A Multi-Criteria, Multi-Modal Passenger Route Advisory System*

Foo Hee Meng, Lao Yizhi, Leong Hon Wai
School of Computing, National University of Singapore

Lau Hoong Chuin
Kent Ridge Digital Labs, Singapore

Abstract

Route advisory in land transportation is a major component of many intelligent transportation systems (ITS) and is a major area of study in the field of transportation research. However, route advisory in public transportation is made complicated by the existence of multiple modes (e.g. bus, subway), planned arrival schedules and multiple fare structures.

In this paper, we model the public transport route advisory problem as a graph-theoretic problem, *viz.* the *Multi-Criteria, Multi-Modal Shortest Path Problem* (MMSPP), and briefly describe an algorithm to solve this problem. We also present a software system called the Route Advisory System (RADS) which was developed based on this algorithm, and runs on the Singapore public transport network.

* This work was supported in part by a grant from the National Science and Technology Board (NSTB), Singapore.

1. Introduction

Singapore aims to develop a world-class land transport system. In September 1995, the Singapore Government established the Land Transport Authority (LTA) to spearhead improvements to the Singapore land transport system. In a white paper published by LTA [LTA96], it was envisioned that "public transport is and will always be the major mode of transport".

Today, Singapore's population of 3.8 million makes 7 million passenger trips per day, of which 63% are made on public transport. By the year 2030, the population is expected to exceed 4.1 million; and the projected number of daily trips will grow to 10 million, 75% of which is estimated to be made on public transport.

Hence, providing an attractive public transport system is the cornerstone of Singapore's land transport strategy. Continual efforts are being undertaken to improve and expand the public transport system, such as the extension of the *Mass Rapid Transit* (MRT) network, the building of *Light Rapid Transit* (LRT) feeder systems, and the improvement of commuter facilities. Interested readers may refer to [LTA98] for more details on LTA's drive to encourage motorists to switch to public transport.

We believe that one way to make public transport more attractive is to provide real-time route-advisory to commuters, over the phone, TV or the Internet, so that they can plan the timing of their trips from home and offices.

The Integrated Transport Management System (ITMS)[1] is a computerized system to link the various private and public transport management systems, and to provide timely information to enhance the efficient use of land transport resources. One important component of the ITMS being developed is the *route advisory system*, which provides real-time route guidance information to motorists. This is a first-generation route advisory system, since the underlying problem (namely, the shortest path problem) has been widely researched in terms of models, algorithms, and deployment devices.

In this paper, we present the design and development of a *second-generation route advisory system*. This system provides *multi-criteria multi-modal* route advisory to commuters. To meet the new requirements, the route advisory system must take into account a number of new problem parameters and constraints.

First, the system has to take into account *multiple modes of transportation* including travel by private cars, buses, the MRT, taxis and so on. Even within a given transportation mode, we may have to further decompose the mode since there may still be significant differences such as that between travel on expressways versus travel within housing estates.

With multiple transportation modes, the system also has to incorporate *transfers between modes*. Examples include the MRT-to-bus transfer which involves the travel from the MRT station to a nearby bus stop, or the car-to-MRT transfer which involves parking the car and

[1] ITMS is an ongoing joint project between LTA and KRDL (Kent Ridge Digital Labs).

travel to the MRT station. Even within a single transportation mode, we may need to model time delay factors like delays at intersections or parking delays.

Second, the system must incorporate *multiple objective functions* (or multi-criteria) to offer commuters a variety of criteria for route guidance such as route *with minimum number of mode transfers, shortest travel time*, or *minimum total cost* of travel. Each of these criteria may be of the utmost importance to a commuter at a given time and so the system should provide the option for the commuter to choose from one of these.

To incorporate the above requirements of a second-generation route advisory system, we need to consider the mathematical problem called the *Multi-modal Multi-criteria Shortest Path Problem* (MMSPP). The problem is a computationally complex problem that is, as yet, not well studied in the literature.

This paper proceeds as follows. Section 2 gives a detailed description of the problem and related research work. Section 3 discusses our solution approach in terms of the model and algorithm. Section 4 describes the RADS system that was developed based on our solution approach, and Section 5 concludes with extensions and applications of the system.

2. The Multi-Criteria, Multi-Modal Route Advisory Problem

In this section, we describe the route advisory problem that underlies the core planning engine of the system. We will present a detailed description and formulation of the problem, followed by a survey of the state of the art to solve the problem.

2.1 Problem Description

Before describing the more general multi-modal, multi-criteria route advisory problem, we need to describe the single-mode route advisory problem.

In the *single-mode, multi-criteria route advisory* problem, we deal with a single mode of transportation (eg. private car). Consider a situation where a commuter wants to get from one point to another in a given city by car. We model this path planning problem as *the single-mode route advisory problem* where we are given a *road network* and we want to find a *shortest route* to go from a given *source point* to a given *destination point*.

The "standard" model of a road network is a weighted, directed graph model $G=(V,E)$ (sometimes also called a network model) consisting of a set V of nodes and a set E of directed edges that connect pairs of nodes. In the graph G, the nodes in V are intersections in the road network, while we use an edge $e=(u,v)$ to represent the road segment that connects the intersections u and v in the road network – going from u to v. We also assign a weight, $w(e)$, on each edge e in the graph G. A route in the road network from a source point s to a destination point t will then be a *path* $P(s,t)$ in the graph from the source node s to the destination node t. The "length" of the path $P(s,t)$ is then the sum of the weight of the edges along the path $P(s,t)$ from node s to node t. Then the *single-mode route advisory problem* is modeled as the well-known and well-studied and well-solved *shortest path problem* of finding a shortest path from node s to node t in the weighted, directed graph $G=(V,E)$.

By assigning different meanings to the weight ***w(e)*** on each edge in the graph ***G***, we can use the shortest path problem to help the commuter to obtain the ***s***-to-***t*** route that minimizes different route planning criteria. For example, if ***w(e)*** is the length of the road segment corresponding to the edge ***e***, then the commuter gets an ***s***-to-***t*** route that minimize the total *distance* traveled. For simplicity, we call this the *distance criterion*. On the other hand, if ***w(e)*** represents the *time* taken to travel from ***u*** to ***v*** along the edge ***e=(u,v)***, then the commuter gets an ***s***-to-***t*** route that minimizes the total travelling time. Again, for simplicity, we call this the *time criterion*. In general, the commuter may also specify a combination of these two criteria – for example, 70% time and 30% distance. This gives rise to the name, multi-criteria shortest path problem in which we want a "shortest" path ***P(s,t)*** based on a combination of criteria.

The above problem can be extended to incorporate *multiple modes* of transport (such as buses, subways, light-rail). At the same time, we have to account for *multiple criteria* to be used for path planning such as (a) *minimum total distance*, (b) *shortest traveling times* and (c) *minimum total fare* for the trip. Each of these criteria or a combination of them may be of the utmost importance to a commuter at a given time and so the system should provide the option for the commuter to choose from them.

Incorporating these requirements entails fundamental changes to the underlying problem model. We have named this new model *the Multi-criteria Multi-modal Shortest Path Problem* or MMSPP. In this paper, for simplicity, we restrict the MMSPP to only two modes of public transport, namely, the *public bus* and *subway*. In Singapore, the subway system is called the *Mass Rapid Transit* (MRT) and so we shall refer to subway and MRT interchangeably in this paper. In addition, it is necessary to include "walking edges" to represent walking distances in the following cases: (a) walking from the source to a nearby bus stop or subway station, (b) waking from the final stop to the desired destination, and (c) walking from one stop to a nearby stop in between different modes of transport.

2.1.1 Problem Formulation

We now give a unified problem formulation for both the bus and subway systems that highlights the similarities between the two systems. In general, a public transport system consists of many *service lines*. Each *service line* is identified by a name (e.g.: Bus 197, East-West MRT line), a *source station* (or interchange) and a *destination station* (or interchange).

Each service line consists of a sequence of *stops* along a prescribed route, the first stop is the source station, and the last stop is the destination station. The stops along the service route serve as the passenger pick-up and alighting points. For example, Figure 1 shows the 14 bus line that goes through bus stop B03 and B06, and the MRT East line that starts from station W07 and ends in E5. Note that different service lines and transport modes may share the same stop (e.g. subway-cum-bus-stop complex) and there could be pedestrian pathways allowing commuters to walk from one stop to another (see Figure 1).

In Singapore, there are 2 major bus operators (SBS and TIBS) that service over 3000 bus stops and 300 bus lines island-wide. The subway system consists of 1 subway operator that services 48 stations and 4 MRT lines covering the 4 major axes of the island (North, South, East and West) with over 80km of rail.

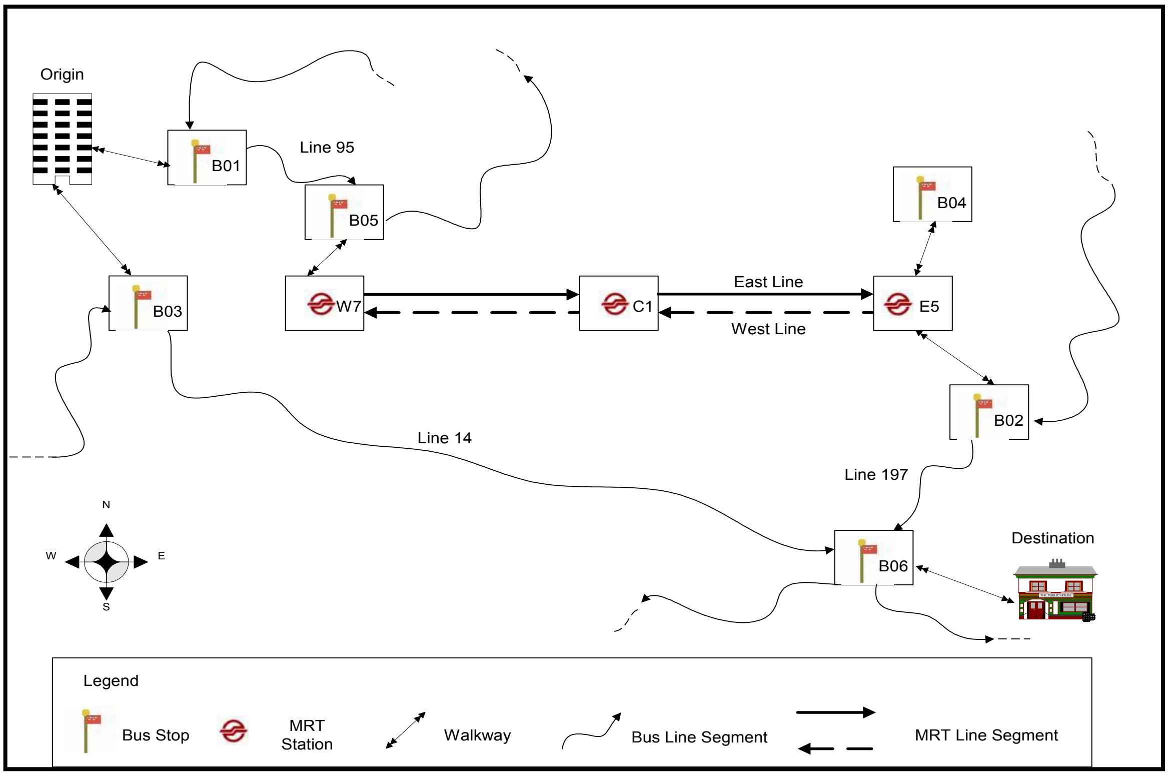

Figure 1 : Subset of Singapore Public Transport Network

For a given service line, there is an arrival schedule that gives for each stop along the route, the first arrival time and the frequency of arrival for the given service line. For instance, 1 bus every 6 to 12 minutes. In general, the frequency may vary during the day, typically higher frequency during peak hours, and lower during off-peak hours. An example of this is shown in Table 1.

Bus Line 14					
Operating Hours					
Weekdays		Saturdays		Sundays	
First Bus	Last Bus	First Bus	Last Bus	First Bus	Last Bus
0545	2330	0545	2330	0545	2330

Time between buses			
0630 to 0900	0901 to 1629	1430 to 1900	After 1900
5-10 mins	6-12 mins	7-11 mins	10-12 mins

Table 1 : Bus Line schedule for line 14

With multiple transportation modes, the route advisory system also has to consider *dynamic transfer between modes*. For example, when transferring from one bus line to another at a bus stop, there is a *waiting time* involved when waiting for the next bus. (To be more precise, the waiting time consists of the time to alight from the bus to the bus stop, and waiting for the next bus to arrive.) Another example is an MRT-to-bus transfer that involves alighting from an MRT station, and walking from the MRT station to a nearby bus stop with the appropriate bus line, and waiting for the next bus to arrive. This waiting time is *dynamic* in the sense that it depends on the time that the commuter arrives at the stop/station as well as the schedule of the line. Thus, the model must account for the dynamic nature of the waiting time as well.

Bus Line 14											
Bus Stop	B10	B11	B14	…	B03	B16	B18	…	B06	…	B23
Fare Stage	1	1.5	1.5	…	6	7	7.5	…	28	…	31

Bus Line 92											
Bus Stop	B21	B01	…	B05	B32	…	…	…	B46	B17	…
Fare Stage	1	2	…	4.5	5	…	…	…	27	27.5	…

Bus Line 197											
Bus Stop	…	B15	…	…	…	…	B02	…	B24	B06	…
Fare Stage	…	2	…	…	…	…	28	…	29	30	…

Table 2 : Bus line fare stage tables

To enable fare computation, we need to model the fare structure for the various service lines. In Singapore, the fare structures of both bus operators are the same. The fare for each trip is not determined by the total number of stops traveled, but rather calculated based on the concept of *fare stages* along the bus route. The fare stage of a given bus line starts from 1 at the source station and increases along the bus route. A fare stage is a rough approximate of distance traveled by the bus. For example, Table 2 shows the route of bus line 14 from B10 to B23 with fare stages going from 1 to 31.

Number of Fare stages between stops	Fare (cents)
0.5 to 4.5	60
5 to 7.5	80
8 to 10.5	100
11 to 13.5	110
14 to 18.5	120
19 to 23.5	130
> 23.5	140

Table 3 : Bus fare table

The fare charged for riding bus line 14 depends not on number of bus stops traveled, but on the number of the fare stages. These exact fare charges are given in Table 3. For example, in Figure 1, if a commuter boards bus line 14 at bus stop B03 (at fare stage 6) and alights from the bus at stop B06 (fare stage 28), then he has traveled 22 (28 – 6) fare stages on bus line 14. From Table 3 we can see that the fare charged is 130 cents.

From Station	To Station	Fare (cents)
W7	C1	120
W7	E5	140
C1	E5	110

Table 4 : MRT fare table

However, the fare charges for the subway depend only on the source and destination stations (ie. it does not depend on the route taken). Hence, the fare charge calculation is simply a lookup table (eg. Table 4). In addition, since there are very few MRT lines in Singapore a similar lookup table can represent the travel times between stations.

From the above description, it is clear that the differing fare structures between the bus and MRT systems add another level of complexity to the MMSPP.

2.2 Related Research Work – State of the Art

We begin with a brief summary of the state of the art in this and related research. The general problem of route planning has been widely investigated in the past decades and there are more than two thousand research papers on related topics (see [Zhao97], [ZhNo98] and [DePa84]). However, most of the research efforts have focused on static, single mode transport systems [Zhao97].

2.2.1 Standard Algorithms for Shortest Path Problems

The static, single mode transport system is generally modeled by a weighted, directed graph where the nodes are (major) intersections and edges represents roads connecting these intersections. There are many algorithms for the associated *shortest route problem* – most notably the celebrated Dijkstra's algorithm [Dijk59], Bellman [Bell58] and Moore's [Moor59] algorithms for the single source problem, and Floyd's [Floy62] algorithm for the all-pairs shortest route problem. There are also some researches on using A* algorithms from AI to compute shortest path on the fly (see [ShII95]). For our purpose, we shall call these algorithms the "standard algorithms". These algorithms generally run in time $O(n^3)$ where n is the number of nodes in the graph and they can be found in many standard textbooks on algorithms and operations research.

While these standard algorithms apply only to the single mode transport problem and so, cannot be directly extended to the general MMSPP problems, they form the basis for analysis and modeling the more general problems. In many cases, these standard algorithms are used as sub-procedures when solving the general problem.

2.2.2 Multi-Mode Transport Systems

More recently, there has been some research on multi-criteria shortest route problems and on multi-mode shortest route problems. However, the research on these more general problems has not matured yet and many of these are based on simplified problem models.

Multi-mode transport systems are generally considered to be generalizations of the single-mode problem in which we extend the graph model by adding new edges to the graph for the various modes of transportation. In addition, edges are also added to represent transfer between modes. A variety of methods are also used to model the transfer costs. Most of the researches focused on single criteria, multi-mode shortest path problem.

Many of the researches reported solve the problem by applying some standard algorithm on an expanded graph. These methods differ slightly in their technical details but almost all of them suffer from *prohibitive problem sizes and running times*. For example a transport system with p stops, m modes and c time intervals results in an expanded graph with $O(pmc)$ nodes and $O(p^2mc)$ static edges for the various modes and $O(pmc)$ edges to represent transfer between modes. The running time of each of the resulting algorithms is between $O(pm^3c^3)$ and $O(p^2m^4c^4)$, far too slow for large scale problems involving thousands of stops. There has been some recent research work reported on speeding up some of the algorithms. Ziliaskopoulos and Mahmassani (see [ZiMa96]) recently reported an overview article on the state-of-the-art of this approach. More recent work is reported in [ZiWa97].

3. Solution Model and Algorithm

In this section, we give an overview of our model and solution of the Multi-criteria Multi-modal Shortest Path Problem (MMSPP). The approach is to start with a simple graph model of the multi-mode public transport system and add to the graph model in order to cater to the issues of traveling time and fare structures in MMSPP. The algorithms for solving the problem will also have to be suitably modified from the standard algorithm. More details of the algorithmic techniques can be found in [Lao99].

3.1 The Basic Multi-Mode Graph Model

The public transport system is first modeled using a graph model $G=(V,E)$ in which each node in V represents a stop. Each service line is represented by a set of labeled, directed edges $e=(u,v)$ that connects two consecutive stops u and v along the route. We also label the edge e with its service line. This means that the graph G is a multi-graph in which between any pair of nodes u and v, there may be multiple labeled edges from u to v corresponding to different service lines connecting stops u and v.

3.2 Modeling Walking Edges

To model walking between "nearby" stops, we add to the graph G, two directed edges $e_1=(u,v)$ and $e_2=(v,u)$ between two nodes u and v in the graph whenever the two stops corresponding to u and v are "within walking distance" of each other. These edges will be labeled as *walking edges*. In our current model, we define two stops to be "within walking distance" if the estimated distance between them is less than a *walking distance threshold*. This threshold is an input parameter to the problem.

In addition, if the user specifies a source point s that is not a stop in V, we will need to add in walking edges from the source point to nearby stops in the graph G. Thus, for each stop v in V that is "within walking distance" from s, we add the walking edge $e=(s,v)$. Similarly, if the destination point is not a stop in V, we will add walking edges from "nearby" stops to the destination point t. Note that these additional walking edges are added in dynamically by the systems since it depends on the source and destination points supplied by the user. These walking edges are also deleted from the model after the end of the route planning.

3.3 Minimizing the Total Distance Traveled

The graph model $G=(V,E)$ thus obtained incorporates multi-mode transport and walking edges. For each labeled, directed edge $e=(u,v)$, we let $w_d(e)$ denote the *distance* from u to v along the labeled route. Then, it is clear that any of the standard algorithms for shortest path problems can be applied on G to compute a s-to-t path that minimizes the *total distance traveled*.

3.4 Modeling Transfer between Modes

To model the transfer between modes, we use a technique called *node explosion* proposed by Spiess and Florian [SpFl89]. Informally, the node explosion technique takes a public transport graph model ***G=(V,E)*** and transforms it into ***G'=(V',E')*** by adding new nodes and edges to represent "alighting" and "waiting" at the stops.

Node Explosion Transformation [SpFl89]: For each service line ***k***, we do the following: (a) for each stop ***v*** along its route, we add a new node ***vk*** to ***V'*** in the expanded graph, (b) replace each directed edge ***e=(u,v)*** in along its route by three new directed edges ***(u,uk)***, ***(uk,vk)*** and ***(vk,v)***. The node ***vk*** is also called the *access point* for line ***k*** at stop ***v***. The edge ***(uk,vk)*** is called a *traveling edge* and represents the commuter traveling on service line from stop ***u*** to stop ***v***. The edge ***(vk,v)*** is called an *alighting edge* to represent the commuter alighting at stop ***v*** from service line ***k***. The edge ***(u,uk)*** is called a *waiting* edge to represent the commuter waiting for service line ***k*** at the stop ***u***.

To illustrate the node explosion transformation, we first consider a commuter traveling on the path ***(u,v)***, ***(v,w)*** on service line ***k*** in the original graph ***G=(V,E)***. In the transformed graph ***G'=(V',E')***, this is represented by the path of traveling edges ***(uk,vk)***, ***(vk,wk)***. In ***G'=(V',E')***, we model a transfer from line ***h*** to line ***k*** at stop ***v***, we by the alighting edge ***(vh,v)*** followed by the waiting edge ***(v,vk)***.

The distance weight $w_d(e)$ = 0 (or negligible) if ***e*** is an alighting or a waiting edge. For a traveling edge ***e=(uk,vk)***, the distance weight $w_d(e)$ is the distance from ***u*** to ***v*** along the route for service line ***k***. It is also easy to see that any standard shortest path algorithm can be used to minimize the total distance traveled.

3.5 Dynamic Transfers and Minimum Traveling Time

We now consider minimization of the traveling time. For each edge ***e*** in the graph ***G'=(V',E')***, we assign a *travel time weight* $w_t(e)$ to denote the time taken to travel along edge ***e***. For a traveling edge ***e=(u,v)***, $w_t(e)$ is the time taken to travel from stop ***u*** to stop ***v***. For an alighting edge ***e=(vh,v)***, $w_t(e)$ = 0 (or negligible). For a waiting edge ***e=(v,vk)***, $w_t(e)$ represents the *waiting time* between arriving at stop ***v*** and boarding the service line ***k***. However, this waiting time $w_t(e)$ is *dynamic*, depending on the time the commuter arrives at stop ***v***, and the schedule of service line ***k***.

In our model, we have made the assumption that the schedule of each service line is fixed and the commuter always boards the next bus/subway of the service line. (These assumptions are reasonable from a commuter's point of view.)

With these assumptions, given the time that a commuter arrives at a stop ***v***, it is possible to compute the waiting time $w_t(e)$ for the waiting edge ***e=(v,vk)***. We showed in [Lao99] that we are able to suitably modify the standard Dijkstra's shortest path algorithm [Dijk59] to minimize the traveling time even with dynamic waiting times. The proof of our result is based on a monotonic property that we prove using the assumption stated above.

3.6 Modeling the Fare for the Bus System

The most complex extension of our graph model arises from the need to model the fare structure of public transport and the need to support multiple criteria in path planning. As the fare structures of the bus and the subway systems are different, different models and algorithms are needed. We first describe modeling of fare for the bus system.

Recall that the fare charge for taking a bus line $\boldsymbol{k}$ is based on the number of fare stages traveled on line $\boldsymbol{k}$, and not on the number of edges. In other words, for any path $(e_1,e_2,\ldots,e_q)$ on bus line $\boldsymbol{k}$, the total fare is not the sum of the "fare" associated with the individual edges e_k's. To successfully model the fare charges for the bus system, we introduce a *bus line expansion transformation* described as follows:

Bus Line Expansion Transformation: For each bus line $\boldsymbol{k}$ with the sequence $(v_1,v_2,v_3,\ldots,v_l)$ of stops, we add the new edges (v_i,v_j) for all i, j with $(j-i)>1$. Note that the edges (v_i,v_{i+1}) already exist in the graph for all i. The expanded graph for line $\boldsymbol{k}$, is the completed directed graph on all the stops of line $\boldsymbol{k}$ (given in sequence). For each edge $e=(v_i,v_j)$, we assign a *fare cost weight* $w_f(e)$ to be the fare to travel from stop v_i to stop v_j along line $\boldsymbol{k}$ (This can be easily calculated -- see Section 2.1.1).

We show that after the bus line expansion, we can use the standard Dijkstra's shortest path algorithm to minimize the total bus fare. In general, a minimum total bus fare path may consist of several bus lines and the total bus fare is the sum of the fare on each of these bus lines.

While the above is a good modeling technique, it cause a serious implementation problem -- the resultant graph after the bus line expansion is very big. Therefore, to circumvent this, we do not store all the "expanded edges". Instead, we modified the standard Dijkstra's algorithm to process these edges as if they are there, thereby simulating the execution of Dijkstra's algorithm on the expanded graph.

3.7 Modeling the Fare for the Subway System

Recall that the fare charge for a subway trip can be done using a simple lookup table since it depends only on the source and destination stations and not on the route taken. Note that this lookup table is similar to the bus line expansion transformation, except that there is no concept of fare stages.

3.8 Integrating the two Fare Models for Multi-Criteria Path Planning

Finally, to integrate the bus and subway fare models, we introduce a hierarchical graph model. The graph G_0 at level 0 is the public transport graph we have defined. The graph G_1 at level 1 is the complete subway network (or collection of connected subway networks). For each subway station $\boldsymbol{v}$, the node corresponding to s in G_0 has a connection to the node $\boldsymbol{v}$ in the graph G_1. This hierarchical model allows the integration of the two fare models as well as multi-criteria path planning using our modified Dijkstra's algorithm [Lao99]. Very briefly, it proceeds as follows: when processing a bus stop, the algorithm will work on the graph G_0. When processing a subway stop, the algorithm will use the graph G_1 to assist in updating the total distances, travel times and fare charges respectively. The details of the algorithms are more complicated and can be found in [Lao99].

The models and algorithms for the MMSPP that forms core engine of our route advisory system has been implemented and tested on public transport data taken from [Tran98]. The implementation is in C++. To speed up the implementation, we have used the LEDA (Libraries of Efficient Data structures and Algorithms) [MeNa99] library.

4. The Route Advisory System (RADS)

The system we have developed to solve the Multi-Criteria, Multi-Modal Route Advisory Problem is called the Route Advisory System or RADS. The system does not dictate which sequence of service lines the commuter has to take but rather interactively assists the commuter to plan his trip by finding and displaying the route that best meets his preferred criteria, the relative weights of which are specified by him.

4.1. The RADS User Interface

Figure 2 depicts the main window of the system. The sub-window on the top right hand, or Map Window, displays the map of Singapore. The user is able to navigate within this map both by selecting a region on the map (zoom) or pan to the north, south, east or west directions by clicking on the arrow buttons located on the top left hand sub-window, the Command Window. The last window of interest is the bottom right or Trip Information Window. It is used to display the detailed sequence of service lines from origin to destination location.

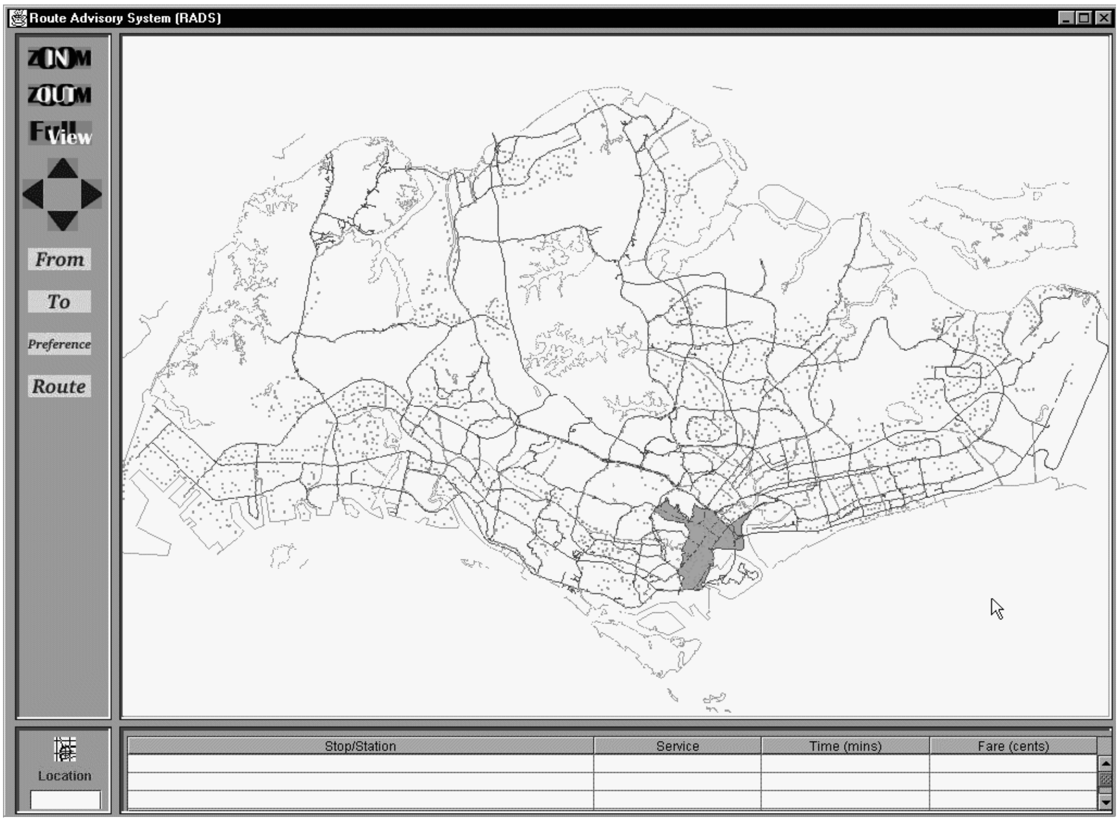

Figure 2 : The main window of RADS

When the "Preference" button in the Command Window is clicked, the "Preference" dialog (Figure 3) appears. Here, the user is able to adjust the relative weights to place on the two criteria: time and fare. In the figure, the user has chosen to find the route that minimizes the total trip time, ignoring the cost incurred. Here, he has also indicated that he wishes to start his trip at 10am.

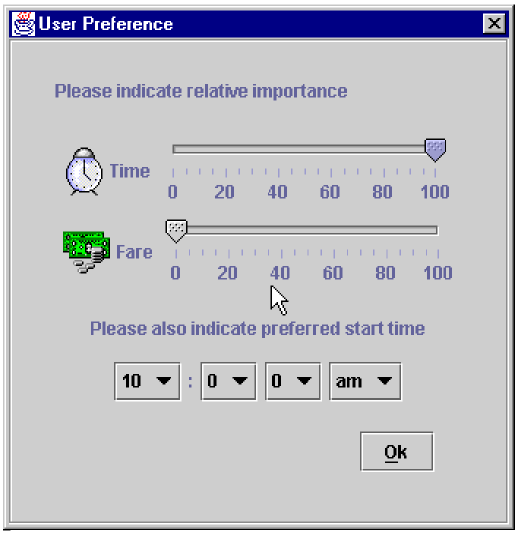

Figure 3 : Preference Dialog

After selecting the origin location by clicking on a location on the map and clicking the "From" button on the Command Window; and selecting the destination location in a similar manner, the user then has to click on the "Route" button and the system will compute and then display the trip, both as a sequence of colored line segments on the Map Window and as textual descriptions on the Trip Information Window (see Figure 4). By zooming in, we can see the one of the multi-modal transfers (Figure 5). In this example, the user is advised to take bus 92 to the bus stop along Buona Vista Road near the Buona Vista MRT station, alight there and walk to the MRT station to take an east bound train.

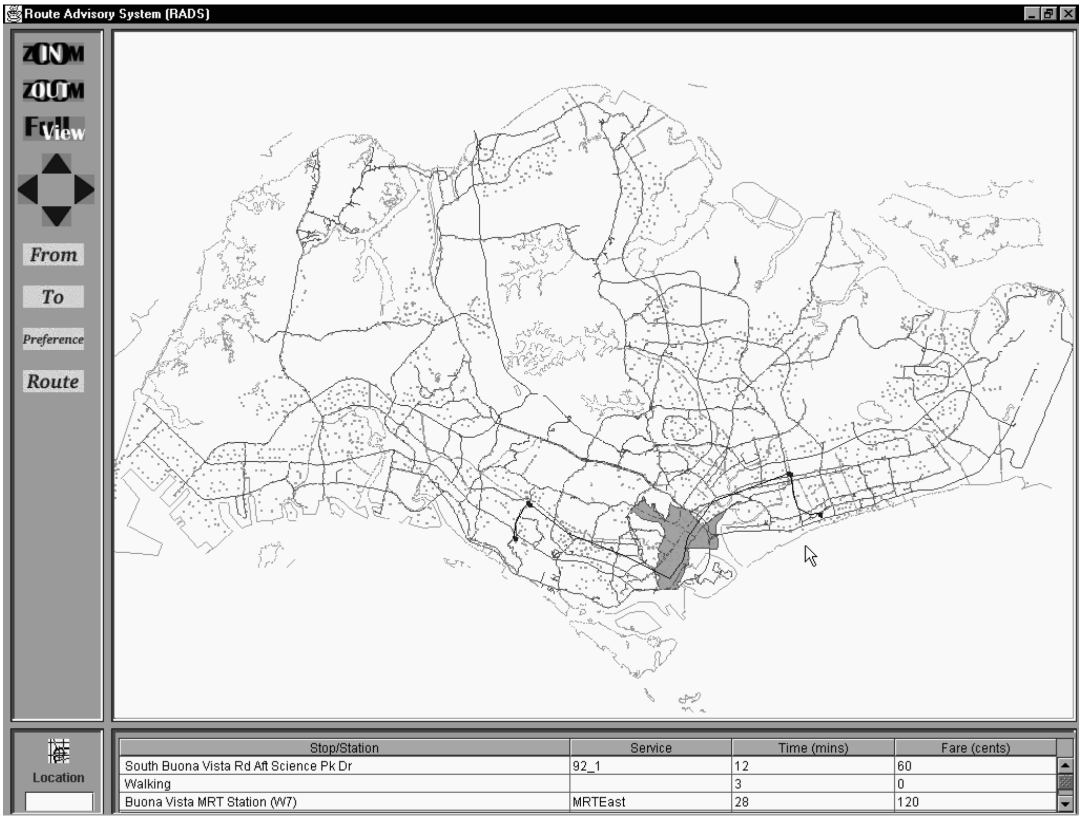

Figure 4 : The RADS main window with trip displayed

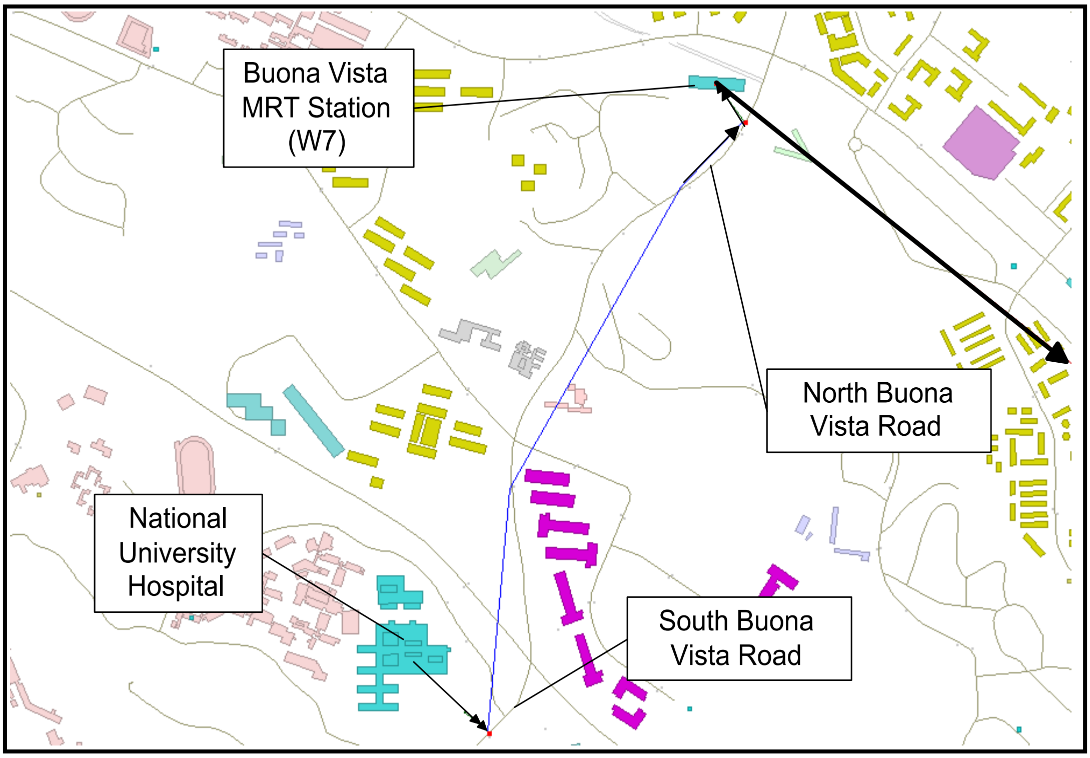

Figure 5 : Portion of multi-modal trip

4.2. The RADS System Architecture

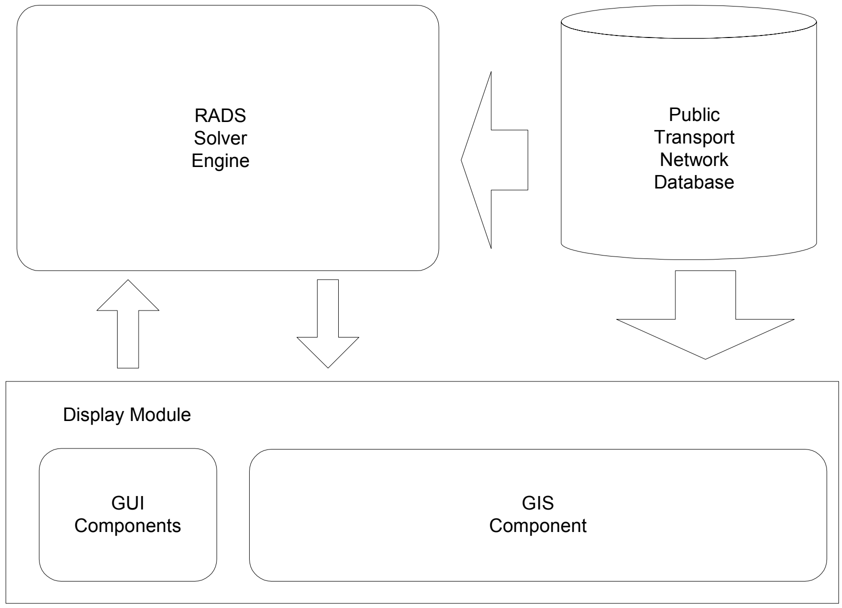

Figure 6 : The RADS System Architecture

In Figure 6 we present the system architecture of RADS which consists of 3 major modules :

a. Display module
b. The Solver Engine module
c. The Public Transport Network database

The Display module provides the graphical front end to the system. It is written entirely in the Java programming language (see [ArGo98]) to facilitate portability across OS platforms. The 2 main sub-modules within this module are:

a. The GUI Components

   These are the components that make up the various windows and dialogs of the system (eg. Preference Dialog, Command Window buttons). Some of these (eg. the Trip Information Table) make use of the Java Swing package found in Version 1.2 of the Java Development Kit (JDK).

b. The GIS Component

   The GIS or Geographic Information System is responsible for the storage and retrieval of the Singapore map data and the display of the map. Data entities stored here include roads and buildings.

The Solver Engine module is responsible for planning the route for the user. This module is implemented in C++ and it makes use of the LEDA (Library of Efficient Data-structures and Algorithms) C++ library [MeNa99]. As the name implies, LEDA provides a sizable collection of data structures and algorithms often found in the combinatorial and geometric computing literature. The library is implemented in C++ with templates, and is designed in an object oriented manner. These data structures and algorithms are used extensively in the Solver Engine in the implementation of the algorithm described in Section 3.

The Public Transport Network Database stores the locations of all bus stops and MRT stations, bus service schedules and MRT network information. This information is used both as input for the Solver Engine module as well as by the Display module.

The entire system occupies 20 Megabytes of disk space of which 500 Kilobytes make up the Solver Engine Module and 17 Megabytes make up the GIS. The average response time of the Solver Engine on a Pentium 200Mhz (with MMX) with 64 Megabytes of RAM running Windows NT 4.0 is 11 seconds.

5. Conclusion

In this paper, we have presented a second-generation passenger route guidance software system for the Singapore public transport network. This system has been carefully designed so that it can be enhanced in a straightforward fashion to incorporate the following:

1. Route advisory under real-time traffic information.

   The system can be extended to provide commuters with dynamic route guidance under real-time traffic information. Real-time information (such as up-to-the-minute bus/MRT arrival information and road traffic conditions) has tremendous impact on the planned optimized route. For example, the travel time along a fastest route may be greatly increased if there is an unforeseen traffic jam along one of the links. In addition, transfer time between modes are also higher susceptible to the traffic load at a given time. For example, the waiting time at a bus stop can vary substantially depending on the number of commuters waiting and riding along similar routes.

2. Extensions to other transportation modes – taxis, LRT, etc – the missing links for a fully integrated public transport route advisory system.

We also believe that the system would provide an important software component for the following applications:

1. Deployment in ITMS, to provide real-time public transport route guidance service.

2. A personalized travel planner to assist a traveler to plan an itinerary for multiple visits that respects individual preferences and constraints.

3. Urban transport planning, e.g. the system is used as a decision support tool for bus/MRT route and service planning, or a tool for devising and evaluating various park-&-ride schemes.

References

[ArGo98] K. Arnold and J. Gosling, “The Java™ Programming Language 2nd Edition”, Addison-Wesley, 1998.

[Bell58] R. E. Bellman, “On a routing problem”, *Quart. Appl. Math.* 16, 1958, pp. 87-90.

[DePa84] N. Deo and C. Y. Pang, “Shortest-Path Algorithms: Taxonomy and Annotation”, *Networks*, Vol. 14, pp 257-267, 1984.

[Dijk59] E. W. Dijkstra, “A Note on Two Problems in Connection with Graphs”, *Numerische Mathematik*, Vol. 1, 1959, pp. 269-271.

[Floy62] R. W. Floyd, “Algorithm 97 : Shortest Path”, *Communications of the ACM* 5, 1962, pp. 345.

[Lao99] Y. Z. Lao, “Multi-Modal Route Planning in Public Land Transportation”, *Honours Project Report*, Dept. of Computer Science, NUS.

[LTA98] Land Transport Authority, Singapore, “Try Public Transport – A Pitch for Motorists to Make the Switch”, *Journeys : a monthly newsletter of the Land Transport Authority*, Issue 18, Oct. 1998.

[LTA96] Land Transport Authority, Singapore, “A World Class Land Transport System”, White Paper, Jan 1996.

[MeNa99] K. Mehlhorn and S. Näher, “The LEDA Platform of Combinatorial and Geometric Computing”, to appear with Cambridge University Press, 1999.

[Moor59] E. F. Moore, “The Shortest Path Through a Maze”, *Proc. International Symposium on the Theory of Switching*, Cambridge, MA, Harvard University Press, Apr 1959, pp. 285-292.

[ShFG96] Shashi Shekhar, Andrew Fetterer and Brajesh Goyal : “A Comparison of Hierarchical Algorithms for Routing in Advanced Traveler Information Systems”, url=http://www.cs.umn.edu/research/shashi-group/

[ShII95] T. Shibuya, T. Ikeda, H. Imai et. al., “Finding a Realistic Detour by AI Search Techniques”, *Second Intelligent Tranportation Systems*, Vol. 4, 1995, pp. 2037-2044.

[SpFl89] H. Spiess and M. Florian, “Optimal Strategies: A New Assignment Model for Transit Networks”, *Transportation Research Part B*, 23B, 1989, pp. 83-102.

[Tran98] Transitlink Corp. Singapore, Transitlink Guide, 1998.

[Zhao97] Y. L. Zhao, “Route Planning Module”, Vehicle Location and Navigation Systems, Artech House, 1997.

[ZhNo98] F. B. Zhan and C. E. Noon, “Shortest Path Algorithms: An Evaluation using Real Road Networks”, *Transportation Science*, Vol 32, No. 1, Feb. 1998, pp. 65-73.

[ZiMa96] A. K. Ziliaskopoulos and H. S. Mahmassani, “A Note on Least Time Path Computation Considering Delays and Prohibitions for Intersection Movements”, *Transportation Research* 30, 1996, pp. 359-367.

[ZiWa97] A. Ziliaskopoulos and W. Wardell, “Design and Implementation of an Intermodal Optimum Path Algorithm for Multimodal Networks with Dynamic Arc Travel Times and Switching Delays”, submitted to *Transportation Science*, 1997