Technical Report on SHRP 2 C10B
Version of DynusT and FAST-TrIPs

PREPUBLICATION DRAFT • NOT EDITED

STRATEGIC HIGHWAY RESEARCH PROGRAM
Accelerating solutions for highway safety, renewal, reliability, and capacity

TRANSPORTATION RESEARCH BOARD
OF THE NATIONAL ACADEMIES
ACKNOWLEDGMENT
This work was sponsored by the Federal Highway Administration in cooperation with the American Association of State Highway and Transportation Officials. It was conducted in the second Strategic Highway Research Program, which is administered by the Transportation Research Board of the National Academies.

NOTICE
The project that is the subject of this document was a part of the second Strategic Highway Research Program, conducted by the Transportation Research Board with the approval of the Governing Board of the National Research Council.

The members of the technical committee selected to monitor this project and to review this document were chosen for their special competencies and with regard for appropriate balance. The document was reviewed by the technical committee and accepted for publication according to procedures established and overseen by the Transportation Research Board and approved by the Governing Board of the National Research Council.

The opinions and conclusions expressed or implied in this document are those of the researchers who performed the research. They are not necessarily those of the second Strategic Highway Research Program, the Transportation Research Board, the National Research Council, or the program sponsors.

The information contained in this document was taken directly from the submission of the authors. This document has not been edited by the Transportation Research Board.

Authors herein are responsible for the authenticity of their materials and for obtaining written permissions from publishers or persons who own the copyright to any previously published or copyrighted material used herein.

The Transportation Research Board of the National Academies, the National Research Council, and the sponsors of the second Strategic Highway Research Program do not endorse products or manufacturers. Trade or manufacturers’ names appear herein solely because they are considered essential to the object of the report.
The National Academy of Sciences is a private, nonprofit, self-perpetuating society of distinguished scholars engaged in scientific and engineering research, dedicated to the furtherance of science and technology and to their use for the general welfare. On the authority of the charter granted to it by Congress in 1863, the Academy has a mandate that requires it to advise the federal government on scientific and technical matters. Dr. Ralph J. Cicerone is president of the National Academy of Sciences.

The National Academy of Engineering was established in 1964, under the charter of the National Academy of Sciences, as a parallel organization of outstanding engineers. It is autonomous in its administration and in the selection of its members, sharing with the National Academy of Sciences the responsibility for advising the federal government. The National Academy of Engineering also sponsors engineering programs aimed at meeting national needs, encourages education and research, and recognizes the superior achievements of engineers. Dr. C. D. (Dan) Mote, Jr., is president of the National Academy of Engineering.

The Institute of Medicine was established in 1970 by the National Academy of Sciences to secure the services of eminent members of appropriate professions in the examination of policy matters pertaining to the health of the public. The Institute acts under the responsibility given to the National Academy of Sciences by its congressional charter to be an adviser to the federal government and, upon its own initiative, to identify issues of medical care, research, and education. Dr. Harvey V. Fineberg is president of the Institute of Medicine.

The National Research Council was organized by the National Academy of Sciences in 1916 to associate the broad community of science and technology with the Academy’s purposes of furthering knowledge and advising the federal government. Functioning in accordance with general policies determined by the Academy, the Council has become the principal operating agency of both the National Academy of Sciences and the National Academy of Engineering in providing services to the government, the public, and the scientific and engineering communities. The Council is administered jointly by both Academies and the Institute of Medicine. Dr. Ralph J. Cicerone and Dr. C.D. (Dan) Mote, Jr., are chair and vice chair, respectively, of the National Research Council.

The Transportation Research Board is one of six major divisions of the National Research Council. The mission of the Transportation Research Board is to provide leadership in transportation innovation and progress through research and information exchange, conducted within a setting that is objective, interdisciplinary, and multimodal. The Board’s varied activities annually engage about 7,000 engineers, scientists, and other transportation researchers and practitioners from the public and private sectors and academia, all of whom contribute their expertise in the public interest. The program is supported by state transportation departments, federal agencies including the component administrations of the U.S. Department of Transportation, and other organizations and individuals interested in the development of transportation. www.TRB.org

www.national-academies.org
SHRP 2 C10B: Partnership to Develop an Integrated Advanced Travel Demand Model with Fine-Grained, Time-Sensitive Networks

Technical Report on SHRP 2 C10B Version of DynusT and FAST-TrIPs

prepared for
Strategic Highway Research Program

prepared by
Cambridge Systematics, Inc.

in association with
Sacramento Regional Council of Governments
University of Arizona
University of Illinois, Chicago
Sonoma Technology, Inc.
Fehr and Peers

August 27, 2013  www.camsys.com
Technical Report on SHRP C10B Version of DynusT and FAST-TrIPs

Technical Report on SHRP 2 C10B Version of DynusT and FAST-TrIPs

prepared for
The Second Strategic Highway Research Program

prepared by
Cambridge Systematics, Inc.
100 Cambridge Park Drive, Suite 400
Cambridge, MA  02140

in association with
Sacramento Regional Council of Governments
University of Arizona
University of Illinois, Chicago
Sonoma Technology, Inc.
Fehr and Peers

August 27, 2013
# Table of Contents

Author Acknowledgements ........................................................................................................ 1

Executive Summary ....................................................................................................................... 2

1.0 Introduction ............................................................................................................................... 1-1

2.0 The DynusT DTA Model ........................................................................................................... 2-1
   2.1 Anisotropic Mesoscopic Simulation (AMS) Model ............................................................... 2-3
   2.2 Gap Function Vehicle-Based Traffic Assignment ............................................................... 2-7

3.0 DynusT Enhancement for the SHRP 2 C10 Integrated Model .................................................. 3-1
   3.1 Transit Simulation ................................................................................................................... 3-1
   3.2 Temporal Domain Decomposition Algorithmic Scheme for Large
       Scale DTA Implementation ................................................................................................. 3-2
       The MIVA Temporal Decomposition Framework ............................................................... 3-4
       Epoch ................................................................................................................................. 3-5
       Projection Period ............................................................................................................... 3-6

4.0 FAST-TrIPs Overview .............................................................................................................. 4-1
   4.1 Fast-TrIPs in the Integrated Model ....................................................................................... 4-1
   4.2 Passenger Assignment .......................................................................................................... 4-5
       Transit Trip-Based Shortest Path (TBSP) ............................................................................ 4-6
   4.3 Passenger Simulation .......................................................................................................... 4-8
       Passenger Movements in Simulation .................................................................................. 4-9
       Intermodal Assignment and Simulation ............................................................................. 4-11

5.0 References ............................................................................................................................... 5-1

6.0 Appendix A: DynusT-FAST-TrIPs Integration Model Execution ........................................... 6-3

7.0 Appendix B: Working with Dataset ......................................................................................... 7-5
   7.1 Create working folder .......................................................................................................... 7-5
   7.2 Rancho Cordova Network Example .................................................................................... 7-8
       Network .............................................................................................................................. 7-8
       Demand .............................................................................................................................. 7-9
       Transit Route ...................................................................................................................... 7-10
List of Figures

Figure 1-1. DynusT-FAST-TrIPs Integration Framework .............................................1-2
Figure 1-2. Interfacing Mechanism in DynusT-FAST-TrIPs Integrated System..........................................................1-4
Figure 2-1. Traffic Simulation, Assignment and Link Volume Estimation Framework in DynusT .................................................................2-2
Figure 2-2. AMS model concept .................................................................................2-4
Figure 2-3. GFV method in conjunction with DynusT .............................................2-9
Figure 3-1. SIR areas with and without bus pullouts ..............................................3-1
Figure 3-2. The SB-DTA algorithmic framework (Chiu, Bottom et al. 2010) ........3-3
Figure 3-3. The MIVA computational scheme implemented within the SB-DTA algorithmic framework (Nava and Chiu 2012) ........3-4
Figure 3-4. The Epoch and Projection Period of the MIVA structure (Nava and Chiu 2012) ..............................................................................3-5
Figure 3-5. The Pseudo code of the MIVA Scheme (Nava and Chiu 2012) ........3-7
Figure 4-1. Fast-TrIPS Algorithmic Structure .......................................................4-2
Figure 4-2. TBSP Algorithm ..................................................................................4-7
Figure 4-3. Transit Passenger Simulation in FAST-TrIPs ..................................4-10
Figure 7-1: Folder Configuration ........................................................................7-6
Figure 7-2. Rancho Codova transportation network in DynusT .......................7-8
Figure 7-3. Rancho Cordova transit network .........................................................7-9
Author Acknowledgements

This work was sponsored by the Federal Highway Administration in cooperation with the American Association of State Highway and Transportation Officials. It was conducted in the second Strategic Highway Research Program (SHRP 2), which is administered by the Transportation Research Board of the National Academies. This project was managed by Stephen J. Andrle, Chief Program Officer for SHRP 2 Capacity.

The primary authors of this report are Drs. Yi-Chang Chiu, Alireza Khani, Hyunsoo Noh, and Brenda Bustillos of the University of Arizona and Dr. Mark Hickman, currently of the University of Queensland, who performed this work while with the University of Arizona. Key contributions were made by Michalis Xyntarakis of Cambridge Systematics, Inc. The Principal Investigator for Project SHRP 2 C10B is Thomas Rossi of Cambridge Systematics.
Executive Summary

The Second Strategic Highway Research Program (SHRP 2) Project C10B, Partnership to Develop an Integrated, Advanced Travel Demand Model and a Fine-Grained, Time-Sensitive Network, is an important step in the evolution of travel modeling from an aggregate, trip-based approach to a completely dynamic, disaggregate methodology. In this project, an existing disaggregate activity based model was integrated with an existing traffic simulation model to create a new, completely disaggregate model.

This report describes the DynusT (Dynamic Urban Systems for Transportation) dynamic traffic assignment (DTA) model and the FAST-TrIPs (Flexible Assignment and Simulation Tool for Transit and Intermodal Passengers) dynamic transit assignment package and how these two model systems interact with each other and with the DaySim activity-based model (ABM) implemented in Sacramento, California to comprise an integrated ABM and DTA model. This work was performed as part of the Strategic Highway Research Program (SHRP 2) Project C10, “Partnership to Develop an Integrated Advanced Travel Demand Model with Fine-Grained, Time-Sensitive Networks.” DynusT is a simulation-based DTA model capable of performing daily regional simulations of large metropolitan areas that involve many millions of trips, a feature necessary for DTA and ABM integration. This document describes in detail the traffic simulation and assignment capabilities of DynusT that captures capacity constraints, congestion, and queue propagation for various types of vehicles including transit vehicles, and allows the generation of time-dependent level-of-service measures that are closer to traffic theory. It also describes in detail the methodology that is used internally by DynusT to determine the time-dependent least-cost path route for each driver, a concept that is described as Dynamic User Equilibrium. FAST-TrIPs is a region-wide dynamic transit assignment model that determines an individual-specific transit route for each transit traveler in the system taking into account published transit schedules and transit vehicle run times that are congestion responsive and are provided by the traffic simulation component of DynusT. FAST-TrIPs deals with both transit-only and park-and-ride trips and is able to maintain multiple constraints associated with activity time-windows and the choice of modes in multimodal travel tours. This report describes how DynusT and FAST-TrIPs interoperate with each other to provide a model system in which the highway and transit assignments influence each other and are based on the same set of LOS variables.
1.0 Introduction

The Strategic Highway Research Program (SHRP 2) Project C10B, Partnership to Develop an Integrated, Advanced Travel Demand Model and a Fine-Grained, Time-Sensitive Network, is an important step in the evolution of travel modeling from an aggregate, trip-based approach to a completely dynamic, disaggregate methodology. In this project, an existing disaggregate activity based model was integrated with an existing traffic simulation model to create a new, completely disaggregate model. Both models were implemented using open source software.

At the same time that travel demand models have been evolving, traffic simulation models, which simulate the movements of vehicles through a highway network, have become more sophisticated due to improvements in computing. The product of SHRP 2 C10B is an integrated model that simulates individuals’ activity patterns and travel and their vehicle and transit trips as the move on a real time basis through the transportation system. It produces a true regional simulation of the travel within a region, for the first time using individually simulated travel patterns as input rather than aggregate trip tables to which temporal and spatial distributions have been applied to create synthetic patterns. A unique feature of this model is the simulation of transit vehicles as well as individual person tours using transit.

The development of the new integrated model has been implemented for the entire Sacramento, California region. The integrated model components include SACSIM, the regional travel model maintained by the Sacramento Area Council of Governments (SACOG), the regional MPO, and DynusT (Dynamic Urban Systems for Transportation), a mesoscopic traffic simulation model developed by the University of Arizona. SACSIM includes and activity based demand model, DaySim. The transit simulation is performed by FAST-TrIPs (Flexible Assignment and Simulation Tool for Transit and Intermodal Passengers), also developed by the University of Arizona. The integrated model also includes the ability to run MOVES, the air quality analysis program developed by the U.S. Environmental Protection Agency.

While the C10B integrated model produces reasonable results for regional travel patterns and behavior, the true value of the model is its ability to provide analysis results that demonstrate sensitivity to policy variables more accurately than models that use aggregate demand or assignment procedures. This sensitivity was tested through a series of policy and project tests conducted by SACOG, using the new integrated model and the existing SACSIM model with aggregate assignment.

The SHRP 2 C10B project has been documented in a series of four reports:

• SHRP 2 C10B Summary Report
This report, the third in the series, describes the theoretical background and methodology for the integration of DynusT, a mesoscopic dynamic traffic assignment (DTA) model, and FAST-TrIPs, a public transit passenger assignment and simulation model. The DynusT and Fast-TrIPs integrated system is designed to be a “loosely coupled” system. This design allows the DynusT and Fast-TrIPs teams to develop and test components in parallel following separate software development cycles. In the SHRP 2 C10B project, the travel demand data are simulated by the DaySim activity-based model (ABM) implemented in Sacramento, California.

The overall FAST-TrIPs model structure is illustrated in Figure 1-1, which depicts the software components as well as the files needed for the DynusT and FAST-TrIPs communication. First, data from the General Transit Feed Specification (GTFS), originally known as the Google Transit Feed Specification, are processed and converted to transit route layouts, stops, and route schedule input files used by FAST-TrIPs. The FAST-TrIPs assignment procedure accepts demand data from DaySim and roadway link travel times from DynusT and assigns travelers to transit paths and transit vehicles. The DynusT DTA model simulates the DaySim trip and tour rosters and the transit vehicles from the FAST-TrIPs component. The network-wide LOS measures that result from the DynusT assignment become inputs to the FAST-TrIPs model in an iterative process until convergence is reached.

Figure 1-1. DynusT-FAST-TrIPs Integration Framework
As shown in Figure 1-2, DynusT and FAST-TrIPS maintain a “loose coupling” integration architecture. This integration architecture is of particular advantage in reducing development risk. This method allows FAST-TrIPS, a completely new development from C10B project, to be prototyped, developed and tested as a stand-alone module, allowing robust and controlled testing and debugging along the development process. As a result, a communications between DynusT and fAST-TrIPS are conducted via flat text files.

In addition to the GTFS files, other inputs to the entire system include the DaySim output files containing trip and tour rosters (sout.dbf) and non-ABM trips such as airport or exogenous OD (exogenous.dbf and air.dbf, ixxi.dbf, etc.) The ft_BST script processes sout.dbf and produces demand_auto.dat. Demand_auto.dat contains the auto trip roster and is further processed into vehicle.dat and path.dat as the main input files for DynusT. The transit route and trip schedule information is organized and prepared as transitRouteSchedule.dat file from GTFS file format.

DynusT takes both the airport demand (processed into demand.dat) and external freight traffic (processed into demand_truck.dat) as the trip roster (vehicle.dat and path.dat) and performs simulation and dynamic assignment until reaching convergence in auto demand.

Once DynusT reaches convergence transit vehicle trajectories (AltTime_transit.dat and AltTime_Intermodal.dat), combining demand_transit.dat are processed and fed into FAST-TrIPS for transit assignment. Essentially AltTime_transit.dat and AltTime_intermodal.dat describe the transit vehicle movements, which can be considered as the supply-side information, whereas demand_transit.dat contains the demand-side information. The transit assignment procedure essentially determines the “assignment” of passengers to various transit route and transit stop choices.

The outcome of FAST-TrIPS procedure determines how many passengers will get on and off at each stop, consequently determines the transit vehicle dwell time at each stop (TransitDwellTime.dat). If FAST-TrIPS is deemed not converged, DynusT is re-run again with the updated transit dwell time as the important input to DynusT.

On the other hand, should the FAST-TrIPS convergence criterion is met, the entire DynusT/FAST-TrIPS integrated procedure is considered completely converged and the system will output necessary skim information in terms of skim.dat, skimXXX.dat. Such information will be fed into DaySim ABM.
Figure 1-2. Interfacing Mechanism in DynusT-FAST-TrIPs Integrated System

This report is structured as follows: a detailed description of the DynusT model is presented in Chapter 2. Chapter 3 describes the enhancements that were made to DynusT to work with the DaySim ABM model. Section 4 provides an in-depth exposition of the theoretical models used in FAST-TrIPs.
2.0 The DynusT DTA Model

The dynamic traffic simulation and assignment model DynusT (Dynamic Urban Systems in Transportation) is a model system that is designed and implemented to perform simulation-based dynamic traffic assignment (DTA) and associated analysis. Due to its unique algorithmic structure and software implementation, it is capable of performing DTA on regional-level networks over a long simulation period. This makes DynusT particularly well-suited for regional level modeling such as regional transportation planning, corridor studies, integration with activity-based models, and mass evacuation modeling. The purpose of this chapter is to present an overview of the theoretical and algorithmic innovations in DynusT.

As shown in Figure 2-1, DynusT consists of two main modules, traffic simulation and traffic assignment. Vehicles are created and loaded into the network based on their respective origins and follow a specific route to their intended destinations. The large-scale simulation of network-wide traffic is accomplished through the mesoscopic simulation approach that omits inter-vehicle car-following details while maintaining realistic macroscopic traffic properties (i.e. speed, density and flow). More specifically, the traffic simulation is based on the Anisotropic Mesoscopic Simulation (AMS) (Chiu, Zhou et al. 2010) technique that calculates a vehicle’s speed from the traffic conditions ahead of the vehicle. Specifically, at each simulation interval, a vehicle’s speed is determined by a speed-density curve, the density being the number of vehicles per mile per lane within a limited forward distance designated as SIR (speed-influencing region).
Figure 2-1. Traffic Simulation, Assignment and Link Volume Estimation Framework in DynusT

What is referred to as the traffic assignment module of DynusT consists of two algorithmic components: a time-dependent shortest-path (TDSP) algorithm and a time-dependent traffic assignment, or routing. The TDSP algorithm determines the time-dependent shortest path for each origin, destination, and departure time while the traffic assignment component selects a route for each driver following some heuristic rules that are proven to lead to approximate user equilibrium, a condition in which each driver has selected the least cost path available to him.

In DynusT, the assignment algorithm maintains the balance of computational efficiency and solution algorithm quality. Innovations in computational efficiency allow DynusT to perform 24-hour assignment, which is a requirement for ABM and DTA integration.

After shortest paths have been calculated and a route choice has been made, all the vehicles are simulated. DynusT uses the time gap between a vehicle’s simulated travel time and the vehicle’s available shortest path time to assess the level of convergence. If the average time gap for all the vehicles in the simulation is small enough (usually 1 to 2%) DynusT terminates and outputs network-wide
LOS measures otherwise it continues iterating between its two models until convergence is achieved.

The rest of this section is structured as follows. Section 2.1 presents the AMS model that powers the traffic simulation component of DynusT. Section 2.2 discusses the traffic assignment algorithm that is deployed to achieve convergence.

2.1 **ANISOTROPIC MESOSCOPIC SIMULATION (AMS) MODEL**

The AMS model was based on a research published in (Chiu, Zhou et al. 2010). The AMS model is developed based on two empirical traffic rules: (1) at any time, a vehicle’s prevailing speed is influenced only by the vehicles in front of it, including those that are in the adjacent lanes; (2) the influence of downstream traffic decreases with increasing distance. These two characteristics define the “anisotropic” property of the traffic flow and provide the guiding principle for AMS model design. Based on the above, for any vehicle \( i \), only the downstream vehicles within a certain distance influence vehicle \( i \)’s speed. This is a similar concept to a stimulus-response type of car-following model, with the distinction that in AMS, the stimulus of a vehicle’s speed response is represented in a macroscopic manner instead of using inter-vehicle distance or speed as in microscopic models.

For modeling purposes, the **Speed Influencing Region** for vehicle \( i \) \((SIR)\) is defined as vehicle \( i \)’s immediate downstream roadway section in which the stimulus is significant enough to influence vehicle \( i \)’s speed response. This concept is further depicted in Figure 2-2. AMS model concept, in which a multi-lane homogeneous roadway segment is considered. The **Speed Influencing Region** \((SIR)\) for vehicle \( i \) is defined as the area (including the lane in which vehicles reside and all the adjacent lanes) in front of vehicle \( i \), where the traffic condition (represented by the density) affects vehicle \( i \)’s speed response. At each simulation clock tick, vehicle \( i \)’s speed is influenced by the density in the SIR. The upstream traffic and downstream traffic outside the SIR does not influence vehicle \( i \). The traffic density in \( SIR_i \), denoted as \( k_i \), is calculated as the number of vehicles present in \( SIR_i \) divided by the total lane-miles of the \( SIR_i \). As such, the unit of \( k_i \) becomes the number of vehicles per mile per lane.
At the beginning of a simulation interval $t$, the prevailing speed of vehicle $i$ during the simulation interval $t$ is determined by Equation (1), which is a non-increasing speed-density relationship function with the following properties (a) density does not influence speed up to a certain density threshold and (b) for density greater than the threshold value increases in density result in decreases in speed. The maximum density in DynusT is the “bumper-to-bumper” density observed in a long, standing-still queue.

The algorithmic steps of an AMS model during simulation are as follows: at each clock tick $t$ (the beginning of a simulation interval), each vehicle’s speed is evaluated based on its SIR density, which is obtained from the previous clock tick $t-1$ through the speed-density ($v$-$k$) relationship of Equation 2-1. The SIR density is calculated based on Equation 2-2 or Equation 2-3, depending on whether the SIR spans over a roadway segment with a different capacity. If the SIR spans a heterogeneous highway section, Equation 2-2 applies; otherwise, the
relationship is simplified to Equation 2-3. Vehicle $i$’s traveling distance at the end of the current simulation interval is obtained by multiplying the prevailing speed with the duration of the simulation interval.

$$v_i^t = \psi(k_{i}^{t-1})$$  
**Equation 2-1**

$$k_{i}^{t-1} = \min\left[k_{\text{queue}}^{*}, \frac{N_{i}^{t-1}}{m_{i}^{t-1} + n(l-x_{i}^{t-1})}\right]$$  
**Equation 2-2**

$$k_{i}^{t-1} = \min\left[k_{\text{queue}}^{*}, \frac{N_{i}^{t-1}}{n l}\right]$$  
**Equation 2-3**

$i =$ Subscript denoting a vehicle. The index $i$ decreases with vehicles traveling in the same direction on the same link

$t =$ Superscript denoting a simulation interval

$n =$ Number of lanes downstream of lane add/drop

$m =$ Number of lanes upstream of lane add/drop

$l =$ SIR length

$v_i^t =$ Prevailing speed of vehicle $i$ during simulation interval $t$

$x_{i}^{t-1} =$ Distance between vehicle $i$ and lane-drop (open) at clock tick $t-1$

$k_{i}^{t-1} =$ Density of the SIR for vehicle $i$

$N_{i}^{t-1} =$ Number of vehicles present in SIR, excluding vehicle $i$

$v_f =$ Free-flow speed in the speed-density relationship

$\psi: k \rightarrow v =$ Non-increasing speed-density function specifying the $v$-$k$ relationship, where $\psi(0) = v_f$ and $\psi(k_{\text{queue}}) = 0$

$k_{\text{queue}} =$ Queue density, $\psi(k_{\text{queue}}) = 0$

During the AMS simulation, each vehicle maintains its own desired speed and SIR. Individual vehicles’ traveling distances are therefore likely to differ, even though they are on the same link. This feature is different from certain previous models (Jayakrishnan, Mahmassani et al. 1994; Balakrishna, Koutsopoulos et al. 2005), in which all moving vehicles on the same link travel at the same speed. This characterizes the AMS model as a vehicle-based mesoscopic model having a greater degree of resemblance with car-following-based microscopic models.
The major difference between AMS and car following models is that in AMS, a vehicle’s speed adjustment at each simulation time interval is governed by the SIR density, which is a macroscopic measure of all the vehicles present in the SIR region, instead of an inter-vehicle measure between the target vehicle and the leading vehicle(s).

Since the SIR moves with each vehicle during simulation, it can be anticipated that in the AMS model, the vehicle advancing mechanism is generally independent of the mathematical representation of the roadway network (i.e. size/length of cell/segment/link). AMS simulation results generally remain stable regardless link lengths are shorter than the simulation interval multiplied by the speed.

AMS handles queue formation/discharge in a natural and straightforward manner. When the maximum value of $k_{queue}$ is reached vehicles get zero speed $v = \phi(k_{queue}) = 0$. When density in the SIR region decreases affected vehicles speed up. This mechanism allows for clear representations of substantial or transient queue formation or discharge. When a free-moving vehicle approaches the end of a queue, its speed gradually approaches the speed of the vehicles at queue’s tail.

Equation 2-1 is further extended to simulate traffic flow in uninterrupted flow facilities under various configurations, such as homogeneous highways, non-homogeneous highways and temporary blockage, by incorporating different SIR and density $k_i$ calculations. In the case of homogeneous highways, $k_i$ is calculated as the number of vehicles present in the SIR divided by the total lane-miles of the SIR (Equation 2). When lane drops or lane additions occur within the SIR, the total lane-mile of SIR is the sum of lane-miles of separate sections, as shown in Equation (3) and depicted in Figure 2-2. AMS model concept (where $m = \text{the number of lanes at the beginning of the section and } n = \text{the number of lanes at the end of the section}$).

In the case of a lane drop or a point bottleneck ($n < m$), the SIR density of a vehicle gradually increases (and hence speed reduces) as it approaches the bottleneck. When $n = 0$, a complete blockage occurs; this can be applied to either the point blockage or red-light signal indication. On the other hand, in the case of discharging from a bottleneck, as a vehicle approaching the open-up of the bottleneck, the density reduces and speed increases gradually.

In this project, the speed-density function takes the following two Greenshield types of function forms.
2.2 GAP FUNCTION VEHICLE-BASED TRAFFIC ASSIGNMENT

In the assignment, the proposed gap function vehicle-based (GFV) algorithm adopts a “gradient projection” concept, where path flow updates are comprised of both a gradient and a step size. The algorithm also takes advantage of the vehicle-based simulation, allowing reassigned selected individual vehicles with better paths to improve their travel times. The gradient projection approach is common in constrained nonlinear programming and has been applied in many classical static and dynamic traffic assignment works (Dafermos and Sparrow 1969; Florian and Nguyen 1974; Sheffi 1985; Smith 1993). The “route-swapping” heuristics developed in more recent years have been found to be related to the same concept, except that in the route-swapping heuristics the step size is usually a pre-determined “swapping rate” and the direction is linearly proportional to the travel time difference between the current vehicle travel time and the shortest path travel time (Smith and Wisten 1995; Huang and Lam 2002; Szeto and Lo 2006). In a more recent study the swapping rate was proposed to be the ratio of the difference of the individual path travel time to the shortest path travel time (Lu 2007; Lu, Mahmassani et al. 2008).

In the GFV method used in the SHRP 2 C10 model, the step size is in relation to the relative gap value calculated for all the paths between each origin-destination-departure time \((i, j, \tau)\) triplet at iteration \(l\). Similar in concept to prior studies, the GFV method leads to a smaller step size with a smaller relative gap value. The gradient determines the search direction, where the “search direction” means those paths to be updated with more or less vehicles. For each \((i, j, \tau)\) triplet, paths are sorted according to the average travel time, and vehicles traveling on each path are also sorted according to decreasing experienced travel time. Note that the assignment interval is normally much longer than the simulation interval. Therefore, vehicles departing within the same assignment

\[
v^{cal} = \begin{cases} v_f, & k \leq k_b \\ v_f \left[1 - \left(\frac{k - k_b}{k_q - k_b}\right)^\beta\right]^\alpha, & k_b \leq k \leq k_q \end{cases}
\]

Equation 2-4

\[
v^{cal} = \begin{cases} v_f, & k \leq k_b \\ v_f \left[1 - \left(\frac{k - k_b}{k_q - k_b}\right)^\alpha\right], & k_b \leq k \leq k_q \end{cases}
\]

Equation 2-5
interval, while subject to the same path set, would experience different travel times due to their different departure times during the simulation interval.

Furthermore, at each iteration, once the assignment is completed, the path set $K^{i+1}(i, j, \tau)$ \forall all $(i, j, \tau)$ are released from memory. At the beginning of each iteration, the path set is re-constructed by scanning through each vehicle and assigned path. This may cause a slight increase in computation, but it eliminates the need to retain the path set $K^{i+1}(i, j, \tau)$ \forall all $(i, j, \tau)$ when such information are not used (e.g. in simulation); thus, reducing the peak memory needed during the entire simulation-assignment procedure. It should also be noted that this strategy is likely to deplete high-travel-time paths of vehicles. As a result, such paths will not appear in the next iteration path set.

Different from methods that explicitly solve for the optimal flow distribution among paths within $K^i(i, j, \tau)$ (Chang and Ziliaskopoulos 2004; Lu, Mahmassani et al. 2008), the determination of the step size and gradient are based on a simple joint approximation of the descent direction and step size. This strategy is adopted after a careful consideration of the trade-offs between the solution quality and computational tractability. The methods that explicitly solve for the optimal distribution given $K^i(i, j, \tau)$ require significantly more computation time that may make the algorithm computationally intractable unless a parallel computing scheme is utilized. Overall, the GFV method used in SHRP 2 C10 exhibits satisfactory convergence quality for a reasonable computation time.

A schematic representation of the GFV algorithm as part of DynusT is illustrated in Figure 2-3. The iterative simulation-assignment is initialized with the primary inputs of network loading: demand patterns, time-dependent origin-destination (OD) matrices and initial path assignments. As the AMS model simulates vehicles within the network, evaluations of time-varying link densities, link flows, travel times and speeds are made (Chiu, Zhou et al. 2010). After the initial network loading, the interplay between GFV and the AMS simulation continues until the convergence criterion is met. The convergence criterion is the gap function value, which is further discussed below.
The GFV procedure starts by sorting vehicles for all \( K^l(i,j,\tau) \) based on vehicles’ experienced travel times. Note that vehicles are loaded into the network on links; therefore, the origin node \( i \) for a vehicle refers to the downstream node of the link on which the vehicle is loaded.

In the GFV algorithm, the step size is in relation to the Relative Gap (\( RG \)) value, calculated \( \forall k \in K^l(i,j,\tau) \). Furthermore, the relative gap \( RG_k \) for path \( k \) defined in Equation 2-6 indicates that \( q_v \) is the experienced travel time of vehicle \( v \) (from the downstream node of origin link \( i \) to destination node \( j \)) and \( u_{i,j}^{\tau,\theta} \) is the
calculated time-dependent shortest path travel time for \((i,j,\tau)\) at iteration \(l\) solved by the TDSP algorithm. \(RG_k\) measures the travel time deviation of path \(k\) in comparison with the shortest path.

\[
RG_k = \frac{\sum_{v \in V^l(i,j,\tau,k)} q_v - r_{i,j,k} \cdot u_{i,j}^l}{r_{i,j,k} \cdot u_{i,j}^l}, \forall k \in K^l(i,j,\tau) \tag{Equation 2-6}
\]

The stopping criterion follows Equation 2-7.

\[
\overline{RG} \leq RG^0 \tag{Equation 2-7}
\]

where \(RG^0\) is the user-specified threshold and \(\overline{RG}\) follows Equation 2-8.

\[
\overline{RG} = \frac{\sum_{i,j,\tau,k} \sum_{v \in V^l(i,j,\tau,k)} q_v - \sum_{i,j,\tau} (r_{i,j} \cdot u_{i,j}^l)}{\sum_{i,j,\tau} (r_{i,j} \cdot u_{i,j}^l)} \tag{Equation 2-8}
\]

Next, the time-dependent shortest path is solved. At each iteration \(l\), the flows for each \(k \in K^l(i,j,\tau)\) to be shifted at this iteration is the step size \(\alpha_{i,j}^l\) times the total flow \(r_{i,j}^l\) between the \((i,j,\tau)\) triplet. \(\alpha_{i,j}^l\) is defined as the minimal of two candidate step sizes as shown in Equation 2-9. \(\alpha'\) is the \(RG\)-based step size, calculated based on Equation 2-10; \(\alpha^0\) is the maximum step size. The step size is determined by Equation 2-7 as the \(RG\)-based step size is calculated as average \(RG\) value for all path \(k \in K^l(i,j,\tau)\):

\[
\alpha_{i,j}^l = \min\{\alpha^0, \alpha'\} \tag{Equation 2-9}
\]

\[
\alpha' = \left\{ \frac{\sum_k RG_k}{|K^l(i,j,\tau)|} \right\} \tag{Equation 2-10}
\]

\(|K^l(i,j,\tau)|\) is the cardinality of the set of non-zero flow paths between criterion \((i,j,\tau)\) for iteration \(l\). Paths in \(K^l(i,j,\tau)\) are ordered with decreasing travel time. Note \(K^l(i,j,\tau)\) includes the TDSP solved at the current iteration.

Based on Equation 2-6 through Equation 2-10, one should expect that since the algorithm starts from an all-or-nothing (AON) assignment, \(|K^l(i,j,\tau)|\) is small and \(RG_k\) may be large; therefore, the step size is initially capped at \(\alpha^0\). As iterations increase, \(|K^l(i,j,\tau)|\) also increases while \(RG_k\) decreases. With the step size capped by \(\alpha^0\) at initial iterations, as iterations increase the step size will be then handled by \(\alpha'\) as the size of path sets \(K^l(i,j,\tau)\) would eventually stabilize.
The value of $\alpha'$ generally continues to decrease as the Time-Dependent User Equilibrium (TDUE) solution is iteratively improved.

If the assignment starts from an initial simulation in which vehicles are being loaded following TDUE paths solved from a prior TDUE run, the step size in the initial iteration is likely to be capped by $\alpha'$. This reduces excess fluctuation of the initial assignment away from the TDUE condition since the initial solution is already close to the TDUE condition. Only those $K^l(i, j, \tau)$ exhibiting large gaps are to be shifted with a larger flow amount. This observation implies a relatively stable and quick convergence for a “warm-start” assignment.

Next, the descent direction $d^l_k$ updates the direction of flow adjustment at iteration $l$ using what are considered increasing-flow and decreasing-flow path subsets. $K^l(i, j, \tau)^+$ defines the path subset that will have the increased flow after assignment while $K^l(i, j, \tau)^-$ is the path subset with decreased flow, where:

$$K^l(i, j, \tau)^+ \cup K^l(i, j, \tau)^- = K^l(i, j, \tau)$$

Equation 2-11

After determining the step size, the amount of the total shifted flow is $\alpha^{r,l}_{i,j} \cdot r^r_{i,j}$ by definition. It is important to note that in $K^l(i, j, \tau)$, paths are always ranked in the order of increasing path average travel time $\tau^l_{i,j,k}$, which is:

$$\tau^l_{i,j,k} = \frac{\sum_{v \in V^l(i,j,\tau,k)} q_v}{|V^l(i,j,\tau,k)|}, \forall k \in K^l(i,j,\tau)$$

Equation 2-12

and

$$\tau^l_{i,j,1} \leq \tau^l_{i,j,2} \leq \cdots \leq \tau^l_{i,j,K^l(i,j,\tau)}$$

Equation 2-13

The decreasing-flow path set $K^l(i, j, \tau)^-$, as defined in Equation 2-14 determined by scanning paths $k = 1, 2, ..., \hat{k}$ until the condition defined in Equation 2-15 is met. $\hat{k}$ is the cutoff path in which vehicles in any path $k \in K^l(i, j, \tau)$ whose $\tau^l_{i,j,k} \geq \tau^l_{i,j,k}$ will be considered for reassignment.

$$K^l(i, j, \tau)^- = \left\{ k = \hat{k}, ..., |K^l(i, j, \tau)| \left| \sum_{k=1}^{\hat{k}-1} r^l_{i,j,k} < \alpha^{r,l}_{i,j} \cdot r^r_{i,j} \leq \sum_{k=1}^{\hat{k}} r^r_{i,j,k} \right\}$$

Equation 2-14

Equation 2-14 only determines the decreasing-flow path set; however, because the GFV algorithm is vehicle-based, the step size $\alpha^{r,l}_{i,j}$ specifies a certain number of vehicles that would be reassigned to the increasing-flow path set. For that
reason, all vehicles on path \( k \in K^l(i,j,\tau)^{-}\backslash \hat{k} \) will be assigned to the increasing-flow path set, but only part of those on the cut-off path \( \hat{k} \) will be assigned. \( \hat{\vartheta} \) is defined as the cutoff vehicle on path \( \hat{k} \) such that any vehicle beneath \( \hat{\vartheta} \) is not considered for reassignment.

\[
\hat{\vartheta} = \sum_{k=1}^{\left| k^l(i,j)^- \right|} r_{i,j,k}^{\tau} - \alpha_{i,j}^{\tau} \cdot r_{i,j}^{\tau} \tag{Equation 2-15}
\]

Those vehicles belonging to \( k \in K^l(i,j,\tau)^- \) will be reassigned with one of the paths in the set \( K^l(i,j,\tau)^+ \), which may also include the latest solved TDSP. The re-distribution scheme is depicted in Equation 2-16:

\[
d_k^l = \begin{cases} 
\frac{(RG_{\hat{k}} - RG_k)^\theta}{\sum_{k \in K(i,j,\tau)^+} (RG_{\hat{k}} - RG_k)^\theta} & , \forall k \in K^l(i,j,\tau)^+ \\
\frac{-r_{i,j,k}^{\tau}}{\alpha_{i,j}^{\tau} \cdot r_{i,j}^{\tau}} & , \forall k \in K^l(i,j,\tau)^{-}\backslash \hat{k} \\
\frac{-\hat{\vartheta}}{\alpha_{i,j}^{\tau} \cdot r_{i,j}^{\tau}} & , \hat{k} \in K^l(i,j,\tau)^{-} 
\end{cases} \tag{Equation 2-16}
\]

where \( RG_{\hat{k}} \) is the relative gap value for cutoff path \( \hat{k} \).

For those paths \( k \in K^l(i,j,\tau)^+ \), the number of vehicles is increased by the amount:

\[
\alpha_{i,j}^{\tau} \cdot r_{i,j}^{\tau} \cdot \frac{(RG_{\hat{k}} - RG_k)^\theta}{\sum_{k \in K(i,j,\tau)^+} (RG_{\hat{k}} - RG_k)^\theta} \tag{Equation 2-17}
\]

For a vehicle to be assigned to a path \( k \in K^l(i,j,\tau)^+ \), the probability can be expressed as:

\[
P(k) = \frac{(RG_{\hat{k}} - RG_k)^\theta}{\sum_{k \in K(i,j,\tau)^+} (RG_{\hat{k}} - RG_k)^\theta}
\]

For those paths \( k \in K^l(i,j,\tau)^{-}\backslash \hat{k} \) the change in the number of vehicles is:
For $k = \tilde{k}$, the decrease amount is:

$$-\alpha_{i,j}^{l} \cdot r_{i,j}^{\tau}, \quad \frac{r_{i,j,k}^{l}}{\alpha_{i,j}^{l} \cdot r_{i,j}^{\tau}}$$

Note that $\theta$ is a scaling factor, which can be kept as a pre-defined constant or can be an increasing function of iteration number. The larger $\theta$ is, the more aggressive the assigned flow to the first best paths will be. The value used in C10B project is 2.0.

Lastly, all flow determined to be reassigned are applied to all paths $k \in K^{l}(i,j,\tau)^{+}$ following a nonlinear proportional scheme:

$$r_{i,j,k}^{l+1} = r_{i,j,k}^{l} + \alpha_{i,j}^{l} \cdot r_{i,j}^{\tau} \cdot d_{k}^{l}, \forall k \in K^{l}(i,j,\tau)$$
3.0 DynusT Enhancement for the SHRP 2 C10 Integrated Model

3.1 Transit Simulation

Vehicle simulation under the presence of transit vehicles needs to properly differentiate the situation with or without bus pullouts. As illustrated in Figure 3-1. SIR areas with and without bus pullouts when a bus pullout is present and a transit vehicle resides in the pullout, passersby vehicles' SIR area remains unchanged. On the other hand, without the pullout the stopped transit vehicle typically blocks one traffic lane, creating a temporal blockage to the following traffic stream. The departure from each stop involves different rules for frequency or schedule-based transit. The main difference is that for schedule-based transit operation, a transit vehicle needs to be held until the scheduled departure time if the transit vehicle is still ahead of schedule after boarding and alighting. Such vehicle holding is unnecessary in frequency-based operation.

![Figure 3-1. SIR areas with and without bus pullouts](image)

As previously mentioned, demand is generated from a tour-based travel demand model. Before this information can be utilized for traffic simulation and assignment purposes it must be manipulated to meet the traffic simulation and assignment model’s specific network and demand inputs. In terms of demand, DynusT demand inputs take two forms: 1) the typical Origin-Destination (OD)
table for specified time periods and 2) vehicle and path files. In general, the exogenous travel components (truck, external, and airport vehicle trips) yield OD demand files given diurnal factors while tour/trip records yield vehicle and path demand files.

Generating DynusT’s vehicle demand file is a more involved process since it requires detailed trip information as opposed to OD demand files simply requiring OD and diurnal factors. Examples of this mandatory information include household ID, traveler/person ID, tour/trip ID, origin/destination parcels/points, origin/destination zones, mode choice, travel time, value of time, and arrival/departure time. The purpose of this information is to represent a trip as realistically as possible within DynusT’s node-link based network and context. Examples of this “realistic” representation not only entail correct zone vehicle generation or destinations but most importantly also ensure that a specific person’s trip reaches its destination before his or her next trip (tour) is generated. This instance is usually prevalent in networks with congestion or disruption or where trips are sequenced immediately after one another.

3.2 TEMPORAL DOMAIN DECOMPOSITION ALGORITHMIC SCHEME FOR LARGE SCALE DTA IMPLEMENTATION

In order to support ABM integration, DynusT needs to perform 24-hr simulation and assignment. This large spatial and temporal scale brings forth great computational challenges in algorithm design and implementation. The DynusT research team developed a temporal decomposition scheme called Method of Isochronal Vehicle Assignment (MIVA) that has successfully overcome this computational challenge and enabled DynusT to perform mega scale spatial- and temporal-scale dynamic traffic assignment.

The design concept is to divide the entire analysis period is divided into Epochs (evenly or based on computing loads). Vehicle assignment is performed sequentially parallelly in each Epoch. Vehicles generated in each Epoch are assigned in a parallel (multi-threaded) fashion. This scheme has two principal advantages. First it improves the model scalability by confining the peak runtime memory requirement as the maximum memory usage for individual Epoch regardless of the total analysis period; thus improving the memory efficiency. Secondly, the parallel processing of vehicle assignment further improves the runtime efficiency. A self-turning scheme adaptively searches for the run-time-optimal Epoch setting during iterations regardless of the characteristics of the modeled network. The following explanations are the brief excerpt from Nava and Chiu (2012).

As shown in Figure 3-2, the common algorithmic framework of a typical simulation-based DTA model includes the iterative execution of network loading (simulation), the path set update procedure (time-dependent travel time cost of
routes for all origin, destination, and departure time triplets), and the path set adjustment (DTA) procedure to update vehicle paths. To determine how close the current solution is to the DUE condition, the evaluation of path assignments, by means of simulation, requires checking a defined convergence criterion. The algorithm terminates if the stopping criterion is met.

The proposed Method of Isochronal Vehicle Assignment (MIVA) decouples the time domain between simulation and both the path set update and path adjustment procedures (comprised of both the TDSP and assignment solution algorithm) into forward-sliding time periods which allows the memory requirement for both the path set update and path adjustment to be bounded solely on the length of the determined temporal segment instead of the entire analysis period.

The memory usage for storing time-varying link travel times is of memory size $|I| \times |T|$, where $I$ is the set of links and $T$ is the set of assignment intervals. For storing the time-varying node (intersection) delay is of size $|I| \times |M| \times |T|$, where $M$ is the set of movements. These arrays are of modest size, even for a large network and long analysis period. The TDSP memory requirement, depending on the algorithmic implementation, generally requires the memory for storing many arrays with dimension $|I| \times |M| \times |J| \times |T|$, where $J$ is the set of destinations (or zones if the destination is the centroid of the zone). For example, a network with 20,000 links, 1,000 zones, 5 movements per intersection, and 100 departure intervals would need $1 \times 10^8$ elements to store information for one array, let alone the multiple arrays typically needed during full network TDSP computation. The applied TDSP algorithm for this research is label-correcting with complexity $O(nmT^2)$, where $n$ is the number of nodes, $m$ is number of links, and $T$ is number of assignment intervals, it is apparent that the TDSP computational time will grow polynomially with network size and the analysis period. The memory usage for the assignment procedure, although varied by

![Diagram of simulation process](image-url)
implementation, typically would require a significant amount of memory to store time-dependent path set for each origin-destination-departure time triplet. The memory usage is linear in the temporal domain, but could be large if each path is stored in terms of individual nodes/links comprising the path.

The MIVA Temporal Decomposition Framework

Shown in Figure 3-3, the MIVA scheme is denoted by two inter-associated time periods: *Epoch* and *Projection Period*. The MIVA scheme decouples the temporal domain of the analysis period (also termed simulation period) into sequential segments of equal length called *Epochs*. For vehicles departing within a single Epoch, the arrival times to their destination are used to estimate the time period, known as the *Projection Period*, in which the *T* domain for the TDSP algorithm is defined for path set update for vehicles departing within the current Epoch. At the end of one Epoch, all TDSP and assignment-related memory is de-allocated, and then re-allocated for the next Epoch. The MIVA scheme then slides the path set update and adjustment operations from one Epoch to the next until completing all Epochs. As a result, the memory usage during the entire path set update and adjustment operation is only a function of the Epoch length.

![Figure 3-3. The MIVA computational scheme implemented within the SBDTA algorithmic framework (Nava and Chiu 2012)](image-url)
Epoch

The Epoch is the partitioned period that acts as the temporal segment for the path adjustment procedure, meaning the TDSP and assignment procedure is bounded solely by the length of the Epoch. An Epoch consists of multiple assignment intervals (interchangeably termed as departure intervals as assignment is performed for vehicle departing at the same departure interval). An aggregation interval pertains to the time interval in which traffic data, i.e., time-dependent link travel times and intersection delays, are averaged to be the input for the TDSP. The assignment interval is bounded by the number of simulation intervals. A simulation interval is defined as the time resolution that traffic simulation states are updated. An assignment interval is a multiple of simulation intervals, and, in the same manner, an Epoch is a multiple of assignment intervals, as shown in Figure 3-4.

Let $H$ be the simulation period and $h$ be the length of the assignment/departure time interval in which $H$ is discretized resulting in a time discrete model. Let $T = \{\tau^1, \tau^2, ..., \tau^{H/h}\}$ be the set of departure time intervals. Let the length of each Epoch (number of assignment intervals) be in terms of the integer number of assignment intervals $b = H/h/n$ where $n$ is the pre-specified total number of Epochs within $H$. Let $E = \{e^1, e^2, ..., e^n\}$ be the set of Epochs. Let $e^s = \{\tau^{(s-1)b+1}, \tau^{(s-1)b+2}, ..., \tau^{sb}\}, \forall e^s \in E$ be the set of assignment intervals for Epoch $e^s$ containing $b$ number of assignment intervals. In each Epoch domain there is a set of departing vehicles $V^{e^s}(i,j,\tau) \subseteq V$, where $V = \{v^1, v^2, ..., v^{|V|}\}$. Those vehicles $v \in V^{e^s}(i,j,\tau)$ are assigned based on the TDSP solved over the Projection Period associated with $e^s$, which will be further explained in the next section.

**Figure 3-4.** The Epoch and Projection Period of the MIVA structure (Nava and Chiu 2012)
Projection Period

The Projection Period, $P(e^s)$ is defined as the set of assignment intervals for each Epoch $e^s$. Let $P(e^s) = \{\tau^{(s-1)b+1}, \tau^{(s-1)b+2}, \ldots, \tau^{sb}, \tau^{sb+1}, \ldots, \tau^{sb+y}\}$, $\forall e^s \in E$ be the set of assignment intervals contained in the Projection Period for Epoch $e^s$. By definition, the start of the Projection Period is synchronized with each associated Epoch. However, the Projection Period is extended beyond the end of the Epoch by $\{\tau^{sb+1}, \ldots, \tau^{sb+y}\}$ as shown in Figure 3-4. This temporal extension is to allow the TDSP to solve for the later arrival times of those vehicles departing toward the end of the Epoch.

It is intuitive to set the limit of the Projection Period based on the latest arrival time of all vehicles departing within the Epoch, as beyond this limit, link travel times and intersection delays would not be needed for the current Epoch’s TDSP calculation. However, binding the Project Period limit based on the latest arrival time may be too conservative and thus include too many additional assignment intervals if the latest arriving vehicle’s travel time is likely to improve in the next iteration because (1) it is assigned with a new path, and/or (2) other vehicles assigned with a better path would also improve the overall traffic condition and improve all vehicles’ travel times. With this being recognized, the length of the Projection Period can be defined as a ratio of total vehicles based on ranked experienced arrival times. This means vehicles belonging to $V^e(i,j,\tau)$ are sorted by increasing experienced arrival times. The Projection Period length is then defined based on a predefined ratio. For example, a 0.95 means that the end of the Projection Period is set at the 95th percentile of all arrival times. In other words, $P(e^s) = \{\tau^{(s-1)b+1}, \tau^{(s-1)b+2}, \ldots, \tau^{sb}, \tau^{sb+1}, \ldots, \tau^{sb+y}\}$, where $h \cdot \tau^{sb+y} \geq G(\varphi)$, where $G$ the increasing arrival time profile and $\varphi$ is the predefined ratio, $0.0 \leq \varphi \leq 1.0$. One should also expect that $P(e^s)$ may vary due to different levels of congestion experienced by vehicles in different Epochs. Vehicles departing in an Epoch corresponding to off-peak hours would have a shorter Projection Period than those corresponding to peak hours due to more severe congestion during peak hours even though the Epoch lengths are identical.

After solving the TDSP, the path becomes available over the domain $P(e^s) = \{\tau^{(s-1)b+1}, \tau^{(s-1)b+2}, \ldots, \tau^{sb}, \tau^{sb+1}, \ldots, \tau^{sb+y}\}$. The computational requirement to maintain $V^e(i,j,\tau)$ in memory rather than for $V$ is the major advantage of the MIVA computational scheme. Memory is only allocated to the TDSP of size $P(e)$ rather than the entire simulation period. For the given $V^e(i,j,\tau)$ set, the path set adjustment, i.e., the DTA solution algorithm procedure, is performed which is bounded by a given $e$. Once path adjustment is completed $\forall \nu \in V^e(i,j,\tau)$, the traffic data and TDSP calculations for the current Epoch $e$ are de-allocated from memory, and the MIVA scheme continues on to the next Epoch.
The following describes the MIVA algorithmic structure within the SBDTA framework:

Step 0  **Initialization:**
- Iteration count, \( l = 0 \)
- Given \( n: b \) is integer, determine \( E \)
- Initiate SBDTA algorithm
- Time-varying OD demand tables
- Instantaneous shortest paths
- All-or-nothing path assignments

Step 1  **Network Simulation:**
- Execute network simulation
- Acquire resulting time-dependent link and path travel times

Step 2  **Check Convergence:**
- If Convergence criteria < \( \varepsilon \), then:
  - Stop; Exit SBDTA algorithm
- Else:
  - \( l = l + 1 \)
  - Continue to Step 3

Step 3  **MIVA Preparation**
- De-allocate all simulation-related data structures except those used by MIVA and assignment

Step 4  **MIVA Enter:**
- For \( e \in E \),
  - Determine \( V^e(i,j,\tau) \) and allocate memory based on
  - Sort \( V^e(i,j,\tau) \) by arrival time and obtain \( \hat{g}(\cdot) \)
  - Establish \( P(e) \) based on \( G \), set percentile, and \( \varphi \), from \( V^e(i,j,\tau) \)

Step 5  **Path Set Update:**
- Perform path set update (TDSP algorithm) for \( P(e) \)

Step 6  **Path Set Adjustment:**
- Perform path set adjustment procedure for \( V^e(i,j,\tau), e \in E \)

Step 7  **MIVA Exit:**
- De-allocate memory for \( V^e(i,j,\tau) \)
- If \( e = n \), then:
  - Go to Step 1
- Else:
  - \( e = e + 1 \)
  - Continue to Step 4

Figure 3-5. The Pseudo code of the MIVA Scheme (Nava and Chiu 2012).
4.0 FAST-TrIPs Overview

For modeling the transit component, FAST-TrIPs is divided into two main sub-modules, transit assignment and simulation. Above all, the transit assignment sub-module plays the role of passenger assignment for given origin-destination (OD) pairs. For assigning transit passengers for the OD pairs, a trip-based shortest path model (Noh, Khani et al. 2011; Khani, Hickman et al. 2012; Khani, Lee et al. 2012; Noh, Khani et al. 2012; Noh, Khani et al. 2012) is utilized by searching for a feasible path on each O-D pair. The assigned passengers including their paths are given to and simulated through the transit simulation sub-module in FAST-TrIPs. During the simulation, experienced arrival and departure times of transit vehicles are used to simulate boarding and alighting of passengers, considering transfers and other components (i.e., walking and waiting). Each passenger’s trajectory (i.e. experienced path) is recorded, and dwell time for each transit route is calculated as a function of the boardings and alightings at each stop. Results of the simulation are used in the next iteration of auto-transit vehicle simulation and are also fed back to the activity-based model in the next global iteration for updating the demand.

FAST-TrIPs has an intermodal functionality which is embedded in the two sub-modules mentioned above. It is capable of assigning and simulating the intermodal passengers in a mixed environment, modeling these movements for auto and transit passengers. The intermodal model consists of a park-and-ride assignment model for individual tours, a transit assignment and transit simulation model for the transit portion of the tour, and an interface with DynusT for the auto assignment and simulation.

4.1 FAST-TrIPs in the Integrated Model

In the big picture, in terms of integration, FAST-TrIPs interfaces with DynusT as well as connecting with the DaySim activity-based model. Before discussing the FAST-TrIPs model in detail, the overall functional location of the FAST-TrIPs tool in the integrated architecture should be noted. The high-level modular architecture for transit in the C-10B project is given in Figure 1-1, and Figure 4-1. Fast-TrIPs Algorithmic Structure briefly depicts the FAST-TrIPs operation.

To operate the FAST-TrIPs module, or DTA module in a higher level, there are several critical inputs to FAST-TrIPs. To support the transit and intermodal travel within DynusT, FAST-TrIPs reads as input the database of traveler tours and trips from DaySim. This processing of the trip and tour rosters allows FAST-TrIPs to identify both the transit trips with walk access and egress, as well as trips with auto access or egress. The latter are intermodal trips in the sense that they require simulation of both modes of travel (transit and auto) within FAST-TrIPs and DynusT.
Specifically, for the purposes of FAST-TrIPs, the output from DaySim includes information for each trip segment and subsequent activity for each individual, containing origin, destination, travel mode including differentiation for different transit access modes, and time at which travel begins, for trips leaving from the primary tour destination, or time at which travel ends for trips destined to the primary tour destination.

This last choice, of departure time or arrival time, makes use of DaySim’s protocol that trips going toward the primary tour destination are constrained to arrive by the start of the activity at that destination. Also, trips departing from the primary tour destination cannot depart before the scheduled end of the activity at that the primary tour destination.

Passengers are assigned to a path in the transit network according to the protocol discussed later. In addition to the assignment of transit trips, FAST-TrIPs also performs an intermodal assignment, for which an estimate of auto travel times is necessary. This allows FAST-TrIPs to choose the optimal combination of an auto segment and a transit segment for an intermodal trip. The auto travel times in this assignment are obtained from either a time-dependent auto shortest path or the most recent auto skims of the network loading and simulation in DynusT. This is indicated on the right side of Figure 1-1. DynusT-FAST-TrIPs Integration Framework.
Once the intermodal assignment has been performed, FAST-TrIPs provides DynusT with the auto portion of any intermodal trip, allowing the simulation of the auto trip through either:

- The assignment of an origin, destination (a park-and-ride lot), and time of arrival at the park-and-ride lot, for trips transferring to transit headed to the primary tour destination; or

- The assignment of an origin (park-and-ride lot), destination, and time of departure from the park-and-ride lot, for trips returning from the primary tour destination.

The auto portion of these intermodal trips is simulated in DynusT. In addition, the trip and tour records from DaySim are updated to enforce sequential trip behavior: in the simulation, transit trips cannot begin before the auto actually arrives at the park-and-ride lot, and vice-versa.

Of course, in parallel, DynusT reads the same trip and tour rosters from the intermodal assignment and determines the assignment of pure auto trips to associated paths and travel times in the network. Especially for intermodal cases, intermodal assignment will create the auto and transit part of these intermodal trips as deciding the optimal park-and-ride lot. The auto portion of the intermodal assignment is added in the DynusT simulation, and the transit portions are loaded to the transit assignment in FAST-TrIPs. These assigned auto trips from the intermodal demand are then simulated in DynusT.

As a second input shown in Figure 1-1, FAST-TrIPs uses Google’s General Transit Feed Specification (GTFS) files. The GTFS files currently allow a transit agency to provide its routes and schedules to Google Maps. However, this same route and schedule data are often made publicly available by transit agencies, allowing others to develop applications using these data. The Sacramento Regional Transit District is one of the many transit agencies providing GTFS data to Google and to the public.

The GTFS files contain the geographic representation of routes and stops (typically in GIS shape files). The data also contain either (1) the formal schedule of service (in the case of GTFS); or, (2) the frequency information (in the case of traditional line files), associated with each transit route and direction. The GTFS data or line files are converted into route networks that are compatible with the DynusT road network. This process is partly automated, using existing shape files for the road and transit networks, but considerable manual processing may be necessary to adjust the network to ensure that road segments are consistent and that transit stop locations are placed at appropriate locations in the road network. Finally, the schedule (the so-called “stop-times” in GTFS) also serves as input to the transit assignment. The transit network and service data should ideally be based on the GTFS. This is useful because the actual service schedule and individual stops can be modeled explicitly. This provides a more dynamic modeling of the transit passenger behavior than a traditional 4-step model. The GTFS data can be used to represent the base year, perhaps by making some
manual adjustments to the existing (2010 for the C10 implementation) GTFS schedule data to make it comparable to the base year.

The detail by which transit routes are defined in a schedule-based format such as GTFS may pose problem when dealing with future year forecasts. A transit network for the future will require designating routes, stops, and schedules. This may be easily adapted from existing schedules if only more modest changes are envisioned. However, for more significant changes in the transit network, this could require significant effort to develop appropriate GTFS data. Whether to develop this GTFS input, or simply using a future line file (from a 4-step model), would be a decision likely made jointly by the local MPO (like SACOG) or the local transit agency (like Sacramento Regional Transit).

For the auto and transit vehicle simulation, as shown at the bottom of Figure 1-1, DynusT-FAST-TrIPs Integration Framework, the auto, transit, and intermodal assignments are fed directly into a network loading and simulation, using DynusT’s simulation capability. The simulation of vehicle movements in the network, at a fine level of temporal and spatial resolution, is handled directly within DynusT; this includes specific movements of both auto and transit vehicles in the network. There is appropriate logic in DynusT for the management of mixed vehicle types in the traffic stream, as explained earlier in Section 2, with each vehicle type having its own particular speed-density characteristics. The simulation also allows for the appropriate movements of transit vehicles at stops, including: (1) a vehicle pulls out of the traffic stream at bus pull-outs, but remains in the traffic stream for curbside stops; (2) dwell times allow for passengers to alight and to board, depending on passenger type (pedestrian, bicycle, wheelchair); (3) a bus may bypass a stop if no passengers want to board or alight (so-called “hail stop” operation); and, (4) the vehicle may hold until the scheduled departure time if a vehicle is running ahead of schedule (“running hot”).

The simulation of passenger movements in the transit network, however, is handled directly within FAST-TrIPs. The interaction between DynusT and FAST-TrIPs is through the link travel times that DynusT provides to FAST-TrIPs and the bus arrival, dwell, and departure times that FAST-TrIPs provides to DynusT. At the beginning of the simulation, FAST-TrIPs provides an estimate of the dwell times for each transit vehicle at each stop. During the simulation in DynusT, the transit vehicle dwells at the stop for this period of time, and then, if holding is needed, the vehicle is held until the scheduled departure time. The final simulation outputs from DynusT include the actual vehicle arrival and departure times from each stop. These outputs, in turn, can be used in the assignment to adjust passengers’ path choices, to reflect issues with service reliability or to confirm vehicle adherence to the schedule.

The assignment and simulation of transit vehicles and passengers is described in the following subsections.
4.2 Passenger Assignment

Transit assignment is the process that determines a path, or set of connecting sub-paths, for a passenger to get from their origin to their destination. DaySim itself determines the origins and destinations for the travelers, and feeds this information into the assignment. FAST-TRIPS handles transit passenger and intermodal passenger assignment, while DynusT handles auto assignment.

From the output of DaySim, intermodal and transit trips are read directly by FAST-TRIPS. The required inputs for transit assignment in FAST-TRIPS include the passenger origin, the passenger destination, and the time of departure at the origin or the time of arrival at the destination. FAST-TRIPS has a transit shortest path algorithm, called trip-based shortest path (TBSP), which finds the path from the origin to the destination of the passengers.

Algorithmically, transit networks have an important feature in the types of nodes. That is, in transit networks, rerouting (i.e. a transfer to another route) can be done only at certain stops. For example, when a passenger is on board, he/she may not consider alighting from a vehicle to transfer to another route at every stop. More particularly, it does not happen where there is no other route available at a stop, or where other stops are not within walking distance from that stop. Therefore, one may consider any variation in the path at transfer stops only. This property leads to the generation of a hierarchical transit network with transfer stops at the higher level, where the non-transfer stops in the path search algorithm may be disregarded. In this case, instead of having multiple stops and multiple links between two transfer stops, the transfer stops can be connected by a single transit trip, with associated departure and arrival times. However, the non-transfer stops are maintained in the network, as they can be used for access and egress points.

In the network preparation phase, transfer links are generated between a pair of stops if:

1. The distance between the stops is less than a certain value (e.g. 0.25 mile; and
2. There is at least one route which serves one of the stops but not the other.

After generating transfer links, transfer stops are defined to construct the network hierarchy. A transfer stop is defined as a stop from which the passenger has the option of transferring to another route. With this definition, a stop is defined as a transfer stop if it is located on more than one route, or it has at least one inbound or outbound transfer link.

Using this simple model, the hierarchical transit network, which can be very useful in transit path algorithms, is generated.

A new network representation which is suitable for modeling schedule-based transit systems was used. This network structure is called trip-based and is defined by a graph $G(N, P, T)$ where $N$ is the set of nodes (or stops), $P$ is the set of
transit trips where each trip belongs to a route \( r \) in \( R \) (set of transit routes), and \( T \) is the set of transfer links. For each trip there is a list \( S(p) \) which contains the stops served by the trip as well as the associated arrival and/or departure time of the transit vehicle at that stop. Also, for each stop, there is a set \( A(n) \) containing the trips serving the stop. The main advantage of the trip-based network representation over node- or link-based structures is that the connection between two stops can be established by a single trip \( p \) if they are located on a same trip, while in both node-based and link-based networks the connection between any two stops is made using sequence of links.

**Transit Trip-Based Shortest Path (TBSP)**

The trip-based network has the advantages of stop connectivity, dynamic representation of service, and hierarchical structure of transfer stops. Based on these properties, together with the data availability through GTFS, and the behavior of transit users, a TBSP is developed. TBSP is a labeling algorithm in the schedule-based transit systems, exploiting the trip-based network format. Therefore, it has the advantage of processing a subset of stops and finds the shortest path in a shorter amount of time compared to traditional shortest path algorithms. The general form of the algorithm is shown in Figure 4-2. TBSP Algorithm; it can be either label-setting or label-correcting, but the label correcting form is used for the application in this project.
1- **Initialization:**
2- Get the origin (O) and the departure time (τ)
3- \(i = O, l_i = 0, p_i = \Phi, a_i = \tau, m_i = \Phi, SE = \{i\}\)
4- \(l_j = \infty, p_j = -\infty, a_j = \infty, m_j = \Phi; \forall j \neq i\)
5- **Termination Criterion:**
6- If \(SE = \Phi\), stop!
7- **Stop Selection:**
8- Select \(i = \text{Argmin}_j \{l_j\} j \in SE\), if Label Setting
9- Select \(i = \text{The first stop in } SE\), if Label Correcting
10- \(SE = SE \{i\}\)
11- **Updating the labels:**
12- \(\forall t \in T(i)\):
13- If \(l_i + t_j < l_j\):
14- \(l_j = l_i + t_j, a_j = a_i + t_j, p_j = i, m_j = "T"\)
15- \(SE = SE \cup \{j\}\)
16- \(\forall r \in R(i)\):
17- Select \(p = \text{Argmin}_q \{w_{iq}\} d_{iq} \geq a_i\)
18- \(\forall j \in S(p)\) with \(seq_j > seq_i\):
19- If \(l_i + w_{ip} + v_{ijp} < l_j\):
20- \(l_j = l_i + w_{ip} + v_{ijp}, a_j = d_{jp}, p_j = i, m_j = p\)
21- If \(j \in N_i\):
22- \(SE = SE \cup \{j\}\)
23- Break the loop!

---

**Figure 4-2. TBSP Algorithm**

The variables in Figure 4-2. TBSP Algorithm are defined as follows:

- **PAT** = Preferred arrival time to stop \(i\)
- **seq** = Sequence number of stop \(i\) on trip \(p\)
- **dp** = Departure time of trip \(p\) at stop \(i\) (usually the same as arrival in printed schedules and GTFS data)
- **vijp** = In-vehicle time from stop \(i\) to stop \(j\) using trip \(p\)
- **tij** = Transfer time from stop \(i\) to stop \(j\) (typically equals to the walking time between two stops)
- **ai** = Arrival/departure time label of stop \(i\)
- **wp** = Waiting time at stop \(i\) for trip \(p\), equal to the difference between the departure time of trip \(p\) at stop \(i\) and the arrival time label of stop \(i\)
- **li** = Label of stop \(i\), equal to the travel time (or cost) from the origin stop to stop \(i\) in forward algorithms, and the travel time (or cost) from stop \(i\) to the destination stop in backward algorithms
- **pi** = Predecessor stop of \(i\)
The mode (trip number or transfer link) used to reach stop i in forward algorithms, or to leave stop i in backward algorithms.

The utility (a function of travel time or cost) of trip p at stop i to reach the destination.

The set of transfer links at stop i.

The set of routes at stop i.

The set of trips at stop i.

Scan Eligible list, containing the stops with temporary labels.

The algorithm in Figure 4-2. TBSP Algorithm is a forward labeling algorithm starting from the origin with \( \tau \) as the planned departure time (PDT). With very few modifications, a backward algorithm can be developed with search from the destination in a backward direction. In this backward case, a preferred arrival time (PAT) is used for the destination. In the proposed forward algorithm, the label of each stop is the earliest arrival time.

In general, a generalized cost function can be used to generate different variations of the shortest path algorithm. In this case, different weights are applied to different components of the trip. This approach is used to model the inconvenience of transfers and waiting times compared with in-vehicle time. Assuming the weight \( \alpha_k \) is applied to the \( k \)-th element of the passenger trip, the least-cost path algorithm is developed based on the shortest path algorithm with changes as shown below:

13. If \( l_i + \alpha_w w_{ip} + \alpha_v v_{ijp} < l_j \):
14. \( l_j = l_i + \alpha_w w_{ip} + \alpha_v v_{ijp}, a_j = d_{jp}, p_j = i, m_j = "T" \)
19. If \( l_i + \alpha_w w_{ip} + \alpha_v v_{ijp} < l_j \):
20. \( l_j = l_i + \alpha_w w_{ip} + \alpha_v v_{ijp}, a_j = d_{jp}, p_j = i, m_j = p \)

The result of the TBSP is a shortest path tree from the origin to all destinations. So the path to a specific destination will be found by tracking the predecessors from the destination stop to the origin stop. The path is then attached to the passenger and is passed to the simulation model.

### 4.3 Passenger Simulation

The passenger simulation model is a high resolution model, capable of simulating the path taken by individuals in the transit and intermodal networks. The main inputs are the paths generated in the transit assignment sub-module. In fact, there are three categories of data inputs to the passenger simulation:

- **Transit Network**, including stops, routes, and schedule
- **Transit Vehicle Simulation Results**, including the actual arrival/departure of transit vehicles at each stop
- **Passengers**, including information about each passenger and his/her assigned path

There are two main modules to capture the behavior of passengers and their interactions with transit vehicles. The first module captures access, egress, and transfer behavior of passengers. In the same way the simulation captures the movement of a passenger from his/her alighting stop to either his/her destination or the next boarding stop (in case of a transfer). The detailed information of the passenger’s trip is recorded. The second module takes care of the boarding and alighting of passengers whenever a transit vehicle arrives to a stop. Therefore, an event-based simulation is used for this part, when an event is the simulated arrival of a transit vehicle to a transit stop. Considering factors such as the number of boarding and alighting passengers and type of transit vehicle, a dwell time value is calculated for the transit vehicle at each stop. For each transit vehicle, based on the type of route, a capacity is assumed. All of the information regarding the boarding, alighting, and passenger load of the vehicles is written in the output files and can be used in the next iteration. The model also has post processing functions for calculating skim tables, summary statistics and convergence measures.

**Passenger Movements in Simulation**

The primary engine that performs simulations is the simulation engine within DynusT, which handles auto and transit vehicle movements. However, FAST-TrIPs itself handles passenger movements within transit network.

During the simulation itself, FAST-TrIPs and DynusT’s mesoscopic simulator do not communicate directly. Rather, the “simulation” of the passenger movements is done through post-processing of the vehicle trajectories from DynusT. This design decision was made so as to maintain the computational efficiencies of DynusT but yet still have a means of determining passenger movements. This passenger simulation process is illustrated in Figure 4-3.
As described previously, DynusT mesoscopic simulator handles all vehicle movements, including autos and transit vehicles. Among many other outputs, DynusT generates vehicle trajectories for the transit vehicles that are simulated, and it also records the times that autos depart their origins and arrive at their destinations. These outputs, shown on the left side of Figure 4-3. Transit Passenger Simulation in FAST-TrIPs, serve as the critical inputs to the transit and intermodal travel simulation in FAST-TrIPs.

In FAST-TrIPs, these data are used in a simulation of transit passenger travel. This simulation likewise follows a high temporal resolution, using a 1-second interval. With this as the simulation clock, the simulation methods in FAST-TrIPs handle the bookkeeping associated with the persons on board each vehicle and at the transit stops. This is done through a set of lists keeping track of passengers on a vehicle and passengers in a stop at any point in time.

The movement of passengers in each time interval follows the following transit assignment logic:

- A passenger who arrives at a stop during that time interval, from a given access mode (walk or auto access), is loaded into the queue at the stop. Note that passengers using auto access are not generated until the DynusT outputs indicate that the auto has arrived at the park-and-ride lot.

- A transit vehicle that arrives at a stop during that time interval is processed, in the sense that alighting passengers are removed from the
vehicle and boarding passengers are added to the vehicle. The first method processes the passengers on the vehicle \( k \). Those passengers on board whose final stop is \( s \) will be taken off this list and processed as an alighting passenger. Statistics for the alighting passenger (access time, egress time, in-vehicle time, transfer time, number of transfers, etc.) are written out to a file to compute experienced transit skims and summary statistics for the transit mode. A second method processes the list of passengers in the queue at the stop \( s \). Those passengers desiring to board this vehicle \( k \) are then placed on board the vehicle, in a first-in first-out basis, until either all passengers desiring to board have been admitted on board, or the vehicle capacity is reached. In the latter case, passengers are denied boarding due to the capacity constraints.

- An alighting passenger is sent to the destination, by a given egress mode, or to a transfer stop.

As these movements occur, FAST-TrIPS keeps track of the time associated with access, waiting, on board, transferring, etc. in order to generate passenger statistics. Once passengers arrive at their destinations, their travel statistics are accumulated, and the passengers’ experienced skims from the transit network are also generated. These skims and transit passenger summary statistics follow the structure of similar outputs for vehicle trips from DynusT. These in turn can be used to affect the transit passenger assignment in the next iteration. In addition, transit vehicle operating statistics (on-time performance, travel times, hours in operation, etc.) can also be derived from the trajectories as outputs from the DynusT simulation.

In the DynusT - FAST-TrIPS integration model, three types of convergence apply. DynusT and FAST-TrIPS, have their own individual convergence methods that use a relative gap measure to assess user equilibrium. DynusT utilizes the simulation-based relative gap measure. The convergence of FAST-TrIPS is estimated by the number of passengers who are denied from boarding due to capacity constraints. Finally, for the combined DynusT-FAST-TrIPS model, a relative gap measure comparing dwell time changes from one iteration to the next is applied.

**Intermodal Assignment and Simulation**

The intermodal assignment and simulation model in FAST-TrIPS models passenger tours with a combination of auto and transit modes, i.e. passengers who drive from their origin to access transit routes have the opportunity to select their transit path among a big set of transit options. Travelers are constrained to park their cars at the transit stations something that limits their feasible choices to park-and-ride facilities only. Furthermore, they have to return to the same park-and-ride to pick up their car in the return portion of the tour. These constraints make the intermodal assignment a difficult problem containing different choice problems.
In the SHRP 2 C10 implementation, the intermodal problem has been decomposed to a tour-based park-and-ride choice (called the intermodal assignment model) and transit/auto path choice model (similar to auto or transit passengers). Given a passenger tour, including the activities and time windows between the activities, the intermodal assignment model finds the optimal park-and-ride location considering the travel cost in the whole tour. More specifically, the park-and-ride location has to:

1. Be reachable in the time window between two consecutive activities; and
2. Have the minimum tour travel cost compared to other park-and-rides.

The algorithm that solves the intermodal problem is a combination of a shortest path in the auto network using planned departure time (PDT) from origin to all park and rides, and a backward transit shortest path to the destination using the preferred arrival time (PAT). This gives the travel cost of the bi-modal trip from the origin to the destination. With the similar concept travel cost for the second half of the tour is calculated and the optimal park-and-ride is assigned to the tour with a specific arrival time to make the mode transfer. Then the trips with auto or transit mode are extracted and fed to the transit assignment model in FAST-TrIPs or DynusT. To make the algorithm computationally efficient, a trip-based shortest path is used for the transit network which takes into account the service schedule in GTFS.

In the transit assignment, intermodal passenger trips are modeled similarly to other passenger trips with the difference that either the origin or the destination is a park-and-ride location. The assignment model generates the path for passengers and the passengers are simulated at the same time with other passengers. One important thing to be considered in simulating intermodal passengers is the actual arrival to the park-and-ride and the transfer to transit mode. Because the first part of an intermodal trip is traveled by auto, an auto trip is simulated in DynusT considering the roadway congestion, and the actual arrival of the passenger to the transit station is recorded. This time is used for the simulation of the transit part and the feasibility of transit path is verified before starting the simulation. Finally, FAST-TrIPs produces the outputs of the intermodal tours in separate files from transit and sends them back to the demand model.
5.0 References


Noh, H., A. Khani, Y.-C. Chiu, M. Hickman and B. Bustillos (2012). Integration of Dynamic Traffic Assignment with Schedule-Based Transit Assignment and Simulation of
Congested Multimodal Networks. The 4th International Symposium on Dynamic Traffic Assignment, Martha’s Vineyard, MA.


6.0 Appendix A: DynusT-FAST-TrIPs Integration Model Execution

As starting with DST_FT_module.py, successive sequential steps to run the DynusT-FAST-TrIPs integration model are processed as follows.

**Step 0: ft_BST.py -> ft_Intermodal.exe**

**Step 1: DynusTx64.exe**

**Step 2: FAST_TrIPs.py -> ft_Assignment.exe -> ft_Simulation.exe**

If not converged yet, go to Step 1. Else, stop.

More details are as follows:

**Step 0: ft_BST.py**

To initiate the main DynusT-FAST-TrIPs integration model, DST_FT_module starts with the boot-strapping process, ft_BST.py which first reads veh_sout.dbf as input files and converts it to text file, to make the file reading more efficient in DynusT and FAST_TrIPs. Then, it runs intermodal assignment (ft_intermodal.exe) which

- separates auto demand and transit demand (modes 6 and 3 respectively) and writes them down in separate files for DynusT and FAST-TrIPs use
- runs an initial intermodal tour assignment (with static auto travel times and printed transit schedule) and splits intermodal tours into transit and auto trip segments, and append them into the auto and transit demand files.

Therefore, after running the python script, demand files are generated for auto (including auto trips and auto part of intermodal tours) and transit (transit trips and transit part of intermodal tours) which are used for assignment in DynusT and FAST-TrIPs respectively.

**Step 1: DynusTx64.exe**

The DynusT module mainly assigns and simulates auto demand including ABM trips, exogenous trips and auto trip in intermodal trip. For integrating with
FAST-TrIPs module, critical transit vehicle arrival information (AltTime_transit.dat) is created.

**Step 2: FAST_TrIPs.py**

FAST_TrIPs module consists of three components; transit assignment, passenger simulation, and intermodal assignment. These components are designed in separate files (ft_Assignment.exe, ft_Simulation.exe, and ft_Intermodal.exe) and are executed sequentially as explained below:

- **Transit Assignment (ft_Assignment.exe)**

  This component assigns passenger trips along the searched transit path which typically shows boarding and alighting stops, utilized transit trips (or vehicles), and access and egress times. Transit paths are assigned to transit passengers so that they can be simulated in the next component (passenger simulation).

- **Passenger Simulation (ft_Simulation.exe)**

  Using the passenger assignment information (ft_passenger.dat) and simulated transit vehicles’ information (AltTime_transit.dat), passengers are simulated on the transit network from their origin to destination, so that passenger flow on each transit trip (vehicle) is determined. After running this component, transit-related outputs are generated and dwell times (in TransitDwellTime.dat) are fed back into DynusT module in an iterative process. Convergence of the system is also tested in this component and the result is printed in ft_convergence.dat.

- **Intermodal Tour Assignment (ft_Intermodal.exe)**

  The intermodal tour assignment is the same function used in ft_BST.py, which assigns intermodal tours to optimal park and rides and splits transit and auto segments for the next iteration. The difference is that in this step, the results of simulation (time-dependent travel time and experienced transit vehicle trajectories) are used in the algorithm. In the latest iteration of DynusT-FAST-TrIPs, the intermodal assignment is not required and transit passenger simulation is the last step of the model.
Appendix B: Working with Dataset

7.1 CREATE WORKING FOLDER

For the integrated DynusT-FAST-TrIPs model, each executable file is assigned to a specific folder as shown in Figure 7-1. Basically, the data folder (given any desired name) includes all the main functions (executable files) of the model and the data set input files except those related to demand from the activity-based model. More specifically, the python master code, called DST_FT_module.exe is in this folder along with DynusT executable and FAST-TrIPS executable files (ft_Intermodal.exe, ft_Assignment.exe and ft_Simulation.exe). In addition, two FAST-TrIPS python scripts (ft_BST.py and FAST_TrIPs.py) which call the main functions also need to be placed in this folder.

A folder called VehicleDemandGen should be created in the main folder, in which all other python scripts are placed to generate demand files from activity-based model. The two sub-folders in VehicleDemandGen folder are assigned for the input files and output files.

The main input file for the DynusT-FAST-TrIPs model is veh_sout.dbf which is located in the Input Files sub-folder, and all the demand-related files are placed in the same folder after generation in the bootstrapping step. Under the same token, DynusT output files need to be copied to the Output File sub-folder. Especially, Output Files folder is temporarily activated in terms of file transfer. The outputs (i.e., vehicle.dat and path.dat) generated by DaySim2DynusT-MAINprog-S.py in VehicleDemandGen are temporarily stored in Output Files folder then transferred to the working folder and utilized by DynusT during runs.

Aside from hosting demand file, the Input Files sub-folder also hosts information mapping files for DynusT and DaySim attributes as well as defined corridors or paths of interest and several DynusT network files. The mapping files are of importance for the following reasons: 1) they define DynusT to DaySim zone, node, and parcel equivalents and 2) coordinate information is used for distance calculation. The defined corridor data provides for easy access information for the purpose of post-processing simulation data. Lastly, the DynusT origin and destination network files will provide information for trips.
As mentioned, the python files in the VehicleDemandGen folder read and translate DaySim trip information and prepares it as DynusT input files. To execute such processes, intermediate “mapping” files (found in Input Files sub-folder) for DynusT and DaySim attributes must also be developed. In essence, this ensures trips will start, originate, and terminate at the DaySim designated times and locations but within DynusT. Within the DynusT simulation a person making a trip will be assigned a vehicle, once this trip reaches its destination the assigned vehicle will exit the simulation. A person making multiple trips within the 24 hour simulation period will be assigned into several vehicles; only one of those vehicles will be simulated at once.

Error! Reference source not found. illustrates typical files (not exhaustive list) for DynusT and FAST-TrIPS.

Table 7-1. Input File List in “Working folder”

<table>
<thead>
<tr>
<th>DynusT</th>
<th>Network Data (network.dat)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Movement Data (movement.dat)</td>
</tr>
<tr>
<td></td>
<td>Vehicle Generation Data (Origin.dat)</td>
</tr>
<tr>
<td></td>
<td>Vehicle Exit Data (destination.dat)</td>
</tr>
<tr>
<td></td>
<td>Signal data (control.dat)</td>
</tr>
<tr>
<td></td>
<td>Simulation and Assignment Configuration Data (system.dat)</td>
</tr>
<tr>
<td></td>
<td>Model Scenario Data (scenario.dat)</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Work Zone Data (workzone.dat)</td>
</tr>
<tr>
<td></td>
<td>Incident Data (incident.dat)</td>
</tr>
<tr>
<td></td>
<td>Vehicle Trip Schedule Data (vehicle.dat)</td>
</tr>
<tr>
<td></td>
<td>Vehicle Route Data (path.dat)</td>
</tr>
<tr>
<td></td>
<td>Advanced Parameter Data (parameter.dat)</td>
</tr>
<tr>
<td>FAST-TrIPs</td>
<td>ft_input_stops.dat</td>
</tr>
<tr>
<td></td>
<td>ft_input_routes.dat</td>
</tr>
<tr>
<td></td>
<td>ft_input_trips.dat</td>
</tr>
<tr>
<td></td>
<td>ft_input_stopsTimes.dat</td>
</tr>
<tr>
<td></td>
<td>ft_input_accessLinks.dat</td>
</tr>
<tr>
<td></td>
<td>ft_input_transfers.dat</td>
</tr>
<tr>
<td></td>
<td>ft_input_transitVehicles.dat</td>
</tr>
<tr>
<td></td>
<td>ft_input_park-n-rides.dat</td>
</tr>
<tr>
<td></td>
<td>ft_demand_auto.dat</td>
</tr>
<tr>
<td></td>
<td>ft_demand_transit.dat</td>
</tr>
<tr>
<td></td>
<td>ft_passengers.dat</td>
</tr>
<tr>
<td>DynusT-</td>
<td>networks.dat</td>
</tr>
<tr>
<td>FAST-TrIPs</td>
<td>AltTime_transit.dat</td>
</tr>
<tr>
<td></td>
<td>AltTime_interModal.dat</td>
</tr>
<tr>
<td></td>
<td>travelerVehicleMapping.dat</td>
</tr>
<tr>
<td></td>
<td>travelerVehIntermodal.dat</td>
</tr>
<tr>
<td></td>
<td>TransitRouteSchedule.dat</td>
</tr>
<tr>
<td></td>
<td>TransitDwellTime.dat</td>
</tr>
</tbody>
</table>
Table 7-2. Input file list in “VehicleDemandGen - Input Files”

<table>
<thead>
<tr>
<th>File Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node-LatLong.geo</td>
</tr>
<tr>
<td>Parcel-LatLongZone.geo</td>
</tr>
<tr>
<td>parcelMOD.txt</td>
</tr>
<tr>
<td>DSTtoDYSzones.txt</td>
</tr>
<tr>
<td>PathInput.txt</td>
</tr>
<tr>
<td>pathInputNAME.txt</td>
</tr>
<tr>
<td>xy.dat</td>
</tr>
<tr>
<td>network.dat</td>
</tr>
<tr>
<td>origin.dat</td>
</tr>
<tr>
<td>destination.dat</td>
</tr>
<tr>
<td>veh_sout.dbf</td>
</tr>
</tbody>
</table>

7.2 **RANCHO CORDOVA NETWORK EXAMPLE**

Rancho Cordova, CA is an eastern suburb of Sacramento, CA. The DynusT Rancho Cordova network was created in late August in 2010 and is a ‘subarea network cut’ from the regional Sacramento model. The DynusT network is shown in Figure 7-2 below.

**Network**

This network is composed of 121 zones, 452 nodes, and 864 links. Like the downtown model, the Rancho Cordova network must first be cleaned/fine-tuned in order to bring it from its original state, macroscopic model network, to mesoscopic model compatibility.

![Figure 7-2. Rancho Cordova transportation network in DynusT](image-url)
For Rancho Cordova area, the transit network consists of 163 stops, 170 bidirectional transfer links, and 205 trips for five bus routes. The Sacramento light rail transit (Gold line) is excluded in this dataset. The transit data is shown in Figure 7-3.

![Figure 7-3. Rancho Cordova transit network](image)

### Demand

Exogenous trip tables (airport, external, thru auto & truck, and 2- & 3-axle commercial vehicles) from the travel demand model were converted to DynusT origin-destination (OD) demand table input. The conversion produced two temporal OD files, demand.dat and demand_truck.dat, with defined DynusT formats (explained in demand.dat description memo). The following table summarizes the generated demands.

Table 7-3. Rancho Cordova exogenous demand statistics

<table>
<thead>
<tr>
<th>File Name</th>
<th>Trips</th>
<th>Total Trips</th>
</tr>
</thead>
<tbody>
<tr>
<td>demand.dat</td>
<td>• airport: 12,262</td>
<td>435,637</td>
</tr>
<tr>
<td></td>
<td>• thru auto: 220,489</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• external: 202,886</td>
<td></td>
</tr>
<tr>
<td>demand_truck.dat</td>
<td>• 2- &amp; 3-axle: 83,388</td>
<td>89,266</td>
</tr>
<tr>
<td></td>
<td>• thru truck: 5,878</td>
<td></td>
</tr>
</tbody>
</table>
Similarly, trip information from the activity-based model (DaySim) is converted, generating a vehicle.dat file with defined DynusT formats (explained in vehicle.dat description memo). The following table summarizes the generated demands. As previously mentioned and observed, the DaySim and DynusT values are different, due to the filtering trips modes that were not solely auto (e.g., ‘shared ride’ or ‘drive alone’) such as ‘drive-transit-walk’, ‘bike’, ‘walk’ along with other modes.

Table 7-4. Rancho Cordova trip roster demand statistics

<table>
<thead>
<tr>
<th>File Name</th>
<th>Total Trips</th>
</tr>
</thead>
<tbody>
<tr>
<td>DaySim trip roster</td>
<td>140,099</td>
</tr>
<tr>
<td>vehicle.dat</td>
<td>117,795</td>
</tr>
</tbody>
</table>

Transit Route

The operating transit system in Rancho Cordova is a part of Sacramento Regional Transit. There are 5 bus routes, and a light rail line which cover Rancho Cordova area and the connection between other parts of Sacramento regional area (i.e., some of the routes are extended out of Rancho Cordova). To construct the transit network, General Transit Feed Specification (GTFS) data of Sacramento Regional Transit is used and the required data is extracted from it for Rancho Cordova. The data was accessed in November 2010 through [http://www.gtfs-data-exchange.com](http://www.gtfs-data-exchange.com) and 2011 transit service was used for the network preparation.

The transit routes serving Rancho Cordova are bus routes # 21, 28, 72, 74, and 75, and a part of LRT route number 507 (Gold line). Then by mapping the GTFS routes in GIS and comparing with the auto network and the area boundary, the transit routes were cut to keep the segments inside the area only. In the same way, transit stops inside the area were selected. Since the transit network is used both in DynusT (for transit vehicle simulation in congested traffic network), and FAST-TrIPs (for assignment and simulation of transit passengers), transit routes had to be coded into the DynusT network. In other words, the path for each route (including its directions and variations) were found, and stops were mapped into DynusT links, so that DynusT could read the transit network, generate buses like a type of vehicle, and simulate them in the model. This process were done in a semi-automatic way, so that paths were found by mapping GTFS shape files (containing geographic information about routes and stops) into the auto network, and finding the best path that can represent the transit route. Obviously, the results were not perfect and many checks and corrections were done manually to accommodate transit routes in the DTA network (e.g., adding some missing links in the DynusT model, on where transit vehicles travel). At the same time, the GTFS data, including stops, routes, trips, and stop-times (detailed schedule of transit vehicle trips) were prepared in appropriate formats to be read by FAST-TrIPs.