Title

Enhancements To A Simulation Framework For Analyzing Urban Traffic Networks With Atis/atms

Authors

Jayakrishnan, R.
Rathi, U.
Rindt, C.
et al.

Publication Date

1996

CALIFORNIA PATH PROGRAM
INSTITUTE OF TRANSPORTATION STUDIES
UNIVERSITY OF CALIFORNIA, BERKELEY

Enhancements to a Simulation Framework for Analyzing Urban Traffic Networks with ATIS/ATMS

R. Jayakrishnan, Unmesh Rathi, Craig Rindt, Ganesh Vaideeshwaran
University of California, Irvine

California PATH Research Report
UCB-ITS-PRR-96-27

This work was performed as part of the California PATH Program of the University of California, in cooperation with the State of California Business, Transportation, and Housing Agency, Department of Transportation; and the United States Department of Transportation, Federal Highway Administration.

October 1996

ISSN 1055-1425

Enhancements to a Simulation Framework for Analyzing Urban Traffic Networks with ATIS/ATMS

R. Jayakrishnan
Unmesh Rathi
Craig Rindt
Ganesh Vaideeshwaran

Institute of Transportation Studies
University of California at Irvine
Irvine, CA 92717

Final Report, Project MOU-84
Partners for Advanced Transit and Highways (PATH)

July 1994

Acknowledgements

The authors wish to acknowledge the support received from the California- Department of Transportation and the PATH program for this research effort. We also acknowledge the support from the Federal Highway Administration for some of the research performed at the University of Texas at Austin mainly on the DYNASMART program development. Professor Wilfred Recker, director of the Institute of Transport Studies, University of California at Irvine, and Professor Michael McNally, Assistant Professor, University of California at Irvine, were instrumental in facilitating the research performed jointly with the Anaheim ATMS research testbed project. The authors also acknowledge Prof. Hani Mahmassani and Mr. Ta-Yin Hu of the University of Texas at Austin (who were deeply involved in the development of DYNASMART during the first year of this research effort) for several helpful research suggestions. The authors, however, remain solely responsible for the research results reported herein.

Abstract

The research reported herein further develops the DYNASMART simulation program developed at the University of California, Irvine with PATH funding, in informal collaboration with a project funded by the Federal Highway administration at the University of Texas at Austin. The DYNASMART program resulted from the doctoral dissertation of the principal investigator of this PATH project at UT Austin, and has now been developed to include full-scale traffic signal control as well as freeway ramp control. The model is capable of simulating large urban networks under various ATMS and ATIS strategies, and include carefully designed modules for driver responses to information and for capturing the dynamics of the network paths. The model has been implemented on work station as well as mainframe computational platforms. This research has developed a User-friendly front-end for the editing large input files as well as for run-time displays of traffic conditions. The research also included the simulation study of a network in Orange County, California, where the benefits from candidate ATIS and ATMS strategies were evaluated. The DYNASMART model has evolved into a very flexible tool that can be applied to evaluate information and control strategies in realistic urban networks in an efficient manner.

CONTENTS

LIST OF ILLUSTRATIONS

PATH Goal Statement

The research reported herein is a part of the Partners for Advanced Transit and Highways, PATH, within the Institute of Transportation Studies, at the University of California. PATH aims to increase the capacity of the most used highways, to decrease traffic congestion, and to improve safety and air quality. It is evolutionary and voluntary. It is a cooperative venture of automakers, electronic companies, local, state and federal governments, and universities.

Disclaimer Statement

The contents of this report reflect the views of the authors who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the STATE OF CALIFORNIA *or* the FEDERAL HIGHWAY ADMINISTRATION. This report does not constitute a standard, specification, or regulation.

Chapter 1
INTRODUCTION

1.1 Overview of Research

This report presents the work carried out for the second year of the PATH project for developing useful simulation tool for the analysis of urban traffic networks operating with ATIS (Advanced Traveller Information Systems) and/or ATMS (Advanced Traffic Management Systems). The first year of the project (PATH MOU 39, Jayakrishnan et al, 1993) developed an early version of a simulation model that had originated as part of a doctoral study at the University of Texas, to a fully functional simulation program. The second year of the research project (PATH MOU 84) dealt mainly with the user-friendly enhancements to this simulation framework, DYNASMART. DYNASMART (Dynamic Network Assignment Simulation Model for Advanced Road Telematics) is a simulation-assignment model for studying the effectiveness of alternative information supply strategies as well as alternative information/control system configurations, for urban traffic networks with ATIS/ATMS. Simulation of about 75,000 vehicles in networks of up to 2000 links with 10 paths from each node can be carried out using this model. However, after the early experience in preparing data sets for the simulation runs, it was readily apparent that a graphic user interface was a necessary component for the simulation system to be useful for practical application to realistic networks. In this report, we discuss our research to develop such capabilities with the DYNASMART simulation system.

1.2 Motivation

As DYNASMART accepts large data sets as input, preparation of data sets becomes a very involved and cumbersome task. The data preparation includes detailed signalization data for signalized intersections, other intersection-specific data, the connectivity and geometry data for the whole network, the capacity reduction data for the links with lane closures, etc. This is a very tedious and time consuming job. To give an example, the network around the Central Business District (CBD) of Austin, Texas has about 650 nodes and about 1700 links. This means

that the link connectivity data is 1700 lines long, with each line showing the upstream and downstream node numbers. The node signalization data set needs one line per phase, and could thus be about 2000 lines long. The node numbers have to be carefully typed in, as DYNASMART does not check for impractical links (as an example, if a node number is typed in incorrectly, DYNASMART will assume that there is a direct connection, such as an express highway, between two nodes which are possibly far apart !). Research during the first year of the project amply demonstrated such problems.

Another requirement for productive use of the simulation framework is the ability to monitor the simulation at run-time. If the density and speed characteristics of all the links are written out for all the simulation time steps, it creates tremendous volumes of output files, which does not easily show the locations and time periods of congested conditions. Again, graphical displays of network conditions makes this extremely easy for the user.

Motivated by the above concerns, a user-friendly environment for data preparation and run-time display was developed during this research project. This was achieved in the form of a graphic interface which we refer to as the front-end. The front-end provides utilities such as pull-down menus for the easy input of signalization and other ATMS related data. It also provides facilities for creating and editing data relevant to the nodes, links and zones in the network, as well as the capability to provide default signalization data and to replicate data for several intersections. In addition, the front-end also has capabilities to interact with the simulation process during the run time, controlling it as well as displaying the intermediate results.

1.3 Overview of the report

A description of the simulation framework, DYNASMART, is given in chapter 2. Some of the details in chapter 2 can also be found in a previous PATH report which dealt with the development of DYNASMART (Jayakrishnan et al., 1993). That report, though written as a final report to the first year research, also reported some of the research on the second year project reported on here, which was well underway at that time. Thus it included most of the research on enhancements to DYNASMART, such as the ability to accept externally specified

path data. Due to the overlap in the research efforts on the two projects, it is appropriate to give a complete description of the model in chapter 2. The development of the graphic user interface was completed well after the first year report was prepared, and the front-end is discussed in detail in chapter 3 of this report. Results from the DYNASMART simulation of ATIS in two different urban networks of reasonable sizes, namely that of Anaheim, CA and Austin, TX, are provided in chapter 4, along with a discussion of how different modelling approaches and network contexts can result significantly different conclusions. The importance and usefulness of the external path input capabilities developed during the second year research become apparent from the results in chapter 4. We complete the report with some conclusions in chapter 5.

Chapter 2
MODELLING FRAMEWORK

2.1 Introduction

DYNASMART is an assignment-simulation modelling framework designed to assign time-varying traffic demands and model the corresponding traffic patterns to evaluate overall network performance of an Advanced Traveler Information System (ATIS) and/or an Advanced Traffic Management System (ATMS). Though DYNASMART is a descriptive evaluation tool in its current form, it is evolving towards a model that may be executed on-line in quasi real-time to support the functions of the system controller in the ATIS/ATMS. DYNASMART moves vehicles individually according to macroscopic traffic relations and microscopic driver behavior rules.

DYNASMART (DYnamic Network Assignment Simulation Model for Advanced Road Telematics) was developed specifically for studying the effectiveness of alternative information supply strategies as well as alternative information/control system configurations, for urban traffic networks with ATIS and/or ATMS. This simulation program models, in an integrated fashion, the three main components of such systems: (1) the response of the drivers to the information/control, (2) the nature of the traffic flow that results from driver responses and applied network control and (3) the dynamics of the routes in the network (in terms of the changing travel times on them) which affect the driver and control system decisions.

At present, this model is primarily a descriptive analysis tool for the evaluation of information supply strategies, traffic control measures and route assignment rules at the network level. This means that it does not attempt to find optimal configurations of ATMS/ATIS systems, but rather studies the effectiveness of given configurations. However, further research is underway to incorporate this model into such optimization frameworks which are expected to operate in real-time or quasi real-time. In its current form, this model can be a valuable tool that would help avoid expensive real-life experimentation and present us with opportunities to extract valuable insights on the complex dynamics of networks under information using computer simulations.

2.2 Conceptual Model

The conceptual model is shown in figure 2.1, and more details of the program components are shown in figure 2.2. The simulation is deterministic, and follows constant time increments. As figure 2.1 shows, the vehicles are generated according to specified time-dependent Origin-Destination zonal demands, and moved in the network (the "Link pass 1" in figure 2.1). The information supply system is assumed to use a path database for information/guidance to drivers reaching route decision points (i.e., network nodes) during the simulation time step. Based on the simulated driver responses to this information and the nodal flow constraints (based on the signalization) the vehicle allocation into various network links are modelled (the "Node Pass" in figure 2.1). This is followed by moving those vehicles in the links that they have just entered (the "Link Pass 2" in figure 2.1), which in turn determines the ensuing traffic flow conditions in the network at the end of the time step. Then the path database is updated ("Path processing" in figure 1), the time step is incremented, and the simulation proceeds. The details of the various components are discussed in the following sections.

In the initial version of DYNASMART, the information center (ATIS computer) is assumed to provide route information using current information only. This means that future conditions are assumed not to be predicted while the routes are displayed to the driver. As reliable prediction algorithms become available, such capabilities will be added to the framework. In addition, ongoing research is using the model itself as a model of prediction, to provide anticipatory route guidance information to the drivers.

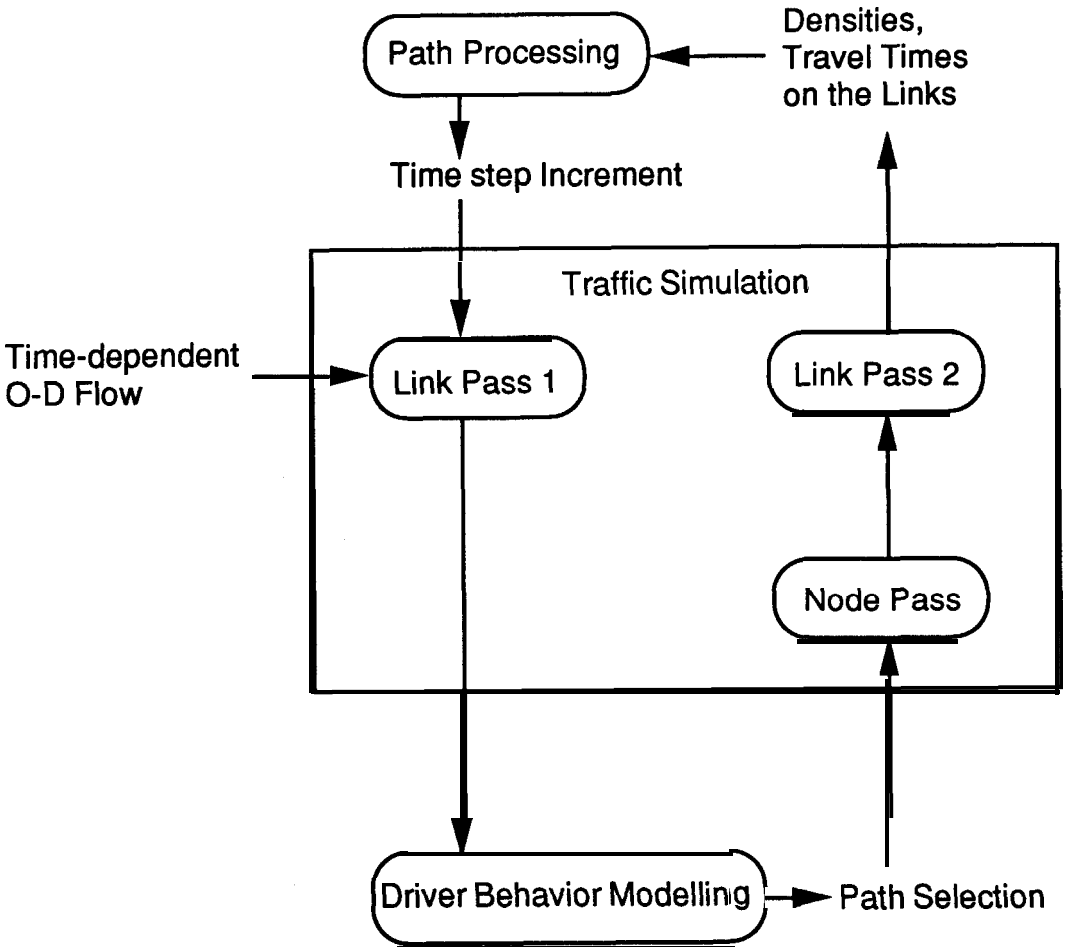

Figure 2.1 DYNASMART model: Conceptual Structure

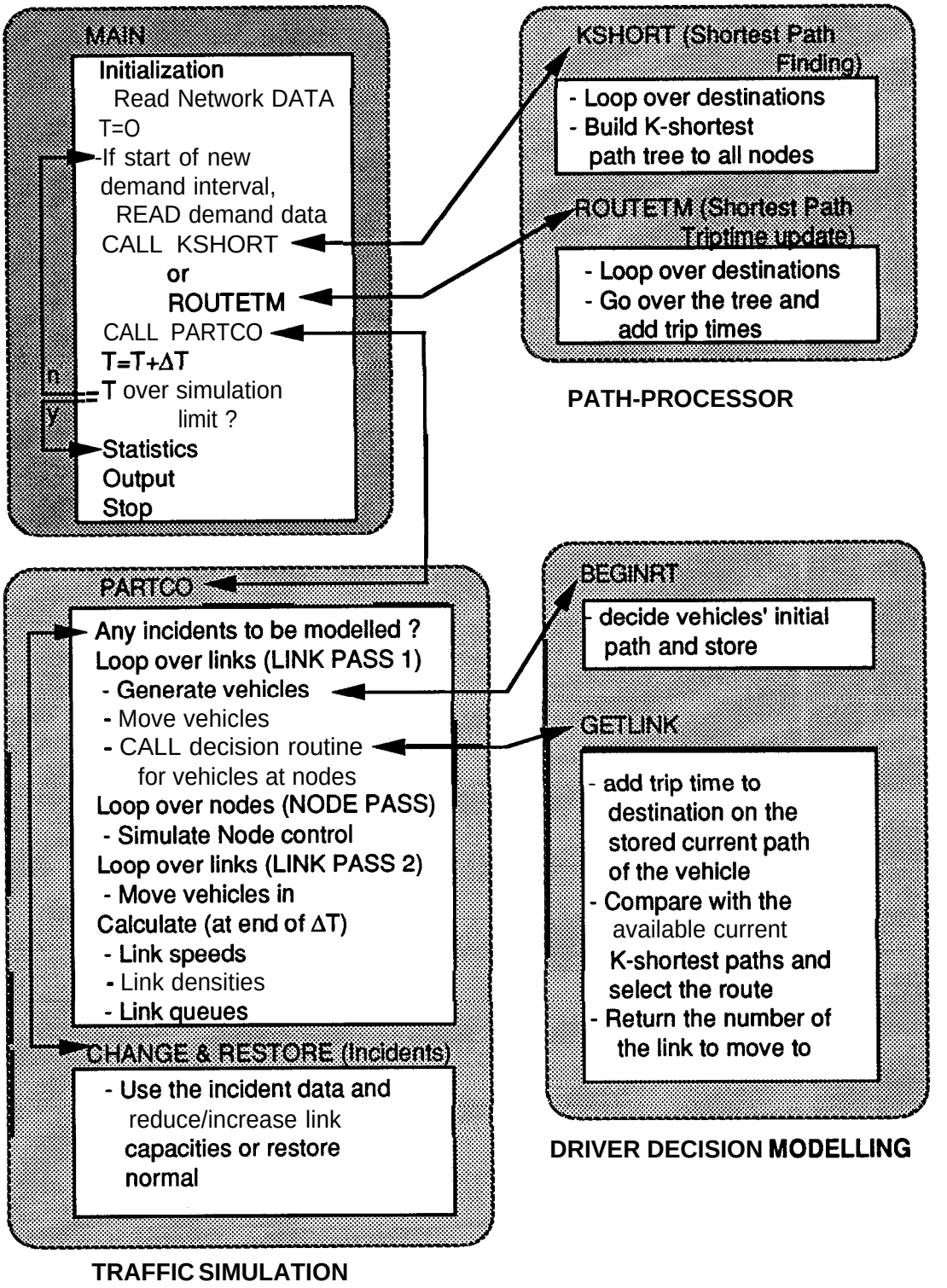

Figure 2.2 Communication between the DYNASMART Modules

2.3 Traffic Flow Simulation

The traffic flow simulation component grew out of the macroparticle simulation model, MPSM (Chang, Mahmassani and Herman, 1985), originally developed for simulation of single highway commuter traffic in a manner similar to plasma simulations (Leboeuf et al., 1979). This model simulated traffic on discrete segments in a one-pass manner, moving vehicles from each segment to its downstream segments. The approach has been extended to simulate a highway network with intersections, which requires the modelling of capacity-constrained vehicle allocation into each segment from segments incident on its upstream node. It is possible to simulate each physical link in the network using multiple segments to capture traffic flow dynamics more correctly. For example, a two-mile physical link on a freeway may be split into four half-mile segments for modelling purposes, and each of the four would be a "link" in our terminology. The traffic simulation details are discussed next.

Traffic flow model: The approach has elements from both macroscopic and microscopic models that have been developed previously. The traffic is represented by the conservation equation,

$$\frac{\partial q}{\partial x} + \frac{\partial k}{\partial t} = g(x,t) \qquad (1)$$

where, q = flow (vehicles/hour)

k = density (vehicles/mile)

g = net vehicle generation rate (vehicles/hour/mile)

In conventional macroscopic models, this is usually coupled with a speed-density relationship and the identity $q = kv$ (where v is the average speed), and solved numerically using discrete time steps and discretized highway segments. DYNASMART, however, moves the vehicles according to the prevailing local speeds and keeps track of their positions, which means that there is no

need to use the $q = kv$ relationship to find the link-to-link vehicle flux for solving the above conservation equation. The vehicle flux across link boundaries is based simply on the number of vehicles reaching the link boundary during each time step, and the movement constraints at the link boundary. This approach can capture shockwave propagation on a physical link if it is split into multiple virtual links.

The vehicle movement during each time step is based on the speed in the link resulting from the density at the end of the previous time step. All moving vehicles in a link move at the same speed during each time step, depending on the density. The speed-density relationship currently used is a modified version (Chang et al., 1985) of the well-known Greenshield's equation. A specified minimum speed at jam density ensures that the simulation does not "shut down" due to zero speeds. The relationship used is,

$$v = v_0 + (v_f - v_0)(1 - k/k_j)^{\alpha} \qquad (2)$$

where, v_0 = a user-specified minimum (jam) speed,

v_f = free-flow speed of the highway segment,

k_j = density at the jam speed, and

a = a user-specified parameter.

Each vehicle is considered a point particle. DYNASMART also provides the option of moving a bunch of vehicles as a "macroparticle" for reducing the computational effort, as in the original MPSM program. However, keeping a particle size of one is more appropriate, when individual drivers' responses to information are simulated.

Unlike in microscopic models, there is no simulation of lane-changing maneuvers or car-following. Neither do we use any platoon dispersion equation to model the headway variations among vehicles. Also, the vehicle positions are not constrained by vehicle lengths. However, the vehicle lengths are properly accounted for when the end-of-link queues are modelled. These approaches are essential to keep the computations manageable, especially in the case of reasonably large networks, which is where DYNASMART can be effectively applied to study

information strategies. Lanes themselves are not explicitly simulated, but the number of lanes specified for each link determines the effective lane miles on the link and thus the variations of density and speed on the links.

Traffic Generation and Initial Path Assignment: The traffic generation is based on the specified zone-to-zone demand during each demand subinterval. Vehicles are generated on the-links which are specified to be "generation links". First, the total specified generation during a subinterval from each zone is calculated from the O-D demand data. This translates to the number of vehicles to be generated during each time step in each zone. When a vehicle is generated according to this rate, it is immediately assigned to a randomly selected generation link in the zone, thus splitting the generation equally among such links. Each vehicle's destination is determined based on the trip distribution fractions (calculated from the O-D data) used as destination probabilities. As each zone with trip ends has a specified destination node, the destination node of each vehicle is determined. Note that the specified destination node of a zone can be a virtual centroid connected via high-speed links to a few other nodes in the zone, or can be a regular node.

At the time of generation, each vehicle is randomly tagged to be equipped for information or otherwise, based on the specified fraction of equipped drivers. An initial path is assigned to each driver. This could be from among the k-shortest paths stored after the load-up period or any paths externally specified in the data (e.g., paths corresponding to a user-equilibrium).

Link pass: During this procedure, the vehicles on the link are moved according to the prevailing speed calculated using the above speed-density relationship and the prevailing density. The prevailing density is the density at the beginning of the time step, which is in turn calculated according to the discretized form of the conservation equation (equation 1) using the density, inflows and outflows of the link in the previous time step. The vehicle positions are updated in a straight-forward manner,

$$x_j^t = x_j^{t-1} - v_i^t \cdot \Delta t, \quad \text{if } (x_j^{t-1}/v_i^t) > \Delta t \qquad (3)$$

$$x_j^t = 0, \text{ otherwise}$$

where,

x_j^t = Distance of vehicle j from the link-end at the end of time step t

At = Simulation time step length (say, 0.1 minute)

v_i = Speed in link i that the vehicle j is on, during time step t

Thus the vehicle positions are kept as distances from the end of the link, and the vehicles are moved at the current link speed. If a vehicle reaches the link-end during the current time step, it is positioned at the end of the link. Such a vehicle may move to the next link (as determined by the driver's decision) and travel for the rest of the time step, or it may join a queue if the node pass determines that it cannot move to the next link. The arrival time of vehicles at the link-end are stored temporarily for use after the node pass. Vehicles that are selected to enter a new link after the node-pass, are moved during a second link-pass, for the remaining fractional time step.

Node pass: The link-to-link transfer of vehicles is carried after the node-pass, which simulates the traffic control features at the nodes and determines the number of vehicles that can move to downstream links. The driver response simulation is also performed here to obtain the driver decisions on which link to move to. The output from the node pass includes the number of vehicles that remain in the link-end queue, and the number of vehicles added to and subtracted from each link during this time step. If a node is an intersection, the effective green times for the signal phases are calculated according to the specified signal control (see below for more on the traffic control simulation approach) and the clock time. This determines the upstream inflow constraints for the links leaving the node.

Travel Time Calculation and Queuing: The link travel times are calculated at the end of every time step and communicated to the path-processing component for finding current path trip times. The travel time (T_i^t for link i at time t) consists of the moving time (Tm_i^t) as well as the queue waiting time (Tq_i^t). That is,

$$T_i^t = Tm_i^t + Tq_i \quad (4)$$

The moving time is calculated based on the current link speed (v_i^t) and the available length of the link, (which is the length of the link, (L_i) minus the length used by the queue), as,

$$Tm_i^t = \frac{L_i - Q_i^t * l_c}{V_i^t}, \quad (5)$$

where Q_i^t is the average queue length across the lanes and l_c is the assumed vehicle length. Queuing occurs at the end of the links. However, the vehicles move till the back of the queue at prevailing speeds, and the time spent moving in the queue is included in the queue waiting time. The queue waiting time (Tq_i^t) is calculated by dividing the queue length by a moving average of the discharge rate over a specified number of time steps.

Incidents: Incidents are modelled by reducing the capacity (in terms of effective lane miles) on specified links by specified fractions. Any number of such incidents can be simulated by specifying the starting and end times and the capacity reduction factors for each incident. When the effective lane-miles is reduced for a link during the simulation, the calculated densities increase instantly. If it is more than the maximum allowed density, the vehicles are still moved at jam speed, till the density falls below the maximum.

2.4 Traffic Control Simulation

DYNASMART provides the ability to explicitly model an array of control elements, listed in table 1. How each kind of control is simulated is explained in the following subsections.

Table 1. Traffic Control Types in DYNASMART

Surface Street	Freeway System
No Control	Ramp Metering
Stop Signs	Pretimed
Signal Control	Demand-responsive(ALlNEA)
(green, red, amber time,	Variable Message Signs
cycle length, offsets, phases)	Route advisory signs
Pretimed	Route congestion warning signs
Pretimed Coordinated	Speed control signs
Multidial Pretimed	
Actuated (full)	

Link (Approach) Outflow Constraints: The outflow constraints limit the maximum number of vehicles that are allowed to leave the link at an intersection approach. These constraints are described in the following equation which states that the total number of vehicles that can enter an intersection depends on the number of vehicles that reach the end of the link or join the queue during the current simulation interval, At, and the capacity of this approach. By capacity, we refer to the maximum number of vehicles that can go through the intersection under prevailing signal conditions.

$$VO_i = \text{Min}\ \{VQ_i, VS_i\} \tag{6}$$

where, for link i,

VO_i = Maximum number of vehicles that can enter the intersection during At. (Note, however, that they may not enter unless the receiving link's inflow constraints

are also met),

VQ_i = Number of vehicles that are available to move out of the link during AT,

VS_i = Maximum number of vehicles that can enter the intersection during the At, based on the green time provided,

VS_i is equal to $G_i * S_i$ where G_i is the effective green time during the At and S_i is the saturation flow rate. Note here that only certain movements may be receiving green at the approach during a given phase, and hence the effective green is calculated accordingly. The effective green time is also reduced if a green phase ends during the current At.

Link Inflow Constraints: The inflow constraints determine the maximum number of vehicles that are allowed to enter a link. These constraints bound the total number of vehicles from all approaches that can be accepted by the receiving link, they include the maximum number of vehicles from all upstream links wishing to enter link j, the constraint due to the available physical space on link j and the section capacity constraint of link j.

$$VI_j = \text{Min}\{\sum_{i \in U} V_{ij}, VE_j, C_j * \Delta t\} \quad (7)$$

where, for link j,

VI_j = the number of vehicles that can enter the link,

U = set of inbound links into link j,

V_{ij} = number of vehicles that are ready to move from link i to link j,

VE_j = Available space in the link (in terms of the number of additional vehicles that will result in jam density), and

C_j = inflow capacity, which is normally 1800*(number of lanes on link j) vph.

Note that $\sum_{\text{all } j} V_{ij} = VO_i$ here. The vehicles are moved one-by-one into link j in a first-come-first-served manner, and counters are kept to ensure that the outflow and inflow constraints are all satisfied.

Unsignalized Intersections: This is the case of intersections with stop control and yield control. As there is no signal timings involved in this case, all the movements receive equivalent green times in the ratio of the critical volumes during each time step.

$$GE_i = \frac{CVQ_i}{\sum_{\text{all } k} CVQ_k} \Delta t \tag{8}$$

where,

GE_i = Equivalent green time for approach i for the current At.

CVQ_i = Critical vehicle volume in queue for approach i, and

At = Simulation time step.

Signal Control: Signal control includes pretimed signal control, pretimed coordinated control, multidial pretimed control and actuated signal control. Since DYNASMART uses a fixed time increment simulation approach, all the signal operations can be readily modelled based on the time clock. The number of phases as well as the offset, green time, red time and amber time of every phase along with the movements receiving green during every phase, need to be specified in order to model the signals. Because DYNASMART is not intended as an optimizer of signal control, the model user has to prepare offsets obtained from other models to coordinate arterials or the network as a whole.

In the case of the pretimed control, only the allowed movements receive green during each *At.* When the phases change during a time step (currently At = 6 seconds), the effective green times are calculated according to the portion of At that either phase receives.

The case of actuated signal control presents certain complications as DYNASMART is designed to use macroscopic relationships, with no intention of modelling the headways between individual vehicles with platoon dispersion models. This means that keeping counters for individual vehicle actuations at the detectors may not give accurate results. DYNASMART uses an appropriate macroscopic method that determines equivalent green times that are updated to reflect prevailing approach volumes. Thus green splits are apportioned according to the Webster's rule, in the ratio of the measured arrival flow rates. The essential features of actuated signal control, namely "Max Out" and " Gap Out" are reflected through constraints on the calculated green times, as below. If the required green time is larger than the maximum green time or smaller than the minimum green time, the maximum or minimum green time is assigned, respectively. The following relationships are used:

$$G_i = \frac{CV_i}{\sum_{\text{all i}} CV_i} (C - L) \tag{9}$$

subject to

$$\text{Actual } G_i = \text{Min } \{ G_i, \frac{CV_i}{S_i} \}$$

$$\text{Min Green} \leq G_i \leq \text{Max Green}$$

where,

CV_i = Critical volume for phase i ,

C = Default cycle length,

L = Total assumed lost time per cycle

S_i = Saturation flow rate.

These calculations are performed at the end of each cycle. This modelling is sufficiently accurate as it allocates green time in response to the congestion in the network, and has been

considered in the past as a viable approach to capture actuated control with macroscopic models (Sheffi, et al., 1982).

Freeway Control: Any link in the network can be specified to be a freeway link. When the freeway links are in series, the node-pass treats the nodes between such links as if they have continuous green. In addition, the node saturation flow rate is fixed at a much higher value, so that the freeway traffic flow is governed strictly by the continuum equation and speed-density relationship, without nodal constraints. This is very important in capturing the shock-wave propagation on the freeways.

DYNASMART includes modelling of some freeway management techniques, such as ramp entrance control, and variable message signs. Entrance ramp control aims to limit the rate of vehicles entering the freeway in order to maintain the quality of the freeway traffic. Pretimed ramp metering as well as traffic-responsive metering can be modelled.

Pretimed ramp metering is modelled similar to any other signals in the network with appropriate allocation of green times to reflect the specified entry rate. The traffic-responsive ramp metering is modelled according to the current flow conditions. A local feedback control rule, ALINEA (Papageorgiou, 1991), is implemented in DYNASMART currently, along with the options to model the two conventional techniques, namely occupancy control and demand-capacity control. In ALINEA, the signal (green time) rate is set as follows:

$$Rate(T+1) = Rate(T) + C1*(C2 - OCC) \quad (10)$$

where,

$Rate(T)$ = Ramp rate (vehicles/minutes) for specified time period T.

occ = Occupancy at specified freeway detector

C_1 = A specified control parameter (default = 0.32), and

C_2 = A specified control parameter (default = 0.20)

The distance of the freeway detector from the merge point of each controlled ramp has to be

specified, and the occupancy is estimated at each location macroscopically based on the effective length of the detector and the assumed average vehicle length.

Variable Message Signs (VMS) are another part of the modelling system. Three major functions currently identified for VMS are speed advise, route advise, and route congestion warning. Though DYNASMART has the facility to accept external data on speed advise, as well as externally specified paths (one at a time for each VMS) for route advise or congestion warning, the driver response to them is based on specified probabilities to accept the advise. Much further research is needed on driver response to VMS before these facilities in DYNASMART can be refined for practical use.

Left - Turn Movement: Left-turn movement is a critical factor causing traffic delay in urban networks; however, it is difficult to model the left turn movement in a macroscopic simulation model, and hence this is an issue that has been carefully considered in DYNASMART. The left-turn capacity is determined by several factors: opposing volume, number of lanes on the opposing approach, and green time for the phase under consideration. The modeling process is summarized as follows :

1. Calculate the left turn capacity;

   This capacity can be calculated under different situations :

   a) Protected left turn phase : based on the saturation flow rate.

   b) Permissive phase : from established tables (we currently use the data from the Highway Capacity Manual of 1985).

2. Calculate an average number of left-turn vehicles and reduce the saturation flow rate for straight and right-turn approaches.
3. Follow outflow-inflow constraints to transfer vehicles from link to link.
4. Calculate the left turn delay for the path processing module. This is done by keeping a moving average of the queue clearance rates over a specified number of time steps. The current queue delay is calculated based on this rate and the current queue length.

2.5 Modelling of Network Path Dynamics

The relative attractiveness of alternative paths can rapidly change in networks with ATMS/ATIS, due to the drivers' decisions on which routes to drive on, and the changing supply characteristics engendered by the kind of real-time control strategies envisioned under AT'MS. Two different aspects need to be modelled: 1) the routes shown periodically by the controller or ATIS and 2) the routes that drivers have in their minds (which, at a given time, may be a path displayed to the driver *at an earlier time).* This directly translates into the necessity for the simulation framework to store (1) the current alternative paths from various nodes to the various destinations with their trip times, and (2) the current paths of the individual drivers. These two sets of paths need not have one-to-one correspondence. The reason is as follows: when a driver makes a decision at a node, it is based on a comparison between the current path in his/her mind and the paths displayed at that time, where the displayed path set may no longer contain the current path in the driver's mind. Thus we need to store one path for each driver, in addition to the displayed current paths. In DYNASMART, the routes at the network level are stored efficiently using predecessor pointers as they are contemporaneous paths which are part of the shortest path trees. The routes in the minds of the individual drivers, however, are stored as separate lists, as they are not contemporaneous across the drivers (i.e., the current path of different drivers are selected at different times from different shortest path trees even when they are going to the same destination).

In addition to the above, DYNASMART also provides the option of storing externally specified paths. This important capability is needed for the realistic modelling of the driver selection of initial paths, which can have significant implications, as discussed below in the section on illustrative results. Those drivers not receiving any information would be selecting a route which is not based on complete information on the prevailing traffic conditions. Their selections may be based on "historic" patterns that they have learned over time. Such paths can be specified exogenously to DYNASMART. These paths are input as node-lists with a "time-stamp" showing when each string become active. Thus, it is possible to use time-dependent equilibrium paths between O-D pairs, externally found using appropriate programs. We have successfully used the paths that CONTRAM assignment program finds for the vehicle-packets

at various times. The user also has the option of storing the k-shortest paths at a specified time during the simulation (say, at the end of the initial network load-up period) and routing all drivers to initial paths which are picked from these, either randomly, or according to origin-based path choice rules.

The ability to store externally input paths and to simulate particular driver assignments to specific paths is essential for the use of DYNASMART as a simulator in the context of procedures for solving time-dependent network assignment problems. For instance, a central controller seeking an optimal assignment of vehicles to paths would be interested in a system-optimal (SO) time-dependent assignment. DYNASMART has been used as the traffic simulator in algorithms to solve the SO assignment (Mahmassani and Peeta, 1993).

One significant reason for the flexibility of DYNASMART in modelling various driver response mechanisms and various route information supply strategies is its ability to find and store multiple paths efficiently, rather than a single shortest path for each O-D pair, such as in other simulation programs. This is necessary to model complex decision mechanisms such as selection from multiple discrete alternatives, and advanced display mechanisms such as multiple paths shown by a non-prescriptive guidance system. Thus, k-shortest paths (shortest, 2nd best, 3rd best etc.) are found and stored at specified intervals from all the nodes to all the destinations. Two different routines are incorporated: (1) a k-shortest path finding routine and (2) a k-path trip time update routine. Note that the latter routine is an order of magnitude faster than the former. The user can specify the intervals at which the former routine is used to find the k-shortest paths and can specify the trip times on these same paths to be updated at various subintervals between successive path-finding operations. This results in efficient use of computer time resources because the paths in the set of k-shortest paths between each O-D pair drop out of the set far less frequently than change positions among themselves within the set, as network conditions change. It is indeed possible that in the real-world ATIS centers, different computers could be handling the time-intensive shortest path finding operation, and the faster operation of updating the path conditions, and DYNASMART can simulate such cases.

Finding the k-shortest paths

It has been our experience that the path-processing component requires up to even two-thirds of the computation time for networks of reasonable sizes; for this reason, this is a subject of continuing development and refinement in DYNASMART. In the initial version of DYNASMART, the k-shortest path algorithm was implemented using a direct modification of the well-known label-setting algorithm (see Minieka, 1978, for instance) by associating a k-vector label with each node. During every iteration of the algorithm, one label among the label vectors of all the nodes is finalized. The cost labels are stored in a binary heap (Fox, 1978; Tarjan, 1983) for efficiency. During every iteration of the label-setting algorithm, the minimum is removed from the heap, and the heap is rearranged using very efficient operations. This results in a worst case computational efficiency of $O(Nklog_2Nk)$ as opposed to $O(k^2N^2)$ for a label-setting algorithm based on list-minimization (where, N is the number of nodes). The k-shortest path building is repeated with each destination node as the root.

Because label-correcting algorithms (see Minieka, 1978) are known to perform better than label-setting algorithms for some traffic networks (Van Vliet, 1978) even though their worst case performance is theoretically worse, a k-shortest path label-correcting algorithm using a "deque" (double ended queue) data structure has also been developed and implemented (Ziliaskopoulos, 1992). DYNASMART can be used with either a label-correcting (deque) or a label-setting (heap) implementation. The current implementation of the shortest path algorithm does not lend itself to efficient vector processing; however, multiprocessor versions have been developed for building trees rooted at different destinations using different processors.

Updating the k-shortest paths

The trip times on the network links vary during the simulation and hence the current trip times over the paths have to be found by summing the trip times on the constituent links after each simulation time step (or as specified by the user). Updating the trip times over the paths found earlier using the k-shortest path algorithm can be achieved an order of magnitude faster

than finding the paths themselves. As significant stretches are common to numerous paths (which is especially true in the case of k-shortest paths in urban networks), it is important that aggregation of link trip times not be repeated over such stretches. To accomplish this, a two-pass updating technique is used in DYNASMART. Note that the shortest paths from one root node to all other nodes form a tree, the second shortest paths form trees that branch off nodes on the shortest path tree, the third shortest path trees branch off nodes on earlier trees, and so on. So the path trip times are summed along successive shortest path trees. In addition, any time we reach a node to which path trip times have already been accumulated, we stop adding trip times along those paths, thereby efficiently preventing multiple updating of common stretches. This updating can be carried out efficiently by pre-ordering the tree-traversal. It can be shown that this is an $O(kN)$ update process. For vector machines such as the CRAY supercomputer, the above technique of path update is not very efficient, as the process is sequential and cannot be vectorized easily. Vectorizable procedures using list storage instead of predecessor-tree storage have been tested, but have not been found better than the pre-ordered tree-traversal procedure.

Storage of Paths and Memory Considerations:

The node-arc connectivity is stored using a backward star data structure, by grouping all the arcs incident on each node in a list (of size equal to the number of the arcs), and by keeping pointers to the location of the first incident arc to each node. The backward star is used because the k-shortest paths *are* constructed *backward* from each destination node to every other node in the network. The trees are rooted at the destinations rather than at the origins (note: any node can be an origin of a path), because the number of destination (centroid) nodes is considerably smaller than the number of nodes. These paths form trees and so are stored using efficient predecessor pointers, which require one storage location for every path from each node to each destination.

In addition, the path that each driver perceives that he/she is driving on (his/her current path) is stored as a list. As these paths across the drivers going to the same destination do not form a single tree, they cannot be stored using predecessors. Hence separate list arrays are used

for their storage. Similarly, the dynamic equilibrium paths (to the same destination) that are read in externally also need not form a single tree, and hence have to be stored as separate list arrays. The largest arrays in DYNASMART are these two list arrays. For example, in the Anaheim network of 430 nodes and 38 destinations, 80,000 vehicles have to be generated for simulating 90 minutes of traffic. As the longest possible reasonable paths have about 50 nodes, this translates to 430x38 x50 = 817,000 locations for external path storage and 80,000 x50 = 4,000,000 locations for storing the drivers' current paths. Other storage techniques that make use of the fact that many paths are much shorter than the maximum possible path length, are being developed now.

2.6 Modeling of Driver Response

It is assumed that for different alternative designs of the information/routing system, the basic information that is ultimately available to the drivers will include the actual or predicted travel times on alternative routes, or in some cases on the "best" routes determined by the system. However, it is not reasonable to assume that the drivers will always follow the "best" routes displayed, and hence behavioral rules on driver response must be incorporated in a flexible simulation framework. Sufficient research has not been conducted in the past to determine the nature of driver response and to develop useful models. Experimental evidence presented by Mahmassani and Stephan (1988) suggests a boundedly-rational model. This means that drivers may be looking for gains from route switching only outside a threshold, within which the results are satisfying and sufficing for them. In other words, a driver may decide to stay on the current route unless he/she expects sufficient benefits from an alternative route. A simple rule has been currently implemented in DYNASMART that captures this behavior.

$$\delta_j(k) = 1 \text{ if } TTC_j(k) - TTB_j(k) > \max(TTC_j(k) \bullet qj, \tau_j) \qquad (11)$$
$$= 0 \text{ otherwise}$$

where, for driver j at node k, 1 indicates a route switch and 0 indicates no switch.

$TTC_j(k)$	=	Trip time from node k to destination on current path
$TTB_j(k)$	=	Trip time on the best alternative path
η_j	=	Relative indifference threshold fraction.
τ_j	=	An absolute minimum trip time advantage that the driver requires for a route switch

Here the driver's indifference threshold is a fraction of the remaining trip time on the current path. This fraction (η_j) is currently assumed to be distributed across the driver population according to a user specified distribution. The model also assumes that drivers will always require a minimum trip time improvement (τ_j) to consider switching, especially when the trip time on the current route itself is small.

This decision routine in DYNASMART implements a simple response mechanism, but is designed to be flexible for future modification. Another candidate mechanism used to model their switching to an alternate route is a probabilistic discrete choice model based on the travel times on alternate routes (if such times are displayed by the information system). This is easily incorporated in the program because the k-shortest path approach stores multiple paths. Another special case is that of myopic switching, where the user always switches to the current best path if different from the current path (such as in prescriptive route guidance). This case can be modelled by setting the indifference threshold to zero, forcing all drivers to select the shortest trip-time path shown.

2.7 Program Capabilities

DYNASMART has so far been implemented on two different computer platforms: the CRAY YMP supercomputer and the SUN SPARC II work station, both with 64 MBytes of primary storage. As the code is written in standard portable FORTRAN 77, it is expected to run on other platforms, especially as RAM storage continues to become cheaper. Simulation of up to 75,000 vehicles in networks of up to 2000 links with 5 paths from each node to each

destination centroid can be achieved on a SUN SPARC-II work station faster than real-time. The capability to attempt larger problems directly depends on the available RAM size.

The program capabilities include:

1) Macroscopic modelling of traffic flow dynamics such as congestion formation and shock wave propagation; tracking of locations of individual drivers.
2) Modelling of different traffic control strategies (freeways, surface streets, signalized intersections, ramp entry/exit etc.)
3) Modelling of prescriptive/normative guidance as well as descriptive guidance with trip time information on alternative routes.
4) Modelling of various aspects of the controller such as frequency of updates of the network route information database.
5) Modelling of individual drivers' responses to information in the case of prescriptive guidance and their selection from a set of paths rather than a single shortest path; random assignment of driver behavioral characteristics. Flexibility to incorporate alternative behavioral rules.
6) Modelling of specified capacity-reducing incidents at any specified time and location in the network.
7) Modelling of traffic with only a fraction of the vehicles equipped to receive information.
8) Capability to carry out simulations based on externally specified dynamic equilibrium paths for drivers not equipped to receive information.
9) Several levels of output statistics for the system, for individual drivers as well as for groups of drivers (equipped drivers, unequipped drivers, drivers on certain O-D pairs etc.). Statistics include average trip times, distances, average speeds etc. The number of route switches made, as well as the average fractional distances at which the successive switches are made are calculated. A few other statistics for insights on the path dynamics are also available.

Chapter 3
THE FRONT-END

3.1 Introduction

Though DYNASMART is a flexible simulation tool, as is evident from our explanation of the model, like any other large piece of software, DYNASMART takes large amounts of data as input. This makes the data set preparation a tedious and time consuming job. For better organization and better understanding of the input data, a user interface that facilitates data entry, is necessary. The graphics front-end to DYNASMART evolved primarily out of this need. The front-end not only improves the aesthetics of the core simulation program but also improves the functionality of the simulation software, thereby enhancing the practical usefulness of the model. The utility of the front-end is however not restricted to the convenience in data input. The front-end also makes it possible for the program and the user to exchange information during run time with a bidirectional communication protocol. This feature enables us to fine-tune the data during a simulation without restarting the entire simulation run. The display of intermediate simulation results improves the user's confidence in the integrity and validity of the data set. The user can choose to interrupt, pause or modify certain simulation variables at run time.

The development phases of the front-end are discussed below followed by a detailed description of the user friendly features and capabilities.

3.2 Development Phases

The front-end to DYNASMART was implemented using the Motif library which is based on X-Windows. The broad acceptance of X as a standard for distributed windowing systems in the industry and academia, the visual similarity of the Motif GUI with the popular non-X GUIs (e.g. OS/2 Presentation Manager, MS-Windows) and the availability of Motif UIMS software for rapid prototyping were the influencing factors for this choice. Now we describe the three phases of development of the interface.

In phase 1, the front-end functions as a data input module. The collected data is

formatted, organized and deposited into a set of files. The file system is used as a rendezvous point between the simulation process and the front-end process. The names of the files in which the data is deposited are all pre-determined.

In phase 2, the file system acts as an information exchange point. The simulation process reads the input data from the file system. The input data is common to both front-end and DYNASMART. The communication is still one way, from the front-end to DYNASMART. This modification will allow DYNASMART to run on a powerful machine with the front-end running on a smaller machine.

In phase 3, the communication is bidirectional. The simulation program periodically outputs its intermediate results to the front-end through the file system and the front-end appropriately displays these results. Another feature is the ability for the user to issue signals to the simulation process to pause, upon which the user can inspect and/or modify the variables of the simulation process. As altering the simulation data at run time can lead to consistency problems, the simulation process can be paused only in certain states and only certain data items can be modified.

3.3 Front-end features

The front-end has various features to facilitate data input. A given data set can be selected using the directory display as shown in figure 3.1. If a new data set is being input, or if an existing data set is selected, the front-end comes up with a screen having a canvas (whose size can be specified), a palette of drawables and a menubar on top. This is shown in figure 3.2. All non-drawing activities are performed through various sub-menus of the top menubar. These are discussed in detail later. The palette of drawables, with its various icons, is used to create the nodes, links and the zones. There is also an option to select these items on the canvas. A description of the palette of drawables and the top menu bar as well as their functions is given in the following sections followed by a discussion on the specification of intersection-signalization data and the link characteristics.

3.3.1 Palette of drawables

The palette of drawables has five icons as shown in the figure 3.3. The "Node" icon is used to create a node at any point on the canvas. Similarly, the links between nodes can be created using the "Link" and "2-Link" icons. The "Link" icon is used to create a one-way link from a selected node to another selected node. The "2-Link" icon is used to create a two way link. This means that the road has vehicles flowing in both directions, and the front-end creates a separate link for each directions (as is the convention with DYNASMART). A zone can be specified by selecting the "ZMark" button from the palette and then clicking on the nodes to be included in that zone. The user will be warned if inclusion of the same node in multiple zones is attempted. The "ZEnd" icon is used to specify the centroid (destination) node for a particular origin zone. Thus the specification of nodes in a zone is always followed by the specification of the centroid node, at which point the front-end recognizes that all the nodes for that zone have been selected. The "Select" icon is used to select an object (ex. nodes, links or zones) in the canvas. Thus the palette of drawables enables one to easily draw even a very large network which would otherwise require considerable manual labor.

3.3.2 The top menu bar

All non-drawable activities are performed using this top menu bar. It has four pull-down menus the details of which are shown in fig 3.4. The file operations (opening, loading and saving), edit control operations (undo, cut and paste) etc. are performed using the various submenus of this menu bar.

"File" menu: Any network data file can be opened using the "Open" option of the "file" submenu, which brings up the directory, as shown in figure 3.1. The front-end automatically brings up the network that is specified by the data file. The "New" option of this menu is used to create new networks and hence new network data files. The palette of drawables is used to create a new network or add new objects (links, nodes or zones) to an existing network. At any time during the preparation of a new network or during the editing of an existing network, it can be saved using the "Save" or "Save as" options of this menu. The last option of this menu "Quit" is used to exit from the front-end. As is easily seen, these are the most common conventions that

users of Macintosh and Windows-based PC software are very familiar with.

"Edit" menu: Any editing operation on the network is done using this menu. Any object (link, node or zone) can be deleted by selecting the particular object and choosing the "delete" option of this menu. The "Clear" option clears the window. The "Refresh' option is used in conjunction with the "Grid" option of the "Other" menu. This is used to clear off the grid pattern that appears on the canvas so as to have a plain background.

"Select" menu: The "Select" menu is used to get information about a link, node or zone of a network. Once the "Node" option of this menu is chosen, a dialog box pops up, which gives the labels for all the nodes in the network. On choosing a particular label, information about that particular node is displayed on a separate window, as shown in figure 3.10. This information has all the node-specific data, which are mostly signalization data as discussed in sec. 3.3.3. Similarly, the "Link" option is used to obtain information about the link-specific data discussed in section 3.3.4. The "Zone" option brings up a dialog box listing the zone numbers for all the zones in the network, as in figure 3.1. On choosing a particular zone, details such as the nodes in that zone and the destination centroid for that zone is displayed, by highlighting the nodes in the zone.

"Other" menu: This menu is used to control the simulation run, as well as certain display characteristics. The "Grid" option of this menu is used in conjunction with the "refresh" option of the "Edit" menu to obtain or clear the grid pattern which appears on the canvas. The "Stop" option is used to stop the simulation program, DYNASMART, temporarily. When this option is chosen, a signal is send to the DYNASMART Unix process, which stops running and the front-end is ready to obtain intermediate results. The results are written on to the network data file which is common to both DYNASMART and front-end. Front-end then reads the data from the file and indicates the change in the variables (eg. traffic density on a link) graphically. Thus the user is given a better overall picture of the flow in the network as the simulation is proceeding. The "Exec" command is used to send a signal to the temporarily interrupted simulation program (DYNASMART) to start running again. The "Magnify" option is used to

magnify a portion of the network. This is particularly useful if the network represented is very large such that it is bigger than the entire area of the canvas when it can be viewed only by scrolling. The "Shrink" option is used to shrink back the portion of the network that is magnified. The "Magnify" and "Shrink" options could also be used to concentrate on specified portions of the network where ATMS control options are being tested. See figures 3.5 and 3.6 for an example of an area in the Anaheim network where the "magnify" option is used

3.3.3 Signalization Data

The front-end enables one to specify the traffic control and other intersection-specific data. The data relevant to a node (intersection) can be specified by just "shift-clicking" on the node. This action brings up a dialog box (see figure 3.8) that has node-specific data such as:

1) Node no:
2) Type of traffic control
3) Number of signal phases
4) Signal cycle length.

The node number is used to specify a label to a particular node. This feature is useful for recall, search and location of a node by name. Any number can be assigned to a node though it is a usual practice to assign them in an ascending order starting from 1 (which reduces the maximum node number in DYNASMART and saves storage). A node is uniquely identified by its node number. Any data relevant to a node is retrieved using the node number. This is done using the "Select" option as explained before. The user is warned if the same node numbers is assigned to two different nodes.

The type of traffic control is specified using the second option in the dialog box. There are four types of control that can be specified. They are: no control, yield, stop, pretimed and actuated signal control. Different numbers ranging from 1 to 5 are used to specify the type of control. The numbering is the same as in the conventional DYNASMART input file: 1 is used to specify that the intersection has no control, 2 for yield control, 3 for stop control, 4 for pretimed signal control and 5 for actuated signal control.

The number of signal phases for a node is specified by the third option above. Each signal cycle is divided into phases and these phases are numbered. Note that a signal phase

exists only for a signalized type of control. So a value is specified for this option only if the type of control (see last paragraph) is either 4 or 5. A signal cycle usually has four phases. The fourth option is used to specify the signal cycle length.

Once the above node-specific data is specified the data related to each phase in the cycle, for a signalized type of control, has to be specified. This is automatically taken care of by the front-end. Every time the node-specific data is specified/altered, the front-end checks for the type of control in the node-specific data. If the control is a signalized type, then depending on the number of phases (n), front-end brings up n different dialog boxes for the specification and/or modification of the data associated with each phase in the signal cycle. Each dialog box asks for information such as:

1) Offset green time
2) Actual green time
3) yellow time
4) Green approaches
5) Upstream green nodes.

The offset green time for a particular phase is specified using the first option. This information is used by DYNASMART to model a coordinated type of control. The second option is used to specify the actual green time for a signal phase and the yellow time for that phase is specified using the third option. The fourth option is for specifying the number of approaches receiving green during the signal phase; i.e., the number of movements that are allowed during the phase. The movements themselves are specified using the last option. This option is used to specify the upstream nodes of the links from which the traffic has green during the phase of concern. These upstream nodes are denoted by their corresponding node numbers. Here, the user has the option of typing the node number, or to go to the display canvas and selecting the node. Note that the version of DYNASMART developed during the PATH MOU-39 research did not provide the option of protected left-turn-only phases etc., which means that the movements could be specified with only the upstream nodes of arriving traffic. This is being modified currently, and the front-end will include the option to specify both the from-node and the to-node of each movement, in the future.

Thus the front-end facilitates the specification/modification of the intersection-specific as

well as the signal phase-specific data for a node. All these data relevant to a node, once specified/modified, are written on to a network data file by the front-end. These data are organized and stored in a specific format by the front-end so that the file system can serve as a the supplier of input data for the simulation process.

3.3.4 Link-characteristic data

Similar to the node data, the link data can also be specified using the front-end. The dialog box for specifying link data is brought up by choosing the "Select" icon from the palette and "shift-clicking" on the link for which the data is to be specified/modified (See figure 3.9). The dialog box for link data has details such as:

1) Length
2) Volume source
3) Number of lanes
4) Free flow speed
5) Saturation flow.

The first option is used to specify the length of the link. The volume source for a link is specified by the second option. The volume source for a link is given a value, 1 if the volume generated on that link is from the zone of the link's downstream node, a value of 2 if it is generated from the link's upstream node and a value, 0 if no volume is generated on the link. This information is used by DYNASMART to specify the zone in which the link is present. The third option is used to specify the number of lanes in a link and the free flow speed associated with a link is specified by the fourth option. The free flow speed is the maximum possible speed of the traffic in a link. The saturation flow is specified using the last option. This is the flow after the start-up delays, when a queue clears through the downstream intersection.

All these data specified/modified for the links are automatically written on to the network data file by the front-end. This data file, as explained before, contains information about the nodes too. The data file also contains information about the total number of zones and the destination centroid (node) corresponding to each zone. Thus the front-end facilitates the creation of a network data file which contains information about the nodes, links and zones, organized in a specific format, as required by the DYNASMART simulation tool. This file system can now

be used as an input to DYNASMART.

3.3.5 Traffic Demand Input

It is to be noted here that the traffic demand is specified as an external input to DYNASMART. So, in addition to the file system created by front-end, DYNASMART also requires traffic demand in the network as input. This demand data is specified through an external file. Thus an O-D matrix containing information on the trips between the origin zones and the destination centroids has to be created. This information can be got from the planning studies conducted for that network. Also DYNASMART requires a dynamic representation of demand i.e. demand between zones in the network is specified for various time intervals on a day. Thus , an O-D matrix is required for each time interval. So, if there are N zones in a network and if demand is specified for T time intervals then we require T O-D matrices or N^2*T cell values as input.

3.4 Run-time Front-end functions

The utility of front-end is not restricted to creation of the large network data file, which would have otherwise involved a lot of time and labor, but it also has some important run-time functions. These functions make the front-end not only more attractive as an enhancement feature to DYNASMART but also increases its practical usefulness as a simulation tool. These functions are discussed below.

3.4.1 Control of simulation run by front-end

One of the primary advantages of front-end is its ability to communicate to the simulation process during run time. As said before, the “Stop” option of the “Other” menu can be used to stop the simulation process temporarily. On doing this, front-end sends a signal to DYNASMART which is sensed by the later and the program stops running. At this point, DYNASMART is made to output certain specified performance measures (MOE’s) which is graphically displayed by the front-end. The simulation data can also be modified at this point and the interrupted program can be continued with the modified data. The “Exec” option in the

front-end is used to restart the interrupted simulation. Thus the front-end provides a greater control over the simulation process during run time and this enables one to obtain a better overall picture of the traffic conditions in the network during the simulation as opposed to the end of simulation.

3.4.2 Run-time display

The front-end has the capability to graphically display the intermediate results coming from DYNASMART. Even though the front-end was originally written to continuously (i.e., after every simulation time step) display DYNASMART results graphically, this required DYNASMART to stop and wait after updating the display data file till the front-end completed the screen update. As the screen update requires much more time than the time needed to complete a simulation time step in DYNASMART, this delays DYNASMART considerably. Thus it was decided that the screen update would be done only after a user-requested "pause" to DYNASMART. As the DYNASMART text output is available on a separate window, it is easy to see the advancing of the simulation time clock. Thus it is easy to stop the simulation at the appropriate time for an updated screen display, as well as for updating any network data that can be modified.

When DYNASMART is interrupted by the front-end during simulation, some of the results up to that point are written on to the network data file by DYNASMART. The results corresponding to the links or nodes are written at the appropriate data fields in the specified format. For example, the density on the links in the network are expressed as a percentage of the maximum density and the corresponding values are written onto the specific field in the data corresponding to each link in the data file. The front-end reads this value from the file system and displays link colors corresponding to the magnitude of the density value in the link. Three different colors are chosen to represent high, medium and low densities respectively. The corresponding ranges for each category is also displayed on the canvas of the front-end. Thus the user can easily identify the congested links and could decide to change the data or the parameters of the simulation. Also various control options could be tested based on the information obtained during simulation. The speeds in the links and the queues is also be displayed using link colors in a similar manner. Front-end can also display the paths of selected

vehicles. The vehicle to be traced is tagged with a specific number and the links through which it passes are denoted by the specific number. This information is passed on to the file system by DYNASMART which writes the number in the specific field for the link. The front-end reads this data from the file system and displays the vehicle path by coloring the links that form the path.

Thus the front-end enhances the capabilities of the simulation framework and brings DYNASMART closer to being a real time background performance predictor for various research projects. Furthermore, such interactive capabilities together with the run-time functions are essential requirements if DYNASMART is to be calibrated with real time data.

3.5 Default Values

A very important feature of the front-end is the provision of default values by the front-end. These are extremely important while preparing data sets for large-scale networks.

Default signalization is provided only for nodes which have two-way links connected to four other nodes. The cycle starts with a green to the traffic from the node (say, a) with the lowest node number among the four connected nodes. The green is also given to the traffic from the node (say, b) such that the angle a-n-c (where n is the node of concern) is closest to 180 degrees. The important default values are summarized here (Some of these are undergoing modifications, and more defaults are being introduced):

Default demand zone for each link : Value = 2, (i.e, the demand generated on to the link is according to the generation rate of the zone of its upstream link)

Default number of lanes for a link: 2

Default maximum speed: 56 Kph (35 mph)

Default link length: 153 m (500 ft)

Default signal cycle length: 90 seconds

Default number of phases: 4

Default green times: 40 seconds (Or, once the cycle length C is specified, (C − 10) / 2.

If the Cycle length is specified, and the green and yellow times for some phases are given the unspecified phase gets the remaining time)

Default yellow time: 5 seconds

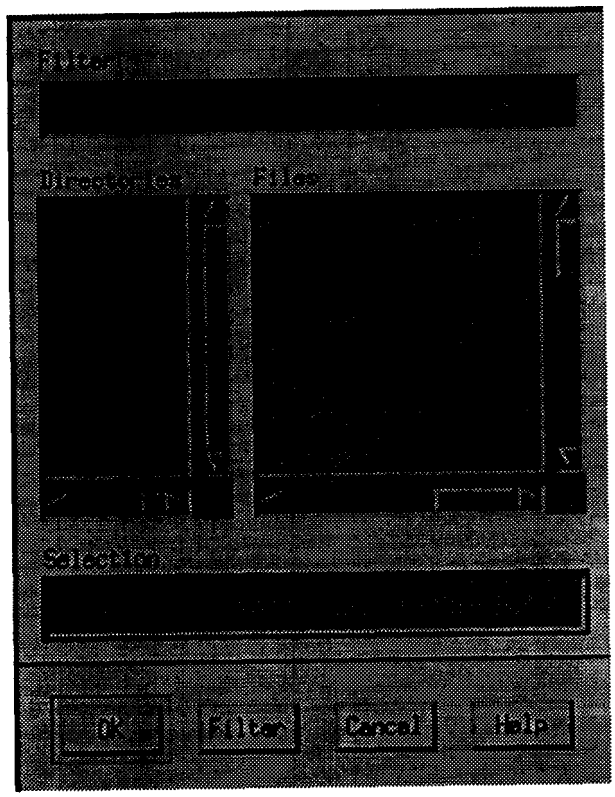

Figure 3.1 The File Directory Display

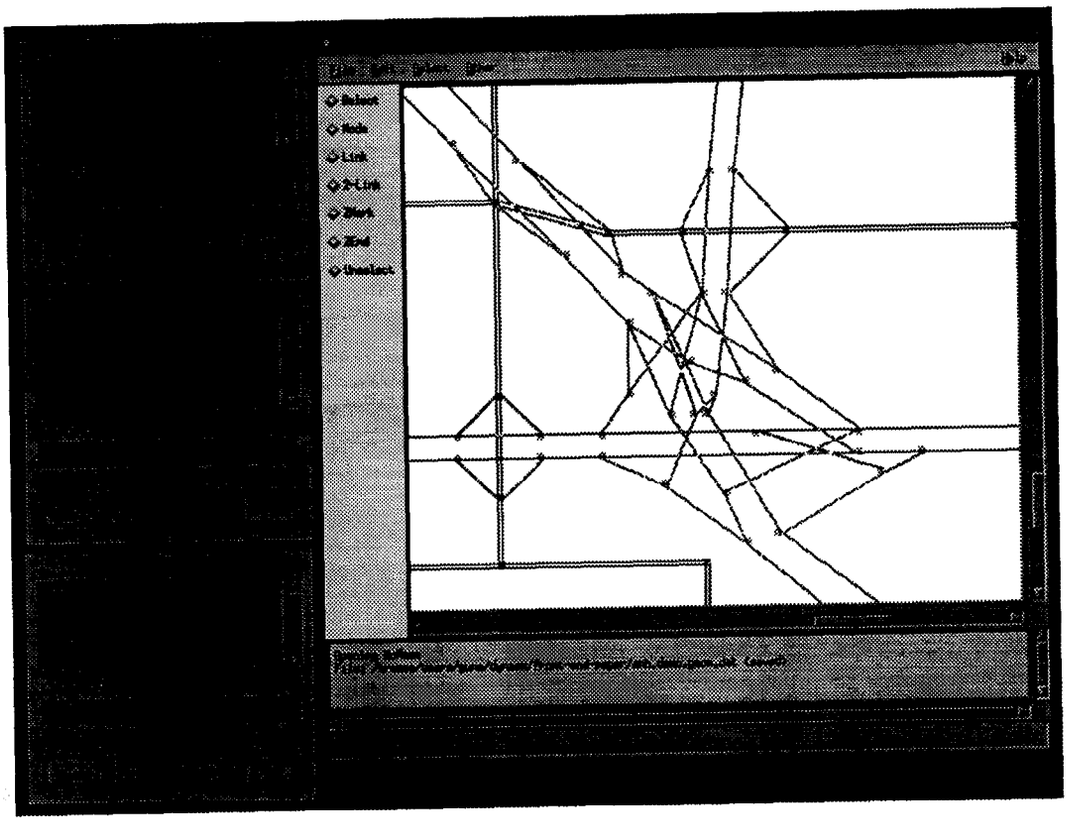

Figure 3.2 The Front-end with Node, Link and Zone Selection Boxes

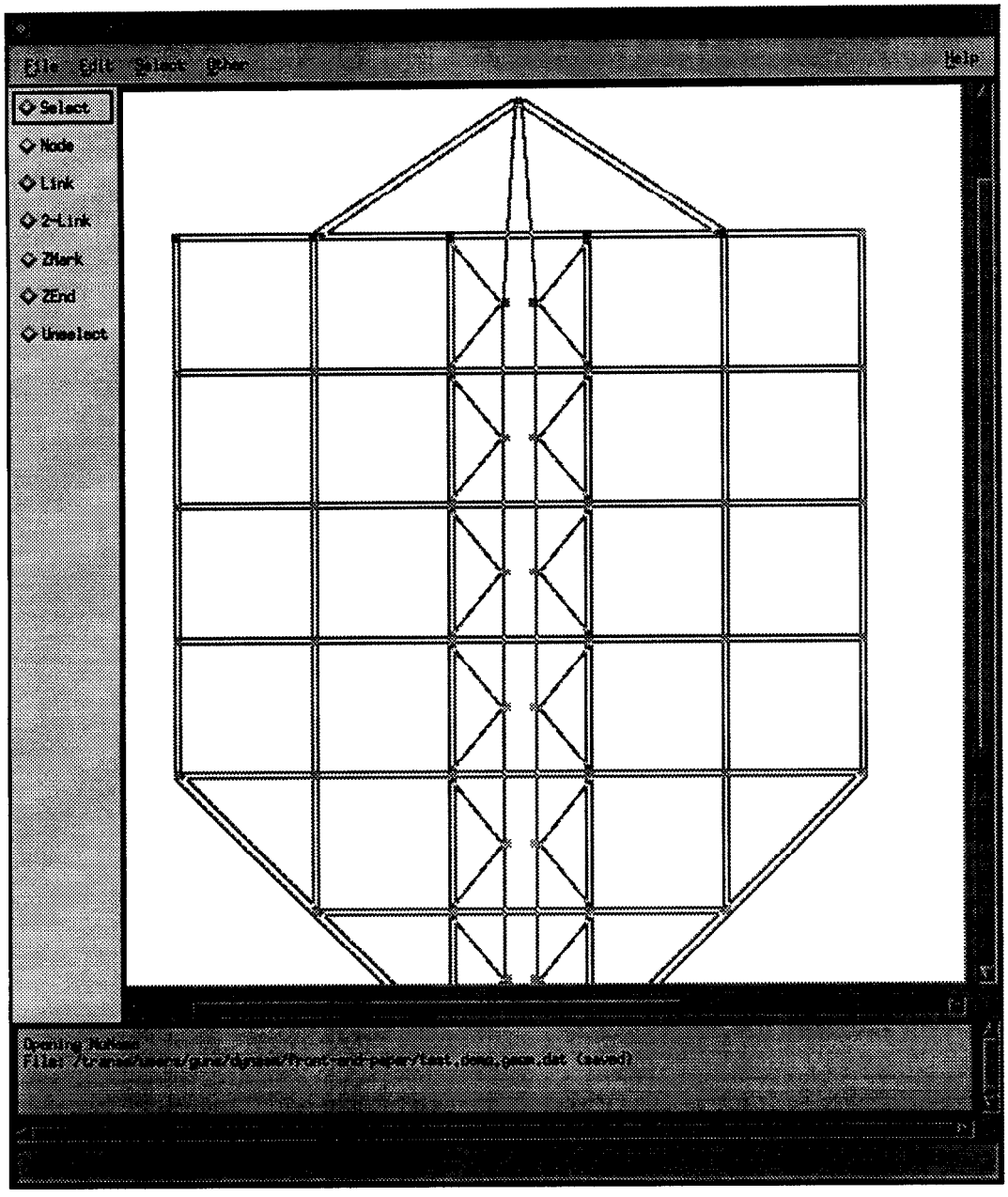

Figure 3.3 The Palette of Drawables (A Test network “Hu-ville” is shown)

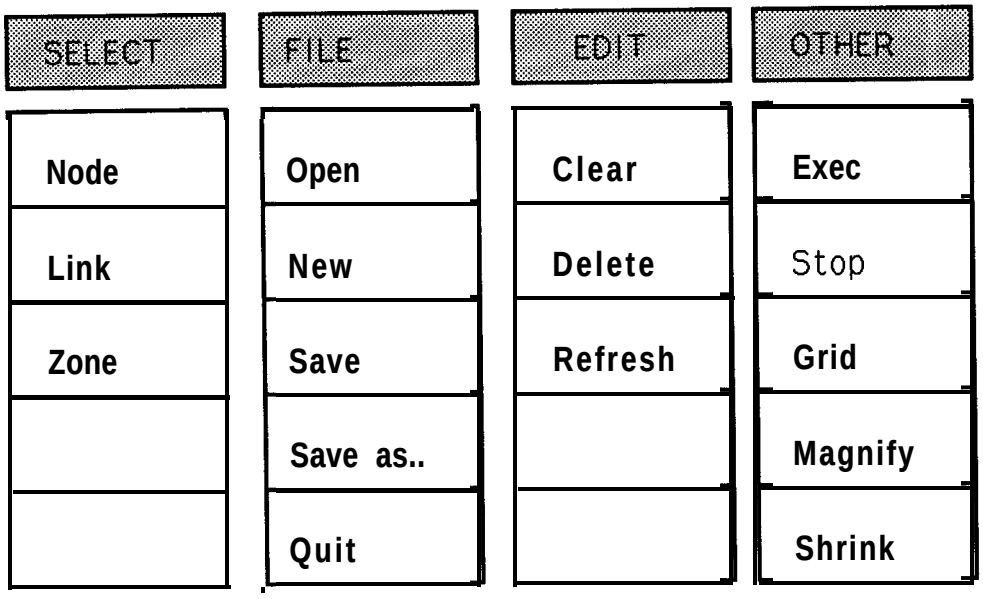

Figure 3.4 Top menu bar pull down menus and their options

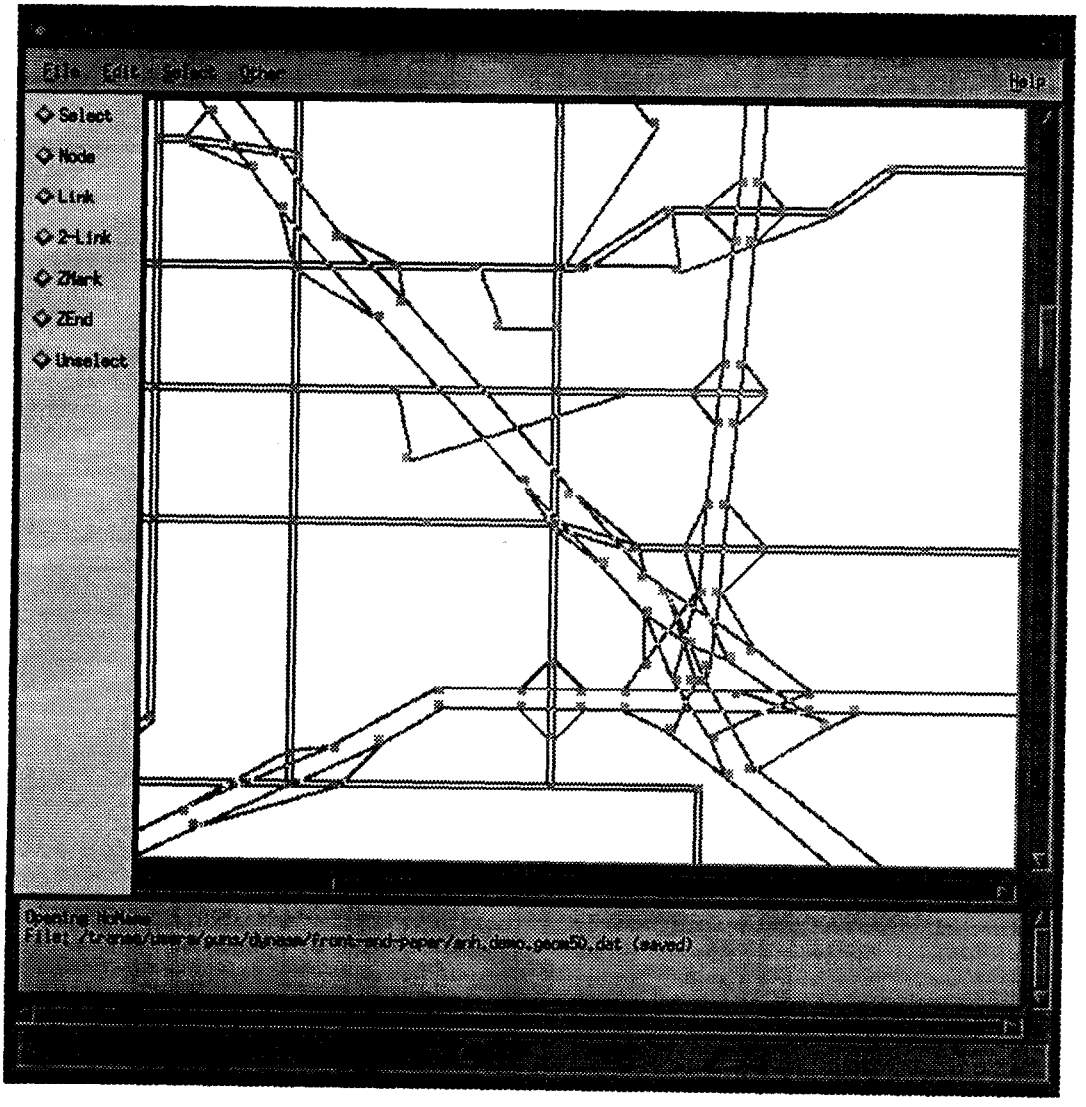

Figure 3.5 A portion of the Anaheim, CA network (at the "Orange Crush" interchange of I-5, I-57 and Hwy-22)

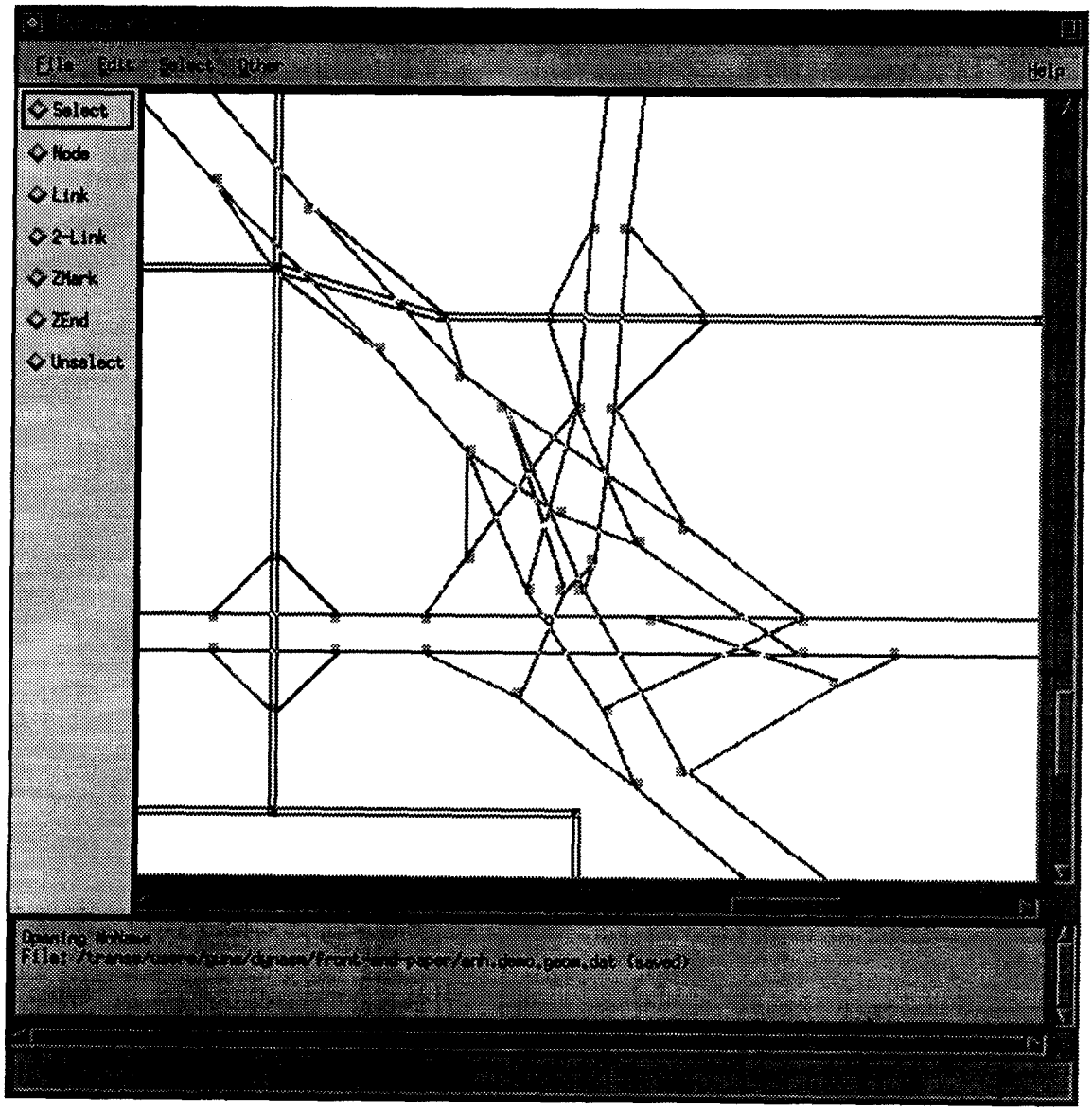

Figure 3.6 A portion of the Anaheim, CA network zoomed in (compare with fig. 3.5)

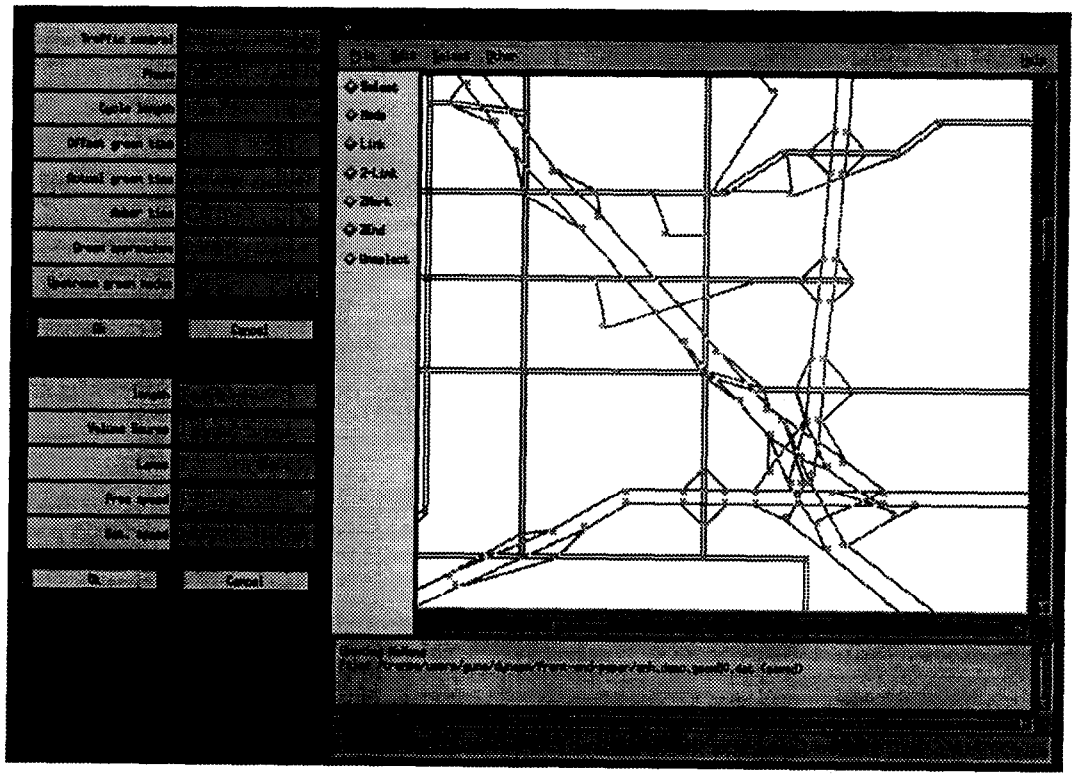

Figure 3.7 Intersection and Link Input Dialog Boxes

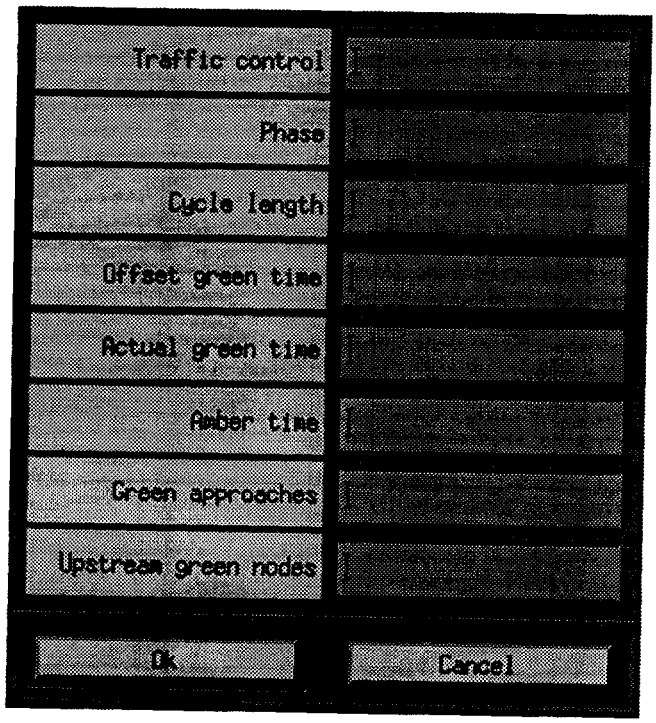

Figure 3.8 Data fields in the Intersection Dialog Box

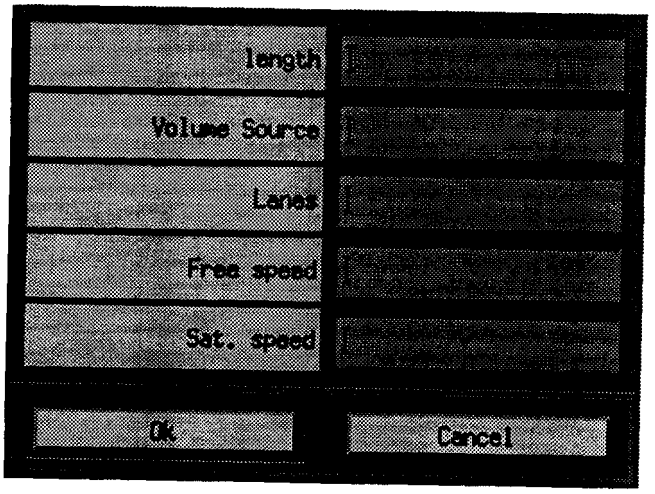

Figure 3.9 Data fields in the Link Dialog Box

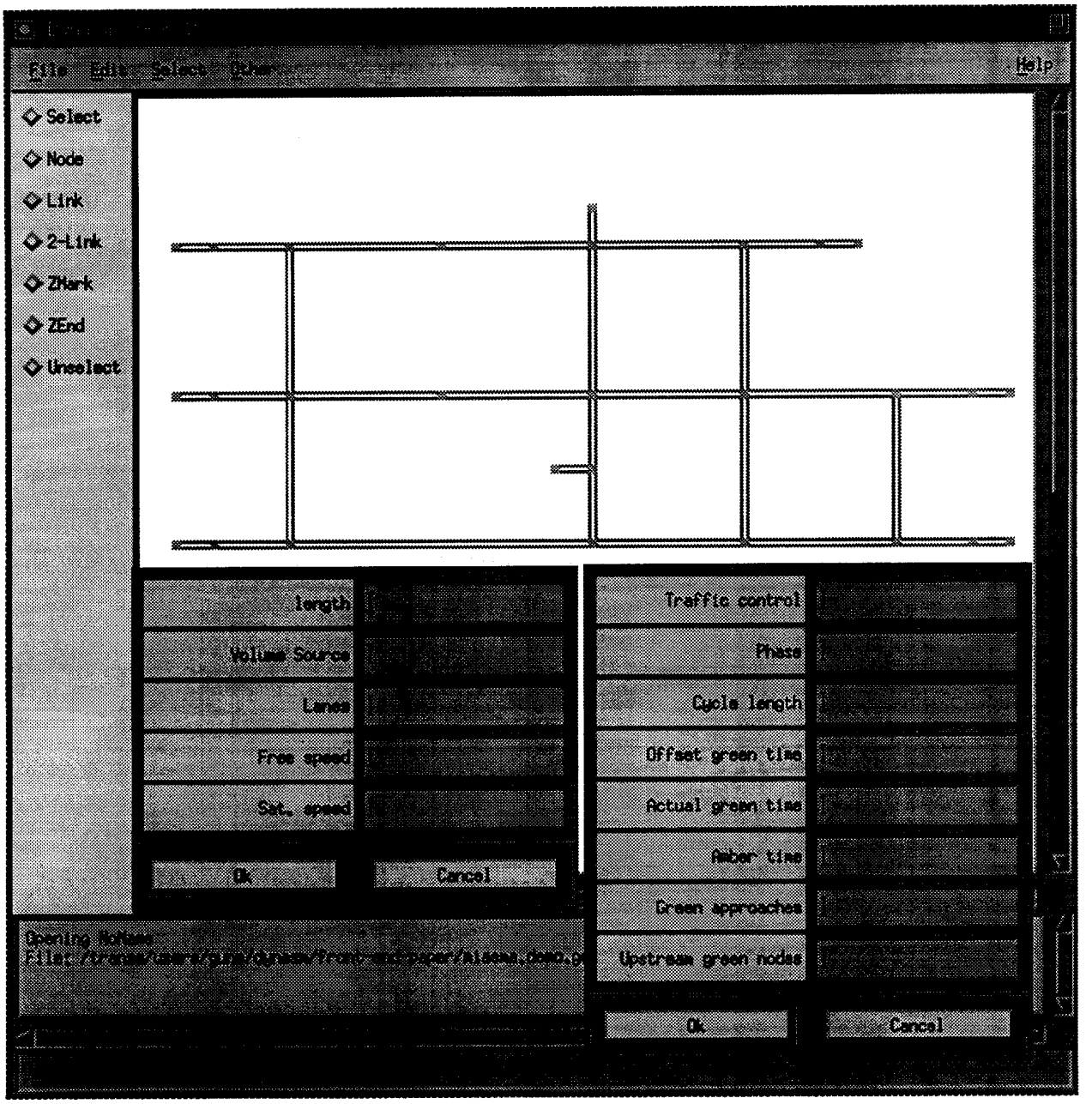

Figure 3.10 Dialog Box display at selected Link and Node on the canvas

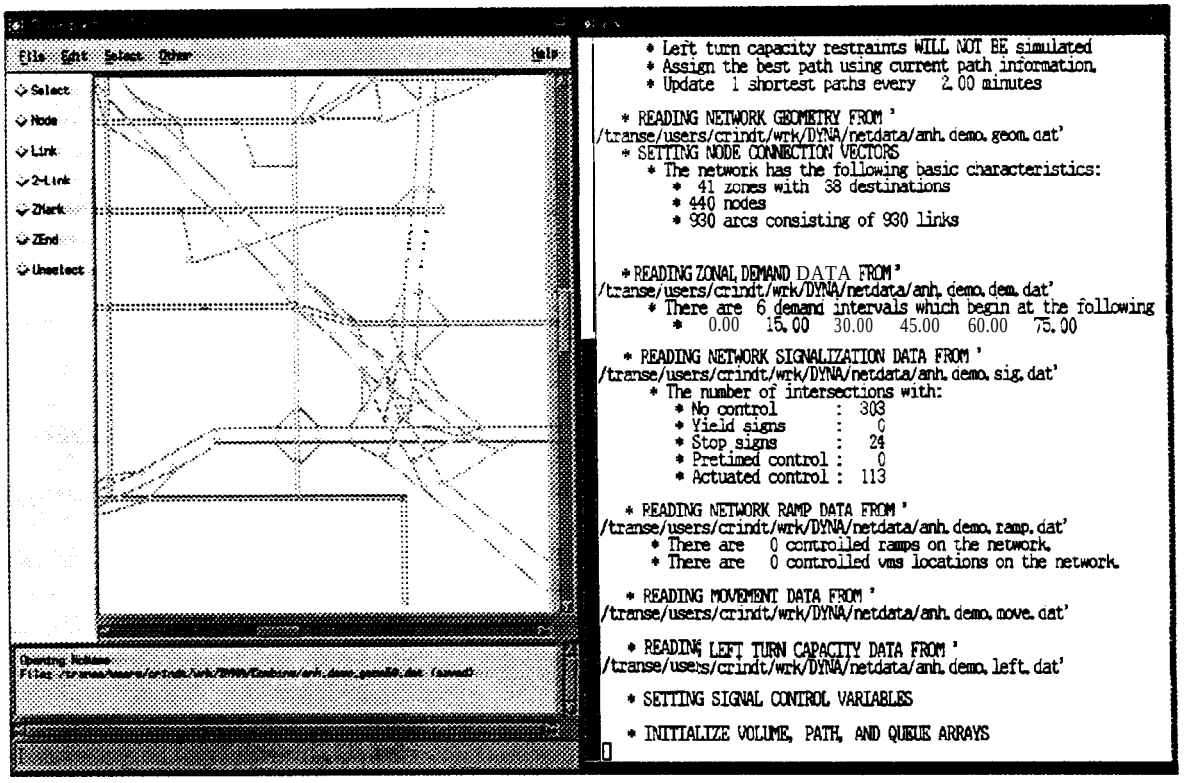

Figure 3.11 Front-end Display and DYNASMART Screen Output on two windows

Chapter 4
REPRESENTATIVE RESULTS

4.1 Introduction

Results are provided for two networks: the core network of Austin, Texas, and the network of Anaheim, California. These results are provided to illustrate the applicability of the approach for studying and evaluating the effects of information supply strategies on realistic networks. The results are intended for illustrative purposes only; not detailed analysis. DYNASMART provides several other measures of efficiency in addition to the aggregate travel times reported here. The discussion is also meant to illustrate the sensitivity of the results to the drivers' initial paths. Extensive DYNASMART simulations will undoubtedly continue to study various ATIS and ATMS strategies, particular traffic demand conditions such as special events, various driver response mechanisms under limited knowledge of paths etc.

4.2 Illustrative evaluation of ATIS strategies

The core network of the city of Austin, TX, consists of 660 nodes and 1750 arcs. The area is roughly rectangular and is bounded by I-35 and Mopac freeways on the east and west sides, by the Colorado river on the south side and by the 26th Street on the north side. The Anaheim city network consisted of 430 nodes and 930 arcs in the area between the SR-22 freeway in the south, the SR-91 freeway in the north, the SR-55 freeway in the east and the Harbor Street in the west. The network also includes the I-5 freeway from the southeast comer to the northeast corner and the SR-57 freeway going north-south through it. These two networks are quite different in characteristics. The Austin network does not provide opportunities for route switching between the freeways while the anaheim network allows some. Realistic zone-to-zone trip matrices were used in both cases; the Austin simulations used the matrix updated in 1988 and the Anaheim simulations were based on a trip matrix estimated using actual counts.

The simulation results in terms of travel time (as a percentage of the base case where no driver receives ATIS information) are shown in figures 4.1 through 4.3 for the Austin network

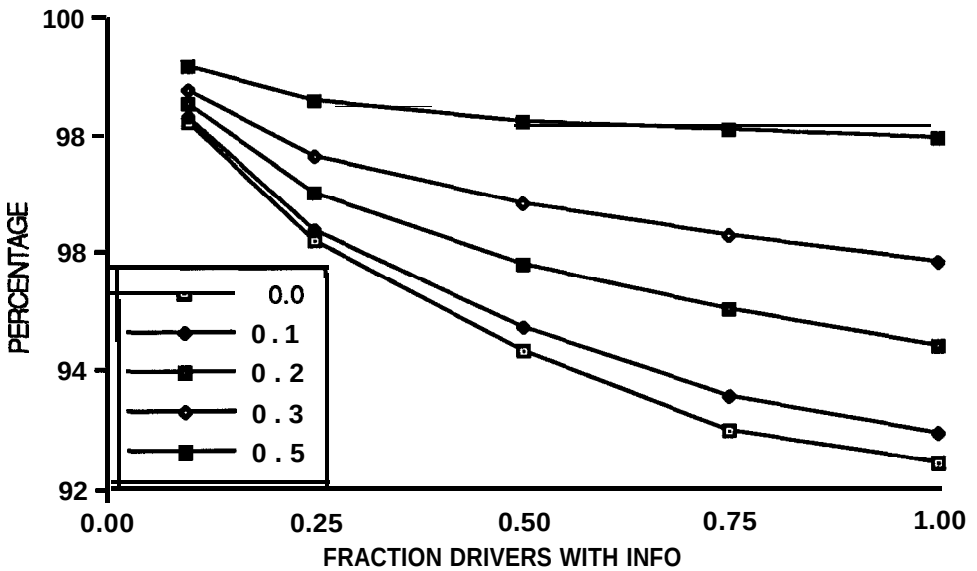

fig 4.1 Average travel time as a percentage of the base case (nobody receiving information) travel time, over all the drivers, for the Austin network. The curves are for different route-switching threshold fraction μ of eqn 11. (μ=0.0 indicates myopic switching).

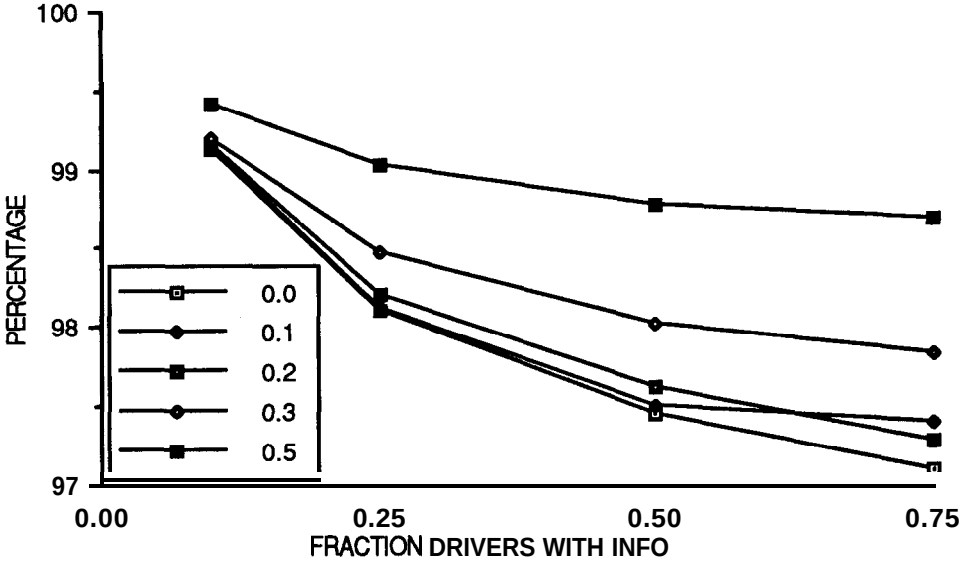

Fig.4.2 Average travel time as a percentage of the base case (nobody receiving information) travel time, for drivers not receiving information, in the Austin network. The curves are for different route-switching threshold fraction μ of eqn 11. (μ=0.0 indicates myopic switching).

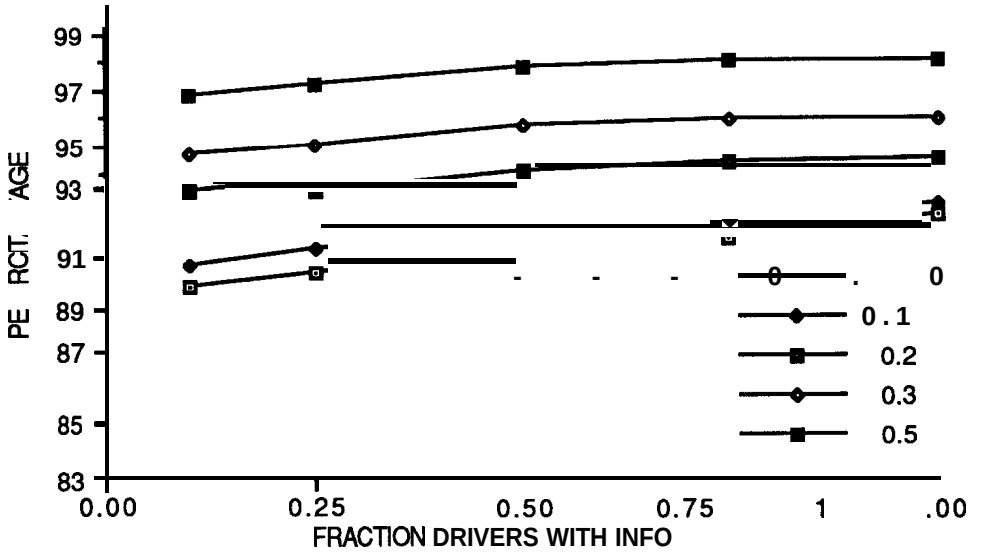

Fig.5. Average travel time as a percentage of the base-case (nobody receiving information) travel time, for drivers receiving information, in the Austin network. The curves are for different route-switching threshold fraction μ of eqn 11. (μ=0.0 indicates myopic switching).

and in figures 4.4 through 4.6 for the Anaheim network. Here the different curves in each graph are for different levels of route-switching propensities (i.e., the average threshold fraction, p, of equation 11). $\mu = 0.0$ is the case of "myopic" switching, and $\mu = 0.5$ is the case of conservative switching. Note also that the y-axis of the graphs are not of the same scale.

These figures show the results separately for the two different driver groups, with and without information, as well as for the total driver population. The differences between the results are primarily due to two factors: (1) in the Anaheim network, the congestion levels were higher than in the Austin network, and (2) for the Austin network, the drivers without information were assumed to be selecting randomly from stored paths (which were the k-shortest paths at the end of the simulation warm-up period), while for the Anaheim network, each driver was pre-assigned an equilibrium path from CONTRAM. The different initial conditions result in entirely different scenarios that are not directly comparable, highlighting the importance of recognizing the underlying assumptions and the network and traffic demand conditions in interpreting the simulation results on ATIS benefits.

As the unequipped drivers between each O-D pair were selecting from a static set of paths in the Austin case and were never changing routes, unrealistic congestion was generated on those paths, with no possibility of dissipation; especially for low market penetration levels. This also means that other paths and thus significant portions of the network remain uncongested. Under these initial conditions, route switching offers good improvement opportunities for the equipped drivers in the Austin network.

In the Anaheim network, the congestion levels are higher, but the initial paths of both equipped and unequipped drivers correspond to a user-equilibrium, providing more efficient utilization of the network than an arbitrary assignment (Mahmassani, et al., 1993). From figure 4.6, we see that in the Anaheim network, the benefits gained by the drivers with information reduce as more drivers receive information, which is due to the higher congestion that develops in the alternative routes that the drivers switch to. On the other hand, this interaction effect is much less pronounced in the Austin network as there is much lower traffic densities in the alternative routes. Other simulations conducted at higher loading levels, resulting in greater network densities in the Austin network, have exhibited similar trends to that of the Anaheim

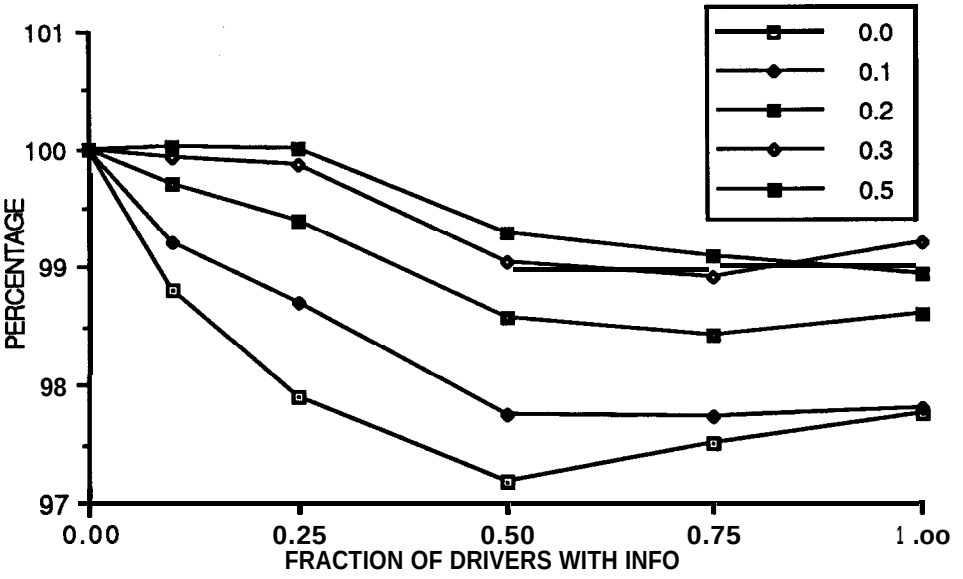

Fig 4.4 Average travel time as a percentage of the base-case (nobody receiving information) travel time, over all the drivers, in Anaheim network. The curves are for different route-switching threshold fraction μ of eqn 11. (μ=0.0 indicates myopic switching).

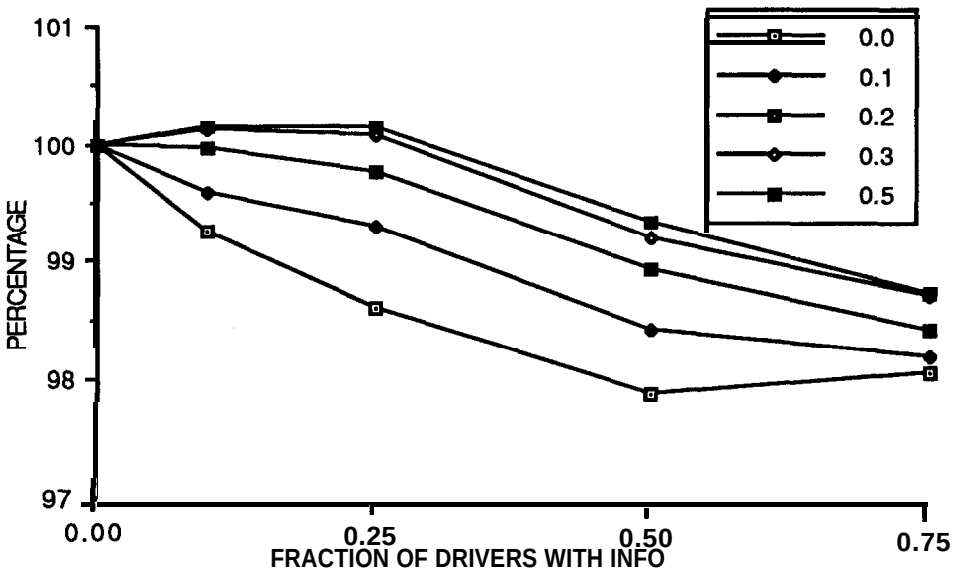

Fig 4.5 Average travel time as a percentage of the base case (nobody receiving information) travel time, for drivers not receiving information, in Anaheim network. The curves are for different route-switching threshold fraction μ of eqn 11. (μ=0.0 indicates myopic switching).

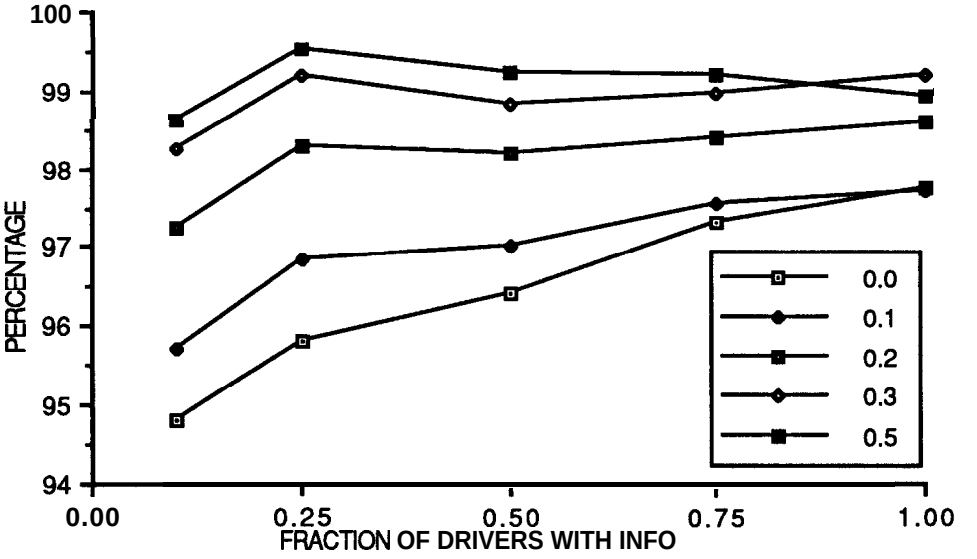

Fig 4.6 Average travel time as a percentage of the base-case (nobody receiving information) travel time, for drivers receiving information, in the Anaheim network. The curves are for different route-switching threshold fraction μ of eqn 11. (μ=0.0 indicates myopic switching).

network (Mahmassani, et al., 1993).

The results from the two modelling scenarios underscore the importance of the initial paths specified for both equipped and unequipped drivers, and brings out the importance of the external path input and storage, such as is possible in DYNASMART. This aspect has not been given careful consideration in some of the earlier modelling efforts on ATIS benefits.

We find (from figures 4.2 and 4.5) that unequipped drivers benefit from information supply, and the benefits increase with the market penetration of information supply. We also find (from figures 4.1 and 4.4) that the system-level average trip time benefits increase with increasing market penetration of information, even though the marginal benefits reduce. In the case of the Anaheim network, we also see the trip times of equipped drivers increasing with higher market penetration, resulting in dramatic decrease in marginal benefits of the information beyond a certain market penetration level, in some cases turning to negative marginal benefits. These results are in line with those reported elsewhere (Mahmassani and Jayakrishnan, 1991) based on simulations using early versions of DYNASMART on small test corridor networks, as well as with those reported by others. It may be noted that the simulations were conducted for different ranges of route-switching propensities (something that the information/guidance controller has no control over) because we lack calibrated driver behavior models, which necessitates studying the robustness of the results under different behavioral assumptions.

It should also be noted that the percentage benefits reported here are for the whole system. The benefits may be much higher for drivers travelling between certain O-D pairs with heavy demand which could have some highly congested paths along with sufficient number of uncongested alternative paths. Recent DYNASMART simulations of ATIS strategies to reduce congestion caused by special events have shown that this is indeed the case (Jayakrishnan, et al., 1993). Several additional evaluation measures can be extracted using DYNASMART, which could be of valuable help in deciding on the implementation plans for ATIS strategies. For instance it is possible to find average travel time per distance for all drivers between each O-D pair, and output the statistics for O-D pairs with the highest value for this measure (say, for the top 5 percent). It is also possible to find relationships between average stopped times and moving times, which could provide measures on network-level traffic quality. Such results are not provided here, and will be published in the near future.

4.3 Comuutation Times

The computation times for DYNASMART have been found to depend predominantly on four factors: 1) The number of vehicles (i.e., the volume level), 2) the number of network links, 3) the k value for the k-shortest paths, and 4) the frequency of the shortest-path finding. Simulation of the Anaheim network (430 nodes, 900 links) for one hour of traffic requires about 2000 CPU seconds on a SUN SPARC-II work station, when k is 3. The time required is about 3000 seconds when the traffic demand level is increased to twice the base value. The same simulations require less than one-third computation time on a CRAY YMP supercomputer. The simulation times for the Austin network (650 nodes, 1700 links) for one hour of simulation with k = 3, requires between 800 and 1200 seconds on the CRAY YMP computer depending on the demand level. These results show that the computation times are certainly acceptable. As the computer hardware continues to become faster, we expect the computation times to decrease further, conceivably by even an order of magnitude in the near future. It is also possible to tailor the program to the specific needs of the user, disabling some of the program modules, so that the computation times can be reduced even further.

Chapter 5
CONCLUSIONS

In this report we have described the DYNASMART simulation program and the graphical user interface developed for the DYNASMART simulation program. DYNASMART is a simulation program for the evaluation of urban traffic networks under Advanced Traveller Information Systems (ATIS) and/or Advanced Traffic Management Systems (ATMS). This simulation program was developed into a comprehensive form during an earlier PATH research project (MOU-39) at the University of California at Irvine, based on a much simpler original version developed at the University of Texas at Austin. DYNASMART simulations can be attempted on reasonably large urban networks, and thus data input and simulation output control are involved tasks, for which a graphical user interface is essential. This was the focus of this research project titled "Enhancement to a Simulation Framework for Analyzing Urban Traffic Networks with ATIS/ATMS", supported by the PATH program of Caltrans.

The user interface that we developed is based on the Motive graphics software and runs on a SUN work station platform. It meets all the conventional criteria for graphical network user interfaces, such as creation and deletion of link, node and zone entities, as well as data input for the nodes (intersections) and links. The default values as well as the duplication features of this front-end to DYNASMART makes network editing and data inputs for large networks much easier.

In addition, the front-end is also used for the display of simulation results during the simulation, which can be used for adjusting some of the simulation data based on the results of the simulation, during the simulation.

The front-end is written in C++ and communicates with the FORTRAN program, DYNASMART, mostly through ASCII data files. This is an advantage, because it retains the separation between the two programs. Both programs can be modified independently of each other, and can be run without the help of each other, as long as the network geometry and other data files are available. This means that, it is possible for DYNASMART to be run on other platforms such as a CRAY supercomputer or an IBM-PC without using the front-end. Alternatively, the data set can be prepared on a SUN work station platform and then ported to

the platform where DYNASMART is implemented. However, if both exist on the SUN platform, additional capabilities such as run-time display, and simulation control (start, stop, pause, restart, etc.) are possible using the front-end. In this case they communicate using Inter Process Communication (IPC) signals from the front-end process to the DYNASMART process.

As the run-time display of simulation results by the front-end is solely based on what is written on to an ASCII file by DYNASMART, such display is clearly possible, even if the two programs are running on different computational platforms, as long as fast file transfer is possible between the systems.

Several DYNASMART simulation were conducted using the front-end, and the benefits from having a user interface was amply demonstrated during this research project. It is expected that this would help the practicing engineers tremendously. We also hope that the DYNASMART system will be calibrated with real traffic data in the near future, so that the complete simulation framework including the front-end can be used for easy and reliable simulations of several ATIS and ATMS options being considered by Caltrans and other Transportation agencies.

REFERENCES

1) Chang, G. L., H. S. Mahmassani, and R. Herman (1985). "A Macroparticle Traffic Simulation Model to Investigate Peak-period Commuter Decision Dynamics," ***Transportation Research Record,*** 1005, pp. 107-120.

2) Fox, B. L. (1978). "Data structures and Computer Science Techniques in Operations Research," ***Operations Research, 26, 5.***

3) Jayakrishnan, R., Rim, J., Cohen, M., Mahmassani, H. S. and Hu, T-Y. (1993). "A Simulation-based Framework for the Analysis of Traffic Networks Operating with Real-time Information," PATH Research Report, UCB-ITS-PRR-93-25, University of California, Irvine.

4) Leboeuf, J. N., Tajima, T. and Dawson, J. M. (1979). "A Magnetohydrodynamic Particle Code for Fluid Simulation of Plasmas," ***Journal of Computational Physics,*** **31, 3,** pp. 379-408.

5) Mahmassani, H. S. and Jayakrishnan, R. (1991). "System Performance and User Response under Real-time Information in a Congested Traffic Corridor," ***Transportation Research-A, 25A, 5,*** pp. 293-307.

6) Mahmassani, H. S. and Peeta, S. (1993). "Network Performance Under System Optimal and User Equilibrium Assignments: Implications for ATIS," presented at the 72nd Annual Meeting of the Transportation Research Board, Washington, D.C.

7) Mahmassani, H. S., Peeta, S., Hu, T-Y., and Rothery, R. A. (1993). "Effect of Real-Time Information on Network Performance under Alternative Dynamic Assignment Rules," ***Proceedings of the Seminar D of the 21st Summer Annnual Meeting of the PTRC,*** pp. 25-

48.

8) Mahmassani, H. S. and Stephan. D. G. (1988). "Experimental Investigation of Route and Departure Time Dynamics of Urban Commuters," ***Transportation Research Record,*** 1203, pp. 69-84.

9) Minieka, E. (1978). ***"Optimization Algorithms for Networks and Graphs,"*** Dekker Inc., New York.

10) Papageorgiou, M., Hadj-Salem, H. and Blosseville, J-M. (1991). "ALINEA, A Local Feedback Control Law for On-ramp Metering," ***Transportation Research Record, No.*** 1320.

11) Sheffi, Y., Mahmassani, H. S., Powell, W. B. (1982). "A Traffic Network Evacuation Model," Transportation Research, **16A, 3,** pp. 209-218.

12) Tarjan, R. E. (1983). ***"Data Structures and Network Algorithms,"*** Monograph, Society for Industrial and Applied Mathematics, Philadelphia, PA.

13) Van Vliet, D. (1978). "Improved Shortest Path Algorithms for Transport Networks," ***Transportation Research, 12,*** pp. 7-20.

14) Ziliaskopoulos, A. and Mahmassani, H. S. (1993). "A Time-dependent Shortest Path Algorithm for Real-time Intelligent Vehicle/Highway System Applications," paper presented at the 72nd annual meeting of the Transportation Research Board, Washington, D.C.