberman 1986), introducing or identifying more inequality constraints in an integer optimization system may lower the level of difficulty for a

solution if these constraints reduce the total number of integer variables of the system. Contemporary mathematical heuristic optimization algorithms effectively reduce the size of a search space by carefully selecting various constraints or upper and lower bounds to isolate or identify sub-spaces most likely to contain optimal solutions and small enough to allow the search for optimal or near optimal solutions in a reasonable amount of time.

In order to tackle transit network design problems of realistic sizes, mathematical optimization approaches for transit network design may require multi-disciplinary research efforts involving research fields/branches in engineering, mathematics, and computer science. Experiences from related application fields where different types of large, practical network optimization problems have been solved successfully can also be borrowed

The application of various GA approaches for transit planning seems to be another example where GA based methods for transit network design have produced limited results even though successful applications of GAs have been reported in many fields that involve combinatorial optimizations. GA based optimization approaches are especially suitable to those practical problems where the environments or conditions are too complicated to be represented or simulated by mathematical formulations, or where the resultant mathematical systems are too large to solve with completeness with any existing computer power. While computing resources required by GAs are spent mainly on generation of feasible solutions and evaluating objective functions (fitness), which can be significant for a large and complicated problem, GAs will always be able to produce feasible solutions in a reasonable timeframe. The computing resources required may also be less intense if evaluation of objective functions and generation of solutions are simple and straightforward. GAs are definitely worth further exploration for application to different transit planning problems in light of the vast successful applications in various engineering, science, and social science fields.

In general, various solution search methods may be categorized as weak or strong solution search methods according to the smoothness requirements on their variables/unknowns, objective functions, or constraints. The classical exhaustive search methods or brute-force methods that attempt to examine as many as possible solution candidates to find the optimal result are the weakest methods since these methods usually do not require any smoothness on the various functions, parameters and unknowns involved in the optimization system. Limitation of such methods is inefficiency and the extremely large amount of computer CPU time needed to obtain reasonable results, which confines these methods to problems with small and finite solution population space. The various gradient-based traditional mathematical optimization approaches are in the strong search method category due to their various smoothness and convexity requirements on objective or constraint functions, problem definition domains, variables or unknowns, and their derivatives. Generally, strong methods are much faster to produce accurate results compared than weak methods. A disadvantage of such strong methods is their limited applicability to complex practical problems due to the strict requirements on smoothness, convexity, and other mathematical conditions required for solution uniqueness, stability, and accuracy. GAs fit somewhere between the weak and the strong methods. They usually require much more time to obtain reasonable results than a gradient-based mathematical strong method, if it exists for a particular problem. At the same time, compared with classical exhaustive, or brute-force, search methods, GAs are much more effective and efficient in term of solution time

and problem solvability. As an alternative method, GAs may be suitable for various combinatorial or mixed integer optimization problems encountered in transit planning field, since for such problems of a realistic size, algorithms based on mathematical optimization techniques are likely to fail to produce any solutions due to computational intractability. In fact, some researchers consider GAs as heuristic type methods that may always give better solutions from a particular solution space or populations.

Other methods in the network optimization research area that may have the potential to become powerful tools for transit network design include fuzzy logic based methods and stochastic methods. In stead of finding an optimal solution with determination, which usually leads to computationally intractability, the goal of stochastic methods is to find an *expected* optimal solution that could be made as close to the real optimum as possible[3]. KBES approaches also have the promise to become a powerful design tool for transit planning if the advantages of different analysis methods as discussed in this report are utilized and integrated with the expertise from experts in multidisciplinary research and application fields. However, the development of such expert system design tool may be costly since it involves coordinated work of a team of experts, possibly over a prolonged period of time.

[3] Conceptually, both stochastic and fuzzy methods should be considered as mathematical optimization approaches.

3. SOLUTION METHODOLOGY

3.1 Introduction

In this chapter, solution methodologies based on mathematical optimization technology are described. The reason for developing a mathematical optimization approach for the transit route network optimization was based on the following considerations. Although always providing solutions for practical problems, the traditional heuristic approaches are in most cases *ad hoc* schemes. They depend heavily on individual planners' past experiences and physical intuitions. The design objectives and/or criteria of conventional heuristic approaches are usually a set of guidelines or requirements established with the natural evolution of existing transit systems. Theoretical justifications for such design guidelines or requirements are usually unclear. Moreover, most transit agencies may already have their own design procedures for TN design based on various heuristic schemes. The benefits to develop a new traditional heuristic approach based on existing design guidelines for TN design seem limited. GA based methods have the potential to tackle complicated TN design problems, but their applications in transit planning fields are limited and are still in the research stage. One of the most serious difficulties of existing GA approaches is the combinatorial explosive solution cost for large TN problems. It seems that GA approaches are options or choices when there are no analytical/mathematical solution means available since, in general, GA methods are slower to obtain results of the same accuracy than analytical/mathematical schemes. Development of good KBES programs is an expensive involved process, thus infeasible for this project. In conclusion, it seems that the development of mathematical optimization based approaches is the most appropriate goal for this study. The main advantages of the mathematical optimization approaches are as follows:

(1) Mathematical optimization problems are usually formulated with relatively more completeness and rigor. In other words, under proper conditions, mathematical optimization approaches may guarantee a global or local optimal result.

(2) Solutions obtained from mathematical approaches are unlikely to be biased to any existing transit systems although initial guess networks may have an effect on which local optimal result an iterative solution procedure will converge to or approach.

(3) Various design guidelines and requirements are represented by mathematical constraints, either equality or inequality constraints. These constraints could be flexibly designed to meet particular requirements of any transit network design problems.

(4) There are a wide variety of solution methods, schemes, or concepts developed or used in operations research, mathematical optimization, graph theory research, or other related fields that may be applied or adopted in the transit network design field, especially mathematical heuristic approaches that have been successfully applied to solve NP-hard problems or other computational intractable problems.

The most difficult task in solving TN design problems with mathematical optimization methods is to deal with the extremely large solution search space obtained from mathematical optimization formulation. The approaches used in this study are based on schemes that gradually

reduce the space to local spaces that are small enough to allow exhaust search with existing computer resources. After exhaustive search of one local solution space, another local space will be generated based on the data obtained from previous search results. The criteria for selecting a new search space include that (a) it is small enough to conduct an exhaustive search and (b) it has the potential to produce better results. The search process will continue until it is unable to find any better results, and, hopefully, the result is a local or global optimal solution.

The main idea behind the methods developed in this study is similar to the branch-and-bound methods in network optimization fields and the rollout algorithms described in (Bertsekas 1998). In branch-and-bound methods, some integer sets and their descendant subsets are discarded if they have no chance of containing optimal results under certain constraints. In the rollout algorithms, in stead of searching through the entire branch-and-bound subsets, the search is performed with a sequence of subsets that are most likely to contain better solutions. In this study, some solution subspaces are excluded from local search spaces if they either violate certain constraints or result in search space sizes that are computationally intractable. In general, the quality of the search results is related to the size of local solution search spaces. In other words, larger local solution search spaces are more likely to produce results that are closer to a global optimum than smaller local spaces. Since searching through larger solution spaces requires more computing resources, the performance of the methods developed in this study will be dependent of the computing resources required. The numerical results presented in the next chapter will show that for several TRN problems, results obtained from this study with an existing high-end PC are at least comparable or better than those obtained from other transit network approaches. With the rapid development of computing techniques and power, the potential to further improve results obtained from the methods developed from this study is promising. The famous Moore's law, which predicted that the computer chip's processing power would double every 18 months, has been confirmed in the last 35 years. It is expected to be true for at least two more decades. Considering such a rapid growth in CPU processor power, the rapid development of multiprocessor and massive multiprocessor computers, as well as paralleled computing techniques, solving TN optimization problems of a realistic size with mathematical optimization techniques seems not to be an intractable task in the future.

The mathematical optimization methodologies were developed based on the following assumptions:

(1) The method should be generally applicable to the design and optimization of a wide range of transit networks in practice. The limitation of this method for practical problems, especially for large transit systems, should stem mainly from the computing resources it requires, not from the basic assumptions, representations, or other problem-dependent issues such as various transit network solution search guidelines in the existing heuristic approaches.

(2) The solution method should be as objective as possible. It should not favor any particular transit network configurations of an existing transit system or certain solution selection guidelines based on individuals' experiences, which are usually problem dependent or the results of historical evolution.

(3) Solutions obtained from this method should give reasonably good results, probably a local optimal solution, in a reasonable amount of time as permitted by the current computer power affordable to most transit agencies in the U.S. The results should improve as the computer resource or power increases, and should approach the global optimum when there is no computer resource limitation.

(4) Algorithms and solution search schemes from this method should be easy to implement or realized in parallel computing frameworks, thus having the potential to take full advantages of the modern parallel computing techniques.

In the remaining sections of this chapter, the solution framework and the corresponding mathematical statements or representations of the transit network optimization methods are described. Sections 3.2 and 3.3 present the mathematical representations of the street network of a transit service coverage area and a transit route and/or a transit route network system, respectively. Section 3.4 gives the mathematical representation of the search space or spaces from which optimal transit routes may be obtained. Sections 3.5 and 3.6 describe certain topology and/or geometrical constraints on transit routes in a transit network system and certain boundary conditions on some transit routes imposed by transit planners, respectively. Constraints related to operating costs are described in Section 3.7.

3.2 Mathematical Representation of Typical Transit Service Area

A transit service area is represented as a set of street nodes and a set of street segments connected to the street nodes. This set of street nodes and segments will be referred to as the street network of the transit service area. More precisely, the definition of a street network consists of three parts:

(1) A set of street segments that are suitable for operating transit vehicles/buses;
(2) Street corners or other points on streets that may be used as transit stops; and
(3) A set of street segment lengths.

In this study, potential transit stops will be referred to alternatively as street nodes, or simply nodes, of the street network. A street segment is the line or arc in the street network that connects two adjacent street nodes. For some applications, it may be more convenient to use travel time between two adjacent nodes as the "length" of the street segment that connects these two nodes. Depending on the requirement of a particular application, a street segment length used in this study may either refer to its geometric length or the time needed to traverse the segment. A sequence of street segments that connects any two nodes and does not intersect itself is called a path or route between these two nodes. Similar to segment length, the shortest path between two nodes may either refer to the path that has the shortest geometric distance between the two nodes or the path that takes the least time to travel. Figure 3.1 illustrates a small street network of a transit service area. The thick, green solid line in Figure 3.1 indicates a path that connects nodes 3 and 28; the blue dashed line and the red dotted lines are two alternative paths. Each of the potential transit stops, or nodes, is assigned a unique integer number as an identifier. The numbering of nodes is arbitrary. The location of a node may be described in a list (or table)

that associates each node number with the name of its corresponding physical location as shown in Table 3.1.

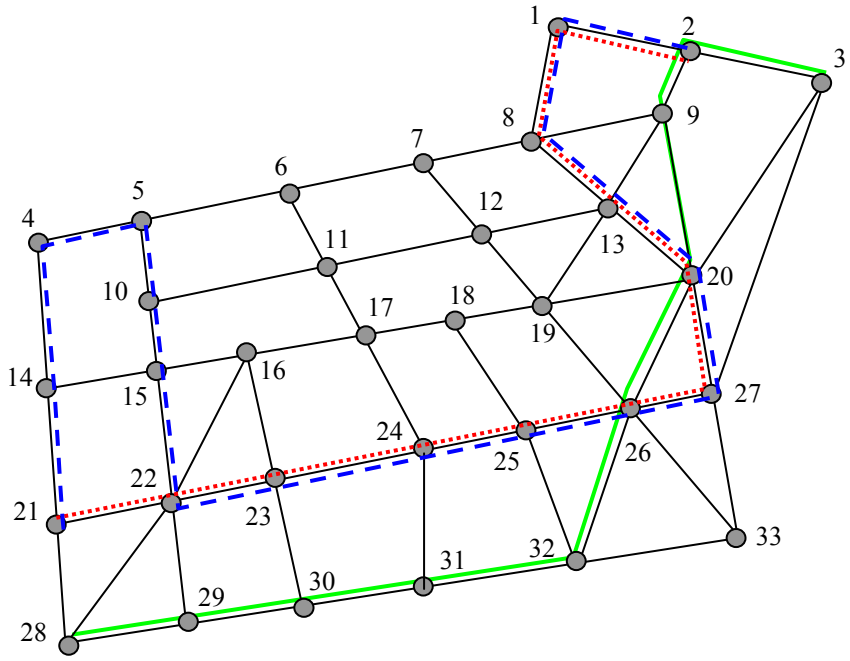


Figure 3.1 Street Network of a Sample Transit Service Area

Table 3.1 Location Description of a Sample Street Network Nodes

Street node numbers	Location names
1	City government center
2	Airport terminal A
3	South station
4	University student center
…	…

A street segment or arc[4] may be defined by its two end nodes and its length (or travel time), i.e., $\boldsymbol{a}_i(n_{i1}, n_{i2})$ where n_{i1} and n_{i2} are the starting and the ending nodes of street segment $\boldsymbol{a}_i$, respectively. In general, street arcs $\boldsymbol{a}(n_1, n_2)$ and $\boldsymbol{a}(n_2, n_1)$ may not be the same if a street segment is a one-way street or travel times are different on the same segment in the two opposite directions. In such cases, the street network is called a directed network. In this study, only undirected network is considered, in which street arcs $\boldsymbol{a}(n_1, n_2)$ and $\boldsymbol{a}(n_2, n_1)$ will be considered the same[5]. This assumption is based on the fact that for most transit routes the overall route lengths in both directions are more or less the same, and the distance between any two one-way street segments of a bus route is usually no more than a few street blocks. The assumption of undirected network may be removed in the future, which will only require minor modifications to the algorithms and programs for them to be applicable to directed networks. It is also assumed

[4] In the literature of transit planning, a street segment is also called a street arc or a street link. The terms *arc* and *link* are more often used in network theory in graph theory and operations research.
[5] The methodology described in this study may be easily extended to problems with directed networks.

that there is only one street segment between any two connected adjacent nodes, i.e., the expression $\boldsymbol{a}(n_1, n_2)$ is uniquely associated with one street segment. If there are two street segments between two adjacent nodes, as shown in Figure 3.2(a) where $\boldsymbol{a}'(n_1, n_2)$ and $\boldsymbol{a}(n_1, n_2)$ have the same end nodes, an additional node n_0 will need to be introduced to modify one of these two arcs to recover the uniqueness of the representation. Figure 3.2(b) gives an example of such a modification where arc $\boldsymbol{a}'(n_1, n_2)$ is replaced by two arcs $\boldsymbol{a}(n_1, n_0)$ and $\boldsymbol{a}(n_0, n_2)$.

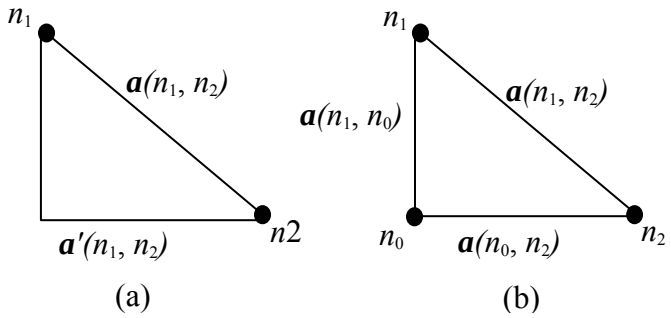


Figure 3.2 Recover the Uniqueness of Street Arc Representation

Another assumption is that the street network is connected, i.e., there are no isolated nodes or sub-networks. Thus for any two nodes, there is a sequence of street segments, or a path, in the street network that connects these two nodes.

In summary, this study deals with transit route network optimization based on a connected, undirected street network system. The following is the mathematical representation of a street network. A street network may be defined by a set of all the street segments/arcs:

$$\boldsymbol{A}^{(m)} = \{\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots \boldsymbol{a}_m\} \quad [3.1]$$

where $\boldsymbol{A}^{(m)}$ denotes a street network of m street segments or arcs, and

$$\boldsymbol{a}_i(n_{i1}, n_{i2}),\ i = 1, 2, \ldots, m \quad [3.2]$$

are the m street segments. Let $\boldsymbol{N}^{(n)}$ denote the set of street nodes,

$$\boldsymbol{N}^{(n)} = \boldsymbol{N}^{(n)}\ \{n_1, n_2, \ldots, n_n\}, \quad [3.3]$$

where n is the total number of nodes. All ending nodes of the street arcs belong to this node set:

$$n_{i1}, n_{i2} \in \boldsymbol{N}^{(n)},\ i = 1, 2, \Lambda\ , m\,. \quad [3.4]$$

A path between any two nodes is defined as a sequence of non-reoccurring nodes (i.e., the path does not intersect itself) in which any two neighboring nodes are connected by an arc,

$$\boldsymbol{p} = \boldsymbol{p}(n_1, n_2, \ldots, n_k) \quad [3.5]$$

A street network may also be represented through adjacency lists (or tables) of the street nodes. For a given node, called the master node, its associated nodal adjacency list consists of all the immediately neighboring nodes. Immediately neighboring nodes are those that can be connected to the master node by one street segment. These segments are called the adjacent segments of the master node. The set of nodal adjacency lists of a street network may be expressed mathematically as follows:

$$\boldsymbol{L}(k) = \boldsymbol{L}(k)\{n_{k,1}, n_{k,2}, \ldots, n_{k,m(k)}\}, \quad k = 1, 2, \ldots, \text{n.} \qquad [3.6a]$$

where, $\boldsymbol{L}(k)$ is the nodal adjacency list or set of the street node k, n_{kj} (j = 1, 2, … $m(k)$) is the node number of the j^{th} neighboring node, $m(k)$ is the number of neighboring nodes in the master node k's adjacency list, and n is the total number of nodes in the street network. In this report, $m(k)$ will be referred to as the *length* of the node k's nodal adjacency list, and $\overline{m}$ will be used to represent the average length of all the nodal adjacency lists in a street network. In some applications, it may be convenient to also include the length of each adjacency arc in the list. In such cases, the nodal adjacency list may be expressed as follows:

$$\boldsymbol{L}(k) = \boldsymbol{L}(k)\{n_{k1}, d_{k1}; n_{k2}, d_{k2}; \ \ldots, n_{km(k)}, d_{km(k)}\}, \quad k = 1, 2, \ldots, \text{n.} \qquad [3.6b]$$

where d_{kj} is the distance from node k to node n_{kj}, i.e., the length of the street arc $\boldsymbol{a}(k, n_{kj})$.

For illustrative purposes, two data files that define the street network shown in Figure 3.1 are given in Tables 3.2 and 3.3. Table 3.2 lists some of the street segment/arc data of the street network, while Table 3.3 presents part of the nodal adjacency list data of the street network. Street segment/arc data and street nodes' nodal adjacency list data are equivalent and either one may be used to define a street network or to derive the other.

Table 3.2 A Sample of a Street Network Data File (Street Segment/Arc File)

Street segment No.	Starting node No.	Ending node No.	Segment length
1	1	2	1125
2	2	3	976
3	4	5	657
4	5	6	589
5	7	8	521
6	8	9	894
7	10	11	982
8	11	12	872
…	…	…	…

Table 3.3 A Sample of a Street Network Data File (Street Nodal Adjacent List File)

Master Node	Adjacent Node									
	Node No.	Arc Length	Node No.	Arc Length	Node No.	Arc Length	Node No.	Arc Length	…	…
1	2	1125	8	1200						
2	1	1125	3	976	9	800				
3	2	976	20	1400	27	1500				
4	5	657	14	710						
5	4	657	6	589	10	320				
6	5	589	7	590	11	410				
…	…	…	…	…	…	…	…	…	…	…

3.3 Mathematical Representation of Transit Routes and Route Network

A transit route may be defined by the sequence of the stop numbers on the route. In Figure 3.1, the sample transit route represented by the green solid line may be expressed by the following sequence of street node numbers:

$$\boldsymbol{r}_i(n_{i1}, n_{i2}, \ldots, n_{ik(i)}) = \boldsymbol{r}(28, 29, 30, 31, 32, 26, 20, 9, 2, 3), \quad [3.7a]$$

where i is the route number, $k(i)$ is the number of stops on this route, and $n_{i1}, n_{i2}, \ldots, n_{ik(i)}$ are the global street node numbers of the stops. For this particular route, these global numbers are 28, 29, 30, 31, 32, 26, 20, 9, 2, and 3. A transit route's stops will also be referred to as the nodes of a transit route, or simply a transit/bus line's nodes.

It needs to point out that in reality, the set of stops of a transit line may not be exactly the same as the street node set consisting of all the street nodes on the transit line because transit vehicles do not necessarily stop at each of the street nodes. One difficulty in using a real transit stop sequence to define a transit route is the lack of uniqueness in route representation since more than one arc will be necessary to connect two adjacent bus stops therefore there may be more than one path between two stops connected by one street arc. To illustrate, consider the route shown as a red dotted line in Figure 3.1. This route is represented uniquely by street node sequence (21, 22, 23, 24, 25, 26, 27, 20, 13, 8, 1, 2). Assume that transit vehicles on this route only stop at street nodes 21, 23, 25, 20, and 2. It may be easy to see that the route stop sequence (21, 23, 25, 20, 2) does not represent the same route uniquely. For example, between stop nodes 25 and 20 on this route, there are two path segments (25, 26, 27, 20) and (25, 26, 20). Unless additional information is provided, by either introducing an intermediate node (e.g., node 27) or requiring a shortest path connection between stops 25 and 20, paths connecting these two stops will not be unique. In this study, for the purpose of representation uniqueness, it is assumed that the transit route stop set and the corresponding street node subset are the same. To refine bus stop locations that may involve either moving or adding bus stops may be handled in post processing, or more appropriately, in the transit route system scheduling/headway optimization process.

A necessary condition for a sequence of street nodes [3.7a] to be a transit route is the connectivity condition, i.e., any two neighboring nodes in a node sequence must correspond to a street arc, that is

$$\boldsymbol{a}(n_{ij}, n_{i(j+1)}) \in \boldsymbol{A}^{(\mathrm{m})}, j = 1, 2, \ldots, k(i) - 1, \qquad [3.7b]$$

where $\boldsymbol{A}^{(m)}$ is the arc space, or the set of the street segments as defined in [3.1]. From equation [3.5] and [3.7], it may be seen that a transit route is also a path of the street network. For a given street network, the set of all possible paths that connect any two nodes of the street network define a path space, denoted as $\boldsymbol{P}_C$, of the street network. The path space $\boldsymbol{P}_C$ will be referred to as the complete path space of the associated street network. $\boldsymbol{P}_C$ is a combinatorial vector space defined on street node set $\boldsymbol{N}^{(n)}$. The goal of the transit route network optimization is to find the optimal path subspace, or the optimal transit network, from the complete path space $\boldsymbol{P}_C$. A transit route network $\boldsymbol{T}^{(l)}$ is defined as a set of transit routes,

$$\boldsymbol{T}^{(\mathrm{l})} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_i\}, \qquad [3.8a]$$

where l is the number of transit routes in this route network system, and

$$\boldsymbol{r}_i = \{n_{i1}, n_{i2}, \ldots, n_{ik(i)}\}, i = 1, 2, \ldots, l \qquad [3.8b]$$

are the l transit routes in the transit route network system. $\boldsymbol{T}^{(l)}$ is a subspace of the path space $\boldsymbol{P}_C$, and $\boldsymbol{r}_i$ ($i = 1, 2, \ldots l$) are members of the path space $\boldsymbol{P}_C$. A transit route network $\boldsymbol{T}^{(l)}$ may also be expressed in the following matrix form:

$$\boldsymbol{T} = \begin{bmatrix} t_{11} & t_{12} & \Lambda & t_{1n} \\ t_{21} & t_{22} & \Lambda & t_{2n} \\ \Lambda & \Lambda & \Lambda & \Lambda \\ t_{l1} & t_{l2} & \Lambda & t_{ln} \end{bmatrix}, \qquad [3.9a]$$

where t_{ij} is the connectivity coefficient between transit line i and street node j and has the following property

$$t_{ij} = \begin{cases} 1, & \text{if street node } j \text{ is a stop or node of transit line } i, \quad i = 1,2,\Lambda\ ,l \\ 0, & \text{if street node } j \text{ is not a stop or node of transit line } i, \quad j = 1,2,\Lambda\ ,n \end{cases}. \qquad [3.9b]$$

In this study, transit routes are also referred to as transit paths, or simply paths.

3.4 Mathematical Representation of Search Spaces for Transit Routes and Network

In this section, a brief description will be given to illustrate the methodology developed in this study to define various solution search spaces from which desired optimal results are sought. Before proceeding, a method based on exhaustive search is described to illustrate the difficulties

in (a) defining a complete solution search space that includes all possible transit network solutions; and (b) developing a global solution search method to find the global optimal solution from the complete solution space.

Assume that one is looking for l optimal transit routes from a street network. To find the optimal solution to this problem, one way is to first list all the possible paths from the street network system. This will define a basic solution search space of the problem. The l optimal transit routes will be formed by paths from the basic search space. Therefore, the search space for finding l optimal transit routes will be a combinatorial search space consisting of all possible unique subspaces, each containing l paths from the original basic path space. In other words, each member of the combinatorial space is an l-subspace of the basic path space. The optimal route network may then be found by evaluating all the members in the combinatorial space and choosing the best one. This is the so-called exhaustive search method.

Theoretically, exhaustive search is the simplest solution search method, and it guarantees a global optimal result. However, it may not be able to produce any optimal solution for practical problems because most time it is infeasible to enumerate and evaluate all path combinations due to the extremely large search space and inadequate computing resources. To illustrate this difficulty, let's consider the search space for a transit network consisting of n nodes. One way to generate the search space is to connect each node pair from the node space $\boldsymbol{N}^{(n)}$ with the shortest path between the two nodes to obtain the shortest path/search space,

$$\boldsymbol{P}_S = \boldsymbol{P}_S\left\{\boldsymbol{p}_{12}, \boldsymbol{p}_{13}, \Lambda\ , \boldsymbol{p}_{(n-1)n}\right\} = \boldsymbol{P}_S\left\{\begin{array}{ccccccc} \boldsymbol{p}_{12}, & \boldsymbol{p}_{13}, & \boldsymbol{p}_{14}, & \Lambda\ , & \boldsymbol{p}_{1(n-1)}, & \boldsymbol{p}_{1n}, \\ & \boldsymbol{p}_{23}, & \boldsymbol{p}_{24}, & \Lambda\ , & \boldsymbol{p}_{2(n-1)}, & \boldsymbol{p}_{2n}, \\ & & \boldsymbol{p}_{34}, & \Lambda\ , & \boldsymbol{p}_{3(n-1)}, & \boldsymbol{p}_{3n}, \\ & & & \Lambda\ , & \Lambda\ , & \Lambda\ , \\ & & & & \boldsymbol{p}_{(n-2)(n-1)}, & \boldsymbol{p}_{(n-2)n}, \\ & & & & & \boldsymbol{p}_{(n-1)n} \end{array}\right\} \qquad [3.10]$$

where, $\boldsymbol{p}_{ij} = \boldsymbol{p}(i, ij_1, ij_2, \ldots, ij_k, j)$ $(i = 1, 2, n-1; j = i+1, i+2, \ldots n)$ represents the shortest path between nodes i and j, and $ij_1, ij_2, \Lambda\ , ij_k$ are the intermediate nodes between the starting node i and ending node j of the path $\boldsymbol{p}_{ij}$. It may be seen that the shortest path space $\boldsymbol{P}_S$ is a subspace of the complete path space $\boldsymbol{P}_C$ of the street network. For simplicity, assume that (a) there is only one shortest path between each node pair in the street network, i.e., subspace $\boldsymbol{P}_S$ is unique, and (b) the optimal results will only occur on shortest paths. These two assumptions are only for illustration purposes and are probably not true in most practical situations. The goal is to demonstrate that even with these two strict assumptions, the size of the resultant search space is already of an astronomical magnitude. As an example, consider the small street network shown in Figure 3.1, where $n = 33$. With assumption (a), the number of paths in search space $\boldsymbol{P}_S$ contains, according to expression [3.10], n_s shortest paths:

$$n_s = \frac{n \cdot n - n}{2} = \frac{33 \times 33 - 33}{2} = 528. \qquad [3.11]$$

In other words, the path space $\boldsymbol{P}_S$ contains 528 paths. For a transit network optimization problem involving l bus routes, the space $\boldsymbol{P}_S$ must be further expanded into a combinatory search space,

$$\boldsymbol{C}_{\boldsymbol{P}_S}^{(l)}\{\boldsymbol{P}\} \equiv \boldsymbol{C}_{\boldsymbol{P}_S}^{(l)}\{\boldsymbol{P}_1^{(l)}, \boldsymbol{P}_2^{(l)}, \Lambda\ , \boldsymbol{P}_{l_c}^{(l)}\} = \left\{\ \boldsymbol{P}^{(l)}\{\boldsymbol{p}_1, \boldsymbol{p}_2, \Lambda\ , \boldsymbol{p}_l\}\ \middle|\ \text{for all } \{\boldsymbol{p}_1, \boldsymbol{p}_2, \Lambda\ , \boldsymbol{p}_l\} \subseteq \boldsymbol{P}_S\ \right\} \qquad [3.12]$$

where $\boldsymbol{P}_i^{(l)}$ (i = 1, 2, …, l) are the l-subspaces of the space $\boldsymbol{P}_S$, i.e., each subspace $\boldsymbol{P}_i^{(l)}$ contains l shortest paths of $\boldsymbol{P}_S$, and l_c is the total number of independent l-subspaces of the space $\boldsymbol{P}_S$. Two l-subspaces $\boldsymbol{P}_i^{(l)}$ and $\boldsymbol{P}_j^{(l)}$ are independent if at least one path is in $\boldsymbol{P}_i^{(l)}$, but not in $\boldsymbol{P}_j^{(l)}$. Each l-subspace $\boldsymbol{P}_i^{(l)}$ could be a possible solution of the l-route transit network. The total number of independent l-subspaces in the combinatory space $\boldsymbol{C}_{\boldsymbol{P}_S}^{(l)}$, l_c, is given by

$$l_c = C_{n_s}^{l} = \frac{n_s(n_s-1)(n_s-2)\,\Lambda\ (n_s-l+1)}{l(l-1)(l-2)\,\Lambda\ 2\cdot 1} = \frac{n_s!}{l!\,(n_s-l)!} \qquad [3.13]$$

The above number could be large even for relative small n_s and l. For the small street network in Figure 3.1, where n_s = 528, several $l_c = C_{n_s}^{l}$ are calculated for some l numbers to illustrate this point:

$$\begin{aligned}
&l = 1, \quad l_c = C_{528}^{1} = \frac{528!}{1!\cdot(528-1)!} = 528 \\
&l = 2, \quad l_c = C_{528}^{2} = \frac{528!}{2!\cdot(528-2)!} = \frac{528\cdot 527}{2} = 139{,}128 \\
&l = 4, \quad l_c = C_{528}^{4} = \frac{528\cdot 527\cdot 526\cdot 525}{1\cdot 2\cdot 3\cdot 4} \approx 3.2\times 10^{9} \\
&l = 10, \quad l_c = C_{528}^{10} = \frac{528!}{10!\cdot(528-10)!} \approx 4.3\times 10^{20} \\
&l = 20, \quad l_c = C_{528}^{20} = \frac{528!}{20!\cdot(528-20)!} \approx 8.1\times 10^{35}
\end{aligned} \qquad [3.14]$$

It may be seen that for transit networks that have four or more routes ($l \geq 4$), $l_c = C_{n_s}^{l}$ is already in astronomical orders. If the two simplified assumptions (a) and (b) made earlier about the street network are removed, the solution search spaces will grow even larger. Assumption (a) assumes that there is only one shortest path between any node pair, which may not be true for some street networks. For example, it may be seen from the street network shown in Figure 3.3 that any path starting from node i, going only in rightward and/or upward directions, and ending at node j will be a shortest path between these two nodes. Figure 3.3 provides two such paths between nodes i and j. For this particular street network, the number of shortest paths between any two nodes may be very large. This example shows that in general, the number of shortest paths n_s in a street network's shortest path space, may be much larger than that calculated from equation [3.11], thus resulting in a much larger combinatorial search space with the search space size given by equation [3.13].

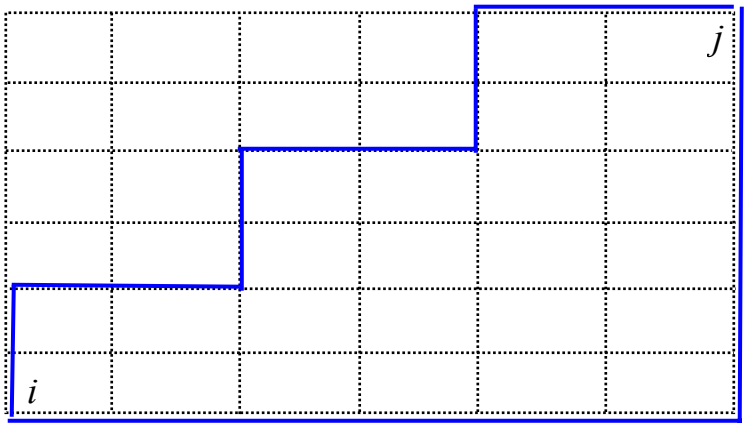


Figure 3.3 Two Shortest Paths between Nodes *i* and *j*

Assumption (b), which claims that optimal routes occur only on shortest paths, may not be true either because the optimal routes depend on both path lengths and the distribution of riders in a street network, as well as some other factors. In other words, an optimal route may not be the shortest path but one that may take longer to travel but serves more transit users. Therefore, tradeoffs between travel time and service coverage must be considered in optimizing a transit network. This means that the search space of a transit route network problem should not be limited to the space defined by shortest paths of the street network. Other alternative paths must also be included in the solution search space. This evidently will result in an even larger solution search space than the already astronomically sized space described by expressions [3.10] through [3.14]. Therefore, it may be impractical to define a complete solution search space, or to develop a solution search scheme intended to find the global optimal from a complete solution search space.

The following is a description of the solution search spaces used in this study that are locally and iteratively defined based on current route network search results. The search space size may be adjusted to stay within the limitation of available computing resources. First, denote the basic path space of a street network as $\boldsymbol{P}_B$. $\boldsymbol{P}_B$ contains all paths that are candidate transit routes. Additionally, denote $\boldsymbol{C}_B^{(l)}$ as the l-combinatorial search space generated from $\boldsymbol{P}_B$, i.e., $\boldsymbol{C}_B^{(l)}$ is made up by all the independent l-subspaces of the basic path space $\boldsymbol{P}_B$. The basic path space $\boldsymbol{P}_B$ is a subspace of the complete path space $\boldsymbol{P}_C$, or $\boldsymbol{P}_B \subseteq \boldsymbol{P}_C$ since being a transit route a path must satisfy certain constraints, such as the minimum and/or maximum route length constraints. Assume that at solution search iteration i, an intermediate transit network result $\boldsymbol{T}_i^{(l)} = \{\boldsymbol{r}_{i1}, \boldsymbol{r}_{i2}, \ldots, \boldsymbol{r}_{il}\}$ is obtained, where $\boldsymbol{r}_{ij}$ (j = 1, 2, …, l) are the l transit routes of $\boldsymbol{T}_i^{(l)}$. It may be seen that $\boldsymbol{T}_i^{(l)}$ is a member of the solution combinatorial space $\boldsymbol{C}_B^{(l)}$, and that $\boldsymbol{r}_{ij}$ (j = 1, 2, …, l) are members of the basic path space $\boldsymbol{P}_B$. Now for iteration i+1, instead of searching through the entire combinatorial space $\boldsymbol{C}_B^{(l)}$, usually of a size of astronomical order, the search space used in this study will be limited to the neighborhood spaces of each route obtained at iteration i. These neighborhood spaces will be referred to as local path search spaces or simply local path spaces.

The following is a description and definition of local path space. A local path space has two components: a master path and a set of paths that are in the neighborhood of the master path. A

local path space is derived from the local node spaces of the master path. Similar to a local path space, a local node space also has two parts: a master node and a set of nodes that are in the neighborhood of the master node. In this study, the first order local node space of a master node is defined as the set of nodes in the master node's nodal adjacency list and the master node itself. For example, node 19's local node space in the street network shown in Figure 3.1 is defined as

$$\boldsymbol{L}^N_{(1)}(19) = \boldsymbol{L}^N_{(1)}\{19|12,13,20,26,18,19\} = \{19\} \cup \boldsymbol{L}(19)\{12,13,20,26,18\}. \quad [3.15]$$

In general, the i^{th} order local node space of a master node k is defined as the set of nodes that can be connected to the master node with i or fewer street segments. The order of a local node space provides a measurement of the degree of localization. The choice of an appropriate order may be made based on the computing resources available, making this method adaptive to the computing limitations. Denote the $(i\text{-}1)^{\text{th}}$ order local node space of a master node k as $\boldsymbol{L}^N_{(i-1)}(k) = \{k_1^{(i-1)}, k_2^{(i-1)}, \Lambda\ , k_{L(k)}^{(i-1)}\}$, where $L(k)$ is the number of nodes in this local node space, and $k_j^{(i-1)}$ is the street node number of the j^{th} node in the local node space. The i^{th} order local node space may be written as

$$\boldsymbol{L}^N_{(i)}(k) = \{k_1^{(i)}, k_2^{(i)}, \Lambda\ , k_{M(k)}^{(i)}\} = \{k_1^{(i-1)}, k_2^{(i-1)}, \Lambda\ , k_{L(k)}^{(i-1)}\} \cup \boldsymbol{L}(k_1^{(i-1)}) \cup \boldsymbol{L}(k_2^{(i-1)}) \cup \Lambda\ \cup \boldsymbol{L}(k_{L(k)}^{(i-1)}) \quad [3.16]$$

where $\boldsymbol{L}(k_j^{(i-1)})$ is the nodal adjacency list of node $k_j^{(i-1)}$. A local node space is a subspace of the street node space $\boldsymbol{L}^N_{(i)}(k) \subseteq \boldsymbol{N}^{(n)}$. As the order i increases, it will approach to the original street node space $\boldsymbol{N}^{(n)}$.

With the above definition of local node spaces, a local path/route space may be derived. First from equations [3.5] and [3.9], in view of equation [3.16], for a given master path $\boldsymbol{p} = \boldsymbol{p}(n_1, n_2, \ldots, n_r)$ one can generate a sequence of local node spaces,

$$(\boldsymbol{L}^N_{(i)}(n_1), \boldsymbol{L}^N_{(i)}(n_2), \Lambda\ , \boldsymbol{L}^N_{(i)}(n_r)), \quad [3.17]$$

where $\boldsymbol{L}^N_{(i)}(n_j)\,(j = 1, 2, \Lambda\ , r)$ is the local node space of master node n_j on master path $\boldsymbol{p}$. The local path space of the master path $\boldsymbol{p}$ is generated or defined in the following steps:

1. Connecting nodes in space $\boldsymbol{L}^N_{(i)}(n_1)$ with nodes in space $\boldsymbol{L}^N_{(i)}(n_2)$ to obtain a set of paths.
2. Extending or connecting those paths obtained in Step 1 to nodes in space $\boldsymbol{L}^N_{(i)}(n_3)$ to obtain a set of paths starting from nodes in space $\boldsymbol{L}^N_{(i)}(n_1)$, passing through nodes in space $\boldsymbol{L}^N_{(i)}(n_2)$, and ending at nodes in space $\boldsymbol{L}^N_{(i)}(n_3)$.
3. Repeating Step 2 to extend the paths to nodes in space $\boldsymbol{L}^N_{(i)}(n_4)$, then to nodes in space $\boldsymbol{L}^N_{(i)}(n_5)$, and so on until the nodes in the last space $\mathbf{L}^N_{(i)}(n_r)$ are connected.

4. The resultant path space obtained from Steps 1 through 3, denoted as $\boldsymbol{L}_{(i)}^{P}(\boldsymbol{p})$, will be defined as the local path space expanded from the master path $\boldsymbol{p}$, or simply, the local path space of path $\boldsymbol{p}$.

Note that in Steps 1 through 3, any intermediate results that are not paths, such as loops, must be removed during the local path generation. To illustrate, a path $\boldsymbol{p}_1$ (the red dotted line) in path space $\boldsymbol{L}_{(1)}^{P}(\boldsymbol{r})$, which is the first order local path space of the master path $\boldsymbol{r}$ (the green solid line), is plotted in Figure 3.1. Path $\boldsymbol{p}_1 = \boldsymbol{p}_1$(21, 22, 23, 24, 25, 26, 27, 20, 13, 8, 1, 2) is generated by connecting nodes 21, 22, 23, 24, 25, 27, 13, 8, 1 and 2, which are, respectively, members of the first order local node spaces of the master path $\boldsymbol{r} = \boldsymbol{r}$(28, 29, 30, 31, 32, 26, 20, 9, 2, 3). For instance, node 21 belongs to the first order local node space of node 28, node 22 belongs to that of node 29, and so on. It may be seen that as the order i increases, a local path space of any master path will approach to or become the complete path space $\boldsymbol{P}_C$. In fact, as i increases, the master path's local node spaces will include any two nodes in street network space $\boldsymbol{N}^{(n)}$, thus the master path's local path space will include any paths between these two nodes.

In the definition of a local path space $\boldsymbol{L}_{(i)}^{P}(\boldsymbol{p})$ in Steps 1 through 4, the paths that connect nodes in local node space $\boldsymbol{L}_{(i)}^{N}(n_j)$ with nodes in local node space $\boldsymbol{L}_{(i)}^{N}(n_{j+1})$ could be any paths in the complete path space $\boldsymbol{P}_C$ of the street network. This will lead to a large path space even for the first order space, e.g., $i = 1$, in $\boldsymbol{L}_{(i)}^{P}(\boldsymbol{p})$, and the resulting path space may be hardly a "local" space. For example, the path segment that connects node 21 with 22 in path $\boldsymbol{p}_1 = \boldsymbol{p}_1$ (21, 22, 23, 24, 25, 26, 27, 20, 13, 8, 1, 2) in Figure 3.1 (the red dotted line) could be replaced by a new path segment (21, 14, 4, 5, 10, 15, 22) to obtain a new path $\boldsymbol{p}_2 = \boldsymbol{p}_2$ (21, 14, 4, 5, 10, 15, 22, 23, 24, 25, 26, 27, 20, 13, 8, 1, 2). The path $\boldsymbol{p}_2$ (the blue dashed line) is also a member of the local path space $\boldsymbol{L}_{(1)}^{P}(\boldsymbol{r})$, but can hardy be considered local to the master path $\boldsymbol{r}$. To deal with this problem, a modified local path space, denoted as $\boldsymbol{P}_{(i)}^{(k)}(\boldsymbol{p})$, of a master path $\boldsymbol{p}$ is defined. The procedure to generate $\boldsymbol{P}_{(i)}^{(k)}(\boldsymbol{p})$ from the master path $\boldsymbol{p}$ is the same as that defined by Steps 1 through 4 for $\boldsymbol{L}_{(i)}^{P}(\boldsymbol{p})$, except that the paths used to connect nodes in local node space $\boldsymbol{L}_{(i)}^{N}(n_j)$ and nodes in local node space $\boldsymbol{L}_{(i)}^{N}(n_{j+1})$ will be taken from a k-level shortest path space $\boldsymbol{P}_S^{(k)}$ that consists of the first k shortest paths between any two nodes in the street node space $\boldsymbol{N}^{(n)}$. A description of algorithms for finding a k-level shortest path space in a network may be found in Shier (1979). Explicitly, the k-level shortest path space $\boldsymbol{P}_S^{(k)}$ may be expressed as

$$\mathbf{P}_S^{(k)} = \mathbf{P}_S^{(k)} \{ \mathbf{p}_{ij}^{(r)}, i = 1, 2, \ldots, (n-1); j = i+1, i+2, \ldots, n; r = 1, 2, \ldots k \}, \qquad [3.18]$$

where $\boldsymbol{p}_{ij}^{(r)}$ represents the r^{th} shortest path between node i and node j. Denote $d_{ij}^{(r)}$ as the length of the r^{th} shortest path $\boldsymbol{p}_{ij}^{(r)}$, one has

$$d_{ij}^{(1)} \le d_{ij}^{(2)} \le d_{ij}^{(3)} \le \Lambda \le d_{ij}^{(k)},\ i = 1, 2, \ldots, (n-1); j = i+1, i+2, \ldots, n \qquad [3.19]$$

For k = 1, one has a 1-level shortest path space $\boldsymbol{P}_S^{(1)}$ as is the case in equation [3.10]. For simplicity, a 1-level shortest path space $\boldsymbol{P}_S^{(1)}$, a 1-level shortest path $\boldsymbol{p}_{ij}^{(1)}$, and distance $d_{ij}^{(1)}$ will refer to as, respectively, a shortest path space $\boldsymbol{P}_S$, a shortest path $\boldsymbol{p}_{ij}^{S}$, and a shortest distance d_{ij}^{S}. The i^{th} order local path space based on a k-level shortest path space, or simply a local path space, will be written as

$$\boldsymbol{P}_{(i)}^{(k)}(\boldsymbol{p}) = \mathbf{P}_{(i)}^{N}(\mathbf{p})\Big|_{P_S^{(k)}} = \mathbf{P}_{(i)}^{N}\left(\mathbf{L}_{(i)}^{N}(n_1), \mathbf{L}_{(i)}^{N}(n_2), \Lambda\ , \mathbf{L}_{(i)}^{N}(n_r)\right)\Big|_{P_S^{(k)}}. \qquad [3.20a]$$

It may be seen that as i and k increase, a local path space of any master path will approach to the complete path space $\boldsymbol{P}_C$. This is because that as i increases, the master path's local node spaces will include any two nodes in space $\boldsymbol{N}^{(n)}$, and as k increases, the master path's local path space will include any paths between these two nodes.

Given the above definition of local path space, the local search spaces of a transit network $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_l\}$ may be defined by the following expression,

$$\boldsymbol{T}_{(i)}^{(k)}\left(\boldsymbol{T}^{(l)}\right) = \boldsymbol{T}_{(i)}^{(k)}\left\{\boldsymbol{P}_{(i)}^{(k)}(\boldsymbol{r}_1), \boldsymbol{P}_{(i)}^{(k)}(\boldsymbol{r}_2), \Lambda\ , \boldsymbol{P}_{(i)}^{(k)}(\boldsymbol{r}_l)\right\}, \qquad [3.20b]$$

where $\boldsymbol{P}_{(i)}^{(k)}(\boldsymbol{r}_j)$ is the local path space of transit route $\boldsymbol{r}_j$, (j = 1, 2, …, l). $\boldsymbol{T}_{(i)}^{(k)}\left(\boldsymbol{T}^{(l)}\right)$ is called the i^{th} order local network search space of the master transit network $\boldsymbol{T}^{(l)}$ based on a k-level shortest path space, or simply a local network space of network $\boldsymbol{T}^{(l)}$. It may be seen from expression [3.24] that $\boldsymbol{T}_{(i)}^{(k)}\left(\boldsymbol{T}^{(l)}\right)$ consists of a sequence of local path search spaces $\boldsymbol{P}_{(i)}^{(k)}(\boldsymbol{r}_j)$ associated with each transit route $\boldsymbol{r}_j$, (j = 1, 2, …, l) in transit network $\boldsymbol{T}^{(l)}$. In a solution stage or in an iteration in a transit network optimization process, $\boldsymbol{P}_{(i)}^{(k)}(\boldsymbol{r}_j)$ will be the search space of the corresponding transit route $\boldsymbol{r}_j$ (j = 1, 2, …, l).

3.5 Approximation of Transit Route Search Spaces

To make the solutions to the transit route optimization problem achievable, certain assumptions have been made in this study. The purposes of these approximate assumptions are (a) to further reduce the size of each transit route's local path search space and (b) to limit the solution path search spaces to those that meet certain route directness constraints.

The local path space of a master path/route described in the previous section is still too large for a practical transit network optimization problem. Figure 3.4 shows a simple one-route transit network in which the blue dashed line represents the master route $\boldsymbol{r}(1, 2, \ldots, n)$. For this simple problem, the number of paths in the local path space $\boldsymbol{P}_{(1)}^{(1)}(\boldsymbol{r})$ of the master route $\boldsymbol{r}$ may be

estimated following the steps below (recall that $\boldsymbol{P}_{(1)}^{(1)}(\boldsymbol{r})$ is the first order space based on 1-level shortest path space):

(1) Connecting nodes 1, 1_a, or 1_b with nodes 2, 2_a, or 2_b, respectively, using shortest path segments, one obtains $3^2 = 9$ path segments: $\boldsymbol{p}(1_a, 2_a)$, $\boldsymbol{p}(1_a, 2_a, 2)$, $\boldsymbol{p}(1_a, 2_a, 2, 2_b)$, $\boldsymbol{p}(1, 2)$, $\boldsymbol{p}(1, 2, 2_a)$, $\boldsymbol{p}(1, 2, 2_b)$, $\boldsymbol{p}(1_b, 2_b)$, $\boldsymbol{p}(1_b, 2_b, 2)$, and $\boldsymbol{p}(1_b, 2_b, 2, 2_a)$. In generating these path segments, if there is more than one shortest path between the two nodes, one may randomly choose one shortest path.

(2) Connecting the path segments obtained from Step 1 to nodes 3, 3_a, or 3_b, respectively, one arrives at $3^3 = 27$ path segments.

(3) Continue the above procedure to the last 3 nodes, n, n_a, and n_b, one obtains 3^n paths that all belong to the local path space $\boldsymbol{P}_{(1)}^{(1)}(\boldsymbol{r})$ of the master route $\boldsymbol{r}$. The green solid line and the red dotted line in Figure 3.4 are two of the 3^n paths in path space $\boldsymbol{P}_{(1)}^{(1)}(\boldsymbol{r})$.

For a practical transit route network, a typical transit route may have node/stop number ranging from 20 to 100, thus leading to a typical local path search space sizes from $3^n = 3^{20} = 3.5 \times 10^9$ to $3^n = 3^{50} = 5.2 \times 10^{47}$. This is a local path search space of only one transit route. Even a local solution search from such a large path space may already have exceeded the limits of current computing resources for most transit agencies.

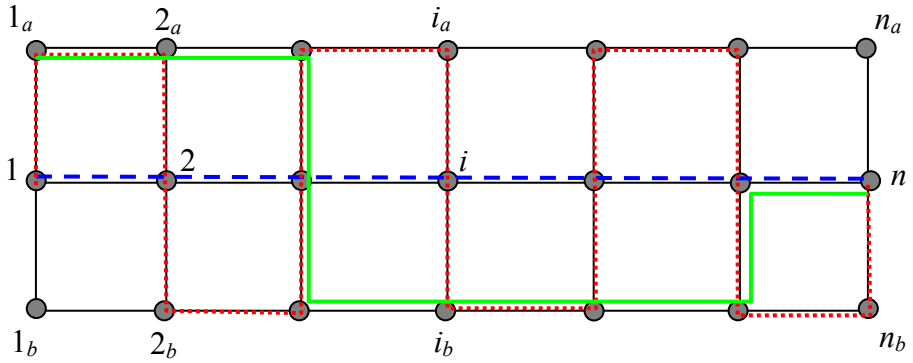


Figure 3.4 Simple One-Route Transit Network

The large size of a local path space is mainly due to the fact that the path space includes many paths that are impractical in terms of route directness. Route directness reflects the deviation of a route that connects two nodes from the shortest path between these two nodes. For example, the transit route indicated by the red dotted line shown in Figure 3.4 is highly unlikely in practice because it is much longer than the shortest path between node 1 and n. One measurement of the directness of a transit route may be

$$d(\boldsymbol{r}) = \frac{\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\frac{d_{ij}(\boldsymbol{r})}{d_{ij}^{S}}}{\frac{n^2-n}{2}}, \qquad [3.21]$$

where $d_{ij}(\boldsymbol{r})$ is the route distance (or travel time) between node i and node j on transit route $\boldsymbol{r}(1, 2, \ldots, n)$, and d_{ij}^{S} is the length (or travel time) of a shortest path between node i and node j of the street network. In general, $d_{ij}(\boldsymbol{r})$ represents the distance (or time) a person travels (or spends) if he/she takes a bus, while d_{ij}^{S} represents the distance (or time) a person travels (or spends) if he/she drives a car or takes a taxi. The route directness value $d(\boldsymbol{r})$ represents the average ratio of the distance/time transit users spend on travel by transit over that if they drive or take taxi between their origin and designation points. Ideally, the optimal directness value of a transit route is $d(\boldsymbol{r}) = 1$. In such a case, all passengers travel along the shortest paths between their origin and designation points. In reality, however, route directness $d(\boldsymbol{r})$ is always greater than one since it will not be possible to provide point-to-point direct services to all passengers. On the other hand, route directness ratio $d(\boldsymbol{r})$ should not be too large as significant route deviation from shortest paths will likely result in loss of ridership due to unreasonably long travel time. Therefore, in this study, instead of searching for solutions in the local space that is derived from the local node spaces of all the nodes on the corresponding master path (ref. equation [3.17]), the local path space to be used is derived from the local node spaces of a few key nodes on the master path that defines a route through a flexible number of piecewise shortest path segments connecting these key nodes. The number of key nodes on a master path may be flexible. Routes defined through small numbers of key nodes will generally result in better route directness and small local path search space, thus leading to fast convergence in solution processes. However, a route with a small number of key nodes may fail to reach or connect nodes with potential large trip demands due to the inflexibility in the route shape. Routes that have larger numbers of key nodes are relatively more flexible to reach more neighboring nodes thus may cover more trips. However, this will also result in larger local path search space thus may require more computing resources. Moreover, route directness may suffer because of the flexibility in the route shape.

The idea of using paths made up by piecewise shortest path segments to represent/approximate a real transit route is a familiar one in continuous field, such as in finite element methods and finite difference methods, where a linear, quadratic, cubic, or even higher polynomial shape functions are used to approximate more complex functions.

The local spaces used in this study to represent a transit route and its neighboring local path space are defined as follows. Figure 3.5(a) shows a transit route $\boldsymbol{r}$ with starting node n_1 and ending node n_2. Assume that this route has been obtained during an iterative solution process, and that better route configurations are being sought based on this intermediate result. Figure 3.5(b) illustrates a two-key-node representation of the master path $\boldsymbol{r}$ where the first and the second key nodes are the starting and ending nodes, n_1 and n_2, of the master path $\boldsymbol{r}$. Nodes n_{11}, n_{12}, n_{13}, and n_{14}, shown in Figure 3.5(b), are adjacent nodes of key node n_1, whereas nodes n_{21}, n_{22}, n_{23}, and n_{24} are adjacent nodes of key node n_2. The local path space of the two-key-node (or simply two-node) representation of the master path $\boldsymbol{r}$ is generated by connecting node n_1 and its adjacent nodes n_{11}, n_{12}, n_{13}, and n_{14}, respectively, with node n_2 and its adjacent nodes n_{21}, n_{22}, n_{23}, and n_{24}. The paths that connect the above nodes are from a shortest path space $\boldsymbol{P}_S^{(1)}$. The local path space of the two-node representation of the master path $\boldsymbol{r}$ may be written as $\boldsymbol{P}_{(1)}^{(1)}\left(\boldsymbol{r}^{(2)}\right)$, where $\boldsymbol{r}^{(2)}$ is the two-node representation of the master path $\boldsymbol{r}$. In general, a local path space of an s-node representation of the master path $\boldsymbol{r}$ will be denoted as

$$\boldsymbol{P}_{(i)}^{(k)}\left(\boldsymbol{r}^{(s)}\right)=\boldsymbol{P}_{(i)}^{(k)}(\boldsymbol{r})\Big|_{\boldsymbol{r}\text{ represented by }\boldsymbol{r}^{(2)}}, \qquad [3.22]$$

where i is the order of the local path space, the superscript k refers to the fact that the shortest paths used to connect key nodes and their nodal adjacency list nodes are from a k-level shortest path space $\boldsymbol{P}_S^{(k)}$, s is the number of key nodes used to represent the master path $\boldsymbol{r}$, and $\boldsymbol{r}^{(s)}$ is the s -node representation of the master path $\boldsymbol{r}$. It may be seen that as the number of key nodes, or s, increases, the local path space $\boldsymbol{P}_{(i)}^{(k)}\left(\boldsymbol{r}^{(s)}\right)$ will approach to the local path space $\boldsymbol{P}_{(i)}^{(k)}(\boldsymbol{r})$, which is the local path space of the master path $\boldsymbol{r}$. For a given transit network $\boldsymbol{T}^{(l)}=\{\boldsymbol{r}_1,\boldsymbol{r}_2,\Lambda\ ,\boldsymbol{r}_l\}$, a local search space based on various numbers of key-node representation is defined as

$$\boldsymbol{T}_{(i)}^{(k)}\left(\boldsymbol{T}^{(l)}\right)=\boldsymbol{T}_{(i)}^{(k)}\left\{\ \boldsymbol{P}_{(i)}^{(k)}\left(\boldsymbol{r}_1^{(s_1)}\right),\ \boldsymbol{P}_{(i)}^{(k)}\left(\boldsymbol{r}_2^{(s_2)}\right),\Lambda\ ,\ \boldsymbol{P}_{(i)}^{(k)}\left(\boldsymbol{r}_l^{(s_l)}\right)\ \right\}, \qquad [3.23]$$

where $\boldsymbol{P}_{(i)}^{(k)}\left(\boldsymbol{r}_j^{(s_j)}\right)$ is a local path space of s_j-node representation for master path $\boldsymbol{r}_j$ (j = 1, 2, ... l). Figure 3.5(c) shows a three-node representation of the master path $\boldsymbol{r}$ where the third key node n_3 and its adjacent nodes n_{31}, n_{32}, n_{33}, and n_{34} are located in an area between the starting and ending nodes of the master path $\boldsymbol{r}$. Figure 3.5(d) shows a four-node and a five-node path representation of the master path $\boldsymbol{r}$. The four- and five-node path representations have been used in the optimization in several of the optimization problems tested in this study.

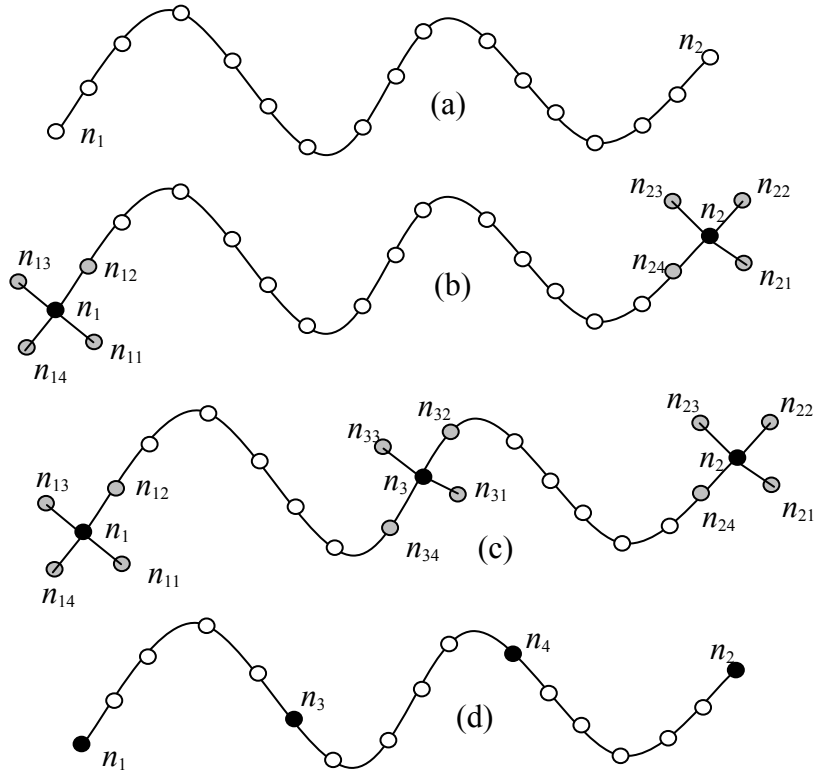
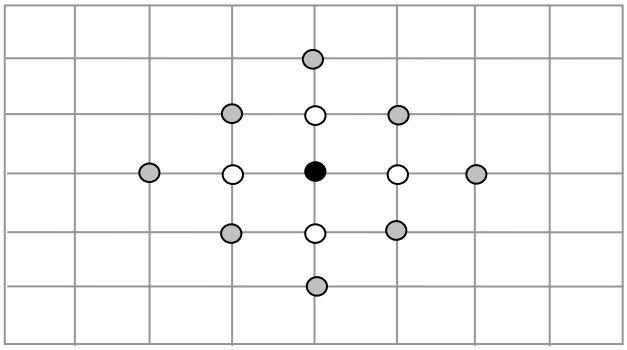
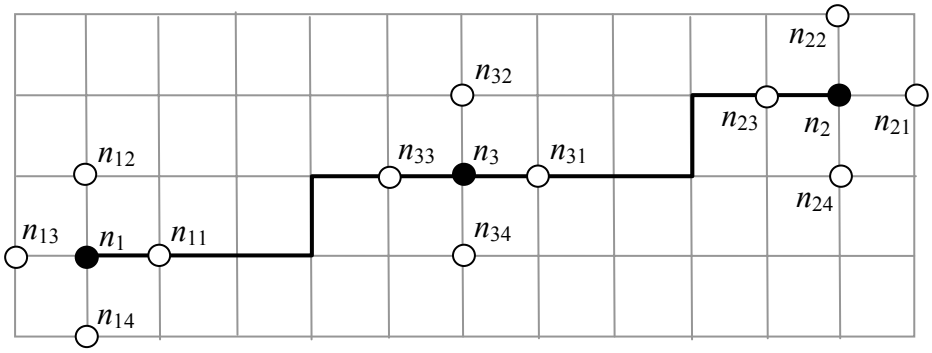


Figure 3.5 Representation of Transit Route with Two to Five Key Nodes

Figure 3.6(a) illustrates a street network (gray lines), a master node (black circle), its first order node space (black and white circles), and its second order node space (black, white, and gray circles). A three key-node representation of a master path (solid line) and the first order local node spaces of key nodes n_1, n_2, and n_3 (black circles and associated neighboring white circles) are illustrated in Figure 3.6(b).



(a) A path, its three key-nodes, and three first order local node spaces



(b) A master node and its neighboring nodes

Figure 3.6 Street Network, a Local Node Space, and a Three Key-Node Local Path Space

As have been indicated before representation of a master path with larger key-node number results in larger search space size, and the associated results (computing resources permitting) are usually better than those obtained from a representation of a smaller number of key-nodes. However, without proper directness constraints, larger number of key-node path representations may lead to paths with unreasonable shapes. Figure 3.7 shows some basic route shapes allowed by different numbers of key-node path representations. For this example, it has assumed that the shortest path between each node pair is a straight line segment. Figure 3.7(a) shows a typical path in a local path space of two-node path representation where the two end nodes are connected with one shortest path segment. Paths obtained from space of two-node path representation have the best route directness. All transit riders on such paths travel on the shortest path segments between their origin and designation points. The disadvantage of such a path is that it fails to cover any nodes outside the shortest path segment thus may result in lower trip coverage. Figure 3.7(b) shows a path configuration from a space of three-node path representation. The extra node in this path representation adds some flexibility to the paths in this path space to cover nodes on one side of a shortest path segment that connects the two end nodes. A path in space of two-node path representation will also be a path in a space of three-

node path representation. In such a case, the third node is located in the shortest path segment between the two end nodes. Figure 3.7(c) shows a case where two nodes are located on each side of the shortest path segment between the two end nodes. For this case, a path from a space of four-node path representation, shown in Figure 3.7(c), may need to cover these two nodes. Figures 3.7(d), 3.7(e), and 3.7(f) further show, respectively, paths in path spaces of five-node, six-node and seven-node path representations. In general, as the number of key-nodes increases, the associated path space will include paths that are more flexible to cover various node distributions although the route directness of a path will decrease as the path's flexibility increases. Additionally, the path space of higher number of key-node path representation will include paths in spaces of lower numbers of key-node path representation. For example, the path shape shown in Figure 3.7(b) is a special case of the path shape shown in Figure 3.7(c) when both nodes n_3 and n_4 are on the same side of the shortest path segment between nodes n_1 and n_2. Various basic path shapes in a path space may be called mode-shapes of the path space. From the above discussion, the path shape in Figure 3.7(a) is the only mode-shape in path space of two-node path representation, the path shapes in Figure 3.7(a) and 3.7(b) are the two mode-shapes in path space of three-node path representation, and Figure 3.7(a), 3.7(b) and 3.7(c) are the three mode-shapes in path space of three-node path representation, and so on.

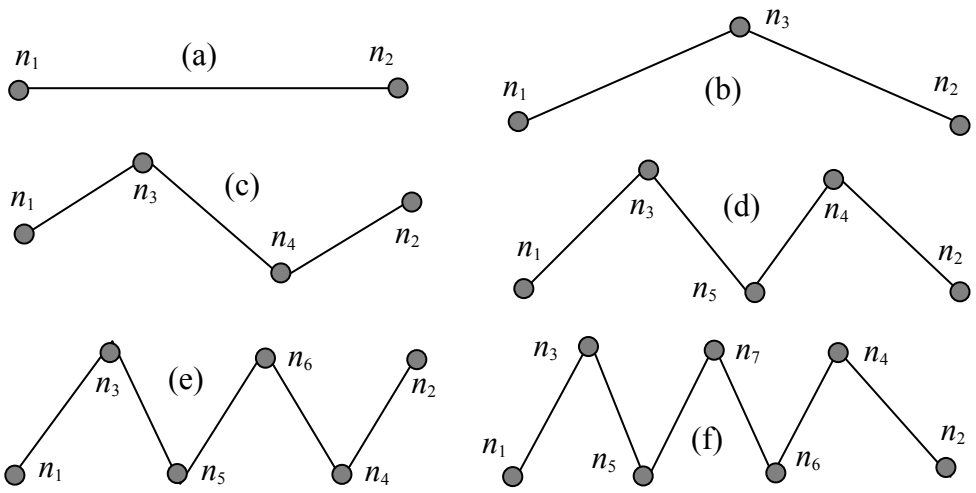


Figure 3.7 Representations of Transit Route with Two through Seven Key Nodes

It may be seen from the above discussion that from a practical points of view, the number of key-nodes used in the representation of local path spaces in a mathematical optimization formulation should be limited since a representation based on a large number of key nodes will include paths that are too flexible to be transit routes due to unreasonable route directness ratios. The selection of an appropriate key-node number for a particular TRN design problem depends mainly on the computing resources available. By selecting a particular key-node number, all paths that could not be represented for the given number of key nodes will be automatically filtered out from the associated local path spaces, resulting in much smaller search spaces. In practice, the key-node number should be such that it is large enough to allow inclusion of as many paths as possible with reasonable route directness, but small enough to result in a tractable solution process.

3.6 Constraints for Transit Routes

This section presents various constraints applied in this study. Identifying and incorporating as many appropriate integer constraints as possible into an integer optimization solution process may be one of the most important steps in solving a large scale TRN design problem since it has a direct impact on the solvability of the whole solution process. For example, in the optimization process of Miami area's transit route network presented in the next chapter, the basic path search space is reduced by more than 30% after applying the constraint on the maximum transit route length. Other constraints, such as the minimum and maximum numbers of bus stops per route, further reduce the search space significantly. Using a path representation based on a specific key-node number described in previous section is another example to separate solution subspaces that contain unfavorable solutions. Applying various constraints to separate and discard solution subspaces that are unlikely to contain valid solutions is a common technique in various branch-and-bound methods in network optimization fields. Such techniques will be employed repeatedly through various solution stages in this study.

There are mainly four types of constraints in transit route network optimization analysis. The first constraint type includes various location constraints imposed on certain transit routes by transit planners. Such constraints must be satisfied during the optimization process, and will be referred to as *location constraints* of a transit route network system. The second type of constraints includes various operational and/or feasibility constraints that are usually based on heuristic guidelines or past experiences. Such constraints are usually given as upper and/or lower bounds that are used to define the ranges of certain transit network parameters. These constraints will be referred to as *heuristic constraints*. Constraints of the first and the second types are simple constraints because incorporating such constraints into a solution framework for TRN optimization is straightforward and may often be used to reduce the solution search spaces significantly through branch-and-bound schemes. The third constraint type includes various composite or functional constraints that rely on other design variables of the network system, and usually require considerable computational efforts to evaluate. The route and network directness constraints are examples of such constraints, where the directness of a route or a route network is a function of route geometry or a route network structure. The fourth type of constraints is those implied in the model formulation or approximation, which may result in the exclusion of certain solution subspaces. These constraints are described in the following subsections.

3.6.1 Location Constraints

The following is a list of location constraints that have been implemented in the computer program developed in this study.

(1) Fixed route constraints

A transit route and all its stops/nodes are specified and may not be changed during the optimization process. Examples of such routes include fixed guideway transit lines, busways, or any routes specified by transit planners to meet certain planning goals.

(2) Prescribed route starting and ending area constraints (two-area constraints)

The general areas of starting and ending points of a transit route are defined by two sets of nodes. A route with such constraints must start from a node inside the starting area, and end at a node inside the ending area. A route starting from a given node, and/or ending at a given node is a special case of such constraint, where the starting area or ending area has only one node. When there is only one node in a specified area, the constraint in the corresponding end is called a fixed boundary constraint; otherwise, it is called a flexible boundary constraint. This constraint is applied when transit planners wish to specify transit routes between two major activity centers. Examples of such routes are those that serve airports, downtown, government centers, railway terminals, universities, shopping malls, etc.

(3) Prescribed route starting, ending, and an in-between area constraints (three-area constraint)

In addition to the starting and ending areas of a transit route, an intermediate area may also be specified. A route with such constraint must start from a node inside the starting area, and pass a node inside the intermediate area, and end at a node inside the ending area. Depending on the number of nodes in a specified area, this constraint may be also called fixed boundary constraint if there is only one node in the area, or flexible boundary constraint there are more than one node in the area. This constraint allows transit planners to specify transit routes between three major civic activity areas.

The two- and three-area constraints described above may also be used in situations where transit planners specify several transit routes intersecting in certain activity areas. Examples of such areas are transit transfer centers. Connecting several feeder bus lines with a major transit line such as a rail rapid transit line will be a special application of these constraints, where the specified area will be defined by some or all of the nodes on the transit line, and routes will be all the feeder bus lines.

3.6.2 Heuristic Constraints

These constraints are used mainly for reducing the sizes of solution search spaces. Such constraints are usually based on either heuristic guidelines, past experiences, or common practices accepted by transit planning communities. The following are the constraints used in this study.

(1) Constraints on the maximum length (or service time) of a transit route.

This constraint defines the maximum length (or service time) for each transit route in a transit route network. The maximum length (or service time) constraint for a transit route is usually based on operational considerations. For example, it is known that it is more difficult to maintain transit service schedules for longer bus line/route (or service hours). Moreover, long operating hours may cause bus driver fatigue thus may result in safety hazards.

Denote $l_r(\boldsymbol{r}_i)$ as the length of transit route r_i, and $l_t(\boldsymbol{T}^{(l)})$ as the total length of transit network $\boldsymbol{T}^{(l)}$. The maximum length constraints for individual transit routes may be expressed as follows:

$$l_r(\boldsymbol{r}_i) \le l_{\max}^R, \; i = 1, 2, \ldots l \qquad [3.24]$$

and for total route length in a transit route network system

$$l_t\left(\boldsymbol{T}^{(l)}\right) = \sum_{i=1}^{i=l} l_r\left(\boldsymbol{r}_i\right) \le l_{\max}^T, \qquad [3.25]$$

where $l_{\max}^R$ and $l_{\max}^T$ are, respectively, two user input parameters to define the maximum route length for each transit route and the maximum total route length of the transit network system.

(2) Constraint on the minimum length (or service time) of a transit route

This constraint sets the minimum length (or service time) for each transit route in a transit route network. This constraint is to prevent short transit lines from being generated because it is known that very short transit lines are usually not cost effective (except certain special purpose shuttle bus lines).

$$l_r(\boldsymbol{r}_i) \ge l_{\min}^R, \quad i = 1, 2, \ldots, l \qquad [3.26]$$

where $l_{\max}^R$ is a user input parameter to define the minimum route length for each transit route.

(3) Constraint on the maximum number of stops/nodes on a transit/bus line

This constraint is to limit solution search spaces to include only paths that have nodes fewer than a user input number, $n_{\max}$. The number $n_{\max}$ may be determined based on past experience or studies or surveys of the existing transit system. A large number of $n_{\max}$ may result in a large solution search space, thus solution difficulty for large transit route network system due to the limitation of computing resources. On the other hand, a small number of $n_{\max}$ may exclude some legitimate solution candidates from the solution search space, resulting in failure in obtaining true optimal solutions. In general, $n_{\max} = 40$ for small or mid transit service systems, and $n_{\max} = 80$ for larger transit service systems may be reasonable values. It needs to be emphasized here that the constraint on the maximum number of nodes is not essential for transit route network optimization problems. It is merely for the purpose of reducing the size of a solution search space thus a more effective solution process. Without this constraint, the maximum number of nodes in a transit route will be determined by the maximum route length (or operation time) constraint.

(4) Constraint on minimum number of stops/nodes in a transit/bus line

Similar to the constraint on the maximum number of nodes, the constraint on the minimum number of nodes is also intended to reduce the size of solution search space, and may also be based on past experience or heuristic guidelines. Denote the minimum number of nodes allowed in a transit route as $n_{\min}$, this constraint may be expressed as $n_{\min} \geq 2$ and $n_{\min} \leq n_{\max}$.

3.6.3 Composite or Functional Constraints

In this study, three transit network service directness constraints are introduced to control and/or evaluate the quality of route network configurations. One is the individual transit route based service directness, called route directness $d(\boldsymbol{r})$, that characterizes the route directness of an individual transit route $\boldsymbol{r}$. A second one, called transit network directness $d(\boldsymbol{T}^{(l)})$, is the transit route network directness that reflects the service directness of the entire transit route network system $\boldsymbol{T}^{(l)}$. Both $d(\boldsymbol{r})$ and $d(\boldsymbol{T}^{(l)})$ are important parameters that reflect the service quality of a transit route network system. A third constraint, referred to as route out-of-direction tolerance constraint $d_o(\boldsymbol{r})$, is based on the concept of Out-of-Direction (OOD) impact index suggested by Welch et al. (1991). The OOD impact index reflects tolerance of riders on a bus route to deviations from main travel path of a fixed route to increase the accessibility or ridership of the route. Welch et al. have observed that as the degree of deviation increases, riders, especially travel time sensitive ones, may choose not to use transit services. There are many ways to define route directness $d(\boldsymbol{r})$, transit network directness $d(\boldsymbol{T}^{(l)})$, and the OOD impact function $d_o(\boldsymbol{r})$. The following is the definition of $d(\boldsymbol{r})$, $d(\boldsymbol{T}^{(l)})$, and $d_o(\boldsymbol{r})$ used in this study.

(1) Transit route directness constraints. There are two ways to define the service directness of a transit route. One is based on the geometry of a transit route that reflects the route's geometrical characteristics. This route directness, denoted as $d^G(\boldsymbol{r})$, will be referred to as the service directness based on route geometry. In general, a transit route with good geometry based directness means that averagely stop/node pairs of the route are connected either with shortest paths of the street network or by paths close to shortest paths between those stop pairs. The other route directness, denoted as $d^R(\boldsymbol{r})$, takes into account of the ridership distribution along the transit route, and will be referred to as ridership based route directness. In general, a transit route with good ridership based directness means that averagely transit riders of the route travel between their origin and designation points along either shortest paths in the street network or paths close to shortest paths between those points. The following is the mathematic definition of the route directness. The route directness of a transit route $\boldsymbol{r}$ based on route geometry is defined as

$$d^G(\boldsymbol{r}) = \frac{\sum_{i=1}^{r-1}\sum_{j=i+1}^{r}\frac{d_{ij}^{(r)}}{d_{ij}^{(S)}}}{\frac{r^2-r}{2}} = \sum_{i=1}^{r-1}\sum_{j=i+1}^{r}\left(\frac{d_{ij}^{(r)}}{d_{ij}^{(S)}}\, w_{ij}^{G}\right), \qquad [3.27a]$$

where

$$w_{ij}^{G} = \frac{1}{\frac{r^2 - r}{2}} \qquad [3.27b]$$

The route directness of $\boldsymbol{r}$ based on route geometry and ridership is

$$d^{R}(\boldsymbol{r}) = \frac{\sum_{i=1}^{r-1}\sum_{j=i+1}^{r}\left[\frac{d_{ij}^{(r)}}{d_{ij}^{(S)}}\left(o_{ij}+o_{ji}\right)\right]}{\sum_{i=1}^{r-1}\sum_{j=i+1}^{r}\left(o_{ij}+o_{ji}\right)} = \sum_{i=1}^{r-1}\sum_{j=i+1}^{r}\left(\frac{d_{ij}^{(r)}}{d_{ij}^{(S)}}\, w_{ij}^{R}\right), \qquad [3.27c]$$

where

$$w_{ij}^{R} = \frac{o_{ij}+o_{ji}}{\sum_{i=1}^{r-1}\sum_{j=i+1}^{r}\left(o_{ij}+o_{ji}\right)} \qquad [3.27d]$$

In expressions [3.27a, b, c, d] r is the number of nodes on the transit route $\boldsymbol{r} = \boldsymbol{r}(n_1, n_2, \ldots, n_r)$, and $d_{ij}^{(r)}$ is the shortest distance (or travel time) between nodes n_i and n_j measured along the transit route $\boldsymbol{r}$. $d_{ij}^{(S)}$ in expressions [3.27a,c] represents the shortest network distance (or time) between the two corresponding nodes, nodes n_i and n_j. In general, one has the relationship $d_{ij}^{(S)} \le d_{ij}^{(r)}$. Variables o_{ij} and o_{ji} in expressions [3.27c] and [3.27d] are coefficients of the origin and designation matrix, or simply, the OD matrix, of the street network,

$$\boldsymbol{O} = \begin{bmatrix} 0 & o_{12} & o_{13} & o_{14} & \Lambda & o_{1(n-1)} & o_{1n} \\ o_{21} & 0 & o_{23} & o_{24} & \Lambda & o_{2(n-1)} & o_{2n} \\ o_{31} & o_{32} & 0 & o_{34} & \Lambda & o_{3(n-1)} & o_{3n} \\ \Lambda & \Lambda & \Lambda & \Lambda & \Lambda & \Lambda & \Lambda \\ \Lambda & \Lambda & \Lambda & \Lambda & \Lambda & \Lambda & \Lambda \\ o_{(n-1)1} & o_{(n-1)2} & o_{(n-1)3} & o_{(n-1)4} & \Lambda & 0 & o_{(n-1)n} \\ o_{n1} & o_{n2} & o_{n3} & o_{n4} & \Lambda & o_{n(n-1)} & 0 \end{bmatrix} \qquad [3.28]$$

where n is the total number of nodes in the street network system. The physical meaning of the OD coefficient o_{ij} is the number of trips originated from node i, and destined for node j. This study deals with fixed transit demand problems, i.e., $\boldsymbol{O}$ is assumed to be a constant matrix, and does not change with transit supply. In practices, fixed transit demand analysis may apply to problems with average trip distribution over certain time durations (e.g. Mandl's problem with daily OD trip distribution), or apply to particular

time periods in a day such as transit peak hours, off-peak hours etc. It should be recognized that in reality, transit demand may depend on transit supply, thus ideally, a more accurate TRN optimization should be carried out in an iterative manner in a cycle of demand estimation and route network design. w_{ij}^G in expressions [3.27a, b] and w_{ij}^R in expressions [3.27c, d] are weighting coefficients, and w_{ij}^G is based on the number of node pairs in the corresponding route or network system, while w_{ij}^R is based on the OD trip distribution over the street network.

It may be seen from expression [3.27a] that physically, the geometry based route directness $d^G(\boldsymbol{r})$ reflects the average ratio of the two traveling distances between each node pair on route ***r***, one is measured along route ***r***, and one is measured along the shortest path of the street network. The value of geometry based route directness $d^G(\boldsymbol{r})$ may represent, in certain sense, an aspect of operating cost of the transit route ***r***. A value of $d^R(\boldsymbol{r}) = 1$ indicates that, averagely, transit vehicles in route ***r*** travel along the shortest paths between route stops, whereas a value of $d^R(\boldsymbol{r}) = 2$ indicates that, averagely, the distances that transit vehicles travel between transit stops are twice of that along the corresponding shortest paths in the street network.

From expression [3.27c], it may be seen that in general, the transit ridership/demand based directness parameters $d^R(\boldsymbol{r})$ reflects an aspect of user cost of the transit system since it represents the average ratio of the distance or time a person spends when traveling between his/her origin and designation points by bus along transit route ***r,*** to the distance/time used if he/she travels along the shortest path of the street network system. A value of $d^R(\boldsymbol{r}) = 1$ indicates that, averagely, passengers in transit route ***r*** travel along the shortest paths between their originating and designating points, whereas a value of $d^R(\boldsymbol{r}) = 2$ indicates that, averagely, passengers in transit route ***r*** use twice of the time of that if they travel along the shortest paths between their originating and designating points.

With the above discussion on transit route directness, the route directness constraints used in this study may be expressed as follow,

$$d^G(\mathbf{r}_i) \le d_r^G, \quad or \quad d^R(\mathbf{r}_i) \le d_r^R, \quad (i = 1,2,\Lambda\ ,l), \tag{3.29}$$

where d_r^G and d_r^R are users defined maximum geometry based and ridership based route directness values or tolerances. The two directness d_r^G and d_r^R may be determined by transit planners based on past experiences, or by survey on existing and potential transit riders to find the maximum values average transit riders may accept. In general, smaller directness values may result in higher transit operating cost. However, larger directness values may not only turn away potential transit riders, but also force existing transit riders to look for other alternatives whenever possible, thus may eventually lead to high operating cost due to lower ridership.

(2) Transit network directness constraints. The transit route network directness has a similar physical meaning as that of the route directness described in the previous section

except that the directness measurement is based on the geometry or ridership characteristics of the entire route network in stead of on individual transit route. Naturally, the computational cost to evaluate the transit route network directness is much more expensive than that for individual route directness. The following is the mathematic definition of the route network directness used in this study. The network directness of a transit route network $\boldsymbol{T}^{(l)}$ based on geometric characteristics is defined as

$$d^{G}\left(\boldsymbol{T}^{(l)}\right)=\frac{\sum_{i=1}^{t-1}\sum_{j=i+1}^{t}\frac{d_{ij}^{(T)}}{d_{ij}^{(S)}}}{\frac{t^{2}-t}{2}}=\sum_{i=1}^{t-1}\sum_{j=i+1}^{t}\left(\frac{d_{ij}^{(T)}}{d_{ij}^{(S)}}\,w_{ij}^{G}\right), \qquad [3.30a]$$

where

$$w_{ij}^{G}=\frac{1}{\frac{t^{2}-t}{2}} \qquad [3.30b]$$

The transit route network directness based on both network geometry and ridership is,

$$d^{R}\left(\boldsymbol{T}^{(l)}\right)=\frac{\sum_{i=1}^{t-1}\sum_{j=i+1}^{t}\left[\frac{d_{ij}^{(T)}}{d_{ij}^{(S)}}\left(o_{ij}+o_{ji}\right)\right]}{\sum_{i=1}^{t-1}\sum_{j=i+1}^{t}\left(o_{ij}+o_{ji}\right)}=\sum_{i=1}^{t-1}\sum_{j=i+1}^{t}\left(\frac{d_{ij}^{(T)}}{d_{ij}^{(S)}}\,w_{ij}^{R}\right), \qquad [3.30c]$$

where

$$w_{ij}^{R}=\frac{o_{ij}+o_{ji}}{\sum_{i=1}^{t-1}\sum_{j=i+1}^{t}\left(o_{ij}+o_{ji}\right)} \qquad [3.30d]$$

In expressions [3.30a, b, c, d], t is the total number of nodes covered by the transit route network system $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots \boldsymbol{r}_l\}$, and $d_{ij}^{(T)}$ is the shortest distance (or travel time) between nodes m_i and m_j measured along the transit route network $\boldsymbol{T}^{(l)}$. Nodes m_i and m_j belong to transit network node set $\boldsymbol{N}_T^{(t)} = \boldsymbol{N}_T^{(t)}\{m_1, m_2, \Lambda\ , m_t\}$, which is made up by all the nodes covered by transit route network $\boldsymbol{T}^{(l)}$**.** In general**,** $\boldsymbol{N}_T^{(t)}$ is a subset of the total street node set $\boldsymbol{N}^{(n)}$, i.e., $\boldsymbol{N}_T^{(t)} \subseteq \boldsymbol{N}^{(n)}$. $\boldsymbol{N}_T^{(t)}$ will be equal to $\boldsymbol{N}^{(n)}$ when the transit route network $\boldsymbol{T}^{(l)}$ covers all the street nodes in the street network system. $d_{ij}^{(S)}$ in expressions [3.30a,c] represents the shortest distance (or time) between the two corresponding nodes m_i and m_j in [3.30a,c]. $d_{ij}^{(T)}$ is the shortest distance/time in the transit network. In general, there is the relationship $d_{ij}^{(S)} \le d_{ij}^{(T)} \le d_{ij}^{(r)}$. Figure 3.7 shows an example of different shortest distance/time values between a node pair through different networks or

route. The shortest distance between node 25 and node 2 is obtained from path (25, 26, 27, 20, 35, 13, 8, 1, 2) if traveling along the red dashed line route; or from the path (25, 26, 20, 9, 2) if traveling along the two-line transit network, i.e., red dashed line and green solid line; or from the path (25, 26, 34, 35, 9, 2) if traveling along the shortest path of the street network system.

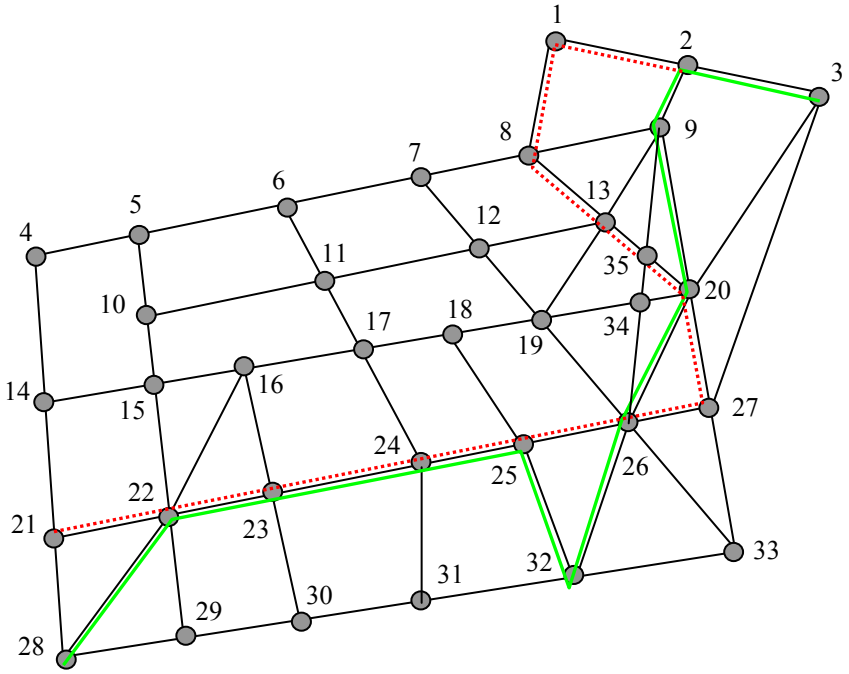


Figure 3.8 Shortest Distances from Different Network/Route System

Variables o_{ij} and o_{ji} in expressions [3.30c] and [3.30d] are coefficients of the OD matrix defined in [3.29]. w_{ij}^{G} in expressions [3.30a, b] and w_{ij}^{R} in expressions [3.30c, d] are weighting coefficients, and w_{ij}^{G} is based on the number of node pairs in the corresponding transit route network, while w_{ij}^{R} is based on the OD trip distribution of the street network. The transit route network directness $d^{G}(\boldsymbol{T}^{(l)})$ in equation [3.30a] has a similar physical meaning as described in previous section for route directness $d^{R}(\boldsymbol{r})$. More specifically, $d^{G}(\boldsymbol{T}^{(l)})$ reflects the average ratio of the two travel distances between each node/stop pair in transit network $\boldsymbol{T}^{(l)}$, one measured along the shortest path via transit route network, and the other along the shortest path in the street network. A value of $d^{G}(\boldsymbol{T}^{(l)}) = 1$ indicates that on average transit vehicles in transit route network $\boldsymbol{T}^{(l)}$ travel along the shortest paths between their OD points, whereas a value of $d^{G}(\boldsymbol{T}^{(l)}) = 2$ indicates that on average transit vehicles in transit route network $\boldsymbol{T}^{(l)}$ travel twice of the distance of that if they travel along the shortest paths between their OD points. $d^{R}(\boldsymbol{T}^{(l)})$ in equation [3.30c] is the transit route network directness that accounts for the ridership/demand distribution along the route and in the network.

Generally the transit ridership/demand based directness parameters $d^R(\boldsymbol{T}^{(l)})$ reflect an aspect of user cost of the transit system while the transit network geometry based directness parameter $d^G(\boldsymbol{T}^{(l)})$ reflects an aspect of operator cost of the transit system.

With the above discussion on transit route network directness, the transit route network directness constraints used in this study may be expressed as follows:

$$d^G\left(\boldsymbol{T}^{(l)}\right) \le d_T^G \;, \quad or \quad d^R\left(\boldsymbol{T}^{(l)}\right) \le d_T^R \;, \qquad [3.30e]$$

where d_T^G and d_T^R are users input maximum transit network directness values or tolerances. d_T^G and d_T^R may be determined in a similar manner as d_r^G and d_r^R. They also have the same implications on the user and operator costs.

(3) Out-of-Direction (OOD) constraints on transit routes. The OOD methodology developed by Welch et al. (1991) is mainly for existing transit route networks. Although it could be a good tool in the post processing stage to fine tune the solution route network, it cannot be directly used in the solution search process developed in this study where routes are constantly changed during search iterations. To incorporate their method into the optimization process developed in this study, the following modification is made. Denote $d_{ij}^{(o)}(\boldsymbol{r})$ as the OOD impact index of node *i* and node *j* of a transit route ***r***, then

$$d_{ij}^{(o)}(\boldsymbol{r}) = \frac{r_{ij}^{(1)}(\boldsymbol{r})\left[l_{ij}(\boldsymbol{r}) - d_{ij}\right]}{r_{ij}^{(2)}(\boldsymbol{r})}, \qquad [3.31a]$$

where $r_{ij}^{(1)}(\boldsymbol{r})$, called the *through ridership* following Welch et al. (1991), is the number of trips or riders traveling on route ***r*** that pass through nodes *i* and *j* without boarding or alighting at any nodes between these two nodes, and $r_{ij}^{(2)}(\boldsymbol{r})$, called the OOD ridership, is the number of trips or riders on route ***r*** that involve either boarding or alighting at nodes between these two nodes. $l_{ij}(\boldsymbol{r})$ is the distance (or time) between nodes *i* and *j* along route ***r***, and d_{ij} is distance (or time) along the shortest path between these two nodes in the street network. The number $l_{ij}(\boldsymbol{r}) - d_{ij}$ in equation [3.32], called the OOD *travel distance* (or time), reflects the increase in travel distance (or time) along route segment of ***r*** rather than along the shortest path segment between nodes *i* and *j*. The physical meaning of OOD defined in [3.32] may be explained as the extra travel distance or time suffered by each through passenger for each OOD passenger. According to Welch et al. (1991), it was believed that in general the maximum value of OOD impact index should not be larger than 15 measured by travel time. Otherwise, the inconvenience to through passengers may have adverse impact on through ridership. In the computer program developed in this study, two user-defined OOD constraint values are used to control the OOD characteristics of individual transit route:

$$d_{ij}^{(o)}(\boldsymbol{r}) \leq d_{\max}^{(o)}, \quad and \quad d^{(o)}(\boldsymbol{r}) \leq d_{ave}^{(o)}, \qquad [3.31b]$$

where $d_{\max}^{(o)}$ is a user-defined maximum OOD impact index value that controls individual OOD route segments on a route, and $d_{ave}^{(o)}$ is another user-defined maximum average OOD impact index value that controls the average OOD characteristics on a route, while $d^{(o)}(\boldsymbol{r})$ is defined as follows,

$$d^{(o)}(\boldsymbol{r}) = \frac{d_{ij}^{(o)}(\boldsymbol{r})}{\frac{r^2 - r}{2}} \qquad [3.31c]$$

r in [3.32c] is the number of nodes in route $\boldsymbol{r}$. The determination of $d_{\max}^{(o)}$ and $d_{ave}^{(o)}$ may be either following certain heuristic guidelines or from survey data Welch et al. (1991).

3.6.4 Implicitly Applied Route Directness Constraints

Some constraints are applied or implied implicitly. Constraints of these types are usually used for the purpose of reducing solution search spaces. For example, when a transit route is represented by k key-node path model, it may automatically exclude some possible paths that may only be represented by higher order key-node models. The selection of k may depend on past experience or certain heuristic guidelines. A larger k will lead to inclusion of paths with poor route directness and large solution search space sizes. Smaller k may result in loss of legitimate solution candidates.

3.7 Optimization Objective Functions

Objective functions are functions of which the values will be maximized or minimized during optimization processes. Depending on particular problems, an optimization process may involve one or more objective functions. In this study, the objective functions considered are various service coverage functions. In a street network system overlaid with a transit route network, a street node i is associated with a given demand, o_{ij} (j = 1, 2, … n), where o_{ij} represents the number of trips originating at node i and destined for node j, and n is the number of nodes in the street network. Nodes i and j are called an OD pair, and o_{ij} is called OD trips from node i to node j and is an element of the OD matrix $\boldsymbol{O}$ (matrix $\boldsymbol{O}$ is defined in [3.28]), also referred to as a demand matrix.

Traveling between any OD pair, e.g., nodes i and j, may or may not involve transfers depending on the configuration of transit route network. If a trip between an OD pair requires no transfer, the trip will be called a zero-transfer trip, while a trip between an OD pair that requires k or fewer transfers will be called a k-or-less transfer trip. A k-or-less transfer trip coverage function, or simply a k-or-less transfer function, is defined as the total number of OD trips that can be accomplished with k or fewer transfers in a transit network service area. The function is evaluated based on the assumption that the network demand is given and that the goal is to optimize the transit route network structure to maximize 0-transfer trips while attempting to

provide services to meet as much demand as possible in a transit service area. In other words, in this study, the interaction between transit service quality and demand is not modeled. In reality, transit demand thus a service coverage function value not only depends on the route network layout or structure, but is also influenced by service frequencies, vehicle time tables, and other service quality factors such as bus stop amenities.

The following is a description of the various transfer coverage functions used in this study. Denote f_k as a k-or-less transfer function, then

$$f_k = f_k(\boldsymbol{T}, \boldsymbol{O}),\ k = 0, 1, 2, \ldots \qquad [3.32]$$

where $\boldsymbol{T}$ and $\boldsymbol{O}$ are, respectively, the transit route network matrix as defined in equations [3.9a,b] and the OD matrix of the system. Expression [3.32] reflects the fact that the value of the transfer trip coverage functions depends on the transit route network configuration and the demand distribution as defined by the OD matrix. Let f_T be the total number of OD trips in a street network OD matrix, i.e.,

$$f_T = f_T(\boldsymbol{O}) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(o_{ij} + o_{ji}\right), \qquad [3.33]$$

then the following function value

$$r_k = \frac{f_k}{f_T} \qquad [3.34]$$

represents the ratio of OD trips that require k-or-less transfers to the total number of OD trips in the demand matrix. Function value r_k will be referred to as k-or-less transfer function ratio, or simply k-or-less transfer ratio. The k-or-less transfer ratio is an indicator of the quality of a transit route network. For example, $r_0 = 0.8$ means that 80% of transit riders can travel between their origin and destination points without any transfers, whereas a value of $r_1 = 0.9$ indicates that 90% of transit riders can travel between their origin and destination points with zero or one transfer. It is easy to see that for a transit route network with fixed total route length or service hours, a greater value of the zero-transfer function or zero-transfer function ratio corresponds to better service quality. For transfer function ratio defined in [3.34], the optimal value is 1.0, while for transfer function in [3.32], the optimal value is f_T. In the computer program there are several options to select the objective functions. These options are described bellow.

3.7.1 Object Function Based on Zero Transfer Trip Coverage

The optimization process will maximize the function value of f_0 (or r_0) under appropriate constraints described in previous sections. Explicitly, the zero-transfer function may be expressed in terms of OD matrix $\boldsymbol{O}$ and transit route network matrix $\boldsymbol{T}$ as follows,

$$f_0 = f_0(\boldsymbol{T}, \boldsymbol{O}) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left[\left(o_{ij} + o_{ji}\right) h_{ij}^{(0)}\right], \qquad [3.35]$$

where o_{ij} are coefficients of OD matrix $\boldsymbol{O}$ and $h_{ij}^{(0)}$ are coefficients with the following property

$$h_{ij}^{(0)} = \begin{cases} 1, \text{ if node } i \text{ and node } j \text{ are directly connected by at least one transit route} \\ 0, \text{ otherwise} \end{cases}. \qquad [3.36]$$

Coefficients $h_{ij}^{(0)}$ will be referred to as zero-transfer coefficients and their associated matrix $\boldsymbol{H}^{(0)} = \boldsymbol{H}^{(0)}\left(h_{ij}^{(0)}\right)$ as zero-transfer matrix. Coefficients $h_{ij}^{(0)}$ may be derived from transit route network matrix $\boldsymbol{T}$ based on the following relationships,

$$\alpha_{ij}^{(0)} = \sum_{k=1}^{l} t_{ki} t_{kj} \text{ , and} \qquad [3.37a]$$

$$h_{ij}^{(0)} = h\left(a_{ij}^{(0)}\right), \qquad [3.37b]$$

where t_{ki} and t_{kj} are coefficients of transit route network matrix $\boldsymbol{T}$ defined in equations [3.9a,b], and h is a Heaviside step function, or simply an h-function, which has the property bellow:

$$\boldsymbol{h}(\boldsymbol{x}) = \begin{cases} 1, & \text{if } \boldsymbol{x} > 0 \\ 0, & \text{otherwise} \end{cases}. \qquad [3.38]$$

The physical meaning of coefficient $\alpha_{ij}^{(0)}$ in equation [3.37a] is the number of transit routes that directly connect street nodes i and j. For a given OD matrix $\boldsymbol{O}$, the optimization process is to select a transit route network $\boldsymbol{T}^{(l)}$ from an appropriate solution space such that the zero-transfer function f_0 in [3.35] reaches a maximum value.

3.7.2 Object Function Based on One-or-Less Transfer Trip Coverage

The optimization process will maximize the function value of f_1 (or r_1) under appropriate constraints. The one-or-less transfer function may also be expressed in terms of OD matrix $\boldsymbol{O}$ and transit route network matrix $\boldsymbol{T}$. Compared with f_0, the evaluation of one-transfer function is much more expensive due to the large number of arithmetic operations involved. The following is a brief description of the relationship between the one-or-less transfer function f_1 and matrices $\boldsymbol{O}$ and $\boldsymbol{T}$. Similar to the zero-transfer function, the one-or-less transfer function may be expressed in terms of OD matrix $\boldsymbol{O}$ and one-or-less transfer coefficients $h_{ij}^{(1)}$ (or matrix $\boldsymbol{H}^{(1)}$),

$$f_1 = f_1(\boldsymbol{T}, \boldsymbol{O}) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left[\left(o_{ij} + o_{ji} \right) h_{ij}^{(1)} \right], \qquad [3.39]$$

where o_{ij} are the coefficients of OD matrix $\boldsymbol{O}$ and $h_{ij}^{(1)}$ are coefficients that have the following property:

$$h_{ij}^{(1)} = \begin{cases} 1, \text{ if node } i \text{ and node } j \text{ are connected by one or two transit routes} \\ 0, \text{ otherwise} \end{cases} \quad [3.40]$$

Coefficients $h_{ij}^{(1)}$ and their associated matrix $\boldsymbol{H}^{(1)} = \boldsymbol{H}^{(1)}\left(h_{ij}^{(1)}\right)$ will be referred to as one-or-less transfer coefficients and one-or-less transfer matrix, respectively. The relationship between $h_{ij}^{(1)}$ and t_{ij} is given by the following equations:

$$h_{ij}^{(1)} = h\left(\alpha_{ij}^{(1)}\right), \quad [3.41a]$$

$$\alpha_{ij}^{(1)} = \sum_{k=1}^{l} \sum_{m=k}^{l} \left(t_{ki}\, t_{mj}\, \alpha_{km}\right), \quad [3.41b]$$

$$\alpha_{km} = \sum_{j=1}^{n} t_{kj}\, t_{mj} \quad [3.41c]$$

where h is the h-function defined in equation [3.38], and $\alpha_{ij}^{(1)}$ are auxiliary coefficients that have the following property:

$$\alpha_{ij}^{(1)} = \begin{cases} \geq 1, \text{ if node } i \text{ and node } j \text{ are connected by one or two transit routes} \\ 0, \text{ otherwise} \end{cases}. \quad [3.41d]$$

Coefficients α_{km} in equation [3.41c] are transit/bus line intersection coefficients that have the property

$$\alpha_{km} = \begin{cases} \geq 1, \text{ if transit lines } k \text{ and } m \text{ intersect each other} \\ 0, \text{ otherwise} \end{cases}. \quad [3.42]$$

Physically, α_{km} represents the number of nodes that appear on both transit line k and line m. It may be seen by comparing equations [3.41] with [3.37] that optimization with one-or-less transfer objective function f_1 involves a much larger number of arithmetic operations than optimization with zero-transfer objective function f_0.

3.7.3 Objective Function Based on Two-or-Less Transfer Trip Coverage

The optimization process will maximize the function value of f_2 (or r_2) under appropriate constraints. Similar to the zero-transfer and one-or-less transfer function cases, the two-or-less transfer function may be expressed in terms of OD matrix $\boldsymbol{O}$ and two-or-less transfer coefficients $h_{ij}^{(2)}$ (or matrix $\boldsymbol{H}^{(2)}$) as follows,

$$f_2 = f_2(\boldsymbol{T}, \boldsymbol{O}) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left[\left(o_{ij} + o_{ji}\right) h_{ij}^{(2)}\right], \quad [3.43a]$$

where o_{ij} are coefficients of *OD* matrix $\boldsymbol{O}$ and $h_{ij}^{(2)}$ are coefficients that have the following property

$$h_{ij}^{(2)} = \begin{cases} 1, \text{ if node } i \text{ and node } j \text{ are connected by three or fewer transit routes} \\ 0, \text{ otherwise} \end{cases} \quad [3.43b]$$

Coefficients $h_{ij}^{(2)}$ and their associated matrix $\boldsymbol{H}^{(2)} = \boldsymbol{H}^{(2)}(h_{ij}^{(2)})$ will be referred to as two-or-less transfer coefficients and two-or-less transfer matrix, respectively. The relationship between $h_{ij}^{(2)}$ and t_{ij} are given by the following equations,

$$h_{ij}^{(2)} = h\left(\alpha_{ij}^{(2)}\right), \quad [3.44a]$$

$$\alpha_{ij}^{(2)} = \sum_{k=1}^{l} \sum_{m=k}^{l} \left(t_{ki}\, t_{mj}\, \beta_{km}\right), \quad [3.44b]$$

$$\beta_{km} = \sum_{i=1}^{l} \alpha_{ki}\, \alpha_{mi}, \quad [3.44c]$$

where h is the h-function defined in equation [3.38], and $\alpha_{ij}^{(2)}$ are auxiliary coefficients that have the following property:

$$\alpha_{ij}^{(2)} = \begin{cases} \geq 1, \text{ if node } i \text{ and node } j \text{ are connected by less then three transit routes} \\ 0, \text{ otherwise} \end{cases}. \quad [3.44d]$$

In equation [3.44c], coefficients α_{ki} and α_{mi} are defined in equation [3.42], and coefficients β_{km} have the following properties:

$$\beta_{km} = \begin{cases} \geq 1, \text{ if transit lines } k \text{ and } m \text{ either intersect or} \\ \quad \text{if they both intersect a third transit line} \\ 0, \text{ otherwise} \end{cases}. \quad [3.44e]$$

Optimization with two-or-less transfer objective function f_2 is even more computational intensive than optimization with one-or-less transfer objective function f_1 due to the great number of arithmetic operations involved in equations [3.44] to obtain all the required coefficients. In fact, numerical tests conducted in this study have indicated that for current high-end PCs, f_2 seems only good for small or some mid sized transit route network optimization problems. For large problems, powerful mainframe computers or special computational means such as multiprocessor parallel computing schemes may be required to obtain results in a reasonable amount of time.

3.7.4 Objective Function of Multiple Objects

There are mainly two methods to deal with optimization problems with more than one object. The first method is to formulate objective functions for individual optimization objects, then create a single composite objective function from these individual objective functions using weighting coefficients. The contribution of a particular objective function to the resulting composite objective function may be adjusted by assign an appropriate weighting coefficient. For example, if all three transfer coverage functions f_0, f_1 and f_2 are to be incorporated into a single objective function, three weighting coefficients α_0, α_1, and α_2 may be introduced to establish the following composite objective function,

$$f_c = \alpha_0 f_0 + \alpha_1 f_1 + \alpha_2 f_2 \qquad [3.45]$$

The magnitude of a particular weighting coefficient with respect to the others will determine the relative importance of the corresponding objective function. Mathematically, solution methods based on a single objective function are easier to formulate if weighting coefficients are given. However, in practice, such methods may be difficult to use since determination of appropriate weighting coefficients may either require in-depth knowledge about the domain or itself be a difficult optimization sub-problem.

The second method does not use weighting coefficients to represent the relative importance of individual objective functions. The solution procedure of such method involves a scheme that first selects potential solutions that meet the most important optimization objective, then from this selected solution pool, seek solutions that also meet the next important objective. The selection process will continue until all the optimization objectives are met. To illustrate the idea of the second method, consider a transit route network design optimization problem. Assume that a transit rider is covered by transit services if he or she can travel from his or her origin to destination with two or fewer transfers (i.e., the person is covered by a two-or-less transfer trip). If the primary design object is the maximization of zero-transfer trip coverage and the secondary object is the maximization of transit network service ridership coverage (i.e., two-or-less transfer trip coverage), then one has the following optimization objectives:

- Maximization of primary objective function: f_0
- Maximization of secondary objective function: f_2

During solution search iteration, solutions that meet the primary objective, i.e., with large zero-transfer function values, will be selected first. From these solutions that have the same zero-transfer function values, solution that meet the secondary objective, i.e., having large two-or-less transfer function values, will be selected next. A list of some of the objective function combinations available in the computer code developed for this study is given below, where $\boldsymbol{f}$ is used to represent the combination of objective functions.

1. $\boldsymbol{f}$ = (k) – Maximization of single objective function f_k (k = 0, 1, 2):
   $\boldsymbol{f}$ = (0): Maximization of zero-transfer objective function f_0;
   $\boldsymbol{f}$ = (1): Maximization of one-or-less transfer objective function f_1;

$\boldsymbol{f}$ = (2): Maximization of two-or-less transfer objective function f_2.

2. $\boldsymbol{f}$ = (i, j) – Maximization of two objective functions with f_i as the primary objective function and f_i as the secondary objective function (i, j = 0, 1, 2, $i \neq j$). For example,
   $\boldsymbol{f}$ = (0, 1): Maximization of zero-transfer function f_0 with one-or-less transfer function f_1 as the secondary function;
   $\boldsymbol{f}$ = (0, 2): Maximization of zero-transfer function f_0 with two-or-less transfer function f_2 as the secondary function;
   $\boldsymbol{f}$ = (1, 0): Maximization of one-or-less transfer function f_1 with zero-transfer function f_0 as the secondary function.

3. $\boldsymbol{f}$ = (0, 1, 2) – Maximization of multiple objective functions with zero-transfer function f_0 as the primary objective function, f_1 as the secondary objective function, and f_2 as the third objective function.

4. $\boldsymbol{f}$ = ((0, 1)) – Maximization of two objective functions with both zero-transfer function f_0 and one-or-less transfer function f_1 as the primary objective functions. In such cases, a route network will be selected during solution search iterations if both f_0 and f_1 have higher values than the existing network. This option is for practical situations where one wants to improve the zero-transfer trip coverage without sacrificing the total service coverage, which is defined in this case by the one-or-less transfer coverage.

5. $\boldsymbol{f}$ = ((0, 2)) – Maximization of two objective functions with both zero-transfer function f_0 and two-or-less transfer function f_2 as the primary objective functions. A route network will be selected if both f_0 and f_2 have higher values than the existing network. This option is for practical situations where one wants to improve the zero-transfer trip coverage without sacrificing the total service coverage defined by the two-or-less transfer coverage.

6. $\boldsymbol{f}$ = ((1, 2)) – Maximization of two objective functions with both one-transfer function f_1 and two-or-less transfer function f_2 as the primary objective functions. A route network will be selected if both f_1 and f_2 have higher values than the existing network. This option is for improving the one-or-less transfer trip coverage without sacrificing the total service coverage defined by the two-or-less transfer coverage. For transit systems that includes rail lines or a trunk-and-feeder system, certain bus lines are designed as feeder lines to serve rail or trunk lines. In such cases, the one-or-less transfer objective function f_1 will be more appropriate than the zero-transfer function f_0.

7. $\boldsymbol{f}$ = ((0, 1, 2)) – Maximization of multiple objective functions with zero transfer function f_0, one-or-less transfer function f_1, and two-or-less transfer function f_2 as the primary objective functions. A route network will be selected if all the three associated functions f_0, f_1, and f_2 have better values than the existing network. This option is for situations where one wants to improve one objective without sacrificing the others.

8. $\boldsymbol{f}$ = (t_2) – Minimization of the transfer directness objective function defined as follows.

$$t_2(\boldsymbol{T},\boldsymbol{O}) = \frac{f_0 + 2(f_1 - f_0) + 3(f_2 - f_1) + \alpha(f_T - f_2)}{f_T}, \quad [3.46]$$

where α is a weighting coefficient to penalize uncovered trips during the optimization process of the transit route network system. Expression [3.46] has the same form as that of [3.45] where the coefficient α has to be determined heuristically or through trial-and-error. The physical meaning of the objective function $t_2(\boldsymbol{T}, \boldsymbol{O})$ is the average number of vehicle boardings that a transit rider has to make to accomplish an *OD* trip. The optimal value of $t_2(\boldsymbol{T}, \boldsymbol{O})$ is 1.0. In such cases, all transit riders board transit vehicles only once, therefore all trips are zero-transfer trips. The denominator in the expression, f_T, is the number of total trips from the transit demand matrix $\boldsymbol{O}$, whereas the numerator is the total passenger boardings in transit network system. The physical meanings of the various terms in equation [3.46] may be explained as follows. For the f_0 transit riders who complete their OD trips without transfers, the number of boardings is f_0 since these people need only to board buses or transit vehicles once. For the $(f_1 - f_0)$ transit riders who complete their OD trips with exactly one transfer, the number of boardings is $2(f_1 - f_0)$ since transferring to another transit vehicle requires an additional boarding. For the $(f_2 - f_1)$ transit riders who complete their OD trips with exactly two transfers, the corresponding number of boardings is $3(f_2 - f_1)$. Any trips that are either not covered by the route network under analysis or require more than two transfers to accomplish are considered as not covered by the transit network service. Therefore, the total uncovered trips in the route network will be $(f_T - f_2)$. To account for the $(f_T - f_2)$ uncovered trips in the measurement, a penalty α is introduced to include a fictitious transit vehicle boarding number $\alpha(f_T - f_2)$. This is to avoid solutions that have a small $t_2(\boldsymbol{T}, \boldsymbol{O})$ value but covers only a small percentage of the demand. The value of the penalty number α needs to be determined by transit planners. For example, one may consider each of the uncovered trips as four vehicle boardings, i.e., $\alpha = 4$. In general, the larger the value α is, the greater importance is given to service coverage in the minimization process. However, α should not be too large relative to other coefficients, as it may result in the optimization process attempting to minimize the single term $(f_T - f_2)$ and neglecting the effects of other factors. In the computer program developed in this study, the default value for α is four, which seems to give reasonable results.

9. $\boldsymbol{f} = (t_1)$ – Minimization of the transfer directness objective function defined below:

$$t_1(\boldsymbol{T},\boldsymbol{O}) = \frac{f_0 + 2(f_1 - f_0) + \alpha(f_T - f_1)}{f_T}. \quad [3.47]$$

The physical meaning of t_1 is similar to that of $t_2(\boldsymbol{T}, \boldsymbol{O})$, defined in [3.46], except that uncovered trips now will include trips that require two or more transfers.

3.8 Solution Search Algorithms

In this section, several solution search algorithms developed in the study are presented. Limitations and advantages of these algorithms are also discussed. In section 3.4, a local transit network solution search space based on branch-and-bound and localization concepts has been represented as (ref. equation [3.23])

$$\boldsymbol{T}_{(i)}^{(k)}\left(\boldsymbol{T}^{(l)}\right)=\boldsymbol{T}_{(i)}^{(k)}\left\{\boldsymbol{P}_{(i)}^{(k)}\left(\boldsymbol{r}_1^{(s_1)}\right),\boldsymbol{P}_{(i)}^{(k)}\left(\boldsymbol{r}_2^{(s_2)}\right),\Lambda\ ,\boldsymbol{P}_{(i)}^{(k)}\left(\boldsymbol{r}_l^{(s_l)}\right)\right\}, \qquad [3.48]$$

where $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_l\}$ is the master transit network; $\boldsymbol{r}_j$ (j = 1, 2, …, l) are the l routes in the network; $\boldsymbol{r}_j^{(s_j)}$ is the s_j-node representation of route $\boldsymbol{r}_j$; $\boldsymbol{P}_{(i)}^{(k)}\left(\boldsymbol{r}_j^{(s_j)}\right)$ is the local path search space generated from $\boldsymbol{r}_j^{(s_j)}$; i is the order of the local space; and k indicates that the local search space is based on a k-level shortest path space (ref. [3.19]). As explained in section 3.4, the search space $\mathbf{T}_{(i)}^{(k)}\left(\mathbf{T}^{(l)}\right)$ may be flexibly defined and expanded to include all possible solutions by increasing i or k, or both. The smallest search space defined by this representation is obtained by setting k = 1 and i =1,

$$\boldsymbol{T}_{(1)}^{(1)}\left(\boldsymbol{T}^{(l)}\right)=\boldsymbol{T}_{(1)}^{(1)}\left\{\boldsymbol{P}_{(1)}^{(1)}\left(\boldsymbol{r}_1^{(s_1)}\right),\boldsymbol{P}_{(1)}^{(1)}\left(\boldsymbol{r}_2^{(s_2)}\right),\Lambda\ ,\boldsymbol{P}_{(1)}^{(1)}\left(\boldsymbol{r}_l^{(s_l)}\right)\right\}, \qquad [3.48a]$$

where the local path space $\boldsymbol{P}_{(1)}^{(1)}\left(\boldsymbol{r}_j^{(s_j)}\right)$ is obtained by connecting adjacent local node spaces generated from nodes of path $\boldsymbol{r}_j^{(s_j)}$ with shortest path segments. Since the solution methods used for most of the application problems or examples in this research are based on the smallest search space defined in [3.48a], for simplicity, the first order, 1-level shortest path space based local network space $\boldsymbol{T}_{(1)}^{(1)}\left(\boldsymbol{T}^{(l)}\right)$ and local path space $\boldsymbol{P}_{(1)}^{(1)}(\boldsymbol{r})$ will be written, respectively, as $\boldsymbol{L}^{T}(\boldsymbol{T}^{(l)})$ and $\boldsymbol{L}^{P}(\boldsymbol{r})$, while the first order local node space $\boldsymbol{L}_{(1)}^{N}(i)$ of a master node i defined in equation [3.16] will be denoted as $\boldsymbol{L}^{N}(i)$. With the above definitions, equation [3.48a] now becomes

$$\boldsymbol{L}^{T}\left(\boldsymbol{T}^{(l)}\right)=\boldsymbol{L}^{T}\left\{\boldsymbol{L}^{P}\left(\boldsymbol{r}_1^{(s_1)}\right),\boldsymbol{L}^{P}\left(\boldsymbol{r}_2^{(s_2)}\right),\Lambda\ ,\boldsymbol{L}^{P}\left(\boldsymbol{r}_l^{(s_l)}\right)\right\}. \qquad [3.48b]$$

This search space now only includes those route networks that are located in the immediate neighborhood of the master network $\boldsymbol{T}^{(l)}$. However, for practical transit network optimization problems, this small local solution search space defined by [3.48b] may be still too large for any solution algorithms to obtain results based on exhaust search schemes. To illustrate, consider a street network $\boldsymbol{A}^{(m)}$ with an average nodal adjacency list length $\overline{m} = 4$ and a transit network $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_l\}$ defined in $\boldsymbol{A}^{(m)}$. Recall, from equation [3.6a], that the length of a nodal adjacency list is the number of nodes in that list, and $\overline{m}$ is the average nodal adjacency list length of the entire street network. Using *two-node* representation, i.e., $s_j = 2$, a local path space $\boldsymbol{L}^{P}(\boldsymbol{r}_1^{(2)})$ generated from a typical master path $\boldsymbol{r}$ in $\boldsymbol{T}^{(l)}$ will have on average a path population of $k_p = (\overline{m} + 1)^2$. Since there are l routes in transit network $\boldsymbol{T}^{(l)}$, the network population in the local network space $\boldsymbol{L}^{T}(\boldsymbol{T}^{(l)})$ defined in [3.48b] will be in the order of $k_T = k_p^l = (\overline{m} + 1)^{2l}$. For a transit network of bus route number l = 50, the population number in network search space $\boldsymbol{L}^{T}(\boldsymbol{T}^{(l)})$ will be in the order of $k_T = (4 + 1)^{2x50} = 5^{100}$, which is impossible to handle by current computer resources in any transit agencies in this country.

To illustrate with a simple example, consider the street network shown in Figure 3.1 where the transit network consists of two routes $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ represented by the green solid line and red dotted line, respectively. The route $\boldsymbol{r}_1$ starts from node 28 and ends at node 3, and both nodes have nodal adjacency list length of 3. Node 28's local node space $\boldsymbol{L}^N(28)$ = {21, 22, 29, 28} consists of node 28's nodal adjacency list nodes 21, 22, 29 and node 28 itself, while node 3's local node space $\boldsymbol{L}^N(3)$ = {2, 20, 27, 3} consists of node 3's nodal adjacency list nodes 2, 20, 27 and node 3 itself. The local path space $\boldsymbol{L}^P(\boldsymbol{r}_1^{(2)})$ based on the two-node representation of the green route $\boldsymbol{r}_1$ is constructed by connecting the four nodes in node 28's local space $\boldsymbol{L}^N(28)$ = {21, 22, 29, 28} with the four nodes in node 3's local space $\boldsymbol{L}^N(3)$ = {2, 20, 27, 3}, respectively, to obtain 4^2 = 16 shortest paths,

$$\boldsymbol{L}^P\left(\boldsymbol{r}_1^{(2)}\right) = \left\{\boldsymbol{p}_{ij}^S\right\}, \quad i = 21, 22, 29, 28, \quad j = 2, 20, 27, 3, \tag{3.49a}$$

where $\boldsymbol{p}_{ij}^S$ represents the 1-level shortest path (ref. equation [3.22]) between nodes i and j. Route $\boldsymbol{r}_2$ represented by the red dotted line in Figure 3.1 starts from node 21, and ends at node 2. Following the same procedure as before, the local path space $\boldsymbol{L}^P(\boldsymbol{r}_2^{(2)})$ for route $\boldsymbol{r}_2$ can be generated by connecting the four nodes in node 21's local node space $\boldsymbol{L}^N(21)$ = {28, 22, 14, 21} with the four nodes in node 2's local node space in $\boldsymbol{L}^N(2)$ = {1, 9, 3, 2}, respectively, to obtain 4^2 = 16 shortest paths,

$$\boldsymbol{L}^P\left(\boldsymbol{r}_2^{(2)}\right) = \left\{\boldsymbol{p}_{ij}^S\right\}, \; i = 28, 22, 14, 21; j = 1, 9, 3, 2. \tag{3.49b}$$

The local transit network search space of the two-route transit network $\boldsymbol{T}^{(2)}$ = {$\boldsymbol{r}_1$, $\boldsymbol{r}_2$} is established by choosing the first path and the second paths, respectively, from the path space $\boldsymbol{L}^P(\boldsymbol{r}_1^{(2)})$ defined in [3.49a], and the path space $\boldsymbol{L}^P(\boldsymbol{r}_2^{(2)})$ defined in [3.49b],

$$\boldsymbol{L}^T\left(\boldsymbol{T}^{(2)}\right) = \boldsymbol{L}^T\left\{\boldsymbol{L}^P\left(\boldsymbol{r}_1^{(2)}\right), \boldsymbol{L}^P\left(\boldsymbol{r}_2^{(2)}\right)\right\} = \left\{ \left\{\boldsymbol{p}_{ij}^S\right\}_{\substack{i=21,22,29,28 \\ j=2,20,27,23}}, \quad \left\{\boldsymbol{p}_{ij}^S\right\}_{\substack{i=28,22,14,21 \\ j=1,9,3,2}} \right\}. \tag{3.50}$$

Since there are 4^2 x 4^2 = 256 different ways to select these two paths from path spaces $\boldsymbol{L}^P(\boldsymbol{r}_1^{(2)})$ and $\boldsymbol{L}^P(\boldsymbol{r}_2^{(2)})$, the local transit network search space of the two-route transit network has a population of 256 (note that some of the networks may have same paths).

From the above discussion, it may be seen that even for a locally defined solution search space, it is still not practical to find a local optimal solution through the evaluation of all the possible solution candidates in the local space. Solution methods that do not require evaluation of all the members in a search space are needed for practical problems.

The following is a description of the various solution search methods developed in this study. The basic assumption of these methods is that the transit demand distribution in a transit network service area has certain continuity or characteristics. In another word, nodes or areas with certain transit trip/demand-distribution values are probably surrounded by nodes or areas with

similar demands. In such cases, it may be more effective in searching of the optimal solution by evaluating paths that pass or near nodes or areas with higher transit demand in stead of evaluating all the paths in a solution space.

Assume that at iteration k, one obtains a transit route network result,

$$\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_i, \ldots, \boldsymbol{r}_l\}, \quad [3.51a]$$

where

$$\boldsymbol{r}_i = (i_1, i_2, \ldots, i_{n(i)}),\ i = 1, 2, \ldots, l \quad [3.51b]$$

are the l routes of the network, and $n(i)$ is the number of nodes on route i. The local transit route network search space of the master network $\boldsymbol{T}^{(l)}$ is, from [3.48],

$$\boldsymbol{L}^T\left(\boldsymbol{T}^{(l)}\right) = \boldsymbol{L}^T\left\{\boldsymbol{L}^P\left(\boldsymbol{r}_1^{(s_1)}\right), \boldsymbol{L}^P\left(\boldsymbol{r}_2^{(s_2)}\right), \Lambda\ , \boldsymbol{L}^P\left(\boldsymbol{r}_l^{(s_l)}\right)\right\}, \quad [3.52a]$$

where, from [3.49a,b],

$$\boldsymbol{L}^P\left(\boldsymbol{r}_i^{(s_i)}\right), \quad i = 1,2,\Lambda\ ,l \quad [3.51b]$$

is the local path search space of the master path $\boldsymbol{r}_i^{(s_i)}$, and path $\boldsymbol{r}_i^{(s_i)}$ is the s_i-node representation of path $\boldsymbol{r}_i$. The goal is to find a better route network result for the next, or the $(k+1)^{\text{th}}$ iteration. The following subsections describe the various methods and the corresponding procedures for selecting better results from local search spaces. For simplicity, only procedures that evaluate/select one route at time will be presented. The procedures to evaluate/select multiple routes at a time are similar except that the evaluation/selection target is a route subspace instead of a single route.

3.8.1 Greedy Search Method I (GS1)

The basic idea and search procedures of this search method may be described through a simple example. Assume that from a previous search process, one obtains an intermediate route network result $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots \boldsymbol{r}_l\}$, and is looking for a better alternative to update/replace one of the routes, $\boldsymbol{r}_j$, in the route network. To illustrate, assume that the path shown in Figure 3.5(a) represents route $\boldsymbol{r}_j$, then the solution search procedure of this method may follow the steps below:

(1) Select a route from the current route network, i.e., route $\boldsymbol{r}_j$ in this case.
(2) Select key-nodes from route $\boldsymbol{r}_j$. For two-node representation, the two key-nodes are the starting and ending nodes of route $\boldsymbol{r}_j$, i.e., nodes n_1 and n_2 in Figure 3.5(b). For three-node representation, the three key-nodes are the starting, the ending, and center nodes of route $\boldsymbol{r}_j$, i.e., nodes n_1, n_2, and n_3 in Figure 3.5(c). Assume the three-node representation of route $\boldsymbol{r}_j$ shown in Figure 3.5(c) is used in this example, and is denoted as $\boldsymbol{r}_j^{(3)}$.
(3) From the three key-nodes n_1, n_2, and n_3 shown in Figure 3.5(c), generate the three corresponding local node spaces:
$\boldsymbol{L}^N(n_1) = \{n_1, n_{11}, n_{12}, n_{13}, n_{14}\}$ – the local node space of node n_1

$\boldsymbol{L}^N(n_2) = \{n_2, n_{21}, n_{22}, n_{23}, n_{24}\}$ – the local node space of node n_2

$\boldsymbol{L}^N(n_3) = \{n_3, n_{31}, n_{32}, n_{33}, n_{34}\}$ – the local node space of node n_3

Note, in this example, the average nodal adjacency list length $\overline{m} = 4$, and number of nodes in all the three local node spaces is $\overline{m} + 1 = 5$.

(4) First, connect nodes in node spaces $\boldsymbol{L}^N(n_1)$ and $\boldsymbol{L}^N(n_3)$ with shortest paths to obtain 5 × 5 = 25 shortest path segments. Then extend these shortest path segments to nodes in node space $\boldsymbol{L}^N(n_2)$ with shortest paths to obtain 25 × 5 = 125 paths. Each of the 125 paths is made from two shortest path segments. These 125 paths is the local path space of route $\boldsymbol{r}_j$ based on three-node representation $\boldsymbol{L}^P(r_j^{(3)})$. In general, a local path space with an s-node representation and an average nodal adjacency list length $\overline{m}$ = 4 will have a path space population of $(\overline{m} + 1)^5$.

(5) Replace route $\boldsymbol{r}_j$ in the existing transit network $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_l\}$, with path $\boldsymbol{r}_{jk}$ in path space $\boldsymbol{L}^P(r_j^{(3)})$ to obtain an updated transit network $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{j-1}, \boldsymbol{r}_{jk}, \boldsymbol{r}_{j+1}, \ldots, \boldsymbol{r}_l\}$, and perform function evaluation of the resulting transit route network. Whenever a better result is obtained, i.e., the objective function value associated with path $\boldsymbol{r}_{jk}$ is better than that associated with route $\boldsymbol{r}_j$, replace $\boldsymbol{r}_j$ with $\boldsymbol{r}_{jk}$, and go to step (1) to start a new search process with the updated route $\boldsymbol{r}_j$. If no better result is found after going through all the 125 paths $\boldsymbol{r}_{jk} \in \boldsymbol{L}^P(r_j^{(3)})$ (k = 1, 2, …, 125), go Step (6) with the original transit network $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_l\}$ in this step.

(6) Select the next route from the transit route network, e.g., route $\boldsymbol{r}_{(j+1)}$, and go to Step (1) to start a new local search process for route $\boldsymbol{r}_{(j+1)}$.

(7) The search process will be considered converged if no better results can be found from all the individual route's local path search spaces.

The basic procedure of **GS1** method now may be summarized as follow. First, select a starting initial route network, and generate a solution search subspace $\boldsymbol{L}_s^T$ of tractable size with existing computing resources. For example, based on the observation that averagely a street node in most transit service areas is jointed by no more than four street segments, one may assume that the average length of a nodal adjacency list in a street network is $\overline{m}$ = 4. The average number of nodes in a local node space of such street network will be $\overline{m}$ + 1 = 5. The search subspace $\boldsymbol{L}_s^T$ will have a solution population of $(\overline{m} + 1)^5 = 5^5$ if it only includes one local path space as is the case described in Step (1) and Step (7), and $\boldsymbol{L}_s^T$ will have a solution population of $(\overline{m} + 1)^{t\times5} = 5^{t\times5}$ if it includes t local path spaces. Second, with exhaust search, find the first encountered path (or paths if $t > l$) from the search subspace $\boldsymbol{L}_s^T$ that gives better objective function value (or values) than the existing one. If the current search subspace $\boldsymbol{L}_s^T$ does not have any better solution, generate a new subspace $\boldsymbol{L}_s^T$ with a new route or routes in route network $\boldsymbol{T}^{(l)}$. Third, generate a new solution search subspace $\boldsymbol{L}_s^T$ from the better path (or paths) obtained from the second step, and start a new search process or iteration. The search process will stop and be considered as converged if no better results could be found.

Numerical results from this study have shown that **GS1** method converges relatively faster than other methods developed in this study. For a convex solution search space, **GS1** method should produce a global optimal result. However, in reality, most transit route network optimization problems are either intuitively non-convex or with unknown convex-ness characteristics. In such

cases, the **GS1** method may suffer rapid and premature converge to one of the local optimal results near initial guess route network.

3.8.2 Greedy Search Method II (GS2)

The main difference between **GS2** method and the **GS1** method is that the **GS2** method will go to the next route, e.g., route $\boldsymbol{r}_{(j+1)}$, and start a new search process whenever a better route alternative is obtained for the current route $\boldsymbol{r}_j$ while the **GS1** method will continue the search process for the current route $\boldsymbol{r}_j$ until it could not find any better alternative path for route $\boldsymbol{r}_j$. Intuitively, the route network layouts obtained from **GS2** method should be more evenly distributed among various transit demand areas than those obtained from **GS2** method. The **GS2** method's search procedures are described below.

Steps (1) through (4) are the same as those for **GS1**.

(5) Replace route $\boldsymbol{r}_j$ in the existing transit network $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_l\}$ with path $\boldsymbol{r}_{jk} \in \boldsymbol{L}^P(\boldsymbol{r}_j^{(3)})$ to obtain an updated transit network $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{j-1}, \boldsymbol{r}_{jk}, \boldsymbol{r}_{j+1}, \ldots, \boldsymbol{r}_l\}$, and perform function evaluation of the resultant transit route network. Whenever a better result is obtained, i.e., the objective function value associated with path $\boldsymbol{r}_{jk}$ is better than that associated with route $\boldsymbol{r}_j$, replace $\boldsymbol{r}_j$ with $\boldsymbol{r}_{jk}$ to obtain an updated transit route network, and go to Step (6).

(6) Select next route from the transit route network, e.g., route $\boldsymbol{r}_{(j+1)}$, and go to Step (1) to start a new local search process for route $\boldsymbol{r}_{(j+1)}$.

(7) The search process will be considered converged if no better results can be found from all the individual route's local path search spaces.

3.8.3 Hill-Climbing Search Method (HC)

From the description of the previous two sections, it may be seen that a greedy-type method, **GS1** or **GS2**, searches for a better route from the local path space $\boldsymbol{L}^P\left(\boldsymbol{r}_j^{(s_j)}\right)$ of an existing route $\boldsymbol{r}_j$. Whenever a better route, denoted as $\boldsymbol{r}_j^{(b1)}$, in local path space $\boldsymbol{L}^P\left(\boldsymbol{r}_j^{(s_j)}\right)$ is found, the existing route $\boldsymbol{r}_j$ will be replaced with $\boldsymbol{r}_j^{(b1)}$, and a new search process will be started with the updated $\boldsymbol{r}_j^{(b1)}$. Although route $\boldsymbol{r}_j^{(b1)}$ is better than the existing route $\boldsymbol{r}_j$, it may not be the best route in local path space $\boldsymbol{L}^P\left(\boldsymbol{r}_j^{(s_j)}\right)$. It may be desirable to use the best route, denoted as $\boldsymbol{r}_j^{(b2)}$, in the local path space $\boldsymbol{L}^P\left(\boldsymbol{r}_j^{(s_j)}\right)$ to replace the existing route $\boldsymbol{r}_j$. This is the basic idea of the **HC** method. The **HC** method may be more computational expensive comparing with greedy type methods since to find the best route $\boldsymbol{r}_j^{(b2)}$ it needs to evaluate all the paths in the local path space $\boldsymbol{L}^P\left(\boldsymbol{r}_j^{(s_j)}\right)$. For the example described in sections 3.8.1 and 3.8.2, the local path space $\boldsymbol{L}^P\left(\boldsymbol{r}_j^{(s_j)}\right)$ (where $s_j = 3$) contains 125 paths. In general, **HC** methods should produce better results than those obtained from the two greedy type methods **GS1** or **GS2**. The following is the search procedure of the **HC** method.

Steps (1) through (4) are the same as those for **GS1**.

(5) Replace route $\boldsymbol{r}_j$ in the existing transit network $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_l\}$ with path $\boldsymbol{r}_{jk} \in \boldsymbol{L}^P(\boldsymbol{r}_j^{(3)})$ to obtain an updated transit network $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{j-1}, \boldsymbol{r}_{jk}, \boldsymbol{r}_{j+1}, \ldots, \boldsymbol{r}_l\}$, and perform function evaluation of the resulting transit route network. Repeat the above process 125 times to obtain the best route $\boldsymbol{r}_j^{(b2)}$ in this step, i.e., the objective function value associated with path $\boldsymbol{r}_j^{(b2)}$ is better than those associated with routes $\boldsymbol{r}_j$ and $\boldsymbol{r}_{jk}$ (k = 1, 2, …, 125). Replace $\boldsymbol{r}_j$ with $\boldsymbol{r}_j^{(b2)}$ to obtain an updated transit route network $\boldsymbol{T}^{(l)}\{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_{j-1}, \boldsymbol{r}_j^{(b2)}, \boldsymbol{r}_{j+1}, \ldots, \boldsymbol{r}_l\}$ (j = 1, 2, …, l), and go to Step (6).

(6) Select the next route from the transit route network, e.g., route $\boldsymbol{r}_{(j+1)}$, and go to Step (1) to start a new local search process for route $\boldsymbol{r}_{(j+1)}$.

(7) The search process will be considered converged if no better results can be found from all the individual route's local path search spaces.

The number of function evaluations required from **HC** method may be estimate as follows. Assume that the average length of a nodal adjacency list in a street network is $\overline{m}$ = 4, and the average number of nodes in a local node space of such street network will be $\overline{m}$ + 1 = 5. For a search subspace $\boldsymbol{L}_s^T$ generated from one master path with key-node number s, the search subspace $\boldsymbol{L}_s^T$ will have a solution population of $(\overline{m} + 1)^5 = 5^5$. For transit route network of l routes, the number of function evaluations corresponding to an iteration loop that includes the exhaust search of all the l local subspaces associated with the l master routes will be$(\overline{m} + 1)^5 \times l = 5^5 \times l$. For r = 5 and l = 80, reasonable numbers for medium or large transit network systems, the number of function evaluations during an iteration loop will be $(\overline{m} + 1)^5 \times l = 5^5 \times 80 =$ 250,000, i.e., the evaluation of object function defined in [3.34] for zero-transfer optimization, or [3.38] for one-or-less transfer optimization, or [3.42] for two-or-less transfer optimization must be performed 250,000 times for one iteration. According to results from tests conducted in this study, an optimization process will typically converge between 10 to 100 iterations.

3.8.4 Fast Hill Climb Method (FHC)

The **FHC** method is an extension of the **HC** method described in the previous section. The differences between the **FHC** method and the **HC** method are the following. In the search process of a typical route $\boldsymbol{r}_j$, the **HC** method will replace the existing route $\boldsymbol{r}_j$ with the best route alternative $\boldsymbol{r}_j^{(b2)}$ obtained from the local path space $\boldsymbol{L}^P\left(\boldsymbol{r}_j^{(s_j)}\right)$, and then start the search process for the next transit route. In the **FHC** method, however, the existing route $\boldsymbol{r}_j$ will not be replaced by the best route alternative $\boldsymbol{r}_j^{(b2)}$ in local path space $\boldsymbol{L}^P\left(\boldsymbol{r}_j^{(s_j)}\right)$. Instead, the best route alternative $\boldsymbol{r}_j^{(b2)}$ for route $\boldsymbol{r}_j$ and the associated objective function value f_j (or values for multi-objective optimization problems) will be saved, and the search process for the next transit route, e.g., $\boldsymbol{r}_{(j+1)}$, will start with the same route $\boldsymbol{r}_j$ in the route network. The local search process will perform l times for all the l transit routes in a transit route network to obtain a set of the *best* alternative routes $\boldsymbol{r}_j^{(b2)}$ (j = 1, 2, …, l), and a set of objective function values f_j corresponding to transit route networks $\boldsymbol{T}^{(l)}\{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_{j-1}, \boldsymbol{r}_j^{(b2)}, \boldsymbol{r}_{j+1}, \ldots, \boldsymbol{r}_{l-1}, \boldsymbol{r}_l\}$ (j = 1, 2, …, l). Finally, the transit route network consisting of all the *best* routes $\boldsymbol{T}^{(l)}\{\boldsymbol{r}_1^{(b2)}, \boldsymbol{r}_2^{(b2)}, \ldots, \boldsymbol{r}_l^{(b2)}\}$ will be evaluated to obtain the objective function f_{l+1} (or values for a multi-objective optimization problem). The best transit route network among the l+1 route networks will be the one that corresponds to the best objective function. Conceptually, the **FHC** method is similar to *the deepest decent search*

method in continuous research fields. For transit service areas where the transit demand distributions have certain continuity characteristics or patterns, the **FHC** method should be able to give better results than those obtained from the other methods discussed in this section. The following is the search procedure of the **FHC** method.

Steps (1) through (4) are the same as those for **GS1**.

(5) Replace route $\boldsymbol{r_j}$ in the existing transit network $\boldsymbol{T}^{(l)}\{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r_l}\}$ with path $\boldsymbol{r_{jk}} \in \boldsymbol{L}^P(\boldsymbol{r_j}^{(3)})$ to obtain an updated transit network $\boldsymbol{T}^{(l)}\{\boldsymbol{r}_1, \ldots, \boldsymbol{r_{j-1}}, \boldsymbol{r_j}, \boldsymbol{r_{j+1}}, \ldots, \boldsymbol{r_l}\}$, and perform function evaluation of the resultant transit route network. Repeat the above process 125 times to obtain the best route $\boldsymbol{r_j}^{(b2)}$ in the step, i.e., the objective function value associated with path $\boldsymbol{r_j}^{(b2)}$ is better than those associated with routes $\boldsymbol{r_j}$ and $\boldsymbol{r_{jk}}$ (k = 1, 2, …, 125). Save the best route $\boldsymbol{r_j}^{(b2)}$ and the corresponding function value f_j in the memory, and go to Step (6) with the original transit network of this step, $\boldsymbol{T}^{(l)}\{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r_l}\}$.

(6) If all the l routes in this transit network have been evaluated, i.e., all the $\boldsymbol{r_j}^{(b2)}$ and f_j (j = 1, 2, …, l) have been obtained, go to Step (7). Otherwise, select the next route from the transit route network, e.g., route $\boldsymbol{r_{j+1}}$, and go to Step (1) to start a new local search process for route $\boldsymbol{r_{j+1}}$.

(7) Perform function evaluation of the resulting transit network $\boldsymbol{T}^{(l)}\{\boldsymbol{r}_1^{(b2)}, \boldsymbol{r}_2^{(b2)}, \ldots, \boldsymbol{r_l}^{(b2)}\}$ to obtain the corresponding function value f_{l+1}. Compare f_{l+1} and f_j, corresponding to network $\boldsymbol{T}^{(l)}\{\boldsymbol{r}_1, \ldots, \boldsymbol{r_{j-1}}, \boldsymbol{r_j}^{(b2)}, \boldsymbol{r_{j+1}}, \ldots, \boldsymbol{r_l}\}$ (j = 1, 2, …, l+1), and select the transit network corresponding to the best function value to replace the original network.

(8) The search process will be considered converged if the function values f_j (j = 1, 2, …, l+1) are no better than the function value f associated with the original transit network $\boldsymbol{T}^{(l)}\{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r_l}\}$ in this iteration.

3.8.5 Integrated Simulated Annealing, Tabu and Greedy Method (ISTG)

In the description of the two greedy type search methods and the two hill climbing search methods, issues regarding the global-ness of the search results from these methods have not been addressed. In fact, all these methods are local search methods, although the two hill climbing methods may produce better results due to larger search spaces. For non-convex optimization system, a local search scheme may suffer from premature convergence to or being trapped into one of the local optimal results near initial guess route network. To avoid premature convergence to poor local optima, one way is to enlarge local search spaces by increase s, the key-node representation number described in section 3.5; or i the order of local space number defined in [3.16] and [3.17]; or k, the number that defined the k-level shortest path space in expressions [3.18] and [3.19]. All these remedies may increase the search space size significantly, thus may require considerable more computing resources. Another way is to include escape schemes in the solution search process to prevent the search process from trapping into local optimal valleys. The integrated simulated annealing, tabu, and greedy search method (**ISTG**) developed in this study combines the greedy search method described in section 3.8.1 with a simulated annealing and tabu search scheme that is capable to escape from any local optima, and theoretically, will visit a global optima eventually with a probability of 1.0.

3.8.6 Tabu Search and Simulated Annealing

The basic idea of escaping from a poor local optimum in the tabu search algorithm is to accept occasionally worse solutions from the local search space (Glover et al. 1993). One difficulty with this idea is that cycling may occur, i.e., the search process may repeat the same sequence of solutions indefinitely. To alleviate this problem, a tabu list is established to keep track of solutions evaluated recently and prevent a list of recent incumbents from reentering the new solution search process. The simulated annealing search scheme is a stochastic process, also aimed to avoid getting trapped into poor local optima. Unlike traditional deterministic search methods where difficult theoretical issues such as convexity, continuity, uniqueness of the optimization system must be addressed, being able to find global optimal results through a stochastic process is not a difficult issue in itself. In fact, under fairly general conditions, one can always expect to find a global solution from solutions obtained by randomly selected initial guess networks. For example, assume that a network system has m local optimal solutions and that one of them is the global optimal solution. Further assume that the chance to converge to this global optimal solution from a randomly selected initial network is $1/m$, then from probability theory, the probability to obtain the global optimal solution from k solution search processes with randomly selected initial networks will be

$$p = 1 - \left(\frac{m-1}{m} \right)^k . \qquad [3.52]$$

It may be seen from [3.52] that as the number of search processes k increases, a global optimal result will be eventually visited (with probability 1) although the converging process may be slow or computationally intractable for problems with large m. The simulated annealing method is a stochastic search process that aims to find a global optimal result faster than these unsophisticated random search methods. The basic idea is inspired by the annealing process in solids where slower cooling temperature process from liquid to solid states usually leads to a lower energy state (Kirkpatrick, 1983). Mathematically, such process can be simulated as a stochastic search process where a local search space will replace an incumbent with probability 1 if it contains solutions with better goal values, and with some probability between 0 and 1 for solutions with worse goal values. The probability to accept a worse solution is proportional to the difference in the goal value, slightly worse solutions have a higher probability of being accepted while much worse solutions have fewer chances to be accepted. In the long run, as the number of search iterations becomes sufficiently large, the search process can escape away from any local optimal, and eventually should visit a global optimal solution.

The basic procedure for a TRN design optimization problem with the boarding functions defined in [3.46] or [3.47] as the objective function begins with a given TRN $\boldsymbol{T}_0$. A solution candidate TRN $\boldsymbol{T}$ is then selected from a local network space $\boldsymbol{T}_{(i)}^{(k)}(\boldsymbol{T}_0)$ defined in [3.20]. The network $\boldsymbol{T}$ is accepted if the associated objective function is improving, that is $t(\boldsymbol{T}, \boldsymbol{O}) < t(\boldsymbol{T}_0, \boldsymbol{O})$, where t is a boarding function defined in [3.46] or [3.47]. Otherwise, $\boldsymbol{T}$ is accepted with probability

$$p = e^{-\Delta/\theta}, \Delta = [\, t(\boldsymbol{T}_0,\boldsymbol{O}) - t(\boldsymbol{T},\boldsymbol{O})] \qquad [3.53]$$

where θ is a positive number representing current temperature[6] of the annealing process. The temperature θ regulates the likelihood of accepting solutions with worse goal values. A higher temperature θ results in a larger probability of accepting worse solutions, while a smaller θ reduces the chances to accept worse solutions. In practice, it is typical to start with a high initial temperature to allow a better chance of escaping from poor local minima, and gradually reduce θ to enhance the selectivity of the search towards improved solutions. The main strategies or features of the **ISTG** method developed in this study are described below.

3.8.6.1 Divide-and-Conquer Heuristic

There are two levels of annealing search processes in the proposed **ISTG** method, one at the network level, and the other at the route local search space level. Directly searching at the network level means searching solutions from a huge combinatorial space and the number of sample evaluations (large enough to suppress responses due to noises) to expose any meaningful characteristics (such as local minimum etc.) from a solution population may be computationally intractable. Dividing a problem into sub-problems of manageable sizes is a common heuristic called "divide-and-conquer" in operations research fields. Parameters and criteria involved in the two levels of search processes include the following:

(1) The initial temperature θ_n, the cooling temperature reduction factor τ_n ($0 < \tau_n \leq 1$), the maximum number of search iterations allowed L_n, and the maximum CPU time allowed for solution search process at network level;
(2) The initial temperature θ_r , the cooling temperature reduction factor τ_r ($0 < \tau_r \leq 1$), and the maximum number of search iterations allowed L_r at route level;
(3) Termination or convergence criterion, i.e., the definition of the "frozen" states in an annealing search process either at the network search level, or at the route search level. The termination condition used in the **ISTG** search method is defined through a probability tolerance P (P_r for route level search, and P_n for network level search) for repeated occurrence of the same goal values during a search process. For example, if the current search result has the same goal value as the previous one, for $P = 0.9$, a search process will be continued with probability $P = 0.9$, or be terminated with the complementary probability $1 - P = 0.1$. In practice, large P values usually lead to lengthy computer running time before convergence, while small P values may result in the search processes failing to escape from local minima located in a flat valley or plateau.

The above parameters and criteria will be further explained through a simple example.

3.8.6.2 Communication of Various Local Solution Spaces

This is an important condition to ensure that a global optimum will be eventually visited with probability 1 in a simulated annealing search process (Hajek 1988). In particular, any local search space must be generated such that it allows a search process to migrate from any local spaces to any other solution spaces. Failing to meet this condition may prevent some solutions from entering the search process, thus failing to arrive at globally optimal results. This condition

[6] For TRN optimization, temperature is a borrowed word from annealing in solids, and has no physical meanings.

is fairly general and easy to meet for most problems in TRN design practice. The locally and iteratively defined local network and path search spaces $\boldsymbol{T}_{(i)}^{(k)}(\boldsymbol{T})$ and $\boldsymbol{P}_{(i)}^{(k)}(\boldsymbol{r})$ defined in expressions [3.20b] and [3.20a] will satisfy this condition if the street network dose not have isolated nodes or sub-networks.

3.8.6.3 Diversification of Search Areas

Diversification of a search process is intended to cover more promising solution spaces in effective and manageable manners. The following are some of the options in **ISTG** search method to enhance the diversity of the method:

(1) Increase the coverage of a local search space by increasing:
   (a) the numbers of key-nodes used to represent master paths (refer to equation [3.22]);
   (b) the level number of the *k*-level shortest path space $\boldsymbol{P}_S^{(k)}$ used to generate local path spaces near master paths;
   (c) the order numbers of local node spaces that define the sizes of master nodes' local node spaces. All the above options result in large local search spaces thus are limited by the computing resources.
(2) Using local search spaces generated from more than one routes will also increase the coverage of local spaces although the resultant local spaces may suffer from combinatorial explosion as the number of routes involved in the local spaces increases.
(3) Applying a high initial temperature θ, or a slow cooling process (by assigning temperature reduction factor τ closed to 1) will also result in diversification.
(4) Using longer tabu queue/list will prevent the search process from cycling the same solution search sequences thus encouraging diversification. Numerical experiments conducted in this study have shown that although simulated annealing algorithm by itself can eventually escape from poor local minima, without the tabu ("forbidden") list to prevent the same solutions from entering the search process, escaping from local minima by random chances may be extremely time assuming, sometimes more than 95% percent of the total solution time.
(5) Increasing N_r, the number of random selected solutions from a global path space. During local search process, **ISTG** allows random selection of N_r solutions outside current local spaces to enhance search diversification.

3.8.6.4 Intelligent Intervention

It has been point out by Fox (1993) that simulated annealing theory does not in principle exclude useful intelligent interventions to optimize the random search process or meet particular requirement in practice, mostly the speed of convergence with available computer sources. Intelligent interventions included in the **ISTG** method are:

(1) temporal and spatial memory of recent samples to avoid cycling of solution search sequence resulted from the incorporation of the tabu queue/list into the annealing search process;

(2) enhancement of communication among local solution spaces by random inclusion of solutions outside a local space into the local search process; and
(3) incorporation of greedy type search strategy into route level local search process.

As mentioned before, each search process or stage in **ISTG** involves only one or a few routes (divide-and-conquer) for the reason of computing resource limitation. At such a route level solution search process, a blind annealing search should eventually converge to the best route in the corresponding local space with probability 1. However, due to requirement of convergence speed, which is usually as important as to find a good optimal result in practice, earlier termination of route level annealing search may be necessary, and the resultant route/routes at such a termination point may not be the best. The greedy type intervention in **ISTG** will replace those termination point results with the best ones visited during the local search process of these route/routes before entering search process of the next route (or routes).

The above intervention will certainly have effects on the random mechanism of the simulated annealing search, thus may have effects on the outcome of the search process, in particular the global-ness of the result. However, numerical experiments conducted in this study have shown that with the same computing resources (CPU time, memory size, etc.), results obtained from **ISTG** search with intelligent intervention are usually much better than those without any treatments. In fact, intervention or treatment to speed up convergence in simulated annealing search seems to be a "must" step for this algorithm since the provable rate of convergence to a global optimum thus far has exponential time complexity, which is inconsistent with many successful applications of this method in practice.

3.8.6.5 ISTG Search Procedure

The following is a brief description of the **ISTG** search procedure developed in this study through a simple example. The route level local search for this simple example involves one route at time. The procedure for route level local search involving more than one route is similar. Assume that during a solution search process, an intermediate TRN result $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots \boldsymbol{r}_l\}$ has been obtained, the current network level annealing temperature is θ_n, and the cooling temperature reduction factor is τ_n. The solution search procedure for the next stage of **ISTG** method involves the following steps:

(1) Select a transit route, e.g., route $\boldsymbol{r}_j$ from $\boldsymbol{T}^{(l)}$. Initialize a local tabu list/queue of length L_t, and set loop number counter $L_r = 0$. Set an initial route level temperature $\theta_r = \theta_n$ and a cooling temperature reduction factor τ_r.
(2) Select key-nodes for route $\boldsymbol{r}_j$. For illustration, assume the three-node representation of route $\boldsymbol{r}_j$ (see Figure 3.6(a)) is used, denoted as $\boldsymbol{r}_j^{(3)}$.
(3) From the three key-nodes n_1, n_2, and n_3 generate three corresponding local node spaces, i.e., $\boldsymbol{L}_{(1)}^{N}(n_1) = \{n_1, n_{11}, n_{12}, n_{13}, n_{14}\}$, $\boldsymbol{L}_{(1)}^{N}(n_2) = \{n_2, n_{21}, n_{22}, n_{23}, n_{24}\}$, and $\boldsymbol{L}_{(1)}^{N}(n_3) = \{n_3, n_{31}, n_{32}, n_{33}, n_{34}\}$. There are 5 nodes in each of the three local node spaces in this example.
(4) Connect nodes in node spaces $\boldsymbol{L}_{(1)}^{N}(n_1)$ and $\boldsymbol{L}_{(1)}^{N}(n_3)$ with the shortest paths in space $\boldsymbol{P}_S^{(1)}$ to obtain $5 \times 5 = 25$ shortest path segments. Then extend these shortest path segments to

nodes in node space $\boldsymbol{L}_{(1)}^{N}(n_2)$ with shortest paths to obtain 25 × 5 = 125 paths. These 125 paths form the local path space of route $\boldsymbol{r}_j$ based on three-node representation $\boldsymbol{P}_{(1)}^{(1)}\left(\boldsymbol{r}_j^{(3)}\right)$.

(5) Selected a path $\boldsymbol{r}_{jk} \in \boldsymbol{P}_{(1)}^{(1)}\left(\boldsymbol{r}_j^{(3)}\right)$. If the path space $\boldsymbol{P}_{(1)}^{(1)}\left(\boldsymbol{r}_j^{(3)}\right)$ is large, choose $\boldsymbol{r}_{jk}$ at random, otherwise select $\boldsymbol{r}_{jk}$ sequentially from space $\boldsymbol{P}_{(1)}^{(1)}\left(\boldsymbol{r}_j^{(3)}\right)$. To enhance diversification, some of the paths may be chosen randomly and occasionally from a global path space. If $\boldsymbol{r}_{jk}$ is already in the tabu list/queue, repeat the selection process of this step. Otherwise, add $\boldsymbol{r}_{jk}$ into the tabu list, and go to Step (6).

(6) Replace route $\boldsymbol{r}_j$ in the existing TRN $\boldsymbol{T}^{(l)}$ with path $\boldsymbol{r}_{jk}$ to obtain $\boldsymbol{T}^{(l)} = \{\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{j-1}, \boldsymbol{r}_{jk}, \boldsymbol{r}_{j+1}, \ldots, \boldsymbol{r}_l\}$, and perform objective function evaluation. If a better result is obtained, replace $\boldsymbol{r}_j$ with $\boldsymbol{r}_{jk}$, reduce temperature by setting $\theta_r \leftarrow \tau_r\ \theta_r$, update the route level loop counter $L_r \leftarrow L_r + 1$, and go to Step (2) to start a new search with reduced temperature. If the result has the same goal value as the previous one, route $\boldsymbol{r}_{jk}$ is accepted with probability p defined in equation [3.53]. If route $\boldsymbol{r}_{jk}$ is accepted, replace $\boldsymbol{r}_j$ with $\boldsymbol{r}_{jk}$, set $\theta_r \leftarrow \tau_r\ \theta_r$, $L_r \leftarrow L_r + 1$ and go to Step (2). If route $\boldsymbol{r}_{jk}$ is rejected, go to Step (7).

(7) Check termination conditions for route level search. Conditions used in this study include (a) maximum number of loops allowed at route level search; and (b) probability tolerance for results with repeated goal value. If a route level search termination condition is reached, go to step (8), otherwise go to step (5).

(8) If local path spaces of all the l routes in TRN $\boldsymbol{T}^{(l)}$ have been searched, begin a new search iteration at network level by setting $\theta_n \leftarrow \tau_n\ \theta_n$, and go to Step (1). Otherwise, select the next route from the transit route network, e.g., route $\boldsymbol{r}_{(j+1)}$, and go to Step (1) to begin a new route level local search for route $\boldsymbol{r}_{(j+1)}$.

The search process will be considered converged and will be terminated if any of the following conditions are met at network solution search level: (a) maximum number of search iterations allowed at network level is reached; (b) maximum CPU time allowed at network level is reached; (c) terminated with the complementary probability 1 - P_n due to the repeated occurrence of network results with the same goal value.

4. NUMERICAL EXPERIMENTS

The methodology for transit route network design optimization developed from this study has been tested through several benchmark problems studied or developed by Mandl (1979, 1979a), Shih and Mahmassani (1994), and Baaj and Mahmassani (1991). The selection of these benchmark problems is based on the following considerations. Firstly, these problems have been well documented with detailed descriptions of street network input data, OD matrix input data and, most importantly, transit route network results and the corresponding objective function values, which can be used for comparison with results from this study. Secondly, methodologies developed by these authors and the corresponding numerical results seem to be well acknowledged in the transit planning research community. Finally, these benchmark problems are real, practical problems, although with some simplification assumptions, making them appropriate for use as benchmark problems to evaluate results from this study. It needs to be emphasized that the comparison between the results from this study and those from the benchmark problems was merely to validate the methodology developed in this study. In fact, the design objectives and design variables involved in this study are different from those used in studies by Mandl (1979, 1979a), Shih and Mahmassani (1994), and Baaj and Mahmassani (1991). In this study, the objective functions are those described in Section 3.6, which reflect a transit system's network directness and transfer directness, and the design variables are integer vectors representing transit routes. The benchmark problems are TN optimization problems, which include both TRN design optimization and certain parts of TNS (transit network scheduling) design optimization problems, and the design objectives are, in addition to various network directness functions, user or operator costs, such as transfer times, waiting times, etc. A more comprehensive comparison of the methods developed in this study with those benchmark problems may be appropriate after work on transit scheduling optimization is completed. Presently, the comparison involves only those design variables and results related to or obtained from TRN optimization. The numerical tests are performed based the Mandl's problem (1979, 1979a) and the Miami-Dade County transit system. The test problems and results are described in the following sections.

4.1 Mandl's Transit Network Problem

Mandl's problem has been discussed in Chapter 2. As shown in Figure 2.2, it consists of a small and dense network of 15 nodes with a total demand of 15,570 trips per day. Each transit node is labeled with an integer and each street segment with the in-vehicle travel time in minutes. The transit demand or OD matrix for the 15 street nodes is also illustrated in Figure 2.2 and provides the average number of passenger trips per day between each transit node pair. Figure 2.3 shows Mandl's final solution route network.

Mandl's problem has been used by Baaj and Mahmassani (1991) and Shih and Mahmassani (1994) as benchmark problems to test their solution frameworks and transit network optimization results. Baaj and Mahmassani (1991) compared three route network layouts that they obtained from their TN optimization processes with three different sets of design parameters. All three showed improvements over Mandl's earlier network results. Shih and Mahmassani (1994) further extended Baaj and Mahmassani (1991)'s work by introducing transit center methodology. Using Mandl's problem as test benchmarks, Shih and Mahmassani (1994) also obtained three

route network layouts from their solution methodology based on three sets of design parameters. The resultant networks demonstrated further improvements over the earlier work by Mandl, and Baaj and Mahmassani. The network layouts produced by Mandl, Baaj and Mahmassani, and Shih and Mahmassani are illustrated in Figures 2.3, 2.5, and 2.4, respectively.

4.1.1 Numerical Experiment 1

The results from this study are given in Table 4.1 along with those from previous authors' work. The comparison is made based on variables that characterize the quality of transit route network configurations, such as the percentages of zero-, one-, or two-transfer trips, route network directness, transfer directness, total route length, etc. The comparisons are limited to those objective function values and design variables that reflect the quality or characteristics of the transit route networks. In Table 4.1, the first row lists the source of the benchmark problems. The second row identifies the solutions to the benchmark problems. While all problems were originated from Mandl's network and demand, they differed in design parameters such as number of routes and total route length. Baaj and Mahmassani produced three solutions for the Mandl's problem, and so did Shih and Mahmassani, with different design parameters or methods. The methods that were used to obtain the results are indicated in the third row, where Mandl means Mandl's method, B&M stands for Baaj and Mahmassani's methods, S&M for Shih and Mahmassani's methods, and **FHC** indicates results obtained from the fast hill climbing solution search method described in Sections 3.8.3. The solutions given here were generated with multiple objective functions $f((0, 1, 2))$, as described in Section 3.7.4. Local search spaces were generated from the three key-node representation. For each solution, the left column provides the statistics for the layout produced in the original study as indicated by the source of the problem, and the right (shaded) column provides the statistics for the results.

Table 4.1 Result Comparison – FHC Method (Existing Network as Initial Guess)

Problem Source	Mandl		Baaj & Mahmassani						Shih & Mahmassani					
Route layout case number	1		1		2		3		1		2		3	
Search Method	Mandl	**FHC**	B&M	**FHC**	B&M	**FHC**	B&M	**FHC**	S&M	**FHC**	S&M	**FHC**	S&M	**FHC**
zero-transfer trips (%)	69.94	76.43	78.6	82.34	79.96	86.64	80.99	82.98	82.59	84.84	87.73	91.78	82.59	89.92
one-transfer trips (%)	29.93	23.57	21.39	17.66	20.04	13.36	19.01	17.02	17.41	15.16	12.27	8.22	17.41	10.08
two-transfer trips (%)	0.13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Uncovered trips (%)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total route length	82	82	126	125	144	144	106	105	124	124	151	152	124	127
Number of routes	4	4	6	6	8	8	7	7	6	6	8	8	6	6
Network directness	1.05	1.05	1.06	1.01	1.01	1.00	1.16	1.09	1.04	1.06	1.03	1.00	1.04	1.06
Transfer directness (t_2)	1.30	1.24	1.21	1.18	1.2	1.13	1.19	1.17	1.17	1.15	1.12	1.08	1.17	1.1
Trips/Length	190	190	124	125	108	108	147	148	126	126	103	102	126	123

The statistics shown in Table 4.1 include percentages of transit trips that could be accomplished with zero transfer (zero-transfer trips), one transfer (one-transfer trips), or two transfers (two-transfer trips). The percentage of trips that were not served by the designed network is also given for each solution. From the table it may be seen that the percentages of zero transfer trips were higher for all solutions produced in this study. Route network performance was also measured by total route length, number of routes, network directness, transfer directness, and trips per unit route length (mile or minute). All the testing problems used in this study were designed to have the same numbers of transit routes and similar total route lengths as the original problems. Recall that the optimal values for network directness ($d^R(\boldsymbol{T}^{(l)})$) and transfer directness[7] are both 1.0. It may be seen that all the results from both the previous work or this study had excellent network directness and in majority of the problems, the results from this study showed slight improvements. The transfer directness, which represents the average number of boardings, from this study again showed improvements. The number of covered trips per unit route length, or trip-length ratio is defined as,

$$R_{T/L}^{(i)} = \frac{f_i}{l_T}, \quad (i = 0, 1, 2, c) \qquad [5.1]$$

where f_0, f_1, and f_2 are, respectively, the zero-, one-or-less, and two-or-less transfer functions defined in Chapter 3; f_c is the number of total covered trips; and l_T is the total length of the route network system. Therefore, $R_{T/L}^{(0)}$, $R_{T/L}^{(1)}$, and $R_{T/L}^{(2)}$ in [5.1] are the trip-length ratios corresponding to the zero, one-or-less, and two-or-less transfer trips while $R_{T/L}^{(c)}$ is the trip-length ratio corresponding to total covered trips. In general, a larger value of $R_{T/L}^{(c)}$ indicate a more effective transit route network layout. However, the ultimate effectiveness of a transit network must also include the outcomes of transit scheduling design and measured by the ratios of number of covered trips to transit revenue hours and revenue miles, which are affected by service frequency setting.

4.1.2 Numerical Experiment 2

Results from this study as shown in Table 4.1 were obtained by using the original solutions of the benchmark problems as the initial guess networks. This means that the optimization methods developed in this study may be used to improve or evaluate an existing route network. For a more objective comparison without any possible biases introduced by using the original solutions to the benchmark problems as the starting point of the search, results were also produced by using initial networks automatically generated by a computer program, OPTNet, which will be describe in Chapter 5. The only requirements in generating these results were that they must have the same number of routes as the original solutions, and the total network route lengths should not be more than 5% different from those of the original solutions. The results are given in Table 4.2, which in general, had performance comparable to those shown in Table 4.1.

[7] In these cases, transfer directness equals to boarding function t_2 since all OD trips are covered by two or less transfer trips.

Table 4.2 Result Comparison – FHC Method (Program Generated Initial Networks)

Problem Source	Mandl's Problem		Baaj & Mahmassani's Problem						Shih & Mahmassani's Problem					
Route layout case number	1		1		2		3		1		2		3	
Search Results	Mandl	**FHC**	B&M	**FHC**	B&M	**FHC**	B&M	**FHC**	S&M	**FHC**	S&M	**FHC**	S&M	**FHC**
zero-transfer trips (%)	69.94	72.25	78.6	85.68	79.96	89.27	80.99	84.20	82.59	87.67	87.73	89.08	82.59	87.12
one-transfer trips (%)	29.93	26.98	21.39	14.32	20.04	10.73	19.01	14.26	17.41	12.33	12.27	10.21	17.41	12.85
two-transfer trips (%)	0.13	0.00	0.00	0.00	0.00	0.00	0.00	1.54	0.00	0.00	0.00	0.00	0.00	0.00
Uncovered trips (%)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total route length	82	82	126	124	144	144	106	105	124	124	151	152	124	127
Number of routes	4	4	6	6	8	8	7	7	6	6	8	8	6	6
Network directness	1.05	1.06	1.06	1.02	1.01	1.00	1.16	1.03	1.04	1.04	1.03	1.04	1.04	1.04
Transfer directness (t_2)	1.30	1.29	1.21	1.14	1.2	1.12	1.19	1.17	1.17	1.12	1.12	1.12	1.17	1.13
Trips/Length	190	190	124	126	108	108	147	148	126	126	103	102	126	123

Table 4.3 provides the detailed data on the route network layouts from the original solutions and solutions produced in this study. In the table, Result 1 means results described in Table 4.1 and Result 2 means results described in Table 4.2.

Table 4.3 Detailed Route Network Structure Layouts from Previous and Current Studies

Problem Source	Route Layout Cases	Results	Route Network Layouts
Mandl	Case 1	Mandl	(0,1,2,5,7,9,10,12), (4,3,5,7,14,6), (11,3,5,14,8), (12,13,9)
		Result 1	(0,1,2,5,7,9,10,12,13), (4,3,5,7,14,6), (11,3,5,7,14,8), (6,9)
		Result 2	(3,1,2,5,7,9,10,12), (0,1,2,5,14,6), (4,3,5,14,8), (11,10,12,13)
B&M	Case 1	B&M	(6,1,4,7,9,10,11), (6,14,5,7,9,12,13), (0,1,2,5,7), (8,14,6,9), (4,3,5,7,9), (0,1,2,5,14,8)
		Result 1	(6,14,7,9,10,11), (14,5,7,9,10,12,13), (2,1,3,5,14,6), (8,14,6,9), (4,3,5,7,9), (0,1,2,5,14,8)
		Result 2	(3,1,2,5,7,9,10), (0,1,2,5,14,6), (4,1,2,5,14,8), (9,10,11), (6,14,5,2,1,3,11), (13,12,10,9,6,14,7)
	Case 2	B&M	(0,1,3,11,10,12,13), (2,5,7,14,6,9), (9,10,12), (9,10,11), (7,9,13), (0,1,3,5), (8,14,5,7,9), (4,1,2,5,14,6,9)
		Result 1	(0,1,3,11,10,12,13), (2,5,7,14,6,9), (9,10,12), (9,10,11), (7,9,13), (0,1,2,5,7), (8,14,5,7,9), (4,3,5,7,9,10)
		Result 2	(1,2,5,7,9,10,11,12), (8,14,6,9), (0,1,2,5,7,14,6), (9,12,13), (4,3,5,14,6,9), (9,10,11), (4,1,2,5,14,6), (7,5,2,1,3,11)
	Case 3	B&M	(9,1,2), (9,10,11), (9,3), (0,1,2,5,7,9), (8,1,4,6,9), (4,3,5,7,9), (0,1,3,4)
		Result 1	(10,12,13), (9,10,11), (9,13,12), (0,1,2,5,7,9), (8,14,6,9), (4,3,5,7,9), (0,1,3,4)
		Result 2	(4,3,1,2,5,7,9), (8,14,6,9), (0,1,2,5,7), (9,12,13), (4,3,5,14,6), (9,10,11), (1,2,5,7,14,6)
S&M	Case 1	S&M	(5,7,9,10,12,13), (6,1,4,7,9,10,11), (6,9,12), (0,1,2,5,7,9), (8,1,4,6,9), (4,3,5,7,9)
		Result 1	(5,7,9,10,12,13), (10,9,7,5,3,11), (6,9,12), (0,1,2,5,7,9), (8,14,6,9), (4,1,2,5,14,6)
		Result 2	(0,1,2,5,7,9), (4,3,5,7,9,10), (13,12,10,9,6,14,7), (8,14,6,9,10), (0,1,2,5,14,6), (9,10,11)
	Case 2	S&M	(2,5,14,6,9,10), (1,2,5,7,14,6,9,10), (9,13,12), (0,1,3,5), (9,10,11), (8,14,6,9), (4,3,5,7,9), (0,1,2,5,7,9,12)
		Result 1	(7,5,2,1,3,11), (3,1,2,5,14,6,9,10), (10,12,13), (0,1,3,5), (9,10,11), (8,14,6,9), (4,3,5,7,9), (0,1,2,5,7,9,10,12,13)
		Result 2	(1,2,5,7,9,10), (8,14,6,9), (0,1,2,5,7), (5,7,14,6,9,12), (4,3,5,7,9,10), (9,10,11), (3,4,1,2,5,14,6), (6,9,10,12,13)
	Case 3	S&M	(5,7,9,10,12,13), (6,1,4,7,9,10,11), (6,9,12), (0,1,2,5,7,9), (8,1,4,6,9), (4,3,5,7,9)
		Result 1	(4,3,5,7,9,10), (12,9,10,11), (6,9,12,13), (0,1,2,5,7,9), (8,14,6,9), (3,4,1,2,5,14,6)
		Result 2	(0,1,2,5,7,9), (4,3,5,7,9,10), (12,10,9,6,14,7), (8,14,6,9,13,12), (0,1,2,5,14,6), (9,10,11)

4.1.3 Numerical Experiments 3 and 4

The last numerical experiment involved the application of the **ISTG** search method to Mandl's problems as defined by Mandl (1979, 1979a), Baaj and Mahmassani (1991), and Shih and Mahmassani (1994). The objective function for all the **ISTG** test problems was the average vehicle boarding function t_1 defined in equation [3.47]. The penalty boarding number for uncovered trips, α, was taken as four, i.e., each trip that could not be accomplished with one or less transfer would be considered as four vehicle boardings. All the test results were obtained with the two key-node representation. The purpose of using the least number of key-node representations was to demonstrate that by introducing the escaping schemes from local optima, the **ISTG** method was capable to produce comparable results obtained from the **FHC** method with higher number of key-node representation.

Table 4.4 Result Comparison – ISTG Method (Exiting Network as Initial Guess)

Problem Source	Mandl		Baaj & Mahmassani						Shih & Mahmassani			
Route layout cases	1		1		2		3		1		2	
Search Method	Mandl	**ISTG**	B&M	**ISTG**	B&M	**ISTG**	B&M	**ISTG**	S&M	**ISTG**	S&M	**ISTG**
0-transfer trips (%)	69.94	76.11	78.61	83.11	79.96	88.83	80.99	82.21	82.59	87.35	87.73	91.01
1-transfer trips (%)	29.93	23.89	21.39	16.89	20.04	11.17	19.01	17.79	17.41	12.52	12.27	8.99
2-transfer trips (%)	0.13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.13	0.00	0.00
Total route length	82	83	126	126	144	144	106	106	124	124	151	151
Number of routes	4	4	6	6	8	8	7	7	6	6	8	8
Boarding function t_1	*1.303*	*1.239*	*1.214*	*1.169*	*1.200*	*1.112*	*1.190*	*1.178*	*1.174*	*1.125*	*1.123*	*1.08*
Transfer directness (t_2)	1.302	1.239	1.214	1.169	1.200	1.112	1.190	1.178	1.174	1.128	1.123	1.090
Network directness	1.052	1.052	1.064	1.064	1.006	1.006	1.157	1.132	1.040	1.049	1.031	1.037

The average boarding objective function (t_1) values are given in the ninth row of Table 4.4 in italic. Results of transfer directness are shown in the tenth row. The transfer directness reflects the overall transfer effective of a network route layout. Note that the differences between the transfer directness values in Table 4.4 from the **ISTG** method and those in Table 4.1 from the **FHC** method were negligible.

Results shown in Table 4.4 were obtained by using the original solutions of the benchmark problems as the initial guess networks. Table 4.5 shows results produced based on program generated initial networks. The only requirements in generating these results were that they must have the same number of routes as the original solutions, and the total network route lengths should not be more than 5% different from those of the original solutions. The results, in general, had performance better or comparable to those shown in Table 4.2 for the **FHC** method.

Table 4.5 Result Comparison – ISTG Method (Program Generated Initial Networks)

Problem Source	Mandl		Baaj & Mahmassani						Shih & Mahmassani			
Route layout cases	1		1		2		3		1		2	
Search Results	Mandl	**ISTG**	B&M	**ISTG**	B&M	**ISTG**	B&M	**ISTG**	S&M	**ISTG**	S&M	**ISTG**
0-transfer trips (%)	69.94	75.72	78.6	89.15	79.96	90.75	80.99	84.84	82.59	91.27	87.73	91.59
1-transfer trips (%)	29.93	23.70	21.39	10.85	20.04	8.93	19.01	14.52	17.41	8.73	12.27	8.41
2-transfer trips (%)	0.13	0.58	0.00	0.00	0.00	0.32	0.00	0.64	0.00	0.00	0.00	0.00
Uncovered trips (%)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Total route length	82	82	126	123	144	140	106	107	124	124	151	152
Number of routes	4	4	6	6	8	8	7	7	6	6	8	8
Boarding function t_1	*1.303*	*1.260*	*1.214*	*1.109*	*1.200*	*1.099*	*1.190*	*1.164*	*1.174*	*1.087*	*1.123*	*1.084*
Transfer directness (t_2)	1.302	1.249	1.214	1.109	1.200	1.096	1.190	1.158	1.174	1.087	1.123	1.084
Network directness	1.052	1.040	1.064	1.006	1.006	1.006	1.157	1.006	1.040	1.037	1.031	1.000

4.2 Miami-Dade County Transit Network Design Problem

Miami-Dade County encompasses an area of about 38 miles by 22 miles, with a population of about 2.3 million. However, unlike some large urbanized cities such as New York City, Boston, or San Francisco, Miami-Dade County has relatively low population and employment densities

and transit ridership. Except in downtown Miami and a few activity centers, transit demands are dispersed thinly over a large area, making it difficult to provide transit services efficiently. However, opportunities are also present. The population forecast has predicted another 1.2 million new residents in the county in the next 20 years (Miami-Dade 2001), which will result in worsening of traffic congestion but an increased and urgent need for better public transit services. The referendum passed in November 2002 by the voters in the county approved a sales tax increase, which will provide much needed funding for highway and particularly transit projects. New rail lines are to be built in the next 30 years and bus services are to be expanded significantly. All these make it imperative to develop tools for transit route network design and optimization to both increase ridership and realize cost savings.

For this study, the street network data were obtained in a GIS format from Miami-Dade County Information Technology Department (ITD) in 2000. There were two sets of street network data: detailed streets and major roads, which are illustrated in Figure 4.1. The street network used for transit route optimization was based on the major roads with some minor streets added based on the existing bus routes. The major street network consisted of 4,300 street segments and 2,804 street nodes. The distance between the two farthest nodes in the network was about 35 miles and the length of the shortest street segment was about 100 feet.

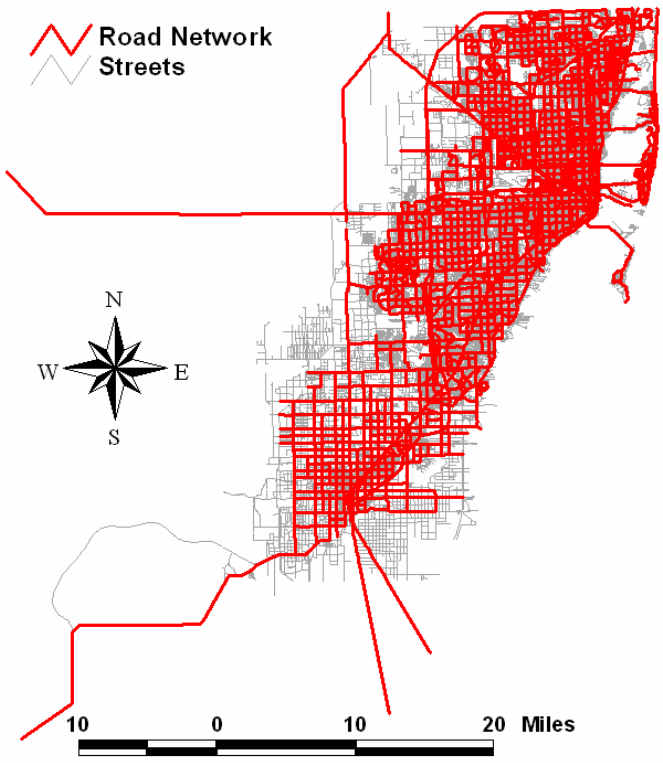


Figure 4.1 Street Network in Miami-Dade Transit Service Area

At the time of this study, Miami-Dade Transit Agency (MDTA) operated 83 transit routes including Metrorail, a rail rapid transit system of 22.5 route miles, Metro-mover, a 4.5-mile downtown automated circulation system, and 81 bus routes. In the optimization process, Metrorail and Metro-mover routes were fixed. For the bus network, the longest route was about 32 miles and the shortest route was about 4 miles. Those two numbers were used as base lines to

set the transit route length constraints in the optimization process. There were about 1,520 route miles. Figure 4.2 shows the existing transit network in Miami-Dade County. The transit route network data shown in Figure 4.2 were obtained from MDTA in year 2002. It may be seen from Figure 4.2 that some areas are not currently served by the existing transit network. The total route mileage of the existing transit network system was 1,520 miles, which also included those of small loops, needed either for transit vehicle turning around at some bus terminal points or for picking up or dropping off passengers at certain activity centers such as shopping centers. If not counting the mileage from these small circular loops, the existing transit network had about 13,00 road miles.

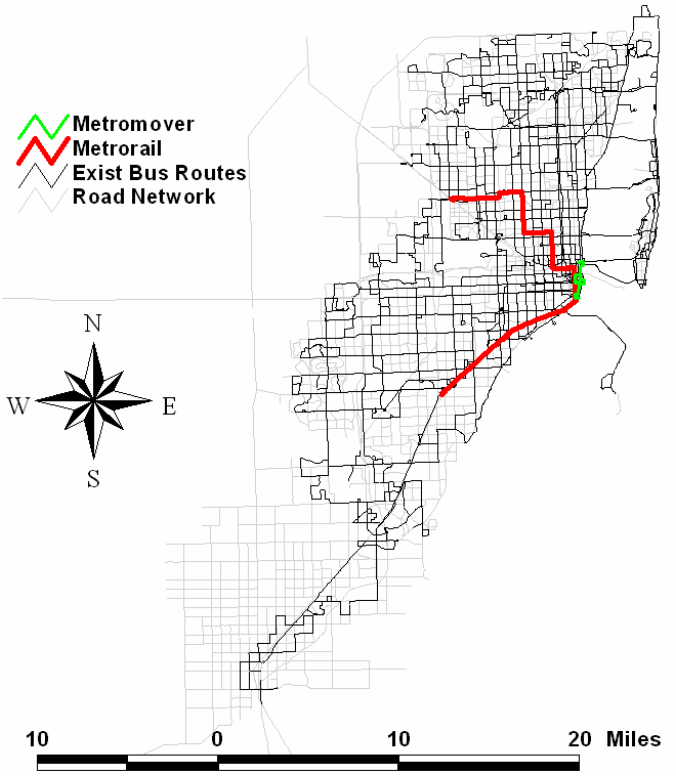


Figure 4.2 Existing Transit Network in Miami-Dade Transit Service Area

The demand matrix used in this study was generated from the 1999 Miami-Dade County FSUTMS[8] model. The demand matrix provided the daily number of passenger trips between each node pair in the street network. The total demand was 162,252 daily transit trips. The distribution of the demand is shown in Figure 4.3. While the FSUTMS model is considered sufficient for the purpose of testing the optimization methodology developed in this study, it is known to have certain limitations. To obtain more accurate OD matrix data that reflect the current Miami Dade transit demand distribution, further efforts are needed to develop transit demand estimates.

[8] Florida Standard Urban Transportation Model Structure

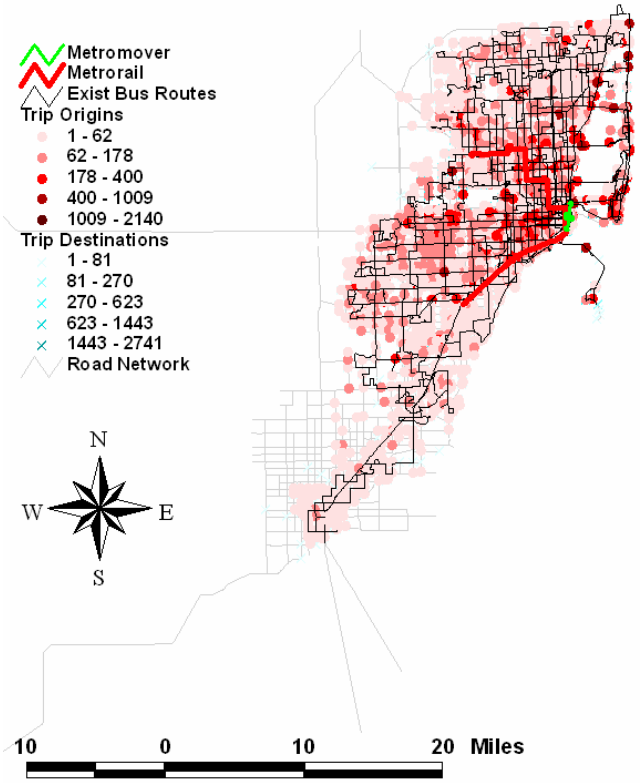


Figure 4.3 Transit Demand Distributions in Miami Dade Transit Service Area

Several methods developed in this study were applied to the problem of transit network optimization for Miami-Dade County, which resulted in a number of potential solutions. In the solution process, depending on the outcomes, certain constraints were adjusted to obtain better or more reasonable results in a reasonable amount of time and input data were modified for more realistic results. In this study, all the numerical results were generated from a high-end PC with a 2.8GHz CPU, 1GB RAM memory, and an 80GB hard disk. Currently, a high-end PC already has 3.06GH CPU with 2GB RAM or more memory. It is observed during the numerical analysis process of this study that the time savings from using a PC with 2.8GH and 1GB RAM was more than 30% compared to a PC with 1.8GHz and 512MB RAM.

4.2.1 Numerical Experiment 1

The route network optimization processes had the following constraints:

(1) The total route length of the route network must be no more than 10% different from that of the existing route network; and
(2) The existing fixed route constraints must be enforced, i.e., the Metrorail, and Metromover routes must not be changed during the optimization processes.

All the results, given in Table 4.6, were generated with up to five-key node representations and with different search methods. The methods, indicated in the second row of Table 4.6, included (refer to Sections 3.8.1 through 3.8.4):

(1) **GS1** – greedy method 1;
(2) **GS2** – greedy method 2;
(3) **HC** – hill climbing method; and
(4) **FHC** – fast hill climbing method.

The performance statistics for the existing transit network are given in the second column in Table 4.6. They are also provided for each of the results produced using the above four methods with two single optimization objectives, maximizing zero-transfer trips (f_0) and maximizing zero and one transfer trips (f_1). The values of the objective functions are given in the shaded cells. The statistics include percentages of trips that can be completed with zero, one, and two transfers, percentages of trips that could be completed with zero, one or less, and two or less transfers, network directness, transfer directness, and CPU time required to obtain the solutions.

Table 4.6 Comparison of Results with Existing Network (Existing Network as Initial Guess)

Network Parameters	Existing Results	Zero transfer objective function f_0				One-or-less transfer objective function f_1			
Solution Method		**GS1**	**GS2**	**HC**	**FHC**	**GS1**	**GS2**	**HC**	**FHC**
Zero-transfer trips (%)	14.28	22.25	22.25	23.82	23.43	18.01	18.67	19.21	18.90
One-transfer trips (%)	40.85	47.46	46.45	47.18	47.94	59.85	60.29	60.87	61.55
Two-transfer trips (%)	10.08	7.98	7.63	6.40	6.39	8.74	7.90	7.94	6.91
One-or-less transfer trips (%)	55.13	69.71	68.71	71.00	71.37	77.86	78.96	80.08	80.45
Two-or-less transfer trips (%)	65.20	77.69	76.34	77.40	77.75	86.59	86.86	88.02	87.36
Total covered trips (%)	65.66	77.86	76.49	77.55	77.89	86.72	86.94	88.13	87.48
Two-or-less transfer trips	105,588	125,810	123,623	125,349	125,918	140,229	140,664	142,547	141,472
Total route mileage	1,278	1,330	1,321	1,345	1,340	1,304	1,317	1,366	1,345
Two-or-less transfer trips per route mile	83	95	94	93	94	108	107	104	105
Total trips	161,944	161,944	161,944	161,944	161,944	161,944	161,944	161,944	161,944
Number of routes	83	83	83	83	83	83	83	83	83
Network directness	1.02	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01
Transfer directness t	1.94	1.82	1.81	1.78	1.78	1.89	1.88	1.87	1.86
Transfer directness (t_2)	2.65	2.30	2.33	2.28	2.27	2.18	2.16	2.13	2.13
CPU Time (hours)	-	0.32	1.72	0.74	0.79	3.79	24.29	14.49	15.76

The best zero-transfer trip coverage result was obtained from the **HC** method with objective function f_0, which had an improvement of 67% over the existing network. The percentage of zero-transfer trips increased from the existing 14.28% to 23.82%. The other three search methods also produced results with improvements ranging from about 56% to 64%. These improvements were achieved with a small increase in the total network route mileages of only 3% to 5% above the existing system.

For objective function f_1, the best one-or-less transfer trip coverage result was obtained from the **FHC** method. The percentage of one-or-less transfer trip coverage increased from the existing 55.13% to 80.45%, which represented a 46% improvement. The corresponding total route length increased only by 5%. The results of the one-or-less trip coverage from the other three search methods with objective function f_1 also showed comparable improvements, with increases ranging from about 41% for method **GS1** to 45% for method **HC**. At the same time, the total

route mileages only increased less than 5%. The one-or-less trip coverage is an important transit service quality indicator since according to survey by Stern (1996), as most transit riders may be only willing to transfer once per trip. Assuming this is true, the one-or-less trip coverage shown in the sixth row of Table 4.4 would be the actual total trip coverage of the corresponding route networks. More specifically, using the case corresponding to the last column as an example, 80.45% of the transit demands in the transit service area might likely to be met, while the rest of the 19.55% trip demands might either be unsatisfied or require two or more transfers. In the latter case the customers who had to make more than one transfer might be lost unless they were captive riders. There are certain cases where two-or-less transfer trip coverage may also be appropriate as optimization objective function. Examples of such situations include the so-called spinal transit route network structures and trunk-and-feeder systems, which have transit centers where some bus routes are designated as feeder lines to serve major transit corridor routes such as light rail and subway lines. Transit corridors such as light rails and subways offer high quality services. As a result, transit users may be willing to use a trunk-and-feeder system even when one or two transfers are required.

The number of covered trips (those that require two or less transfers) per route length is an indicator of the efficiency of the network. The number of covered trips (those requiring two or less transfers) per route length is an indicator of the efficiency of the network. The best result of 108 is produced from method **GS1** with one-or-less transfer function f_1 as the objective function. It may be seen from the 11th row of Table 4.6 that for the same objective function, either f_0 or f_1, the differences between the results from various search methods were insignificant.

All the solutions in Table 4.6 showed excellent network directness. Two transfer directness parameters are given in Table 4.4: $t = [f_0 + 2(f_1 - f_0) + 3(f_2 - f_1)]/f_2$ that represents the average numbers of boardings per transit rider who i able to complete a trip with two or fewer transfers and $t_2 = [f_0 + 2(f_1 - f_0) + 3(f_2 - f_1) + \alpha(f_t - f_2)]/f_t$ ($\alpha = 5$) that takes into account of trip demands not covered by the associated route network. All the solutions had various degrees of improvements in transfer directness t_2, ranging from 24% for methods **HC** and **FHC** (one-or-less transfer objective function) to 12% for method **GS2** (zero-transfer objective function).

The network results from the **FHC** method with up to five key-nodes are illustrated in Figure 4.4. The solid black lines represent results from the zero-transfer objective function f_0, the yellow lines represent the existing bus route network, and the gray dots represent street nodes with transit trip demands. It may be seen that some of these street nodes with trip demands were not covered by transit routes. For this problem, the uncovered trips are 100% - 77.89% = 22.11%. Figure 4.5 further shows the route network results obtained with the one-or-less transfer trip coverage function f_1 as the objective function. The solid lines labeled as **One-FHCS–Five** now represent route network result from the one-or-less transfer objective function using the **FHC** search method with up to five key-nodes. The route network result had, in general, a similar pattern as that in Figure 4.4 except that the total route network coverage increased from 77.89% to 87.48% at a cost of decreasing the quality of transfer directness t. Moreover, the percentage of zero-transfer trips also decreased from 23.43% to 18.90%. One drawback of an optimization process with a single objective function is that the optimization of one network parameter may be at the cost of other network parameters. For problems where the

optimization of more than one network parameters is required, multi-objective functions or with a composite objective function will be necessary.

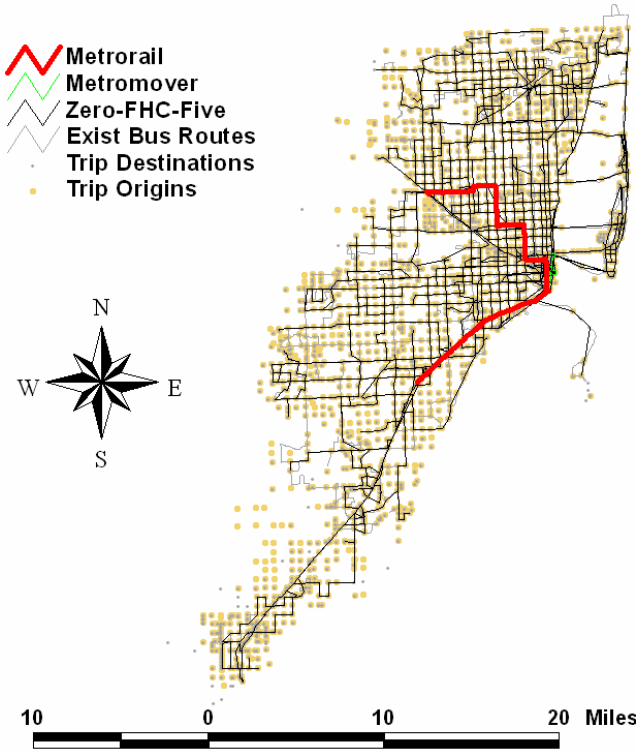


Figure 4.4 Route Network Results from Zero-Transfer Objective Function (Existing Network as Initial Guess)

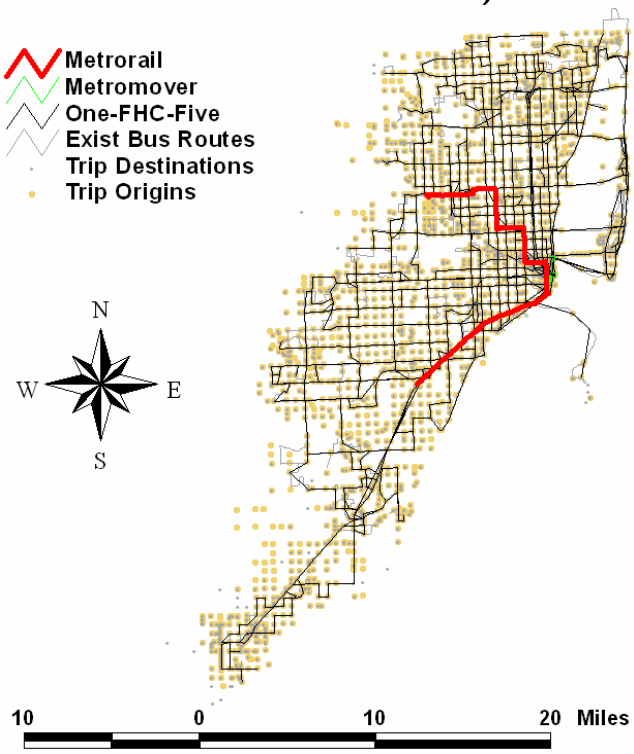


Figure 4.5 Route Networks from One-or-Less Transfer Objective Function (Existing Network as Initial Guess)

In terms of computer CPU times shown in the last row of Table 4.6, method **GS1** seemed to be the fastest method, while method **GS2** the slowest. With a PC of 2.8GH and 1GB RAM, all the optimization processes based on zero-transfer objective function was accomplished within one or two hours, while results based on one-or-less transfer function needed from a few hours for method **GS2** to a couple of days for other methods. Considering the size of these network optimization problems, the improvements to the existing results, and the low cost of high powered PCs, these CPU times are reasonable.

4.2.2 Numerical Experiment 2

The various results presented in Table 4.6 were generated with the existing Miami-Dade transit route network as the initial guess network, which might be biased toward the existing network. In other words, those results might be local optima in the neighborhood spaces of the existing network. For comparison and objectivity, all the optimization problems in Table 4.6 were solved again with the same constraints except that the initial guess network was automatically generated by the program. Table 4.7 presents the numerical results based on the program generated initial guess networks. Results in Table 4.7 show that in general, various design variables and objective functions values obtained from automatically generated initial networks were slightly better than those obtained based on the existing networks (refer to Table 4.6). Although this did not guarantee global optimal results, it did increase the possibility that a global optimal result or a local optimal result in a large solution subspace might be near. In general, to determine whether or not a solution is a global optimal result is a difficult task especially in transit network optimization field where available theories or guidelines seem nonexistent for large complicated network systems. However, statistically, if an optimization process always converges to the same or similar solutions from randomly selected initial guesses, the possibility of such results being close to a global optimal result increases as the number of times that the process is repeated increases.

Table 4.7 Results from Program Generated Initial Networks

Network Parameters	Existing Network	Zero transfer trip coverage function				One transfer trip coverage function			
Solution Method		**GS1**	**GS2**	**HC**	**FHC**	**GS1**	**GS2**	**HC**	**FHC**
Zero-transfer trips (%)	14.28	26.29	26.33	26.87	26.41	20.14	19.58	19.96	19.47
One-transfer trips (%)	40.85	46.41	46.28	47.54	47.36	60.61	61.65	62.29	62.10
Two-transfer trips (%)	10.08	2.21	2.33	2.36	2.48	2.84	2.74	2.82	2.66
One-or-less transfer trips (%)	55.13	72.71	72.61	74.41	73.77	80.74	81.23	82.24	81.57
Two-or-less transfer trips (%)	65.20	74.91	74.94	76.77	76.25	83.58	83.97	85.06	84.23
Total covered trips (%)	65.66	74.92	74.95	76.77	76.25	83.58	83.97	85.07	84.24
Two-or-less transfer trips	105,588	121,316	121,364	124,325	123,482	135,353	135,979	137,752	136,407
Total route mileage	1,278	1,340	1,338	1,344	1,348	1,341	1,335	1,353	1,346
Two-or-less transfer trips per route mile	83	91	91	93	92	101	102	102	101
Total trips	161,944	161,944	161,944	161,944	161,944	161,944	161,944	161,944	161,944
Number of routes	83	83	83	83	83	83	83	83	83
Network directness	1.02	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01
Transfer directness t	1.94	1.68	1.68	1.68	1.69	1.79	1.80	1.80	1.78
Transfer directness t_2	2.65	2.26	2.26	2.22	2.24	2.16	2.15	2.13	2.15
CPU Time (hours)	-	0.39	1.20	0.86	0.65	4.68	18.62	11.15	12.93

Figures 4.6 and 4.7 show two network results given in Table 4.7. Similar to Figures 4.4 and 4.5, both results were obtained from methods **FHC**. The solid black lines represent solutions from, respectively, zero-transfer objective function f_0 in Figure 4.6, and one-or-less transfer objective function f_1 in Figure 4.7.

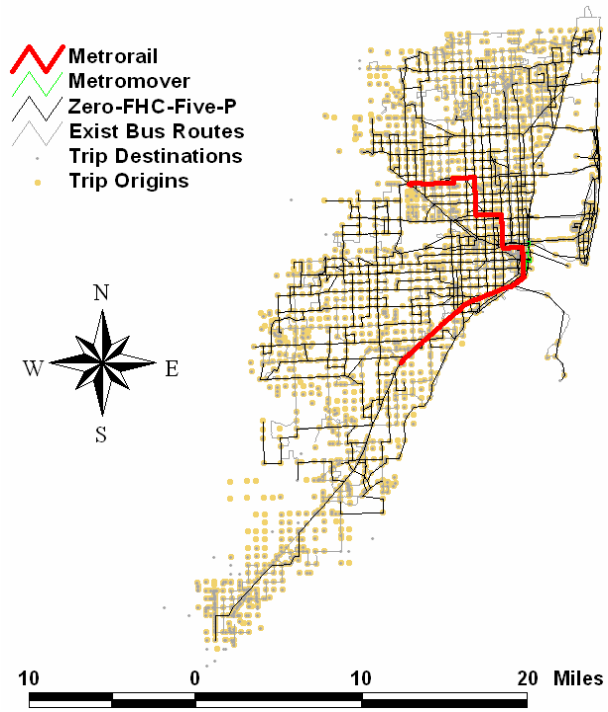


Figure 4.6 Route Networks from Zero-Transfer Objective Function (Program Generated Initial Network)

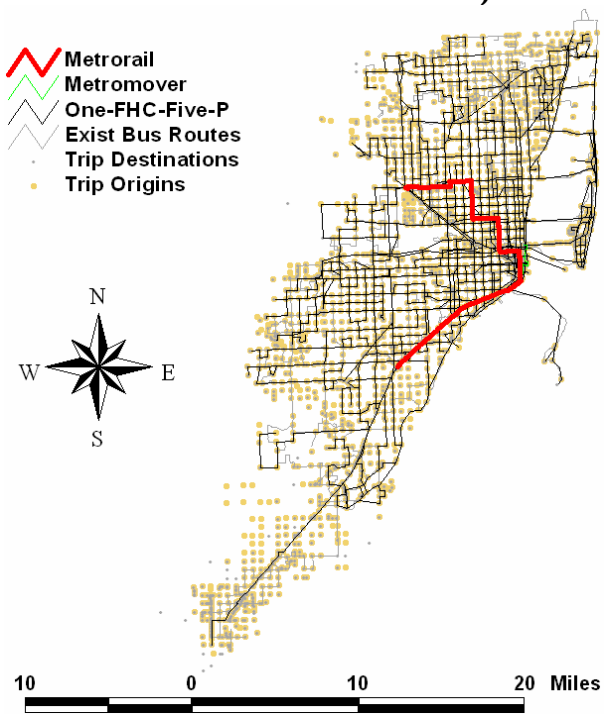


Figure 4.7 Route Network from One-or-Less Transfer Objective Function (Program Generated Initial Network)

As the last set of numerical experiments of this section, Table 4.8 describes results obtained from the minimization of the two network transfer directness objective functions described in section 3.7.4, i.e., the average number of passenger boarding objective function based on one-or-less transfer trips defined in equations [3.47]:

$$t_1 = \frac{f_0 + 2\left(f_1 - f_0\right) + \alpha\left(f_T - f_1\right)}{f_T};$$

and the average number of passenger boarding objective function based on two-or-less transfer trips defined in equations [3.46]:

$$t_2 = \frac{f_0 + 2\left(f_1 - f_0\right) + 3\left(f_2 - f_1\right) + \alpha\left(f_T - f_2\right)}{f_T}.$$

The penalty number α was set at 4, meaning that each uncovered trip would be penalized as four passenger vehicle boardings. For objective function t_1, uncovered trips are those that are not included in one-or-less transfer trips, and for objective function t_2, uncovered trips are those that are not included in two-or-less transfer trips. The optimal value for both functions t_1 and t_2 is 1.0. In such cases, all riders in the transit service area under analysis are covered, and each rider completes his/her trip with one vehicle boarding, or a zero-transfer trip. The advantage of using the average numbers of passenger boardings t_1 or t_2 as objective functions is that it takes into account the effect of zero-transfer, one-or-less transfer, and uncovered trips simultaneously into the optimization process for functions t_1 and t_2. Additionally, it also includes the effect of two-or-less transfer trips into the optimization process for function t_2. The disadvantage of these objective functions is that the selection of the penalty number α is somewhat ad hoc, as it relies on trial-and-error based on previous program results. Moreover, optimization processes based on the average boarding are more computational intensive than those based on single objective function shown in Tables 4.6 and 4.7 because both functions t_1 and t_2 are composite objective functions. Function evaluation of t_1 involves the calculation of the zero-transfer function f_0 and the one-or-less transfer function t_1, while function evaluation of t_2 involves the calculation of the zero transfer function *f0*, one-or-less transfer function f_1, and two-or-less transfer function f_2.

In Table 4.8, statistics for the existing transit system and four optimization results are provided. The shaded cells show the objective function values after the optimization processes. It may be seen that results obtained from the optimization processes had significant improvements over the existing network. Generally, method **FHC** produced slightly better results than method **GS1**. However, the computational cost of method **FHC** was higher than method **GS1**. Additionally, the optimization process for objective functions involving two-or-less transfer function f_2 was quite time consuming. It took 90.41 hours or about four days for method **GS1** to produce results, and 389.67 or about two weeks for **FHC** to complete. Although in terms of real monetary terms, a week of CPU running time on a PC is still considered inexpensive, it may be impractical to experiment with a large number of scenarios and obtain results in a reasonable amount of time. In such cases, the use of a main frame computer, a multi-processor PC, or a multi-computer network may be desirable.

Comparing various network parameter results obtained from objective functions t_1 and t_2 in Table 4.8, it may be seen that for the same search methods, either **GS1** or **FHC**, results from objective function t_1 and those from function t_2 were quite consistent. There were no significant differences in the zero, one-or-less, and two-or-less transfer trip coverages associated with objective functions t_1 and t_2, as were the cases in optimization with single objective function described in Tables 4.6 and 4.7. Figure 4.8 depicts the network optimization results with transfer directness function t_1 using the **FHC** method with up to five key-node route representation, while Figure 4.9 shows the results based on objective function t_2 using **GS1**.

Table 4.8 Comparison of Results with Existing Network (Existing Network as Initial Guess)

Network Parameters	Existing Network	Average passenger boarding function *t1*		Average passenger boarding function *t2*	
		GS1	**FHC**	**GS1**	**FHC**
Objective Function *t1*	2.76	2.14	2.13	2.19	2.10
Objective Function *t2*	2.65	2.12	2.10	2.15	2.04
Zero-transfer trips (%)	14.28	24.22	24.55	22.40	25.70
One-transfer trips (%)	40.85	56.48	56.77	57.05	56.28
Two-transfer trips (%)	10.08	2.85	3.06	4.22	6.63
One-or-less transfer trips (%)	55.13	80.70	81.31	79.45	81.98
Two-or-less transfer trips (%)	65.20	83.54	84.37	83.67	88.61
Total covered trips (%)	65.66	83.54	84.38	83.68	88.72
Zero-transfer trips	23117	39,218	39,754	36,279	41,613
One-or-less transfer trips	89,271	130,684	131,682	128,664	132,754
Two-or-less transfer trips	105,588	135,294	136,634	135,505	143,498
Total covered trips	106,330	135,295	136,641	135,513	143,676
Total trips	161,944	161,944	161,944	161,944	161,944
Total route mileage	1,278	1,350	1,358	1,336	1,355
Two-or-less transfer trips per route mile	83	100	101	102	91
Number of routes	83	83	83	83	83
Network directness	1.02	1.01	1.01	1.01	1.01
Transfer directness	1.94	1.74	1.76	1.78	1.79
CPU Time (hours)	-	7.62	9.85	90.41	389.67

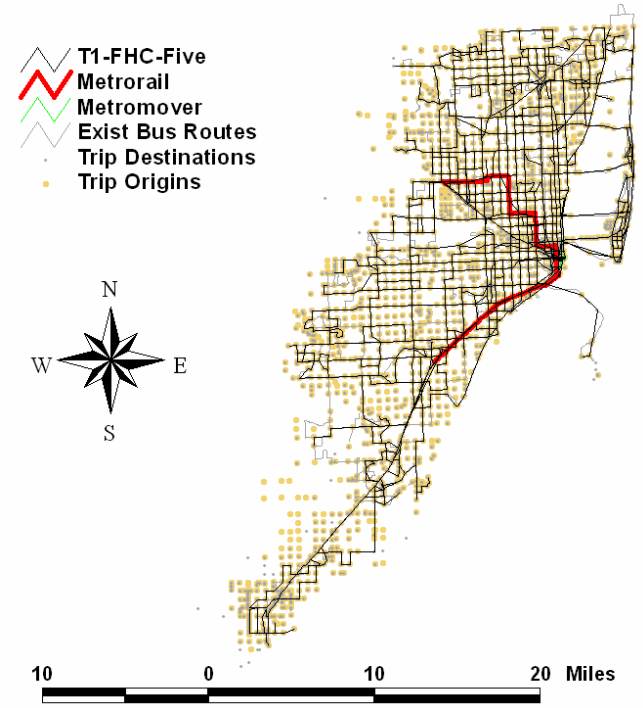


Figure 4.8 Route Networks from Transfer Directness Objective Function t_1

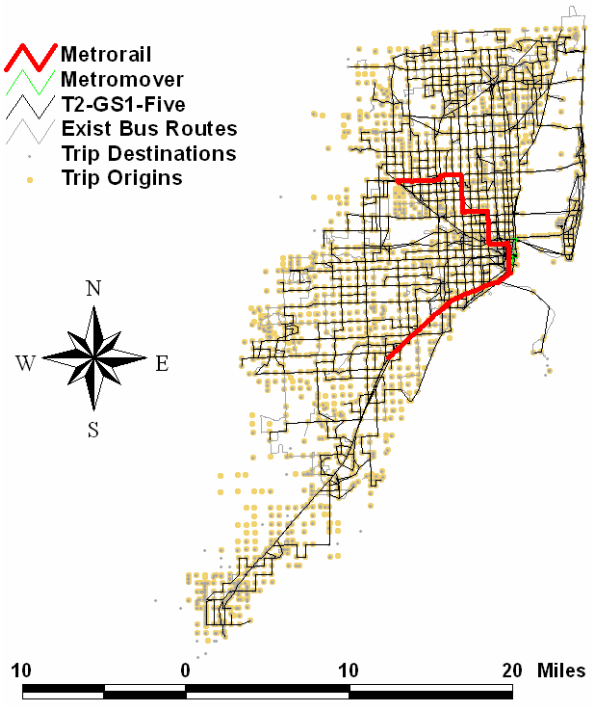


Figure 4.9 Route Networks from Transfer Directness Objective Function t_2

4.2.3 Numerical Experiment 3

The last numerical experiment was based on the **ISTG** method. Recall from section 3.8.5 that the **ISTG** method was developed by incorporating the simulated annealing and tabu schemes

into the **GS1** local search method to prevent the search process from being trapped into poor local optima. For comparison, results from **GS1** were also generated. All the results from both **GS1** and **ISTG** methods were obtained from local path search spaces based on the two key-node representation.

The various parameters used in **ISTG** method for all the test cases are summarized below (refer to section 3.8.5 for detailed description of these parameters):

(1) Length of tabu list/queue $L_t = 250$;
(2) Maximum route level iteration counter $L_r = 1500$;
(3) Maximum network level iteration counter $L_n = 2000$;
(4) Number of randomly selected routes that are outside a local search space $N_r = 50$;
(5) Maximum CPU time = 48 hours;
(6) Initial annealing temperatures for various test cases: $\theta_0 = 1000, 2000, 4000$;
(7) Temperature reduction factors for various test cases (for both route and network levels): $\tau = 0.50, 0.75, 0.95$;
(8) Probability tolerance for search results with repeated goal values (at both route and network levels): $P = 0.50, 0.75, 0.95$.

Table 4.9 presents the results obtained from the **ISTG** method and those obtained from the **GS1** method. For comparison, network parameters associated with the existing network are also presented. The values of objective function t_1 are given in italic. The first row in this table identifies the methods used to generate the numerical results. Three θ_0 values were used for the **ISTG** method represented the three initial annealing temperatures used in the simulated annealing search processes. The second row presents different temperature reduction factors for different test cases. The probability tolerances for all the test experiments were set as the same as the corresponding temperature reduction factors. In such cases, a slow cooling process (τ close to 1) was associated with large probability tolerance ($P = \tau$) for results with repeated goal function values. For each solution method or each simulated annealing process defined by parameter set (θ_0, τ)**,** the corresponding column provides the statistics for the TRN obtained from the optimization process.

From Table 4.9, it may be seen that the value of objective function t_1 obtained from the **GS1** method decreased from $t_1 = 2.76$ in the existing network to $t_1 = 2.22$, an improvement of about 20%, while the best result, $t_1 = 1.972$ or an improvement of about 28% over the existing network, was obtained from the **ISTG** method with initial temperature and temperature reduction factor (θ_0, τ) = (2000, 0.95). The corresponding zero-transfer function f_0 increased from the existing 14.28% to 24.42%, an improvement of about 71%, and the one-or-less transfer function f_1 increased from 55.13% to 89.19%, an improvement of about 62%. These improvements were achieved with a small increase of less 8% in total network route mileage.

Table 4.9 also presents percentages of two-or-less transfer trips and total covered trips. In general, including transit trips involving more than one vehicle transfers in transit service coverage may not be reliable unless the trips are made on a high-speed transit line and its feeder-bus system where transfers between different transit modes are usually much more efficient than at regular bus line intersection points.

Table 4.9 Comparison of Results from ISTG Method, GS1 Method, and Existing Network

Network Parameters	Existing Network	**GS1** method	Integrated Simulated Annealing, Tabu and Greedy Method (**ISTG**)								
			$\theta_0 = 1000$			$\theta_0 = 2000$			$\theta_0 = 4000$		
Temperature Reduction τ	-	-	0.50	0.75	0.95	0.50	0.75	0.95	0.50	0.75	0.95
Boarding Function t_1	*2.755*	*2.218*	*2.077*	*2.044*	*1.980*	*2. 096*	*2.037*	*1.972*	*2.070*	*2.004*	*1.977*
Boarding Function t_2	2.65	2.15	2.03	2.00	1.94	2.05	2.00	1.94	2.02	1.97	1.94
0-transfer trips (%)	14.28	20.45	21.74	21.76	23.86	21.46	23.57	24.42	22.54	24.51	24.32
1-or-less transfer trips (%)	55.13	78.86	85.27	86.92	89.05	84.47	86.39	89.19	85.23	87.55	89.00
2-or-less transfer trips (%)	65.20	85.83	90.38	91.20	92.94	88.78	90.37	92.50	90.14	91.18	92.74
Total covered trips (%)	65.66	85.96	90.40	91.23	92.95	88.79	90.39	92.50	90.15	91.22	92.74
Total route mileage	1,278	1,360	1,360	1,380	1,384	1,363	1,379	1,374	1,381	1,384	1,377
Trips per route mile $R_{T/L}^{(2)}$	83	102	108	107	109	106	106	109	106	107	109
Average Transfers	1.94	1.84	1.82	1.81	1.79	1.81	1.78	1.77	1.80	1.77	1.78
Network directness	1.017	1.008	1.004	1.007	1.006	1.008	1.003	1.003	1.010	1.004	1.005
CPU Time (hours)	-	2.15	5.34	7.49	25.48	4.40	9.25	36.85	5.38	8.87	30.99

The number of covered trips per route mile shown in Table 4.9 was defined as $R_{T/L}^{(k)} = f_k / l_T$, where f_k was the number of trips accomplished with k-or-less transfers, and l_T was the total length of the TRN. As a network efficiency indicator, the best $R_{T/L}^{(k)}$ value was obtained with temperature reduction factor $\tau = 0.95$. The average transfers in Table 4.9 were defined as $[f_0 + 2(f_1 - f_0) + 3(f_2 - f_1)]/f_2$, which was the average boardings per transit rider who could complete a trip with two or fewer transfers. The network directness shown in Table 4.7 represented the average ratio of the distance a person traveled between his/her OD points by using transit service over the distance measured along the shortest path in the street network. It may be seen that all the TRNs shown in Table 4.7 had excellent network directness.

Although differences in results produced by the **ISTG** search method with different combinations of annealing parameters (θ_0, τ) were not significant, it did show a general trend of improvement in objective function values as the initial temperature θ_0 increases and the cooling process slows down, i.e., as the temperature reduction factor τ approaches to 1.0. As was pointed out before, the **ISTG** search method developed in this study is a stochastic process. Statistically, higher initial temperatures and lower cooling rates are expected to produce better solutions although such expectation may not be met on occasions.

Results shown in the second and the last rows of Table 4.9 and also from other testing experiments conducted in this study revealed an interesting fact that it might be more convenient to use CPU time allowance as the termination condition for an **ISTG** search process than the convergent condition associated with the annealing parameter set (θ_0, τ). This was based on the

fact that improvements in objective function values seemed to be more consistent with the amount of CPU time actually used in a search process than the selection of the two annealing parameters θ_0 and τ. Consequently, it seems to be more effective in practice to conduct an **ISTG** search process with a slow cooling rate (τ and P close to 1) and large initial temperature θ_0, while using allowable CPU time as termination condition.

4.3 Summary of Numerical Experiments

In this chapter, the applications of the transit route network optimization methodology developed in this study to two real transit route network design problems are described. The first problem was Mandl's small and dense transit network problem that was based on a real network in Switzerland. Comparison and agreement of results generated from this study with those from previous work may serve as validity tests of the methodology developed in this study. Results obtained from this study showed good improvements over the existing solutions to transit route network design problems. Although the comparison may not be complete in this development stage since the two previous studies involved both transit route network design and transit scheduling design, while this study deals only with transit route network design optimization, the results demonstrated the ability of the methodology to improve a given solution.

The second numerical experiment was a large-scale transit network problem defined based on the Miami Dade County's transit system. The problem included two routes with fixed topology and geometry, 81 bus routes with unknown geometrical and topological characteristics, 2,804 street nodes, and 4,004 street segments, which was perhaps the largest transit network design problem reported in the literature so far. Results obtained from the search methodologies developed in this study showed significant improvements over the existing network system in terms of transfer directness defined by the average number of passenger boardings, network directness defined by the closeness of an average transit rider's travel route to the shortest path between his/her OD points, and the average trip length ratio defined by the number of trips covered by a unit transit route length. Again, although the results should not be interpreted as better than the existing transit network since the demand used in this study might not be accurate, they did demonstrate the ability of the methodologies developed in this study to improve an existing network for a given demand distribution.

The computing resources required for this large-scale network optimization problem was reasonable considering that only a high end PC was used to obtain all the numerical results. For zero transfer objective function, it took less than one day to obtain results. For one-or-less transfer objective function, it might need one or two days. For two-or-transfer objective function, several days of computing time will be needed to obtain results with current PC computing powers. It is believed that the solution time on a high end PC will decrease rapidly with the improvement of PC computing power that has so far obeyed Moore's law, i.e., doubling every eighteen months.

5. COMPUTER PROGRAM FOR TRANSIT ROUT NETWORK DESIGN

In addition to the development of methodologies for transit network optimization, a computer program OPTNet, was designed and implemented. The program was written in FORTRAN with a GIS based user interface implemented in TransCAD. In this chapter, the program design, functions, and user interface are described.

5.1 Implementation of Transit Network Optimization Methodologies

The solution methodology for the optimization of transit route network design described in Chapter 3 has been implemented in a computer program called OPTNet (OPTimization Package for transit Network). As mentioned previously, the methodology and the computer program OPTNet developed in this study are only the products of the first development phase of a complete transit network design project that should include two design stages: transit route network design and transit scheduling design. A transit route network design problem deals with issues related to transit route layout or coverage, while a transit scheduling design problem addresses issues related to transit vehicle headway design and vehicle assignment timetables.

In general, optimization of transit route network design should produce route network structure that allow optimal route network directness and transfer directness. The route network directness of a transit network system is measured in this study by the two directness functions $d^G(\boldsymbol{T}^{(l)})$ and $d^R(\boldsymbol{T}^{(l)})$, defined in equations [3.30a] and [3.30c], and the two transit route Out-of-Direction impact indices defined in equations [3.31a] and [3.31c]. The transfer directness was represented by various transfer objective functions described in Section 3.7 and the average boarding functions defined in [3.46] and [3.47]. Ideally, both transit route network design and transit scheduling design processes should be carried out iteratively into one integrated design process. This has been left for future development.

The computer program OPTNet consists of three separate modules: OPTNet1, OPTNet2, and OPTNet3. The OPTNet1 module generates the shortest path space $\boldsymbol{P}_S$ defined in [3.10] from the user input street network data in a format as shown in Table 3.2. The shortest path search method used in OPTNet1 is based on the label setting approach developed by Dijkstra (1959) and several other researchers, and the code is adopted from the algorithm developed by Gallo and Pallotino (1988).

The OPTNet2 module generates an initial route network $\boldsymbol{T}_0^{(l)}$ from the shortest path space $\boldsymbol{P}_S$ obtained from OPTNet1. OPTNet2 provides two options in selecting the initial network, i.e., the simple and the random network selection methods. Assuming that the path population of the shortest path space $\boldsymbol{P}_S$ is n_s, the simple method first select a path $\boldsymbol{r}_1^{(0)}$ from the n_s paths in shortest path space $\boldsymbol{P}_S$ such that the one-route transit system $\boldsymbol{T}_0^{(1)} = \{\boldsymbol{r}_1^{(0)}\}$ gives the best objective function value (or values for multi-objectives problems). This will need at most n_s function evaluations of the one-route network systems. Second, from the rest of the $n_s - 1$ paths in space $\boldsymbol{P}_S$, select a path $\boldsymbol{r}_2^{(0)}$ such that the two-route network system $\boldsymbol{T}_0^{(2)} = \{\boldsymbol{r}_1^{(0)}, \boldsymbol{r}_2^{(0)}\}$ gives the best objective function value(s). This will require at most $n_s - 1$ function evaluations of the two-route network systems. Repeat the above procedure to add route $\boldsymbol{r}_3^{(0)}$ to obtain transit system $\boldsymbol{T}_0^{(3)} = \{\boldsymbol{r}_1^{(0)}, \boldsymbol{r}_2^{(0)}, \boldsymbol{r}_3^{(0)}\}$ with $n_s - 2$ additional function evaluations of the three-route network systems. Continue

the above procedures to obtain an initial guess transit network with l initial transit routes $\boldsymbol{T}_0^{(l)} = \{\boldsymbol{r}_1^{(0)}, \boldsymbol{r}_2^{(0)}, \boldsymbol{r}_3^{(0)}, ..., \boldsymbol{r}_l^{(0)}\}$. The total number of function evaluations n_t in generating $\boldsymbol{T}_0^{(l)}$ may be estimated as below,

$$n_t < \{ n_s + (n_s - 1) + (n_s - 2) + \ldots + [n_s + (l - 1)] \} \times e(l) = [\, l \times n_s + \frac{l-1}{2} l \,] \times e(l)$$

where $e(l)$ is the number of function evaluations of an l-route network system. The simple method usually can find an initial network much better than that obtained from the random selecting process. However, the computational cost will be high for a large street network. Note from the above expression that although the number of function evaluations n_t is linearly related to the population size n_s of the shortest path space $\boldsymbol{P}_s$, the number of function evaluations may be huge for a large street network. For a street network with n street nodes, the corresponding level-one shortest path space $\boldsymbol{P}_s$ will have a population size $n_s = n\,(\,n - 1\,)/2$. For very large street networks, e.g., $n > 6000$, it may be more effective to start the search process from a randomly selected initial network than using the simple method. It may been seen that an initial guess transit network generated from the OPTNet1 module does not take information from either the existing network system or heuristics design guidelines, thus should not introduce biases into any particular systems.

Starting from an initial route network $\boldsymbol{T}_0^{(l)}$, either generated from the OPTNet2 module or an existing route network, the OPTNet3 module searches for an optimal solution with the methods described in Chapter 3. OPTNet3 produces two files describing the route network optimization output results. One is an ASCII file that describes all the transit routes with street node sequences, and the other is used to display the results in a GIS environment. Both files may be used in post processing, which involves graphic display of optimization results, analysis of the results, generation of system statistics, and interactive modifications by the user.

5.2 TransCAD User Interface for the OPTNet Program

The TRN optimization methodology developed in this study has been implemented in a computer program OPTNet with a TransCAD based user interface. All the functions related to transit network optimization are available from a menu *TRN Optimization*. Through the *TRN Optimization* menu, the OPTNet will accept and/or output graphic or table data interactively with various TransCAD user interface menus, dialog boxes, tools, or functions. The *TRN Optimization* menu consists of the following menu items:

(1) *Prepare TRN Input Data File*;
(2) *Get TRN Names*;
(3) *Get Major Global Control Data*;
(4) *Get Detailed Global Control Data*;
(5) *Get Route Control Data*;
(6) *Data Check*;
(7) *Run OPTNet Programs*;
(8) *Post processing*;
(9) *Continue Optimization*.

In this section, the functions, data files, and definitions of various variables related to the above submenus are described. Sections 5.3 through 5.3 explain the functions of the above menu items.

5.2.1 Prepare TRN Input Data Files

This menu prompts the user for the input files needed by OPTNet. The user must provide these files before starting a TRN optimization process.

A transit route network optimization process may require up to four data files. These include:

(1) A street network data file that defines all the major street segments/links suitable for transit vehicle operation in the transit service area;
(2) An OD trip matrix data file that defines transit demand distribution among various street nodes of the street network;
(3) A route network data file that defines any pre-defined routes in the TRN optimization process, such as fixed routes (guideway lines, transit planners specified routes, etc.) and/or user specified initial guess routes (existing route system etc); and
(4) A route constraint area or location data file if some of the transit routes must start from, end at, or pass through certain pre-defined areas or locations.

These files may be generated either from standard TransCAD user interface tool menus (such as creating network line layer or creating route system table, etc.), or from TransCAD's street network database, or from other software tools. *TRN Optimization* menu tool developed in this study also provides a Graphic User Interface (GUI) tool to generate and/or to edit pre-defined routes and route constraint areas during TransCAD input process.

The current version of the OPTNet program accepts two types of input file formats, i.e., files generated from TransCAD menu tool and the comma- or space-delimitated plain text files. Files generated from other software packages must be converted into TransCAD files or plain text files first. The **PTIDF** menu provides detained step-by-step instruction on converting non-TransCAD files to TransCAD files with standard TransCAD tools. The plain text files are the basic input/output data formats for program OPTNet. In fact, files generated from TransCAD will be converted to plain test format before OPTNet is executed and the results obtained from OPTNet in plain text file format will be converted back to TransCAD graphic or table file formats. The following is a description of TransCAD format and plain text file format for street network, OD matrix, pre-defined route files, and pre-defined route constraint area files.

5.2.1.1 Street Network Input File

Comma- or space-delimitated plain text file format

The street network file has *m* lines of data, where *m* is the number of street segments in the street network. Each line consists of four integer numbers that define a street segment. The meanings of the four numbers, denoted as *Link_ID*, *From_Node_ID*, *To_Node_ID*, and *Link_Leng*, are as follows:

(1) *Link_ID* – street segment identification number. Any integer uniquely associated with a street segment may be used as the ID of the street segment.

(2) *From_Node_ID* – node ID of the starting node of the street segment. A street network consists of a set of street nodes connected by street segments. The street node set is also referred to as the street segments' end node set. A street segment is defined by its two end nodes. Each street's end node is associated uniquely with an integer called street node's or street end node's ID.

(3) *To_Node_ID* – node ID of the end node of the street segment.

(4) *Link_Leng* – length of the street segment. An integer that defines the length of the street segment. The length is generalized length. It may be geometric length, travel time, travel impedance, or any other costs associated with the street segment. In the OPTNet program, available units of length are feet, meters, or minutes. Other length units such as miles, kilometers, or hours will be converted automatically into appropriate units in the TransCAD data input stage.

TransCAD map (.map) file format

The OPTNet will automatically covert a street network line layer defined in a TransCAD map file (.map) into a space-delimitated plain text file described in the previous section. In the file conversion process, the user will be asked to provide the following information regarding the street network data file:

- Name and the corresponding path directory of the TransCAD .map file;
- Layer name of the street network line layer in the TransCAD .map file;
- Field name of the street network's segment/link cost, such as Length, Travel Time, LENGTH, Travel Impedance, etc.;
- Units of the street network link cost, such as mile, foot, minute, hour, meter, etc.

Layer name and field name provided by the user must match those appeared in the TransCAD .map file. The layer and field names are case-sensitive.

5.2.1.2 OD Matrix Input File

The OPTNet program accepts three types of OD matrix distribution: the uniform demand distribution, the random demand distribution, and the user defined OD distribution. The uniform demand distribution assumes that all OD pairs in the street network have the same number of trips, and users will be asked to provide one constant to define a typical OD pair's trips, which will be applied to all other OD pairs. The random demand distribution assumes random demand distribution among OD pairs of the street network. The user will be asked to provide a mean trip number of a typical OD node pair. For example, a number 10 means that the number of trips for an OD pair will be a random number selected from 0, 1, ... , 20. For uniform and random demand distribution cases, the OPTNet program does not require an OD matrix input file. The user will be asked for the uniform or random trip numbers in the *Get Global Control Data* menu.

If a user defined OD trip distribution is to be used, the user is required to provide the input data file to define OD trips for the OD node pairs. The following is a description of two input file formats for an OD matrix.

Comma- or space-delimitated plain text file format

The street network file has k lines of data, where k is the number of node pairs with nonzero demand trips. Each line has three integers that define an OD node pair's demand. The meanings of the three numbers, denoted as *O_Node_ID*, *D_Node_ID*, and *Num_Trips*, are given below:

(1) *O_Node_ID* – node ID of the trip origin node. The node ID must be the same as that used to define the street node (end node) set.

(2) *D_Node_ID* – node ID of the trip designation node. The node ID must be the same as that used to define the street node (end node) set.

(3) *Num_Trips* – number of trips originated from *O_Node* and destined for *D_Node* identified by the two ID numbers.

TransCAD map (.map) file format

The OPTNet will automatically covert an OD matrix file generated from the standard TransCAD menu tool/function into a space-delimitated plain text file described in the previous section. During the file conversion process, the OPTNet will asks the user to provide the information regarding the street network data file, which will be described in the next subsection.

5.2.1.3 Route Network Data File

A route network data file defines any pre-defined routes and/or user specified initial guess routes (existing route system etc). Pre-defined route input files are required only in the following two cases:

(1) The TRN involves fixed routes, i.e., some of the routes must follow exactly the routes (and stops) that the planner specifies;
(2) The solution search process for some routes must be started from user specified initial routes such as routes from an existing route system.

The following is a description of the input file format.

Comma- or space-delimitated plain text file format

The pre-defined route file consists of n data blocks, denoted as B_1, B_2, ... , B_n, associated with the n pre-defined routes, r_1, r_2, ... , r_n. Each data block is made up by the same number of lines as the number of stops on the corresponding route. For example, the data block B_i has n_i lines, where n_i is the number of stops on route r_i. Each line defines a route stop, or a stop associated with a particular route, and has at least three integer numbers:

(1) *Route_Num* – route number of the route stop. This number must be greater than zero but less than or equal to the total number of routes of the TRN system.

(2) *Stop_Node_ID* – node ID of the route stop. The stop must be a street node. *Stop_Node_ID* is the ID of street node used as the stop. *Stop_Node_ID* must be one of the numbers used to defined street nodes or street segment end nodes.

(3) *Route_Stop_ID* – route stop sequence number of a stop. A transit route consists of an ordered sequence of stops. *Route_Stop_ID* defines the sequence number of the stop in the route stop sequence. *Route_Stop_ID* must be greater than zero and less than or equal to the total number of stops on the current route.

Note that for a TRN system of l routes, the user may pre-define any number of routes (less than or equal to l). For each route stop data line, the user may provide more information on the route stop such as the name of the stop, milepost of the stop, etc., as long as the first three integers are *Route_Num*, *Stop_Node_ID*, and *Route_Stop_ID*.

TransCAD Route Table File

The *TRN Optimization* menu interface will automatically covert a TransCAD route table file generated from a standard TransCAD menu, tool, or function to a space-delimitated plain text file described in the previous section. The OPTNet also provides a graphic dialog to create a new or to modify an existing route network system.

5.2.1.4 Route Constraint Area/Location Data File

A node set data file defines any pre-defined node sets. Users will have the options to specify that certain routes must start from, end at, or pass through certain areas or locations defined by the corresponding street node sets.

The following is a description of the input file format.

Comma- or space-delimitated plain text file format

The pre-defined route file consists of n data blocks, denoted as B_1, B_2, ... , B_n, associated with the n pre-defined node sets, s_1, s_2, ... , s_n. Each data block is made up by the same number of lines as the number of nodes on the corresponding node set. For example, the data block B_i has n_i lines, where n_i is the number of street nodes on node set s_i. Each line defines a street node associated with a particular node set, and has at least two integer numbers:

(1) *Set_Num* – node set number of the node set.
(2) *Node_ID* – node ID of the street node. *Node_ID* must be one of the numbers used to defined street nodes or street segment end nodes.

TransCAD Node Set Table File

The *TRN Optimization* menu interface provides a graphic dialog to create a new or to modify an existing route constraint node set data file.

5.2.2 Get TRN Names

This menu item, when selected, will ask the user to input the following information:

(1) Directory name to store all the files generated during a TRN optimization process;
(2) Name of the TransCAD map file that defines the street network.

5.2.3 Get Major Global Control Data

When selected, this menu item asks the user to provide the following information:

(1) Layer name of the street network line layer in the TransCAD .map file;
(2) Field name of the street network's segment/link cost, such as Length, Travel Time, LENGTH, Travel Impedance, etc.;
(3) Units for the street network's segment/link cost, such as mile, foot, minute, hour, meter etc;
(4) Data file format (ASCII or binary);
(5) OD matrix data file input information (name and path);
(6) Pre-defined route data file information (input from user provided data file or create from scratch);
(7) Route constraint node set node file information (input from user provided data file or create from scratch).

Layer name and field name provided by the user must match those appeared in the TransCAD geometric map file. The layer and field names are case-sensitive.

5.2.4 Get Detailed Global Control Data

This menu item, when selected, will ask the user to input various global control data used during the TRN optimization process. There are about 50 control parameters, most of them have been pre-set with default values. The user only needs to edit a few parameters relevant to their particular applications. It is recommended that the user not change a default parameter setting before fully understanding its physical meaning. The following is a list of the control parameters:

(1) *KBin* – input/output file format options (default = Opt1)
- Option 1 = Input/output files between TransCAD and OPTNet in ASCII format
- Option 2 = All files in ASCII format
- Option 3 = All files in binary format

Options 1 and 2 are for advanced users. These two options use readable input/output files, thus allowing OPTNet to be used as a stand-alone network optimization processor. Option 2 may require large memory space for large problems.

(2) *NRoutT* – Number of Active/Inactive Transit Routes (default = 20). Active routes are a set of routes involved in the TRN optimization process, while inactive routes are those not involved in the TRN optimization process. Large scale transit network optimization may take a great amount of CPU time. Even though a large system may be specified, it may be more effective to select a small number of routes as the active routes and test optimization process with this subset of the complete input to check the input/output data and various constraint settings. The full scale optimization process may then start with appropriate parameter settings.

(3) *IndxInRt* – Pre-defined Route/Line Options (default = Option 1)

- Option 1 = No user pre-defined routes
- Option 2 = User pre-defined routes, fixed or initial guess routes

If option 2 is selected, the user will be asked to input the file name and path of the pre-defined route data file.

(4) *InRtMeth* – Initial Network Selection Options (default = Option 2)

- Option 1 = Randomly select initial routes, for very large problems
- Option 2 = Select initial routes through simple evaluation of objective function

These two options are for problems where users do not provide all the initial guess routes through pre-defined route file.

(5) *Lmin* – Globally defined minimum transit route length (default = 1 mile). This constraint parameter defines the lower bound of all transit route lengths except for fixed routes.

(6) *Lmax* – Globally defined maximum transit route length (default = 30 mile). This constraint parameter defines the upper bound of all transit route lengths except for fixed routes.

(7) *Nmin* – Globally defined minimum number of nodes in a route (default = 2). This constraint parameter defines the lower bound of the number of nodes/stops on a transit route except for fixed routes. Route stops/nodes on a route are an ordered street node subset along the route.

(8) *Nmax* – Globally defined maximum number of nodes in a route (default = 100). This constraint parameter defines the upper bound of number of nodes/stops on a transit route except for fixed routes.

(9) *KOpt* – data output options (default = Option 1)

- Option 1 = No detailed data output
- Option 2 = Detailed data output for input data check

All detailed data output files have a .chk file extension and provide detailed explanations of results from various stages in the optimization process.

(10) *KDir* – Transit/street network characteristics options (default = Option 1)

- Option 1 = Undirected network
- Option 2 = Directed network (allow one-way streets N/A)

An undirected street network is one where all street segments are two-way streets. A directed street network involves one-way street segments. A directed network may be approximated by an undirected network if for any one-way street, one can always find a nearby one-way street in the opposite direction with similar cost, length, or travel time.

(11) *KODM* – OD matrix data input option (default = Option 1)

- Option 1 = Uniform Trip Distribution
- Option 2 = Random Trip Distribution
- Option 3 = User-Provided Trip Distribution

If Uniform or Random trip distribution option is chosen, the average number of OD trips between a typical OD pair is required. If User-Provided trip distribution option is chosen, an OD matrix data file must be provided later.

(12) *Walk* – Maximum walking distance (default = ¼ mile). Trips at OD pairs that have travel distances less than the distance specified by "Walk" will be considered as covered trips. The search process will not attempt to provide coverage for those trips

(13) *IteMax* – Maximum number of search iteration at network level (default 1000). Search process will stop and output results when this user specified limit is exceeded.

(14) *TimeMax* – Maximum CPU time (hours, default 2 hours). Search process will stop and output the results when this user specified time limit is exceed.

(15) *Temp* – Initial temperature of the Simulated Annealing Search (default = 2000.0). Large values will help prevent the search process from being trapped into poor local optima. However, large values require more CPU time.

(16) *Tabu* – Tabu list length (default = 50). Large tabu list length will prevent the same solution or solution series from repeatedly entering the search process. However, large tabu list length requires more CPU time and memory.

(17) *NRand* – Randomly selected path list length (default = 50). During a local search, the search process will randomly select solutions from the global solution space to prevent the search process from being trapped into poor local optima. However, large number of random paths requires more CPU time and memory.

(18) *BdPen* – Penalty number of uncovered trips (default = 4). A zero-transfer passenger has one vehicle boarding, while a one-transfer (two-transfer) transit rider has two (three) vehicle boardings. In an optimization process where the boarding objective function only takes into account two-or-less (three-or-less) boarding trips, any trips that cannot be completed with two-or-less (three-or-less) boardings will be assigned a fictitious boarding number, i.e., the penalty number of uncovered trips.

(19) *Tol* – Relative error tolerance (default = 0.0). The search process will be considered as converged if the relative difference between two succeed results is smaller than this tolerance value.

(20) *NFactR* – Temperature decrease factor (in %) at network level (default = 50). In simulated annealing process, to enhance the selectivity and to speed up the search process, the annealing temperature may need to be decreased as the search process proceeds, i.e., $T_{i+1} = \tau \times T_i$, where $\tau \leq 1.0$. However, rapid temperature reduction (small τ value) may result in premature convergence to poor local optima, while slow temperature reduction (large τ value) requires more CPU time.

(21) *NFactE* – Network Level Probability Acceptance (%) for Same Goal Value Solutions (default = 50). At flat local solution valleys or plateaus, some solutions may have the same goal values. Acceptance of some solutions of same goal values may increase the chance to escape from poor local optima. On the other hand, large probability acceptance may require more CPU time.

(22) *RFactR* – Temperature decrease factor (in %) at route level (default = 50). Same as *NfactR*.

(23) *RFactE* – Route Level Probability Acceptance (%) for Same Goal Value Solutions (default = 50). Same as *NfactE*.

(24) *NSame* – Index for Check Repeated Network Results (default = Option 1)

- Option 1 = Do not check repeated network
- Option 2 = Check repeated network

(25) *NRang* –Annealing Search Results Update Options (default = Opt2)

- Option 1 = Update at the end of every network search iteration
- Option 2 = Update at the end of every route search iteration

(26) *MaxIte2* – Maximum Number of Solution Searches in Two-Point Search (default = 2000). This number sets the limit of solution searches in a two key-node local path space (including the associated rollout path spaces). A TRN optimization process searches for best results from local spaces of various route(s) in an iterative manner. It may not be effective to spend too much effort on particular routes' local spaces.

(27) *MaxIte3* – Maximum Number of Solution Searches in Three-Point Search (default = 2000). Same as MaxIte2 except that this is for three key-node local space search.

(28) M*axIte4* – Maximum Number of Solution Searches in Four-Point Search (default = 2000). Same as MaxIte2 except that this is for three key-node local space search.

(29) *MaxIte5* – Maximum Number of Solution Searches in Five-Point Search (default = 2000). Same as MaxIte2 except that this is for three key-node local space search.

(30) *MaxIte0* – Maximum Number of Solution Searches in Simple Search (default = 2000). This number sets the limit of solution searches in a simple local path space of an existing route. A path in an existing route's simple path space may be obtained by extending the existing path to its end points' nodal adjacent nodes.

(31) *GdirA* – Maximum Average Route Directness Based on Geometry (default = 1.5). Average route directness (geometry) represents the average ratio of two travel distances between a pair of route stops on a transit route, one along the transit route, one along the shortest path in the street network. The optimal value is 1.0, when all route segments between route stop pairs in a route are also shortest path segments in the street network.

(32) *RdirA* – Maximum Average Route Directness Based on Ridership (default = 1.5). Average route directness (ridership) represents the average ratio of two travel distances between an OD point pair on a route, one along the transit route and one along the shortest path in the street network, weighted by OD trip distribution along the route. The optimal value is 1.0, when all transit riders on this route travel along the shortest paths between their OD points. Note that the calculation of this ridership based route directness currently only considers riders with both their origin and destination points on the same route. In other words, transferred riders have not been accounted for.

(33) *OdirA* – Maximum Average OOD Route Directness (default = 15.0).

(34) *GdirM* – Maximum Route Directness Based on Route Geometry (default = 2.0). Same as *GdirA* except that this constraint defines the maximum route directness of any or all route segments in a route instead of the average route directness of a route. In other words, the ratio of the two distances, one along the route and one along the shortest path, must be not exceed this constraint value between any two route stops on all routes.

(35) *RdirM* – Route Directness Limit for Route Segments (default = 2.0). This parameter defines route segment directness' upper bound. A route segment is a portion of a route between its two route stops. Any route segments with route directness values larger than the upper bound will be considered as route segments with bad route directness. This constraint must be combined with the tolerance parameter defined in (36).

(36) *RdirMp* – Tolerance (as a percentage) for Route with Bad Route Directness (default = 20). A route will be discarded if more than *RdirMp*% of riders travel with route directness values exceeding the limit defined by *RdirM*.

(37) *OdirM* – Maximum OOD Route Directness (default = 20). The parameter defines the upper bound of the OOD route directness of any route segments in a route.

(38) *RtCtIndx* – Route Constraint Definition Options (default = Option 2)

- Option 1 = Route constraint defined locally, i.e. defined route by route
- Option 2 = Route constraint defined globally, i.e. one set of route constraint data applied to all routes.

If route constraint defined locally, separated route-by-route constraint data will be required.

(39) *RtCtType* – Route Constraint Type Options (default = Option 2)

- Option 1 = Relative Route Constraint Type Based on Existing Routes

- Option 2 = Absolute Route Constraint Type

For relative route constraint type, constraints are based on the corresponding existing route's characteristics. For example, a 20 % route length tolerance constraint limits the route search process to the path space where all paths are no more 20 % longer than the existing route.

(40) *LTol* – Route Length Tolerance (in %) for Relative Constraint Type (default = 10). The length of a route candidate must be no more than LTol % difference from the corresponding initial route.

(41) *NTol* – Route Stop Number Tolerance (in %) for Relative Constraint Type (default = 100). The number of nodes of a route candidate must be no more than NTol % difference from the corresponding initial route.

(42) *GTol* – Geometry Based Route Directness Tolerance (in %) for Relative Constraint Type (default = 10). The geometry based route directness (GdirA and/or GdirM) of a route candidate must be no more than GTol % difference from the corresponding initial route.

(43) *RTol* – Ridership Based Route Directness Tolerance (in %) for Relative Constraint Type (default = 10). The ridership based geometry based route directness (RdirA and/or RdirM) of a route candidate must be no more than RTol % difference from the corresponding initial route.

(44) O*Tol* – OOD Route Directness Tolerance (in %) for Relative Constraint Type (default = 10)

The OOD route directness (OdirA and/or OdirM) of a route candidate must be no more than RTol % difference from the corresponding initial route.

(45) Optimization Objective Function Options (default = Option 3)

- Option 1 = Based on Zero Transfer Trips f_0
- Option 2 = Based on One-or-Less Transfer Trips f_1
- Option 3 = Based on Boarding Function One t_1
- Option 4 = Based on Two-or-Less Transfer Trips f_2
- Option 5 = Based on Boarding Function Two t_2

(46) Solution Search Method Options (default = Option 1)

- Option 1 = ISTG based on greedy search method
- Option 2 = ISTG based on greedy and hill climbing search method

(47) Key-Node Representation Options (default = 1000000001). Basic options include:

- Option 1 = Local path space based on two key-node path representation (code = 1000000000)
- Option 2 = Local path space based on three key-node path representation (code = 0100000000)
- Option 3 = Local path space based on four key-node path representation

(code = 0010000000)
- Option 4 = Local path space based on five key-node path representation (code = 0001000000)
- Option 5 = Simple path space (code = 0000000001)

There are also two combined options:
- Space based on two key-node representation and simple path space (code = 1000000001)
- Space based on two key-node and five key-node representation (code = 1001000000)

(48) Route Directness Constraint Options (default = 0000000000). Basic Options include:
- Option 1 = No transit directness constraint (code = 0000000000)
- Option 2 = Average route directness (geometry) constraint (code = 1000000000)
- Option 3 = Average route directness (ridership) constraint (code = 0100000000)
- Option 4 = Average route directness (OOD) constraint (code = 0010000000)
- Option 5 = Maximum route directness (geometry) constraint (code = 0001000000)
- Option 6 = Maximum route directness (ridership) constraint (code = 0000100000)
- Option 7 = Maximum route directness (OOD) constraint (code = 0000010000)

5.2.5 Get Route Control Data

This menu allows the user to specify various control or constraint data for individual routes. Depending on certain control parameters or options selected in the global control data menu, this menu will ask the user to provide additional data or information. First, if the users chooses Option 2 = [With user pre-defined routes, fixed or initial guess routes] from the global control data input menu (see item (3) in the previous section), the file name and path of the pre-defined route data file will be asked for input. Next, if users chooses Option 1 = [Route constraint defined locally, i.e. defined route by route] from the global control data input menu (item (38) in the previous section), a set of route level control data must be provided. There are 19 route level control parameters, most of them have been pre-set with default values by the program. The user may only need to edit a few parameters relevant to their particular applications. Again, it is recommended not to change a default parameter setting before fully understanding its physical meaning. The following is a list of control parameters in a typical route level constraint set.

(1) *Cont_Set* – Constraint set number. Users may define up to *NRoutT* sets of route constraints, where *NRoutT* is the total number of routes (active or inactive, refer to item (2) of the previous) in the TRN system.

(2) *Rt_Act* – Route activity options:
- Option 1 = Inactive route
- Option 2 = Active route

If Option 1 is chosen, any routes with this constraint set number (*Cont_Set*) will not be included the TRN optimization process.

(3) *Loc_Cont* – Location constraint options (default = Option 1)

- Option 1 = None – no location constraint;
- Option 2 = Fixed – route and its stops are fixed;
- Option 3 = 3_Areas – the starting, ending and an in-between areas of the route are specified;
- Option 4 = 2_Areas – the starting and ending areas of the route are specified.

If Option 3 or Option 4 is chosen, corresponding node sets will be requested later.

(4) *Cont_Typ* – Constraint type (default = Option 2)

- Option 1 = Relative constraint type
- Option 2 = Absolute constraint type

The following constraints, when specified, only apply to routes with this constraint set number (Cont_Set).

(5) *Lmin* – Locally defined minimum transit route length (default = 1 mile).

(6) *Lmax* – Locally defined maximum transit route length (default = 30 mile).

(7) *Nmin* – Locally defined minimum number of nodes in a route (default = 2).

(8) *Nmax* – Locally defined maximum number of nodes in a route (default = 100).

(9) *GdirA* – Locally defined maximum average route directness based on geometry (default = 1.5). Referred to item (31) in the previous section for more description.

(10) *RdirA* – Locally Defined Maximum Average Route Directness Based on Ridership (default = 1.5). Referred to item (32) in the previous section for more description.

(11) *OdirA* – Locally Maximum Average OOD Route Directness (default = 15.0). Referred to item (33) of previous section for more description.

(12) *GdirM* – Locally Defined Maximum Route Directness Based on Route Geometry (default = 2.0). Referred to item (34) in the previous section for more description.

(13) *RdirM* – Locally Defined Route Directness Limit for Route Segments (default = 2.0). Referred to item (35) in the previous section for more description.

(14) *RdirMp* – Locally Defined Tolerance (in %) for Route with Bad Route Directness (default = 20). Referred to item (36) in the previous section for more description.

(15) *OdirM* – Locally Defined Maximum OOD Route Directness (default = 20). Referred to item (37) in the previous section for more description.

(16) *LTol* – Locally Defined Route Length Tolerance (in %) for Relative Constraint Type (default = 10). Referred to item (40) of previous section for more description.

(17) *NTol* – Locally Defined Route Stop Number Tolerance (in %) for Relative Constraint Type (default = 100). Referred to item (41) of previous section for more description.

(18) *GTol* – Locally Defined Geometry Based Route Directness Tolerance (in %) for Relative Constraint Type (default = 10). Referred to item (42) of previous section for more description.

(19) *RTol* – Locally Defined Ridership Based Route Directness Tolerance (in %) for Relative Constraint Type (default = 10). Referred to item (43) of previous section for more description.

(20) *OTol* – Locally Defined OOD Route Directness Tolerance (in %) for Relative Constraint Type (default = 10). Referred to item (44) of previous section for more description.

5.2.6 Data Check

All the global and local input data will be checked at this stage. If there are no input data and/or file location errors, TRN optimization process is ready to start.

5.2.7 Run OPTNet Program

There are five executable files in a TRN optimization process:

(1) Prep1.exe – converts all the TransCAD interface files into text file (ASCII or binary) forms accepted by other executable files;

(2) OPTNet1.exe – generates the shortest path search space;

(3) OPTNet2.exe – generates an initial transit route network if the user does not provide all the initial routes;

(4) OPTNet3.exe – searches for the optimal transit route network starting from the initial guess route system; and

(5) Post1.exe – converts results (in text file form, ASCII or binary) obtained from OPTNet3 into TransCAD data table formats.

For large-scale TRN optimization problems, the process OPTNet3.exe may take hours or days depending on the problem size. It is recommended to start a large-scale TRN problem with small number of key-node representation, low initial simulated annealing temperatures, small temperature reduction factors, small probability acceptance percentages, small CPU time allowance, small iteration limits, and short tabu list length. These measures will make search processes short and produce results that may not be close to optimal. After checking the results obtained from this preliminary program execution and if all the input/output data and the corresponding TRN characteristics are correct or are as expected, some of the control parameters may be reset and a full scale search process may be initiated.

5.2.8 Post Processing

By selecting this menu item, the user will be able to display the results from the TRN optimization process in TransCAD. Currently, the post processing for route network layout optimization includes functions or tools with the following features:

(1) Graphic or tabular display of the layout of the route network and attributes and statistics about the network. The attributes of a network include number of routes, total route lengths, minimum and maximum route lengths, stop names and locations, etc. The statistics include route network directness measures (geometric and/or ridership based), transfer directness parameters such as zero, one-or-less, two-or-less trip coverage functions, and average boarding functions, etc. Transfer statistics at various route intersection points will also be provided, which include, for example, the intersection node or transfer node for each route, the number of passengers originated from or destined at that location, the number of passengers transferring at that location, and the total number of on-and-off passengers at that location, which is the sum of the previous two numbers. Note, the number of transfer passengers is computed based on the best case scenario for the corresponding node. The best case scenario is a transfer node that may serve the largest number of transfer passengers even when some of them may have the options to use other transfer nodes. Transfer information is useful to transit planners who may wish to select certain transit stops as major transfer centers.

(2) Graphic or tabular display of individual routes and their attributes and statistics. Information on route topology, or connection between a given route and others, is also provided. Attribute information includes route length, route segments' and/or route stops' locations and names. Statistics include route directness (geometric and ridership based), OOD (Out-of-Direction), etc.

5.2.9 Save Network Problem

This menu entry allows the user to save the current optimization results, which may then be modified and used as the input for a new iteration of optimization.

6. CONCLUSIONS AND RECOMMENDATIONS

6.1 Summary and Conclusions

There is evidence that transfers and transfer difficulties are major factors that negatively impact overall transit ridership. Therefore, one of the goals of the optimization of a transit route network system is to minimize the number of transfers to improve ridership and the quality of transit services. The primary objective of this work was to develop transit route network optimization methodologies and implement them in a computer program, which should be capable of performing transit route network optimization for practical problems of either small, mid-sized, or large transit service areas in a reasonable amount of time. In this study, methodologies and a computer program were designed to minimize the total number of transfers of a transit network system through the optimization of the route network configuration, which was achieved through the maximization of trips that either need zero transfer or one-or-less transfer depending on the objective function settings. Previous studies on the optimization of transit network design problems are mainly various heuristic approaches where problem dependent design guidelines, criteria, or principles based on past experiences or existing systems must be followed in order to obtain a tractable solution search spaces. Results obtained from various heuristic optimization approaches, although in most cases showing improvements over existing systems, are hardly optimal results, either local or global, which may be attributed to the facts that the search schemes in most of heuristic approaches are *ad hoc* procedures and the corresponding search spaces are usually not clearly defined. Although there are some mathematical optimization based approaches for transit network designs in the literature, most of them are still in the theoretical formulation development stage or may be only applied to ideal or small networks.

Transit network design and optimization, whether including transit scheduling or not, is a complex task. The difficulties in transit network design optimization are inherent from the complex characteristics of transit network design realities, which include combinatorial complexity of network solution spaces, non-linearity of design objective functions or design variables, non-convexity of objective functions or solution/variable spaces, as well as multi-objective nature of the problems. These complexities in the transit network design field thus far seem to preclude development of systematic mathematical optimization methodologies for transit network design problems of realistic sizes. The methodology developed in this research is intended to provide a systematic design tool for the optimization of transit route networks of small, mid-sized or large scale transit service areas. The methodology was tested first through several existing benchmark transit network design problems, and applied to a large scale transit network optimization problem based on the Miami-Dade County's transit system. The main features or contributions of the solution methodology developed from this research may be summarized as follows.

(1) The research developed a systematic mathematical statement of the route network optimization problems including the definition of various objective functions, solution search spaces and constraints commonly used in the transit planning field. The main difference between this study and previous ones is that the mathematical statement and the formulation derived from this study has been implemented as computer program to

solve large, realistic transit route network optimization problems, while other studies usually resort to heuristic approaches or confine their problems to small networks.

(2) Schemes that flexibly define solution search spaces based on available computing resources and optimization problem sizes have been developed. The flexibility is provided through:

- key-node representation of a transit route,
- the order of local node spaces, the order of local path spaces, and the order of local network spaces, and
- the *k*-level shortest path search space.

By selecting the appropriate key-node representation, adjusting the order of local node spaces, local path spaces, and local network spaces, and choosing a k for the k-level shortest path search space, the size of the solution search space may be varied. The flexibly defined local solution search space will approach to the complete solution space as the key-node number, the order of various local spaces, and the number k in the k-shortest path search space increase, although the computing cost to obtain results will also increase as the search spaces expands.

(3) This study developed several local search schemes, which were inspired or based on mathematical heuristic and stochastic methods widely used in operations research, graph theory, or other fields, to obtain results in reasonable time periods with the constraints on the current computing powers. Mathematical heuristics search methods related to the methods used in this study included, e.g., the greedy search methods, the nearest-first search methods, and hill-climbing methods, and the stochastic methods included tabu search, simulated annealing, and greedy search. The fast hill-climbing search scheme is inspired by the gradient based fast decent methods widely used in research fields of continua. The traditional heuristic approaches used in the transit planning field are usually strongly domain or problem dependent and the search schemes are usually ad hoc and, in some cases, a computer simulation or automation of the design procedures used by human planners. The mathematical heuristics search methods are much less domain dependent, and have been used and studied in a wide range of application and research fields. Moreover, mathematical heuristics search methods usually have relative more solid theoretical ground. For instance, Wolsey (1988) has stated that under certain convex conditions, the greedy type search schemes converge to global optima.

(4) The methodology developed in this study is particularly suitable for implementation on parallel platforms. In local search processes, the function evaluations of networks with different path candidates are independent of each other thus allowing multi-threaded parallel function evaluations of possible solution networks in local search spaces. For the fast hill-climbing search method, both the procedure to generate local search spaces for various routes and the function evaluations of various solution networks inside local search spaces are independent processes between different local spaces and between different solution networks. Implementation on parallel computational platforms will result in more effective solution schemes, which promises a great potential for

improvement, especially considering the rapid progress in parallel computing software and hardware.

(5) The methodology developed in this work may also be used as a useful computational tool to evaluate existing transit route networks. Various output parameters such as zero, one-or-less, and two-or-less transfer trip coverage, network directness, transfer directness, and OOD impact indexes may serve as indicators on the qualities of transit networks.

(6) Finally, numerical results obtained from this work showed that the methodology developed in this work appeared to be capable to tackle large scale transit network design optimization problems in a reasonable amount of time. The results from the mathematical optimization approaches, though not guaranteed to be a global optimal, did show significant improvements over those obtained from traditional heuristic approaches.

The methodology developed in this research has been implemented in a prototype GIS based program called OPTNet (OPTimization Package for transit Network). The program was implemented with TransCAD as the front end user interface. It allows the user to specify input data including street network, transit demand (OD matrix), optional route configurations, initial guess routes, fixed routes, pre-specified service areas, optimization parameters, etc. The program can display the input such as streets, initial transit network, and OD matrix graphically as well as the output. The attributes and performance statistics of the optimized transit network are also provided. The user may choose to modify the program output to arrive at a final network configuration by adjusting selected routes and stop locations, or, after making modifications, choose to use the current solution as the initial input into OPTNet for further optimization.

6.2 Recommendations

As mentioned before, a complete transit network design optimization process should include two design optimization components, i.e., transit route network design optimization and transit network scheduling design optimization. The present work dealt with the optimization of transit route network structure in an attempt to find the optimal route network layouts in terms of network directness, transfer directness, and ridership coverage. However, to realize those optimal characteristics allowed by the resultant route network obtained from the route network design stage, the optimization of transit network schedule design should be implemented. Design variables in network scheduling optimization may include vehicle headways and timetables, and the optimization objective functions may be the user cost, operator cost, or a combination of the two. Constraints may include minimum/maximum vehicle headways, passenger waiting times, vehicle load factors, fleet size, and so on. Although traditional heuristic methods may produce a workable transit schedule by following certain guidelines or criteria, it is important to develop effective mathematical optimization methodologies for transit scheduling design since the differences in cost benefits between a workable result and an optimal or even a good result may be significant, especially for large-scale transit networks.

Another important improvement would be to allow the use of travel time instead of travel distance in the optimization process. This is because a shortest path measured by distance may

not be the shortest path measured by travel time. For transit users, they are more sensitive to travel time and less to travel distance. More importantly, unless travel time is used in optimization, travel time savings provided by rapid transit services such as Miami-Dade County's Metrorail and Busway and transit users' preference for these rapid transit services cannot be properly considered and results will not be accurate.

In this study, the street networks were assumed or approximated as undirected networks. This assumption is valid for street networks containing one-way streets if any two corresponding one-way segments of a bus route are close to each other and the lengths (or travel times) of the two segments are more or less the same. This assumption may not be true in some practical situations when the two one way segments are far apart or when travel time on these two segments are noticeably different. Thus it is desirable to extend the current methodology to support optimization problems based on directed street networks.

Because network travel time varies by route and by time of day, to model a transit network accurately with travel time as the cost measure, it is necessary to consider time-of-day models and network optimization for different periods of time. This requires that transit demand for desirable time period during a day is defined. Currently, the accurate estimation of transit OD matrix remains a challenging task. Although several methods based on limited survey data and statistic technologies have been reported in the literature (Tsygainitzky 1979, Simon and Furth 1985, Furth and Navick 1992), they are usually limited to one transit route or one transit corridor, and their validity for use for transit route networks remains unclear. The Automatic Passenger Counters (APC) technology, especially the smart card technology, is a promising ridership and OD data collection means. However, before such technologies become available, additional study will be needed to estimate transit demand and OD matrix. Improvements in OD matrix estimation may be possibly achieved through modeling and the use of existing ridership data. Sample data collected for selected routes may provide good estimates of the spatial distribution and the temporal patterns of transit demand. These estimates may also be extrapolated to areas not being served by transit. One difficulty in using existing ridership data such as boarding and alighting data is the lack of historical data or data in electronic format as most transit properties do not systematically preserve or utilize such data. It is recommended that tools be provided to transit properties to help them preserve such data electronically and allow them to retrieve and analyze the data, and that the potential of utilizing these data for the purpose of helping estimate transit demand be studied.

To improve the speed of the OPTNet program, different strategies and techniques will need to be investigated and applied. Although the computing power of a single processor computer has been increasing rapidly for the past several decades, the full potential of mathematical optimization approaches to find global or near global optimal results for large-scale transit network analysis seems to lie in parallel computing techniques. This is because the power of a single processor computer is limited by two factors: the speed of data exchanges/transfers and the number of microcircuits in a CPU chip. Both of these two factors have limits based on the current knowledge of physics. The speed of data exchanges could not exceed the speed of light while the minimum size of a micro circuit could not be smaller than the molecular of the chip material. Unless there are breakthroughs in physics, these two limitations will determine the bounds that the power of single processor computers cannot exceed. Nowadays, with a parallel

computing platform, computers interconnected via a local network can process data and perform calculations in parallel with speeds of from several to hundreds of thousands times faster than a single processor computer. The computing power of a parallel computer depends on both individual processor's speed and the number of processors in the computers. Although the power of a single computer process is limited, such limit is removed if multiple processors are connected to form a parallel computer. Therefore, implementing OPTNet on a parallel platform is naturally the next development stage for any promising mathematical optimization methodologies.

REFERENCES

Aarts, E. and J. K. Lenstra. (1997). *Local Search in Combinatorial Optimization*, Wiley, N. Y.

Axhausen, K.W. (1984). *Heuristic Transit Network Optimization: Review and Application*, M.S. Thesis, Dept. of Civil and Environmental Engineering, University of Wisconsin-Madison.

Axhausen, K.W. and Robert L. Smith Jr. (1984). "Evaluation of Heuristic Transit Network Optimization Algorithms," Dept. of Civil and Environmental Engineering, University of Wisconsin-Madison. Presented at Annual Meeting of Transportation Research Board, National Research Council, Washington, D.C., January 1984.

Baaj, M. Hadi and H.S. Mahmassani (1990). "TRUST: A Lisp Program For the Analysis of Transit Route Configurations," *Transportation Research Record* 1283, Transportation Research Board, National Research Council, Washington, D.C., pp. 125-135.

Baaj, M. H. and H.S. Mahmassani (1995). "Hybrid Route Generation Heuristic Algorithm for the Design of Transit Networks," *Transportation Research C*, Vol. 3, No.1, pp. 31-50.

Baaj, M. Hadi and H.S. Mahmassani (1992). "Artificial Intelligence-Based System Representation and Search Procedures for Transit Route Network Design," *Transportation Research Record* 1358, Transportation Research Board, National Research Council, Washington, D.C., pp. 67-70.

Baaj, M.H (1990). *The Transit Network Design Problem: An AI-Based Approach*, Ph.D. thesis, Dept. of Civil Engineering, University of Texas at Austin, Austin, Texas.

Baaj, M.H and H.S. Mahmassani (1991). "An AI-Based Approach for Transit Route system Planning and Design," *Journal of Advanced Transportation* Vol. 25, No. 2, pp 187-210.

Bertsekas, D. P. (1991). "Linear Network Optimization: Algorithms and Codes", MIT Press.

Bertsekas, D.P. (1998). "Network Optimization: Continuous and Discrete Models," Athena Scientific, Belmont, Massachusetts.

Bielli, M., M. Caramia, and P. Carotento. (2002). "Genetic algorithms in Bus Network Optimization." *Transportation Research Part C* 10(1), 19-34.

Bookbinder, H. James and A. Désilets (1992). "Transfer Optimization in a Transit Network," *Transportation Science,* Vol. 26, No. 2, pp. 106-118.

Byrne, B. F. (1975). "Public Transportation Line Positions and Headways for Minimum User and System Cost in a Radial Case," *Transportation Research,* Vol. 9, pp. 97-102.

Byrne, H.F. and V. Vuchic (1972). "Public Transportation Line Positions and Headways for Minimum Cost," *Proceedings of the Fifth International Symposium on Traffic Flow and Transportation*, American Elsevier, New York, pp. l47-360.

Caramia, M., P. Carotenuto and G. Confessore (2001). "Metaheuristics Techniques in Bus Network optimization," NECTAR Conference No. 6, European Strategies in the Globalising Markets; Transport Innovations, Competitiveness and Sustainability in the Information Age, 16-18 May 2001, Espoo, Finland.

Carr Smith Corradino (2000). *Southeast Florida Regional Travel Characteristics Study*, Final Report, prepared for Florida Department of Transportation, Districts IV and VI, October 2000.

Ceder, A. and Y. Israeli (1997). "Creation of Objective Functions for Transit Network Design," *Transportation Systems (TS'97), Proceedings of the 8th IFAC/IFIP/IFORS Symposium*, Edited by M. Papageorgiou and A. Pouliezos, Vol. 2.

Chien, S., B. V. Dimitrijevic, and L. N. Spasovic. (2003). "Optimization of Bus Route Planning in Urban commuter Networks." *Public Journal of Transportation* 6(1), 53-80.

Chua, T.A. (1984). "The Planning of Urban Bus Routes and Frequencies: A Survey", *Transportation* Vol.12, pp 147-172.

Chua, T.H. and D.T. Silcock (1982). "The Practice of British Bus Operators in Planning Urban Bus Services," *Traffic Engineering and Control*, pp. 66-70.

Désilets, A. (1989). "Transfer Optimization in a Transit Network," M.A.Sc. Thesis, Department of Management Sciences, University of Waterloo.

Dijkstra, E. (1959). "A note on Two Problems in Connexion with Graphs," *Numer. Math*., Vol. 1, pp. 269-271.

Dubois, D., G. Bel, and M. Llibre (1979). "A Set of Methods in Transportation Network Synthesis and Analysis," *Journal of the Operations Research Society*, Vol. 30, pp. 797-808.

Dym, Clive L. and Raymond E. Levitt (1991). "Knowledge-Based Systems in Engineering," McGraw-Hill, Inc.

Fox, Bennett L. (1993). "Integrating and accelerating tabu search, simulated annealing, and genetic algorithms." *Annals of Operations Research* (41), 47-67.

Furth, P. G. and D. S. Navick (199x). "Bus Route O-D Matrix Generation: Relationship Between Biproportional and Recursive Methods," *Transportation Research Record* 1338.

Gallo, G. S. and S. Pallottino (1988). "Shortest Path Algorithms," *Annals of Operations Research*, Vol. 7, pp. 3-79.

Garey, M.R. and D.S. Johnson (1979). *Computers and Intractability, A Guide to the Theory of NP-Completeness*, Freeman, New York.

Giarratano, Joseph and Gary Riley (1993). "Expert Systems: Principles And Programming," PWS Publishing Company, Boston.

Glover, F. and M. Laguna (1997). *Tabu Search*. Kluwer Academic Publishers, Boston, MA.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Weslay Publishing Co., Reading, Mass.

Gould, Ronald (1988). *Graph Theory*, the Benjamin/Cummings Publishing Company, Inc., Menlo Park, California.

Hajek, B. (1988). "Cooling Schedules for Optimal Annealing." *Math. of Operations Research* (13), 311-329.

HasseIstrom, D. (1981). *Public Transportation Planning*, Ph.D. Thesis, Department of Business Administration, University of Gothenburg, Sweden.

Haupt, R. L. and S. E. Haupt (1998). *Practical Genetic Algorithms*, John Wiley, New York, New York.

Hill, S. D. and M. C. Fu (1995). "Transfer Optimization via Simultaneous Perturbation Stochastic Approximation," Proceedings of the 1995 Winter Simulation Conference, (Eds.) C. Alexopoulos, K. Kang, W. R. Lilegdon, and D. Goldsman.

Hillier, F.S. and Lieberman, G.J. (1986). *Introduction to Operation Research,* Holden-Day, Inc., Oakland, California.

Holland, J.H. (1992). *Adaptation in Natural and Artificial Systems.* 2nd Ed., MIT Press, Cambridge, Mass.

Ignizio, J.P. (1980). "Solving Large-Scale Problems: A Venture Into A New Dimension," *Operation Research* Vol. 31, pp217-225.

Israeli, Y. (1990). *Optimal Routing Allocation to Transfer Demand in Networks*, Doctoral dissertation, Civil Engineering Department, Technion - Israel Institute of Technology, Israel.

Janarthanan, N. and J.B. Schneider (1998). "Development of an Expert System to Assist in the Interactive Graphic Transit System Design Process," *Transportation Research Record,* 1187, Transportation Research Board, National Research Council, Washington, D.C., pp. 30-46.

Kim, D. and C. Barnhart (1997). "Transportation Service Network Design: Models and Algorithms," in *Computer-Aided Transit Scheduling*, N.RM. Wilson (ed.), Springer, New York, August.

Kirkpatrick, S., C. D. Gelatt, Jr., M. P. Vecchi. (1983). "Optimization by Simulated Annealing." *SCIENCE* (220)4598, 671-680.

Kuah, G. K. and J. Perl (1988). "Optimization of Feeder Bus Routes and Bus-Stop Spacing," ASCE *Journal of Transportation Engineering*, Vol. 114, No. 3, pp.

Lampkin, M. and P.D. Saalmans (1967). "The Design of Routes, Service Frequencies and Schedules for a Municipal Bus Undertaking: A Case Study," *Operations Research Quarterly*, Vol. 18, pp. 375-397.

Magnanti, T.L. and R.T. Wong (1984). "Network Design and Transportation Planning: Models and Algorithms," *Transportation Science* Vol. 18, pp. 1-55.

Mandl, C. (1979). *Applied Network Optimization*, Academic Press.

Mandl, C. (1979a). "User Reference and Implementer Manual IANO-System," Institutsarbeit No. 122, Institute for Advance Studies, Vienna.

Mandl, C.E. (1979). "Evaluation and Optimization of Urban Public Transportation Networks", *European Journal of Operational Research*, Vol. 5, pp. 396-404.

Martins, C.L. and M.V. Pato (1998). "Search Strategies for the Feeder Bus Network Design Problem," *European Journal of Operational Research*, Vol. 106, pp. 425-440.

Miami-Dade County (2001). *2025 Long-Rang Transportation Plan*, Miami-Dade County Metropolitan Planning Organization, Miami-Dade County, Florida.

Michalewicz, Z., Vignaux, G.A., and Hobbs, M. (1991). "A Non-Standard Genetic Algorithm for the Nonlinear Transportation Problem," *ORSA Journal of Computing*, Vol.3, No.4, pp. 307-316.

Mor, J.J. and Stephen J.W. (1993). *Optimization Software Guide,* SIAM Publications.

Nemhauser, G.L. and L.A. Wolsey (1988). *Integer and Combinatorial Optimization*, John Wiley & Sons, New York.

Newell, C.E. (1979), "Some Issues Related to the Optimal Design of Bus Routes," *Transportation Science*, 13 p20-35.

Newell, G. F. (1973). "Scheduling, Location, Transportation, and Continuum Mechanics: Some Simple Approximations to Optimization Problems," *SIAM, Journal on Applied Mathematics*.

Newman, D.A., M. Bebendorf, and J. McNally (1983), "Timed-Transfer: an Evaluation of Its Structure, Performance and Cost," Urban Mass Transportation Administration, U.S. Department of Transportation, Washington D.C.

Norojono, O. (1990). *The Optimization of the Public Transport Network in Yogyakarta: A Case Study*, M.S. Thesis, Faculty of Civil Engineering, Dept. of Transportation Planning and Highway Engineering, Delft University of Technology, the Netherlands.

Osman, I. H. and G. Laporte. (1996). "Metaheuristics: a Bibliographies." *Annals of Operations Research* (63), 513-623.

Pappadimitriou, C. H. and K. Steiglitz (1982). *Combinatorial Optimization Algorithms and Complexity*, Prentice Hall, Englewood Cliffs, NJ.

Pattnail, S.B., S. Mohan, and V.M. Tom (1998). "Urban Bus Transit Route Network Design Using Genetic Algorithm", *Journal of Transportation Engineering*, Vol.124, No.4, July/August 1998.

Rapp, M. H. and C. D. Gehner (1976). "Transfer Optimization in an Interactive Graphic System for Transit Planning," *Trans. Rec.* No. 619.

Rea, J.C. (1972). "Designing Urban Transit Systems: An Approach to the Route Technology Selection Problem", *Highway Research Record,* No. 417, pp. 4859.

Sheffi, Yosef (1985), "Urban Transportation Network: Equilibrium Analysis with Mathematical Programming Methods," Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Shier, D. R. (1979), "On Algorithms for Finding the *k* Shortest Paths in a Network," *Networks*, Vol. 9 pp. 195-214.

Shih, Mao-Chang and H.S. Mahmassani (1994). *A Design Methodology for Bus Transit Networks with Coordinated Operations*, SWUTC/94/60016-1, Center for Transportation, Bureau of Engineering Research, the University of Texas at Austin, Austin, Texas.

Silman, L.A., Z. Barzily and U. Passy (1974). "Planning the Route system for Urban Buses," *Computers and Operations Research*, Vol. 1, pp. 201-211.

Simon, J. and Furth, P. G. (1985). "Generating a bus Route O-D Matrix from On-Off Data," ASCE *Journal of Transportation Engineering*, Vol. 3, No. 6, pp. 583-593.

Soehodo, S. and Koshi, M. (1999). "Design of Public Transit Network in Urban Area with Elastic Demand," *Journal of Advanced Transportation*, Vol. 33, No. 3, pp.335-369.

Stern, R. (1996). *Passenger Transfer System Review*, Synthesis of Transit Practice 19, Transportation Research Board, National Research Council, National Academy Press, Washington, D.C.

Toliver, P., B. White, M. Baker, R. Walsh, and J. Jacobson (1989). *Year 2000 Public Transportation Plan, Phase I: Network Design Transit Concepts and Determinants*, Metro Transit Department, Seattle, Washington.

Tsygainitzky, S. (1979). *Simplified Methods of Transportation Planning*, M.S. Thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts.

Van Nes, R., R. Hamerslag, and L. H. Immers (1988). "The Design of public transport network," Paper presented at the 1988 Transportation Research Board Annual Meeting, Transportation Research Board, National Research Council, Washington, D.C., January 1988.

Van Oudheusden, D. L., S. Ranjithan and K. N. Singh (1987). "The Design of Bus Route systems - an Interactive Location-Allocation Approach," *Transportation* 14: pp 253-270, Martinus Nijhoff Publishers, Dordrecht, the Netherlands.

Vignaux, G.A. and Z. Michalewicz (1989). "Genetic Algorithm for the Transportation Problem," *Proceedings of the 4th International Symposium on Methodologies for Intelligent Systems*, North-Holland, Amsterdam, pp. 252-259.

Vignaux, G.A. and Z. Michalewicz (1991). "A Genetic Algorithm for the Linear Transportation Problem," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 21, pp. 445-452.

Welch, W., R. Chisholm, D. Schumacher, and S. Mundle (1991). "Methodology for Evaluating Out-of-Direction Bus Route Segments," *Transportation Research Record 1308*, Transportation Research Board, National Research Council, Washington, D.C., pp. 43-50

Wolsey, L.A. (1988). *Integer Programming*, John Wiley & Sons Inc., New York, New York.

APPENDIX. A BRIEF DESCRIPTION OF MATHEMATICAL OPTIMIZATION

Appendix 1 Mathematical Optimization

The following description of various mathematical optimization problems serves only as informational purpose, and it is not intended as rigorous mathematical statements. More accurate description may be found in, for example (Wolsey 1998), among others.

General Mathematical Optimization Problem

A general mathematical optimization problem statement is the follow.

Maximize/Minimize:
$f(\boldsymbol{x}, \boldsymbol{n})$ for all $\boldsymbol{x}$ in $\boldsymbol{R}$, and $\boldsymbol{n}$ in $\boldsymbol{I}$
Subject to:
Equality constraints:
$g_i(\boldsymbol{x}, \boldsymbol{n}) = 0, i = 1, 2, \Lambda\ , l$
Inequality constraints:
$h_j(\boldsymbol{x}, \boldsymbol{n}) \le 0, j = 1, 2, \Lambda\ , m$.

In the above statement, $\boldsymbol{x} = \boldsymbol{x}(x_1, x_2, \Lambda\ , x_r)$ is a vector (or, more general, a matrix) with continuous variable components $x_1, x_2, \Lambda\ , x_r$, and $\boldsymbol{n} = \boldsymbol{n}(i_1, i_2, \Lambda\ , i_s)$ is a vector (or, more general, a matrix) with integer variable components $i_1, i_2, \Lambda\ , i_s$, and $\boldsymbol{R}$ is a space of vectors with continuous variable components, whereas $\boldsymbol{I}$ is a space of vectors with integer variable components. For example, the continuous vector space $\boldsymbol{R}$ may have the form of $\boldsymbol{R}$: $a_i \le x_i \le b_i$, $i = 1, 2, \Lambda\ , r$, where *ai* and *bi* define the range of the i^{th} variable component of continuous vector $\boldsymbol{x}$; similarly the integer vector space $\boldsymbol{I}$ could have the form of $\boldsymbol{I}$: $n_j \le i_j \le m_j$, $j = 1, 2, \Lambda\ , s$, and *nj*, *mj* define the range of j^{th} integer component of the integer vector $\boldsymbol{n}$.

Traditional or Continuous Optimization Problem

If the general optimization problem does not involve integer variables, i.e., $\boldsymbol{n} = \boldsymbol{0}$, the problem is considered as a traditional or continuous optimization problem.

Integer Optimization Problem

On the other hand, if the general optimization problem involves only integer variables, i.e., $\boldsymbol{x} = \boldsymbol{0}$, the problem is considered as an integer optimization problem.

Mixed Integer Optimization Problem

Mixed integer optimization problem involves both integer and continuous variables, i.e. $\boldsymbol{x} \neq \boldsymbol{0}$ and $\boldsymbol{n} \neq \boldsymbol{0}$.

Combinatorial Optimization Problem

A combinatorial optimization problem is a special case of integer optimization problems. It refers to an integer optimization problem where the integer vector's component set, i.e. the integer set $(i_1, i_2, \ldots, i_s)$ in vector $\boldsymbol{n} = \boldsymbol{n}(i_1, i_2, \ldots, i_s)$ is a subset of a larger integer set or space $\boldsymbol{N}(i_1, i_2, \ldots, i_t)$, $t \geq s$. For example, the space that includes all two-integer subsets of the integer space $\boldsymbol{N}$ (1,2,3,4) is, if order of each pair dose not matter, $\boldsymbol{C}$((1,2), (1,3), (1,4), (2,3), (2,4), (3,4)). Here, $\boldsymbol{C}$ is a combinatorial space generated from the original integer space $\boldsymbol{N}$(1,2,3,4) (the combinatorial space $\boldsymbol{C}$ will be twice big if order of each pair dose matter, e.g. if (1,2) and (2,1) are considered as different sets). The integer vectors defined on this combinatorial space will be $\boldsymbol{n}$(1,2), $\boldsymbol{n}$(1,3), $\boldsymbol{n}$(1,4), $\boldsymbol{n}$(2,3), $\boldsymbol{n}$(2,4), and $\boldsymbol{n}$(3,4). It may be seen that as the size of $\boldsymbol{N}(i_1, i_2, \ldots, i_t)$, and the variables number of vector $\boldsymbol{n}(i_1, i_2, \ldots, i_s)$, i.e. r and t, increase, the corresponding combinatorial space $\boldsymbol{C}$ could be very large.

Mixed Combinatorial Optimization Problem

A mixed combinatorial problem refers to a problem that has continuous variables, i.e., $\boldsymbol{x} \neq \boldsymbol{0}$ and, at the same time, the integer vector $\boldsymbol{n}$ is defined on a combinatorial space $\boldsymbol{C}$ obtained from a base integer space $\boldsymbol{N}$ following certain combination rules.

Appendix 2. Convex Domain and Convex Function

The following is an informal description of convex domains/regions and convex functions. This description is intended to give readers without a background in mathematical programming a general understanding of what "convex" means in optimization and why it is important in certain optimization problems. It is not a mathematical definition or statement. Rigorous definitions of convex domain and convex function as well as their corresponding properties may be found in, for example, Wolsey (1998) and Bertsekas (1998).

For illustration, we only consider 1D, 2D and 3D Euclidean spaces. The basic idea is similar for spaces of other types.

Convex Domains

A domain or region is convex if two points are in the domain then all the points on the straight line segment that joints these two points are also in the domain. Here "in the domain" means either inside the domain or on the boundary of the domain. A domain or region is strictly convex if two points are in the domain then all the points on the straight line segment that joints these two points are inside the domain. Based on the above definition, it is easy to see that the 1D region defined by the blue line segment in Figure A.1(a) is convex, while the 1D region defined by the blue line segments in Figure A.1(b) is not, or is nonconvex.

(a) (b)

Figure A.1 1D Examples of Convex and Nonconvex Domains

In Figure A.2, (a) is a 2D strictly convex domain, (b), (c) and (d) are 2D convex domains, while (e) and (f) are not; (g) is a 3D convex domain, while (f) is nonconvex. Verification of the above may be easily done by drawing a straight line between any two points in a domain and checking whether or not any portions of the line segment are outside the corresponding domain.

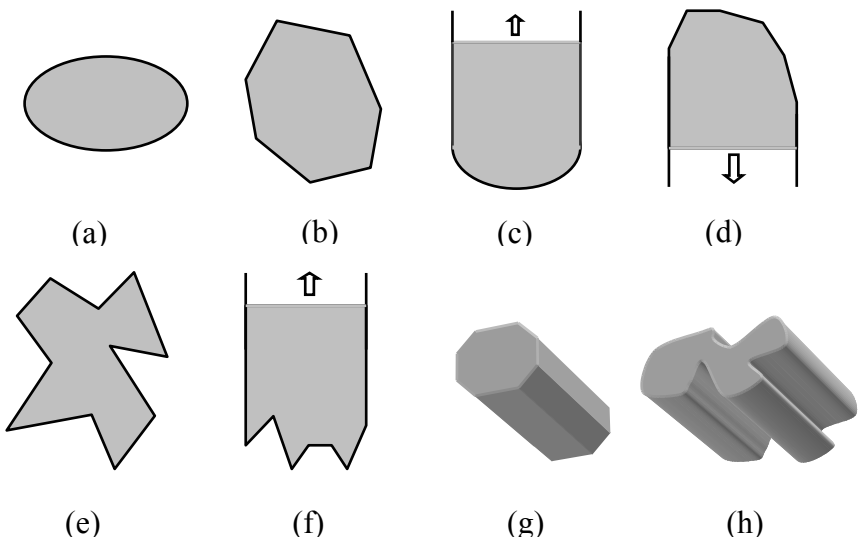


Figure A.2 2D and 3D Examples of Convex and Nonconvex Domains

Convex Functions

A (strictly) convex function is a function line, curve, surface (or a set of function line, curve, or surface segments) that bound or define a (strictly) convex domains/regions. From this definition, the function curves and surfaces of in Figures A.3(a), (b) are convex functions since they bound convex domains, while the rest in Figure A.3 are non-convex functions. Moreover, Figure A.3(a) is a strictly convex function. The properties of convex domains and functions in terms of the solvability of an optimal result may be listed as bellow.

(a) A strictly convex function defined on a convex domain has a unique maximum or minimum point in the domain. The global optimal point may be found by following a favorite direction (e.g. following a cost decreasing direction in TRN design) starting from any initial guess point in the domain. Figure A.3(a) shows a strictly convex function defined on a convex domain. It easy to see from Figure A.3(a) that one may always find the minimum point *b* from any initial guess point in the domain by just following the decreasing direction.

(b) There is a slight different between a strictly convex and a convex function. A convex function defined on a convex domain has a unique maximum or minimum value in the domain. However, there are may be many points that have this maximum or minimum value in the convex domain. Figure A.3(b) shows an example of a convex function defined in a convex domain. It may be see that all the points on segment *cd* of Figure A.3(b) have the same minimum value. Like case (a), an optimal point, i.e. any one of the points on line *cd* in Figure A.3(b), may also be found by following a favorite direction starting from any initial point in the domain.

(c) For nonconvex functions defined on convex domains, there may be many local maximum or minimum points or local optima. Therefore, to search an optimal point from a guess point may only obtain one of the local optimal points. The global optima must be obtained by comparing all the local optima. Figure A.3(c), (d), (e) show various local optimal points for corresponding nonconvex functions. For nonconvex functions, a

search scheme may or may not generate a local optimal since the search process may oscillate among the neighborhoods of several local optima, and not converge to any of them.

(d) Some nonconvex problems may be decomposed into several convex sub-problems. For example, the problem in Figure A.3(c) may be decomposed into two convex sub-problems defined on, respectively, the sub-domain, *ab* and *bc*. In fact, all the nonconvex problems shown in Figure A.3 may be decomposed into several convex sub-problems. Decomposition is one of the mathematical regularities or augmentations to recover the solution uniqueness of a complex system. In such cases, the global optimum is obtained by selecting the best solution from various sub-problems' results. For certain problems the number of sub-problems may be huge. In such cases, in stead of searching all the local optimal solution for each sub-problem, one may estimate or approximate up-bounds or lower-bounds of some of the sub-problems' solution ranges, and excludes those sub-problems that are not likely to have a global optima. To exclude unfavorable sub-problems is also a typical scheme in optimization to reduce the size of solution search spaces. For example, in Figure A.3(g), one only needs to solve the minimization sub-problem in sub-domain *ab*, if knowing that some of the lower bound solution points in both sub-problems *cd* and *ef* are larger than any feasible solution points in sub-problem *ab*.

Convex Domains and Functions

To summary the above discussion on convexity of functions and domains, it is may be seen that (a) and (b) are the most desirable conditions since any feasible local search will always give better result than the initial starting one, i.e. the new result is always closer to the global optimal result than the starting one. The optimization of a convex objective function with a set of constraints that define a convex domain is a convex optimization problem. For convex optimization problems, any feasible heuristic or mathematical optimization solution schemes will always produce a better solution toward the global optima. For nonconvex objective functions defined on convex domains, or convex functions defined on nonconvex domains, as well as nonconvex objective functions defined on nonconvex domains, the outcomes of a solution search scheme are not clear. It may be one of the various local optima, or may not converge at all. To obtain solutions or some approximate solutions from such nonconvex systems/problems, certain mathematical regularities or other augmentations may be needed to recover the convexity, either locally or globally. For example, in Figure A.3(c), the function curve in domain *ae* is nonconvex. However, at the sub-domains *ab*, *bc*, *cd*, and *de*, the corresponding function curve segments are convex. Therefore, one may tackle the nonconvex system by solving all the convex sub-systems, and select the best solution from the results of sub-systems.

Appendix 3 Function Continuity

The convexity of an optimization system mainly deals with issues regarding the uniqueness of a global optimum or a local optimum in a convex subsystem. The existence of such optima, and the cost to obtain an optima solution if it exists are usually related to issues on the smoothness of the objective functions. Considering various 1D functions show in Figure A.3 where the blue line or line segments represent function defined domains. For example, in Figure A.3(h), *a* and *c* are discontinuous points where the curve go to infinite, and there is no optimal values for this problem. Generally, a continuous function (curve, surface etc.) defined on a closed domain always has maximum and minimum points on the domain. An open domain is defined as the domain bounded by its boundary points/curves/surfaces, but not include those points on the boundary points/curves/surfaces, while a closed domain is defined as the open domain plus points on its boundary points/curves/surfaces. For simplicity, only closed domain will be discussed here. A discontinuous point is a point where the function: a) has multiple values, e.g. points *b*, *c*, and *d* in Figure A.3(e); b) goes infinite, e.g. points *a*, and *c* in Figure A.3(h). The smoothness of a function may be classified as bellow.

(a) A function is a discontinuous function if there are discontinuous points in the domain it defined. The function curve in Figure A. 3(e) is a discontinuous function since there are discontinuous points, *b*, *c*, and *d.*
(b) A function is a continuous function if there are no discontinuous points in the domain it defined. The function curves in Figure A. 3(a), (b), (c), (d), (f), and (g) are continuous functions since there are no discontinuous points in their associated domains.
(c) A function is once-differentiable, or order-one smoothness, if a) it is a continuous function; b) its derivative (i.e. slope) function is also a continuous function. The function in Figure A. 3(a) is once-differentiable, while the one in Figure A. 3(b) is not since its slope function has jumps at points *b*, *c*, and *d*. Figure A.3(c) is not once-differentiable since at point *b*, the slope (i.e. the tangent of the slope angle) goes infinite. Similarly, Functions in Figure 3(f), and 3(g) are once-differentiable functions, while those in Figure A.3(d), 3(h) are not (at points *a* and *c*, Figure A.3(h) has infinite slope values).
(d) A function is twice-differentiable, or order-two smoothness, if a) it is a continuous function; b) its derivative (i.e. slope) function is a continuous function; c) its second derivative (i.e. curvature) function is also a continuous function. The functions in Figure A. 3(a) and 3(f) are twice-differentiable, so is that in Figure A.3(g).

The order of smoothness of a function characterizes the relationship of a function or a solution point with its neighborhood points or solutions. Functions with higher order smoothness usually indicates that individual points of such functions shear more common properties or characteristics with their neighboring points. For instance, the neighboring points of the minimum point *b* of the function curve in Figure A.3(a) have lower values relative to other points not in this neighborhood. For discontinuous functions, the relationships of points near discontinuous points may not be clear, or may not have any relation at all (there are functions that are discontinuous everywhere!). For such functions, exhaustive search or certain enumerative methods may be needed to find optimal results. The non-smooth, continue functions (i.e. continue, but not once-differentiable) provide more information between neighboring points, e.g. the difference between two neighboring points should not be too big etc.

For such functions, certain bi-section based methods or upper or lower bound based methods may be more effective to obtain good solutions. Functions with first order smoothness may further provide the direction of where a local/global optima locates since the function curve's tangent slope tell which direction the function increase/decrease, or a possible optima point for a zero tangent slope value (i.e. horizontal tangent line). For example, In Figure A. 3(a), the minimum point *b* has a zero tangent slope, and at other points, the tangent line decreases toward the optimal point *b*. Furthermore, for functions with first order smoothness, various gradient-based methods apply. Those methods search the best solutions iteratively in tangent (or fast decent/accent) directions of points obtained from previous iteration. Generally, higher order smoothness functions allow more effective solution search schemes than those functions with lower order smoothness.

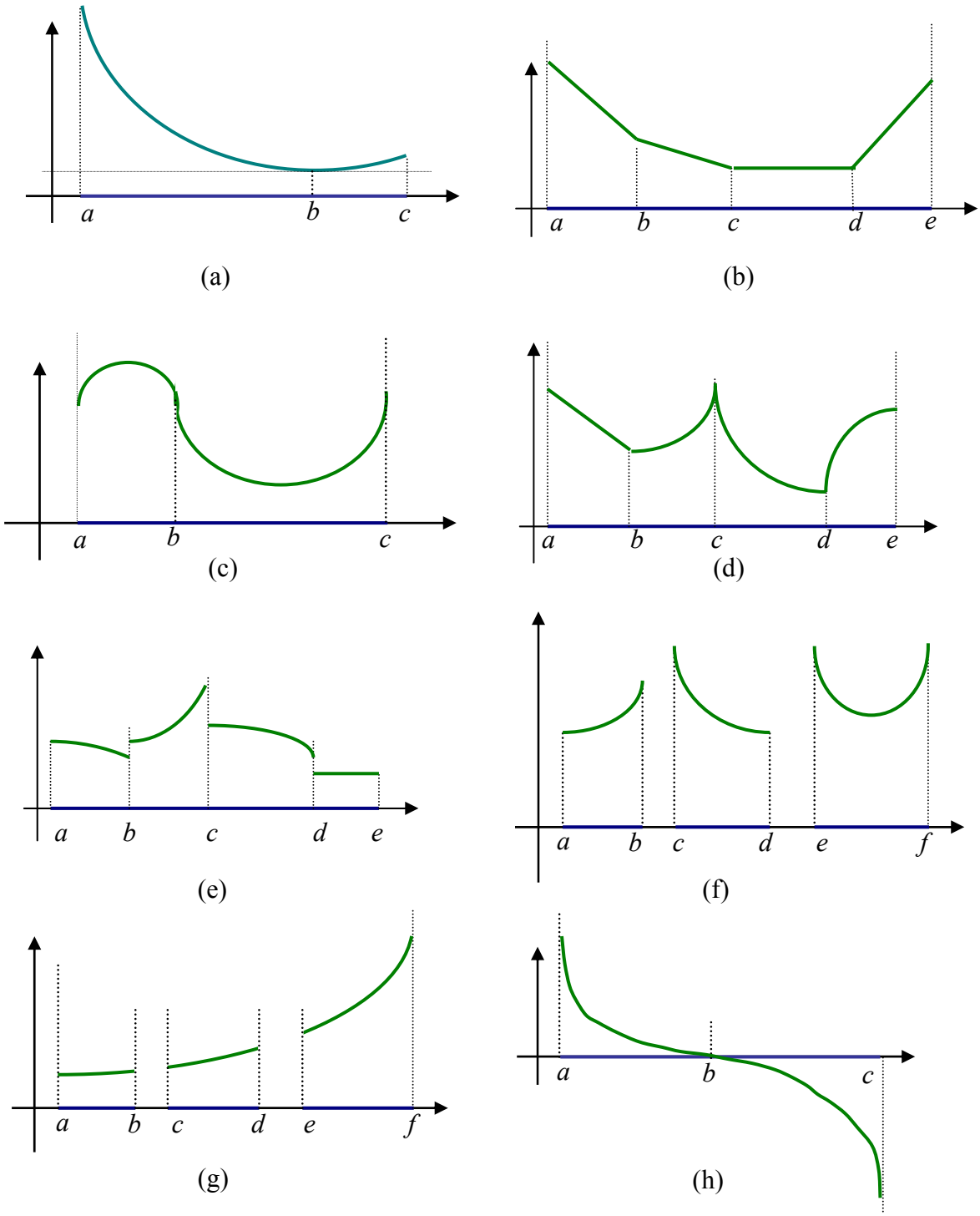


Figure A.3 Functions: Convex/Nonconvex, Continue/Discontinue; and Domains: Convex and Nonconvex.