Data Structure and Workflow of DTALite/NeXTA

Table of Context

1. Introduction

1.1. Motivation

Motivated by a wide range of application needs, such as region-wide emissions analysis and route guidance provision, dynamic traffic assignment models have been increasingly recognized as an important tool for assessing operational performances of those applications at different spatial resolutions (e.g., network, corridor and individual segment levels) and across various analysis temporal regimes (e.g., second-by-second, peak hours, entire day and multiple days). The advances of DTA in this aspect are built upon the capabilities of DTA models in describing the formation, propagation and dissipation of traffic congestion in a transportation network. Planning practitioners have recognized the full potential of DTA modeling methodologies that describe the propagation and dissipation of system congestion with time-dependent trip demands in a transportation network. In April 2009, the TRB Network Modeling Committee conducted a DTA user survey through the FHWA TMIP mail list, which indicated that more than 70% of the 85 respondents plan to apply DTA tools within 2 years. On the other hand, they also clearly identified the following top 5 technical and institutional barriers:

- DTA requires more data than are available or accessible to most users (47%)
- Setting up a DTA model consumed inordinate resource (44%)
- Cost/benefit of implementation is unclear (45%)
- DTA tools take too long to run (35%)
- The underlying modeling approaches are not transparent (35%)

Emerging multi-core computer processor techniques are offering unprecedented available parallel computing power, on most of laptops and desktops currently available in the market. To exploit this paradigm change in computing will require a new software architecture and algorithm design so as to facilitate the most efficient use of emergent parallel hardware. Designed for and implemented on serial single-core processor architecture, most existing DTA software packages have no or very limited distributed computing capabilities.

With both open-source traffic simulation engine and graphic user interface (GUI), the software suite of DTALite + NEXTA aims to
(i) provide an **open-source code base** to enable transportation researchers and software developers to expand its range of capabilities to various traffic management analysis applications.
(ii) present results to other users by **visualizing time-varying traffic flow dynamics** and traveler route choice behavior in an integrated 2D/3D environment.
(iii) provide a **free education tool** for students to understand the complex decision-making process in transportation planning and optimization processes.

1.2. System Architecture

In general, a dynamic traffic simulation-assignment tool needs to read time-dependent origin-destination (OD) demand matrices and then assign vehicles to different paths based upon dynamic link travel time. As shown in Fig. 1.1, The physical process of moving vehicles through a transportation network works in a two-step process. First, a vehicle is moved across a link by a **link traversal** model, and then moved between links at the node by a **node transfer** function. In particular, the link traversal model typically involves speed-density relationship and hard outflow capacity constraints, determined by the number of lanes and link type. The node transfer model involves specific left-turn or through movement capacity determined by green time allocation at signalized nodes or other attributes. Link travel times from the traffic simulator are fed to the **time-dependent shortest path** model for path selection through a certain **route choice utility function** or traffic assignment rules. The new paths are then fed back into the traffic simulator in the next iteration.

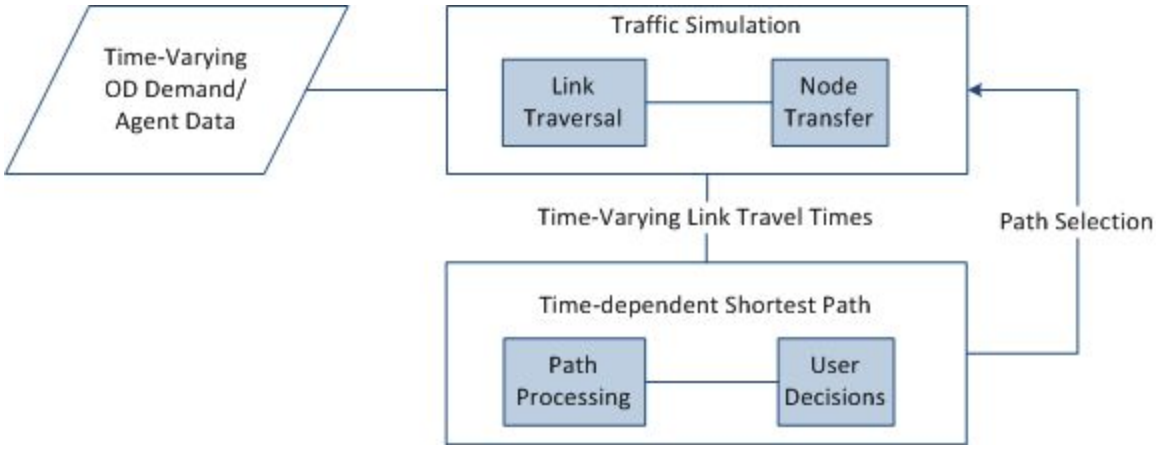


Figure 1.1. Typical Simulation-based Dynamic Traffic Assignment Modelling Framework

The software architecture of DTALite aims to integrate many rich modeling and visualization capabilities into an open-source dynamic traffic assignment model suitable for practical everyday use within the context of an entire large-scale metropolitan area network. Using a modularized design, the open-source suite of **simulation engine + visualization interface** can also serve future needs by enabling transportation researchers and software developers to continue to build upon and expand its range of capabilities. The **streamlined data flow** from static traffic assignment models and common signal data interfaces can allow state DOTs and regional MPOs to rapidly apply the advanced DTA methodology, and further examine the effectiveness of traffic mobility, reliability and safety improvement strategies, individually and in combination, for a large-scale regional network, a subarea or a corridor.

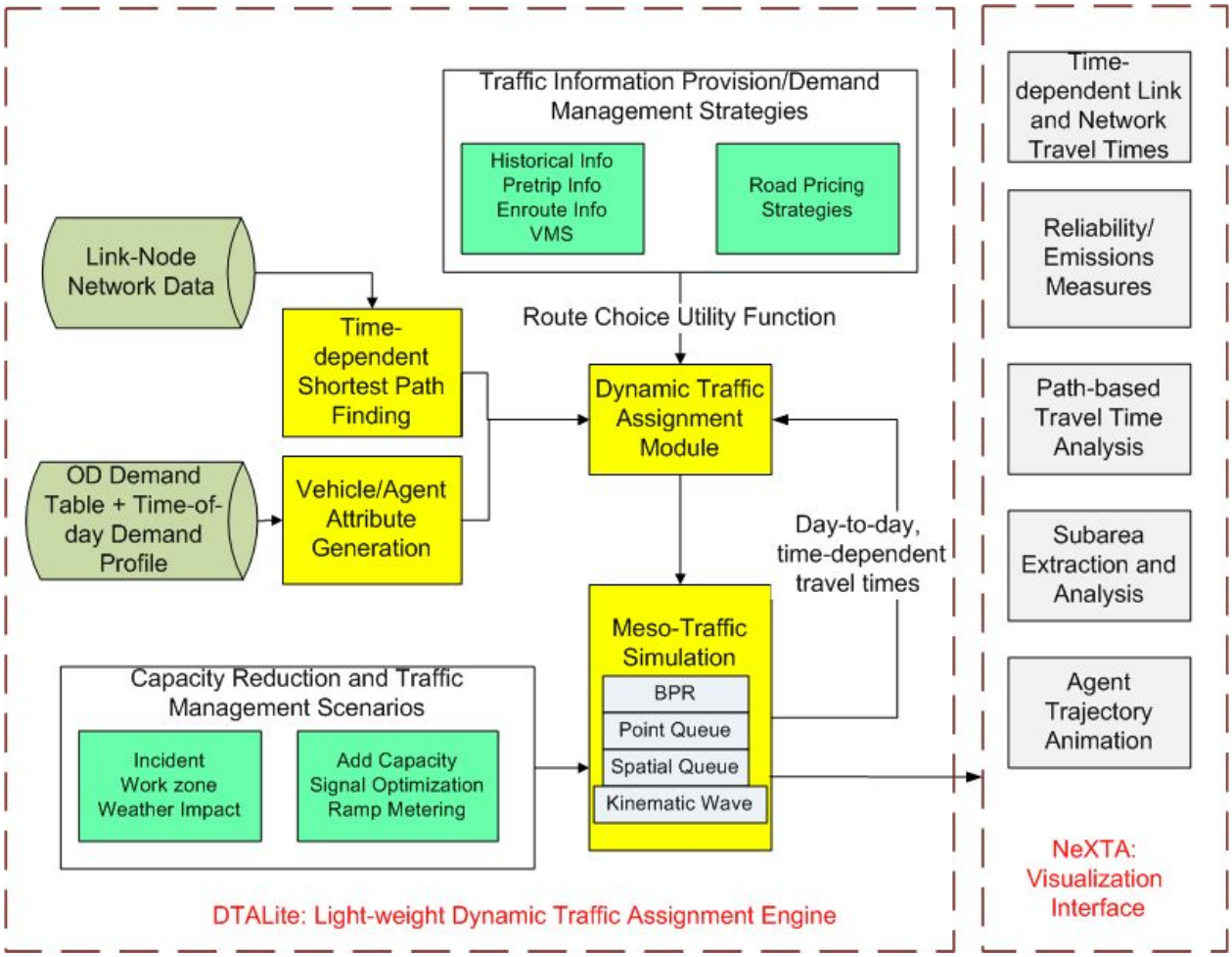


Figure 1.2 Software System Architecture with Key Modeling Components

The overall structure is given in Fig. 1.2, which integrates four major modeling components highlighted in yellow:
(1) **time-dependent shortest path finding** based on a node-link network structure from regional planning models;
(2) **vehicle/agent attribute generation** that converts traditional OD demand matrix in conjunction with additional time-of-day departure time profile as well as possible variable value of time distribution;
(3) **dynamic traffic assignment method** that considers major factors affecting agents' route choice or departure time choice behaviour, such as (i) different types of traveller information supply strategies (e.g. historical information, pre-trip, enroute and on-road variable message signs and (ii) road pricing strategies where the dollar values are converted to generalized travel time through a typical value of time coefficients (e.g. $10 per hour).
(4) a wide range of **traffic flow models** that can take essential road capacity reduction or enhancement measures, such as work zones, incidents and ramp meters.

The traffic assignment and simulation modules are fully integrated and iterated to either capture day-to-day user response or find steady-state equilibrium conditions. Within this simulation-assignment framework, the rich set of output data include traffic MOEs at different spatial and temporal scales, ranging from network, corridor-level and specific links. Typical speed, volume and density measures and agent-based trajectories can be visualized through the NEXTA user interface. The following discussions will focus on how to design flexible and efficient algorithms to model essential traffic dynamics and user behaviour responses.

1.3. Data Flow for Performing Dynamic Traffic Analysis

There are a number of practical challenges in enabling rapid data importing, analysis and exporting for multi-scale dynamic traffic analysis. The typical input data for mesoscopic DTA models include:

Network Data: Existing static traffic assignment network has a few link attributes but very different field naming conversions (e.g. from node, to node vs. A and B); the network topology has been widely coded in GIS format but with very different coordinate systems.

Demand Data: Trip tables at a TAZ level can be imported from regional traffic planning models for different peak and off peak periods, but the corresponding data files are available in many possible formats, such as column-by-column (origin, destination, value) vs. matrix, ASCII text vs. binary files.

Control Data: Signal timing and phasing data are critically needed for typical DTA analysis. While many packages can model pre-timed or actuated signals, very detailed signal timing and phasing data are coded various data format, which is different from commonly used (and very complex) dual-ring structure in a form of Universal Traffic Data Format in SYNCHRO. On the other hand, planners have difficulties to obtain signal timing data for future-year scenarios with forecasted and potentially dramatic changes in traffic volume and turning percentages.

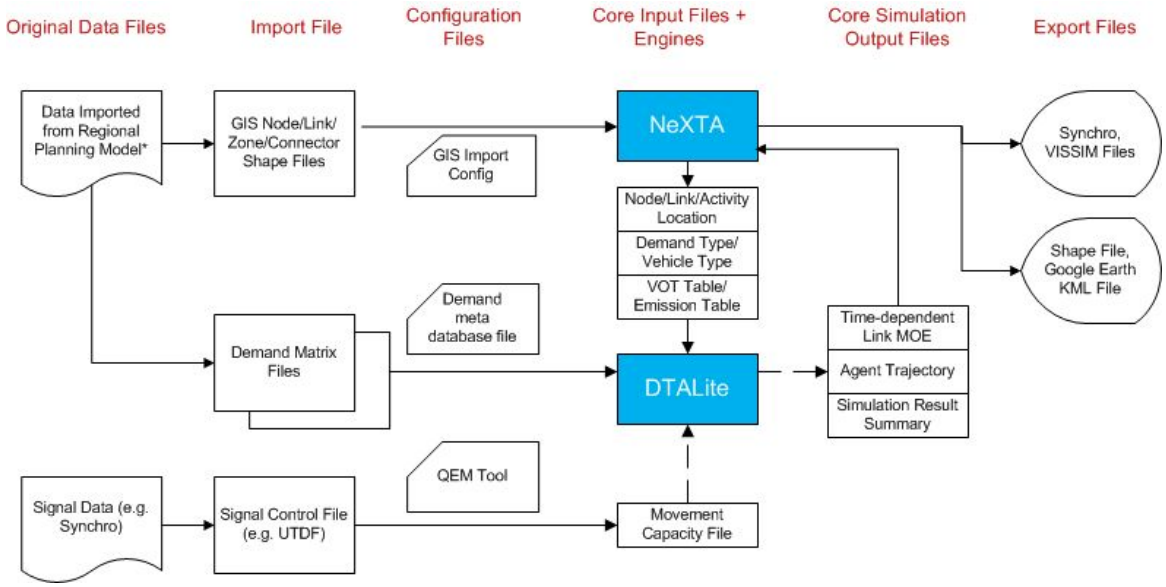


Figure 1.3 Work Flow Chart for Importing, Analyzing and Exporting Data

Fig. 1.3 gives the system importing, analysis and exporting data flow for the bundle of DTALite and NEXTA. In particular, our solution to import GIS data from multiple planning packages is to develop an **open data format** that allows DTALite users to feasibly convert their own data in proprietary format to a unifying data structure and widely used longitude and latitude coordinate system (WGS 84 used by Google Maps, Google Earth and on-line Google Fusion Tables), akin to the Open Document format which creates a standardized file format that allows users to open, edit, and save Microsoft Word documents from other applications such as Google Docs.

In the third step of important data, we use metadata definitions for network and demand files, which can avoid ad-hoc linkages to specific software packages. By integrating a Quick Estimation Method (QEM) as a solid signal control emulator that can provide realistic signal timing and phasing data, we simplify the data preparation process for signalized intersections. The QEM_SIG Excel spreadsheet (available at http://www.learning-transportation.org) is integrated as a computational engine for estimating signalized intersections operation. It relies on the Highway Capacity Manual (HCM) 2010 methodology for signalized intersection analysis and the HCM QEM method, in conjunction with other methodologies for computing parameters of signalized intersections (as described in the Signal Timing Manual (STM), 2008 edition).

Furthermore, the open-source NEXTA package, as a network editing and visualization tool, is made available through this project to display large-scale dynamic traffic analysis results. It plays a critical role for both validating and reporting operational performance at different bottlenecks. As a cross-resolution data conversion utility, NEXTA can export path-based, vehicle-based simulation results across the full range of spatial and temporal resolutions, to typical microscopic simulators such as SYCHRO and VISSIM.

1.4. Simple Introduction to DTALite/NeXTA Software Releases

The latest software release can be downloaded at Google Code. The source code can be downloaded at here with the instruction for compilation.

The basic control and interfaces of NeXTA are described in this user's guide document.

A **working sample project** folder can be found under
Internal_release\sample_data_sets\Salt_Lake_City_West_Jordan - BaseLine
with a step-by-step learning document here.

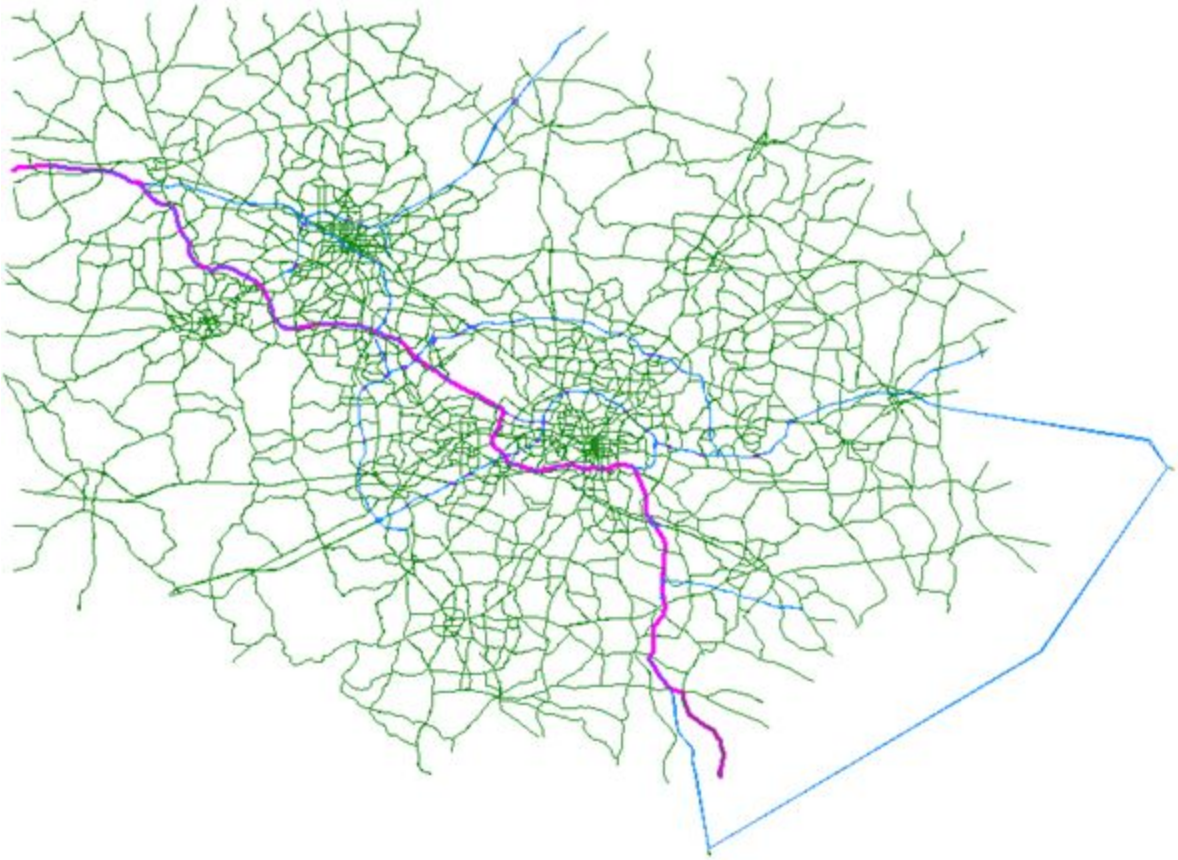
The **software package** includes two software applications: NEXTA as GUI and data hub; DTALite as DTA simulation engine. For each application, there are two versions: 32_bit vs. 64_bit: 32_bit for GIS shape file importing and legacy support for Excel importing.; 64_bit for a large scale network.

There are a number of **subfolders** under internal release folder

- Documentation (for data structure, users guide, QEM tool)

- Default data folder (default data attribute files)
- Sample data sets (real-world test networks)
- Importing sample data sets (GIS files, Excel, Synchro)
- Test data sets (simple networks for testing traffic flow models and other key modelling features)

Hardware requirements. As shown below, a large-scale network provided by the NCSU team has about 2,300 Zones, 9,500 nodes, 20,000 links and 1,900 signals, and 1 million vehicles from 5AM to 10 AM, with 490K household.



The average running time is about 1 hours for 20 user equilibrium iterations on a PC with the following specification.

CPU: Intel i7-2960XM @ 2.70 GHz *8;
Memory: 16.0 GB;
System Type: 64-bit Windows 7

All DTALite **data files** are in CSV format. The files for node, link and zone layers have

geometric fields for importing from and exporting to GIS software, Google Earth, Google Fusion Tables. Each project folder should only store one data set with a project file (*.tnp) that serves as a reference point for locate the other files. DTALite data files have prefix of input_xxx, output_xxx for input and output files. NeXTA can load multiple projects.

List of Internet Resources:

Google Code (for hosting source code, latest release, bug reporting)
https://code.google.com/p/nexta/
Training website (for learning material and user forum)
https://sites.google.com/site/nextadtalitetraining/
www.learning-transportation.org (for hosting general learning material about network modelling)

2. Data Structure of Files

This document describes all input and output files associated with DTAlite/NeXTA package. Each input/output file includes descriptions for required variable names, followed by a short description of their type, purpose, function, interaction with other variables, and the use cases in which the variable is required/not required. Since NeXTA uses DTALite for transportation network analysis, not all variables required as inputs to DTALite are required as inputs for visualization in NeXTA, and not all variables required as inputs to NeXTA are required as inputs to DTALite.

Dynamic traffic assignment encompasses many problem formulation and representation methods, for example, trip-based vs. activity-based, fixed departure time vs. flexible departure time. As the DTA models expand to include greater behavioral and policy realism, the complexity of the model data structure increases. To reach the right balance between the representation details and required data preparation efforts, DTALite adopts an **agent-based, simulation-based mesoscopic dynamic traffic assignment framework** using time-dependent origin-destination matrices. Overall, it is a link-based simulation with capacity constraints. The capacity constraints make calibrating network capacities a very important step in the process of building our models. On the other hand, this modeling framework can be flexibly enhanced to meet specific modeling needs. For example, one advanced user can embed alternative traffic flow models to generate high-fidelity simulation results, or allow traveler-specific value of time attributes to recognize the heterogeneity of the network users in road pricing applications. Fig. 8 shows different groups of data files, namely network, demand and scenario input as well as optional data files describing signal control and movement capacity.

Network data structure defines the basic node-link structure used in DTALite and NeXTA, along with attributes for each link and node. Additionally, nodes are related to zones and activity locations, which can be used to disaggregate trips from zones to nodes and activity locations.

Demand/agent data structure describes the number of trips between zones or nodes. Through a demand meta data file, there are many different ways to define the time-dependent demand inputs: 1) Demand table with starting time and ending time, 2) Demand table with time-dependent 15-min departure time profile, and 3) Input vehicle file. Methods 1 and 2 will generate vehicles in the network based on the time period information provided. The vehicle data file describes all vehicle trips in the network, allowing the user to provide very detailed trip information, such as vehicle type, traveller information type, value of time, vehicle age as well as specific path sequences.

Signal data is related to traffic signal control (movement and phasing tables), and **sensor input data** for origin destination matrix estimation.

Below is a short list of **key features** for DTALite/NeXTA data files and data structure.

- Different layers: different files: node, link, zone, activity locations
- Many model attribute files: node control type, link type, demand type and vehicle type
- Time representation: 24 hour (for demand and sensor data), day number= iteration number, work zone has day attributes (for modelling day-to-day learning)
- Demand meta database file: “Dynamic demand data manager”, read multiple demand files, in different format: column, matrix, agent file, DYNASMART file, different demand loading periods, additional departure time profile
- Scenario setting file: traffic flow model, traffic assignment model, scenario number for multiple scenario runs
- Scenario files for advanced modelling features: work zone, incident, tolling, VMS files
- Sensor data file: for model validation and calibration, different time period
- Output files: simulation summary, network MOE, link MOE, trajectory file

Group I: Network files

High-level introductions:

- **A generic network used for DTALite and NeXTA includes a set of four layers: node, link, activity location and zones.**
- **The specific file names are input_node.csv, input_link.csv, input_zone.csv and input_activity_location.csv.**
- **A node has a certain control type such as pretimed signals or 4-way stop signs. A cycle length is required for signalized intersections/nodes.**
- **A link is defined using upstream node and downstream node ids, with essential attributes such as length, speed limit, number of lanes and capacity, typically required for static traffic assignment and meso-scopic traffic assignment.**
- **A zone can be a typical Traffic Analysis Zone (TAZ) that contains one or multiple centroids. The zone number is the index used in the OD demand table. The geometric field for a zone is not required for running DTA simulation, especially for external zones/stations.**
- **The activity location** file serves as a mapping layer that maps zones to individual

nodes. Activity locations are a special case of nodes for generating and attracting trip demand. In DTALite's agent-based simulation method, an agent is a vehicle and it is generated based on an OD demand table from zone i to zone j with a certain vehicle type. An agent's path is computed from one activity location in zone i to another activity location in zone j. One activity location can be a centroid (typically used in traditional static traffic assignment) or a physical node in the urban network.

- The node and link layers can use arbitrary **coordinate system**, but a WGS 84 (long/lat) coordinate system is preferred to export data to Google Earth/Google Map.
- The **geometric attribute** of node, link and zone layers is coded in a field of "geometry" for individual records. The "geometry" field uses the format required by Google Fusion Tables, so one can easily import node, link and zone data to Google Earth/Google Maps.
- NeXTA provides a number of methods to **import network data** from GIS planning packages in generic shape file format, Excel files, Openstreetmap OSM format and Synchro UTDF format.
- A user can also manually **create a new network** from the scratch by using a click-and-draw method based on a background image file.
- There are also **optional network layers** such as subarea, movement and path,which are mainly used in specific applications, but not required for running basic DTA simulation.

input_node.csv

Field Name	Description	Sample Value
name	Optional for visualization only	Main street @ Highland Dr.
node_id	Node identification number	1001
control_type	Intersection control type, defined in input_node_control_type.csv. Typical range: 1-7. E.g. 5: pretimed signal.	5
control_type_name	The text name corresponding to the control type number in the control_type field. This text is automatically generated based on the numeric value in field control_type when a user saves the network in NeXTA.	pretimed_signal
cycle_length_in_second	The signal cycle length for a specific node, unit: second.	120

signal_offset_in_second	The starting time for a cycle with respect to the global simulation clock. Default value: 0; unit: second.	0
x	Longitude or horizontal coordinate in any arbitrary geographic coordinate system.	100
y	Latitude or vertical coordinate horizontal coordinate in any arbitrary geographic coordinate system	200
geometry	Text string used to describe node location (typically in WGS84 geographic coordinate system)	<Point><coordinates>-111.979958,40.703899</coordinates></Point>

Minimum set of fields: node_id, x and y. The geometry field can be generated from NeXTA based on fields x and y. The control_type (specifically signal location) can be identified by NeXTA through menu->tools->traffic control tools->generate signal locations.

input_link.csv

Field Name	**Description**	**Sample Values**
name	Optional for visualization purposes	Main Street
link_id	Optional link identification number, generated from NeXTA	101
from_node_id	Upstream node number of the link, must already defined in input_node.csv	2
to_node_id	Downstream node number of the link, must already defined in input_node.csv	3
link_type_name	Optional text label for visualization and data checking purposes, generated from NeXTA based on field link_type and name defined in	Minor arterial

	input_link_type.csv	
direction	Identifies the direction of travel on the link. Default value = 1. When -1, NeXTA reverses from_node_id and to_node_id for correct orientation. When 0 or 2, NeXTA automatically converts link into two one-way links.	1
length	The length of the link (between end nodes), measured in units of miles.	1.0
number_of_lanes	The number of lanes on the link	2
speed_limit_in_mph	Speed limit on defined link in units of miles per hour. Unit: mph	20
saturation_flow_rate_in_vhc_per_hour_per_lane	The saturation flow rate is only used when the downstream node is signalized control, when DTA simulator discharges vehicles during a green time period in a cycle. Default value = 2000.	1900
lane_capacity_in_vhc_per_hour	Maximum service flow rate for each lane on the link, in vehicles per hour. This capacity value is not used used for signalized control with a positive cycle length defined. Typical values are 1800 for a freeway lane, 900 for an arterial lane, 1500 for a single on-ramp lane.	1500
link_type	Link type identification number, corresponding to link functional class (freeway, ramps), must be defined in input_link_type.csv	7
jam_density_in_vhc_pmpl	Jam density (in vehicles per mile per lane), input for two traffic flow models (spatial queue and Newell's simplified kinematic wave model). Unit: number of vehicles per mile per lane: default value 180 for freeway	180

	links, 190 for arterial street links.	
wave_speed_in_mph	Backward wave speed in miles per hour, only used in Newell's simplified kinematic wave model to define the vehicle storage space on a link: Default = 12 mph	12
effective_green_time_length_in_second	It is only used when the downstream node is signalized control. unit: second.	60
green_start_time_in_second		Value ≥ 0
AADT_conversion_factor		Default = 0.1
mode_code	Optional: Indicates which types of traffic (transit, pedestrian, car, etc.) can use a link	
grade	Optional: Roadway grade. default value = 0	0
geometry	Text string used to describe link shape and location (typically in WGS84 geographic coordinate system). The initial value can be empty, and NeXTA will generate the text string based on the coordinates of upstream and downstream nodes.	<LineString><coordinates>4165.673828,23656.343750,0.0 5207.092773,23656.343750,0.0</coordinates></LineString>
original_geometry	Optional text string used to describe link shape and location. This field is typically imported from the original GIS link layer before offsetting the links in NeXTA. That is, the coordinates in field geometry will be affected when offsetting two directed links in opposite directions.	<LineString><coordinates>4165.673828,23656.343750,0.0 5207.092773,23656.343750,0.0</coordinates></LineString>
transit_travel_time_in_min transit_transfer_time_in_min transit_waiting_time_in_min transit_fare_in_dollar	These optional fields are only used for transit assignment	

BPR_alpha_term BPR_beta_term	These optional fields are only used for static traffic assignment. Typical values are 0.15, 4.	
number_of_left_turn_bays	Optional: The number of left turn bays on the link. NeXTA uses the value here to overwrite the number of lanes for left-turn movement in AMS_movement.csv file.	1
length_of_bays_in_feet	Optional: Length of the left turn bays on the link, in units of feet. Not used in the current version of DTALite	200
number_of_right_turn_lanes	Optional: Not used in the current version of DTALite	0
from_approach	Optional: Indicates the direction from which vehicles enter the link. Generated by NeXTA for microscopic simulation (e.g., VISSIM). Typical values: N = North, S = South, E = East, W = West	N
to_approach	Optional: Indicates the direction in which vehicles leave the link, which is the opposite direction of from_approach. Generated by NeXTA for microscopic simulation (e.g., VISSIM): Typical value: N = North, S = South, E = East, W = West	S
reversed_link_id	Optional: Identifies the link ID for the link between the same two nodes, but with opposite travel direction. Generated by NeXTA for microscopic simulation (e.g., VISSIM)	102

Remarks:

Requirements: the from_node_id, to_node_id fields must be defined in input_node.csv. If the

downstream node is a signalized intersection, the effective_green_time_length_in_second field is required.

Minimum required set of fields: from_node_id, to_node_id, length_in_mile, number_of_lanes, speed_limit_in_mph, lane_capacity_in_vhc_per_hour, link_type.

Fields can be generated or populated by NeXTA:
geometry fields can be imported from GIS shape files or generated based on the coordinates of upstream and downstream nodes. direction = 1 by default. saturation_flow_rate_in_vhc_per_hour_per_lane =2000 by default. jam_density_in_vhc_pmpl is 180 by default for freeway links, 190 for arterial links.
wave_speed_in_mph is 12 by default.
effective_green_time_length_in_second can be computed as cycle length*lane_capacity/saturation flow rate. E.g. 120 second cycle length * 900 lane capacity / 1800 saturation flow rate = 60 as effective green time.

input_zone.csv

Field Name	Description	Sample Value
zone_id	Zone identification number. A zone number used in OD demand table files must be defined here first. Zone numbers are not required to be consecutively sequential.	1001
production	Reserved for trip distribution step and not used in DTALite.	
attraction	Reserved for trip distribution step and not used in DTALite.	
geometry	Optional: Text string used to describe zone location for visualization (typically in WGS84 geographic coordinate system).	"<Polygon><outerBoundaryIs><LinearRing><coordinates>-111.907241,40.802401,0.0 -111.870550,40.789266,0.0 -111.822437,40.768850,0.0 -111.907241,40.802401,0.0<coordinates> </LinearRing></outerBoundaryIs></Polygon>"

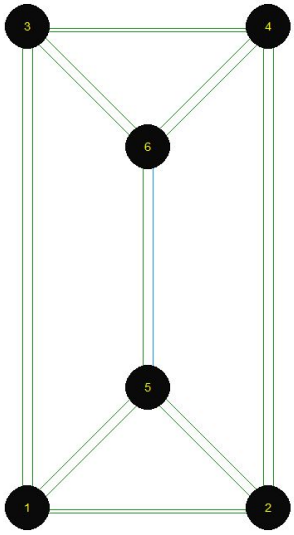
input_activity_location.csv

Field Name	Description	Sample Value

zone_id	Zone identification number	1001
node_id	Node identification number associated with the zone identification number in the same row. Must defined in input_node.csv.	1001
external_OD_flag	Used to identify the type of activity location as non-external (default = 0) or external (-1 or 1). When 0, acts as both origin and destination.	0

Remarks:
Requirements: A zone used in OD demand tables (with a positive number of trips generated) must contain at least one node as it activity locations. node_id must be defined in input_node.csv.

Example Files for 6-node network



input_node.csv

name	node_id	control_ty	control_ty	cycle_leng	signal_off	x	y	geometry
	1	0	unknown_	0	0	0	0	
	2	0	unknown_	0	0	1.8314	0	
	3	0	unknown_	0	0	0	3.6628	
	4	0	unknown_	0	0	1.8314	3.6628	
	5	0	unknown_	0	0	0.9157	0.9157	
	6	0	unknown_	0	0	0.9157	2.7471	

input_link.csv

name	link_id	from_nod	to_node_i	direction	length_in	number_c	speed_lin	saturation_flow_ra
(null)	0	1	2	1	2	2	35	2000
(null)	0	1	3	1	4	2	45	2000
(null)	0	1	5	1	1	4	35	2000
(null)	0	2	1	1	2	2	35	2000
(null)	0	2	4	1	4	2	45	2000
(null)	0	2	5	1	1	2	35	2000
(null)	0	3	1	1	4	2	45	2000
(null)	0	3	4	1	2	2	35	2000
(null)	0	3	6	1	1	2	35	2000
(null)	0	4	2	1	4	2	45	2000
(null)	0	4	3	1	2	2	35	2000

link_type	jam_dens	wave_spe	mode_code	grade	geometry
4	120	12			<LineString><coordinates>0.000000,-0.015151,0.0 1.831400,
2	120	12			<LineString><coordinates>0.015151,0.000000,0.0 0.015151,3
4	120	12			<LineString><coordinates>0.010714,-0.010714,0.0 0.926414,
4	120	12			<LineString><coordinates>1.831400,0.015151,0.0 -0.000000,
2	120	12			<LineString><coordinates>1.846551,0.000000,0.0 1.846551,3
4	120	12			<LineString><coordinates>1.842114,0.010714,0.0 0.926414,0
2	120	12			<LineString><coordinates>-0.015151,3.662800,0.0 -0.015151
4	120	12			<LineString><coordinates>0.000000,3.647649,0.0 1.831400,3
4	120	12			<LineString><coordinates>-0.010714,3.652086,0.0 0.904986,
2	120	12			<LineString><coordinates>1.816249,3.662800,0.0 1.816249,-
4	120	12			<LineString><coordinates>1.831400,3.677951,0.0 -0.000000,

input_zone.csv

zone_id	productio	attraction	geometry
1			<Polygon><outerBoundaryIs><LinearRing><coordinates>-0.59300,0.66112,0.0 0.3240
2			<Polygon><outerBoundaryIs><LinearRing><coordinates>1.30652,0.39257,0.0 2.29558
3			<Polygon><outerBoundaryIs><LinearRing><coordinates>-0.59300,4.06716,0.0 0.4746
4			<Polygon><outerBoundaryIs><LinearRing><coordinates>1.03797,4.19161,0.0 2.36108

input_activity_location.csv

zone_id	node_id	external_OD_flag
1	1	0
2	2	0
3	3	0
4	4	0

Group II: Advanced type definition files (default file provided)

High-level introductions:

- DTALite and NeXTA aim to accommodate and preserve **flexible data types** defined by users. We do not impose limitations on the number of link types, demand types, and users can use their existing link type, demand type as is to maintain a seamless connection from their existing static planning models to the new DTA tools.
- **Typical definitional files** include input_node_control_type.csv, input_link_type.csv, input_pricing_type.csv, input_demand_type.csv.
- Definitional files for **vehicle emission modeling** include input_vehicle_type.csv, input_cycle_emission_factor.csv, input_vehicle_emission_rate.csv, input_base_cycle_fraction_of_OpMode.csv. Those files are provided by Dr. Chris Frey's team at NCSU as part of MOVES Lite.
- The definitional file for **agent-based road pricing modeling** includes input_VOT.csv.

- The definitional file for **safety planning** include input_crash_prediction_model.csv.
- Each definitional file has its **default file** stored in folder Internal_release\default_data_folder. When creating a new data set, a user does not need to create those files manually and NeXTA can fetch all default files to the current project folder automatically.
- A number of **mapping fields** are provided to relate the user-defined type to the essential data type used in DTALite. E.g. link type 100 is mapped to the freeway type in DTALite through a code of f . A demand type of 3 is mapped to the truck type in DTALite as pricing type of 2.
- DTALite performs an agent-based routing for road pricing applications. with the following individual generalized cost function. To create **discrete VOT distribution** quickly, the VOT value in file input_VOT.csv can be set as 50% of hourly rate (Concas and Kolpakov, 2009), based on an easy-to-obtain income distribution per trip purpose.

$$Cost = Travel\ Time * VOT + Toll$$

input_node_control_type.csv

The input_node_control_type table defines the control type of nodes in the network in terms of control type name, unknown control, no control, yield sign, 2way stop sign, 4way stop sign, pretimed signal, actuated signal and roundabout. This file is required when using the network import tool, and the control type field is read from the node shape file.

control_ty	unknown_	no_contrc	yield_sign	2way_stop	4way_stop	pretimed_	actuated_	roundabout
control_ty	0	1	2	3	4	5	6	100

input_link_type.csv

The input_link_type table allows users to define their own specific link types, as long as the flag variables are correctly used to identify how the different link types are connected/related (e.g., freeways connect to arterials using ramps). Only one flag may be used for each link type. Link types can also be used to determine how links are visualized in NeXTA. This file is required when using the network import tool to interpret the link type field in the link shape file.

Field Name	Description	Sample Value
link_type	Link type identification number	100
link_type_name	Optional: Name label assigned to link type in the same row, used for visualization purposes in NeXTA	freeway

type_code	A text character which identifies which type of link is mapped to the link type identification number. f = freeway, h = highway/expressway, a = arterial, c = connector, r = ramp, t = transit, w = walk	f
default_lane_capacity	According to the link type, the lane capacity assigned by default to new links created in NeXTA.	1900
default_speed_limit	According to the link type,the speed limit assigned by default to new links created in NeXTA.	60
default_number_of_lanes	According to the link type, the new of lanes assigned by default to new links created in NeXTA.	3

default link type file:

link_type	link_type_name	type_code	default_lane_capacity	default_speed_limit	default_number_of_lanes
1	Freeway	f	1800	65	3
2	Highway/Expressway	h	1450	50	3
3	Principal arterial	a	1000	40	3
4	Major arterial	a	900	35	3
5	Minor arterial	a	850	30	2
6	Collector	a	650	25	1
7	Local	a	600	20	1
8	Frontage road	a	1000	45	2
9	Ramp	r	1300	30	2
10	Zonal connector	c	2000	100	2
100	Transit link	t	1000	40	1
200	Walking link	w	1000	5	1

input_pricing_type.csv

The input_pricing_type.csv file defines the pricing type in the assignment for road pricing applications.

Field Name	**Description**
pricing_type	Pricing type id. DTALite uses the following convention: 1: single occupancy vehicle, 2: high occupancy vehicle, 3: truck, 4: intermodal travelers using transit. Please do not change this order.

pricing_type_name	Pricing type name
default_VOT	Default value of time for a given pricing type. This value is not used when the file input_VOT.csv is provided, which contains more detailed distributions for VOT.

default pricing type file:

pricing_type	default_VOT
1	10
2	20
3	30
4	10

input_demand_type.csv

The input_demand_type table is used to define the characteristics for different demand types for the trips in demand files such as input_demand.csv. There are three different demand types by default (1 = SOV, 2 = HOV, 3 = Trucks), but additional types can be defined in the table (e.g., trip purpose – HBW, HBO, etc.).

Field Name	**Description**	**Sample Value**
demand_type	Demand type identification number	1
demand_type_name	Optional: Name label assigned to demand type in the same row, used for visualization purposes in NeXTA	SOV
average_VOT	Average Value of Time (in units of dollars/hour) assigned to the demand type in the same row. This value is not used when the file input_VOT.csv is provided, which contains more detailed distributions for VOT.	10
pricing_type	Pricing type identification number, only used for tolling applications. See definitions for input_pricing_type.csv	1
percentage_of_pretrip_info	Percentage of vehicles with pre-trip travel time information. Affects routing behavior in	5

	DTALite.	
percentage_of_enroute_info	Percentage of vehicles with en-route travel time information. Affects routing behavior in DTALite. Drivers with historical information = 100 – percentage_of_pretrip_info – percentage_of_enroute_info.	5
percentage_of_vehicle_type1	Percentage of vehicles of vehicle type 1 for the demand type in the same row. Percentages in row should sum to 100.	80
percentage_of_vehicle_type2	Percentage of vehicles of vehicle type 2 for the demand type in the same row. Percentages in row should sum to 100.	20
percentage_of_vehicle_type#	Additional columns (with incremental #) can be used when more vehicle types are defined.	0

Default file:

demand_t	demand_t	average_\	pricing_ty	percentag	percentag	percentag	percentag	percentag	percentag	percentage_of_vehicle_type5
1	SOV	10	1	0	0	80	20	0	0	0
2	HOV	10	2	0	0	80	20	0	0	0
3	truck	20	3	0	0	0	0	30	30	40

input_vehicle_type.csv

The input_vehicle_type table is used to define different vehicle types for emissions analysis.

Field Name	**Description**
vehicle_type	Vehicle type identification number
vehicle_type_name	Name label assigned to vehicle type in the same row
rolling_term_a rotating_term_b drag_term_c source_mass	Fields used for calculating vehicle-specific power (VSP)
percentage_of_age_0 percentage_of_age_5 percentage_of_age_10	Percentage of vehicles in specific age group

percentage_of_age_15	

Default file:

vehicle_type	vehicle_type_name	rolling_term_a	rotating_term_b	drag_term_c	source_mass	percentage_of_age_0	percentage_of_age_5	percentage_of_age_10	percentage_of_age_15
1	passenger car	0.156461	0.00200193	0.000492646	1.4788	5.9059	47.9479	28.3283	17.8179
2	passenger truck	0.22112	0.00283757	0.000698282	1.86686	2.9949	43.372	26.3805	27.2526
3	light commercial truck	0.235008	0.00303859	0.000747753	2.05979	3.0918	44.2272	25.9693	26.7117
4	single unit short-haul truck	0.561933	0	0.00160302	7.64159	3.994	51.9218	23.4647	20.6195
5	combination long-haul truck	2.08126	0	0.00418844	31.4038	3.9991	51.9888	23.495	20.5171

The data structure of files input_crash_prediction_model.csv, input_cycle_emission_factor.csv, input_vehicle_emission_rate.csv, input_base_cycle_fraction_of_OpMode.csv is not listed here, as it should be kept with the default values.
Sample files:
input_vehicle_emission_rate.csv

vehicle_ty	opModeID	meanBase	meanBase	meanBase	meanBase	meanBaseRate_HC_(g/hr)
1	0	68371.1	4913.603	0.05385	2.36609	0.039171
1	1	52728.1	3789.393	0.008979	4.05557	0.000418
1	11	85288.1	6129.372	0.146868	6.52187	0.022892
1	12	106704	7668.461	0.155233	2.82379	0.02085
1	13	153460	11028.66	0.363034	9.76815	0.052262
1	14	194390	13970.16	0.657844	14.2137	0.072532
1	15	234348	16841.81	1.18797	20.8813	0.103686

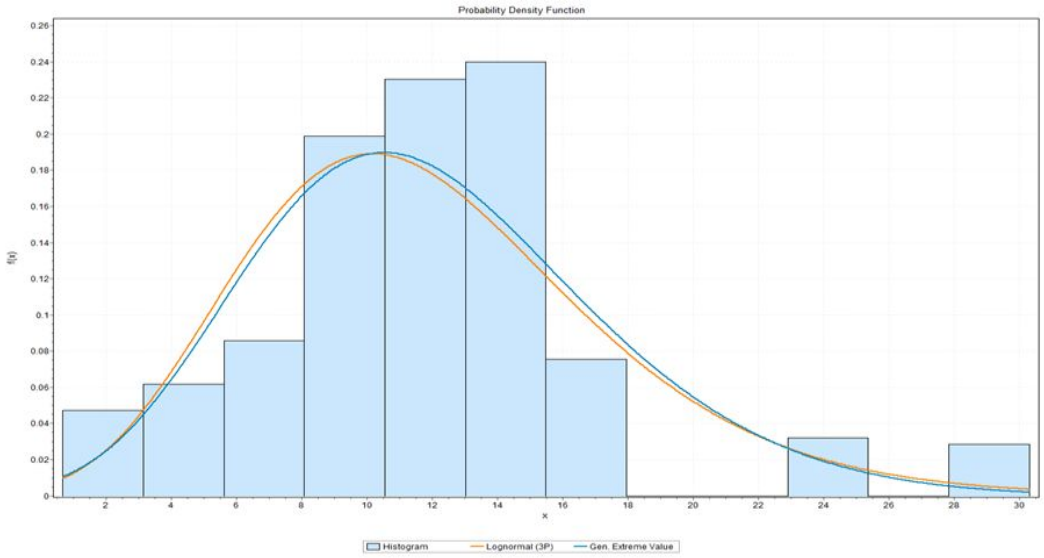
input_VOT.csv

Field Name	**Description**	**Sample value**
demand_type	Demand type identification number	1
VOT_dollar_per_hour	Value of Time in dollars/hour	20
percentage	Defines the percentage of travelers in of a specified demand type with a specified VOT. Used to describe the VOT distribution for the demand type in the same row.	2

Sample file:

demand_type	VOT_dollar_per_hour	percentage
1	0	2.749
1	5	12.206
1	10	10.007
1	15	12.619
1	20	1.278
1	25	0
1	30	1.141
1	0	0.687
1	10	3.052
1	20	2.502
1	30	3.155
1	40	0.32
1	50	0
1	60	0.285
1	0	3.436
1	10	15.258
1	20	12.509
1	30	15.773
1	40	1.598
1	50	0
1	60	1.426
2	0	2.749
2	5	12.206
2	10	10.007
2	15	12.619
2	20	1.278
2	25	0

The above data defines a discrete distribution of VOT shown in the following figure.



Group III: Configuration files

High-level introductions:

- DTALite and NeXTA aim to accommodate and preserve **flexible demand file format** used by users. DTALite can use the demand file as is.
- DTALite accepts **multiple demand file types,** such as 3-column, NXN matrix, agent-base file, DYNASMART format
- Through a very flexible **demand meta database** configuration file, a user only needs to provide specific information to map demand type and demand loading time periods from their existing demand table to the new DTA tools.
- To create **time-dependent demand loading patterns, a user can define loading ratio for each 15 min interval.**
- **DTALite uses a 24-hour demand horizon /representation to facilitate 24-hour dynamic traffic assignment and compare simulated results with sensor data easily (defined in the format of 24-hour horizon)**
- **DTALite allows continuous runs of multiple scenarios to compare user-defined MOEs quickly.**
- **Typical configuration files include meta demand database, scenario setting, MOE settings.**

input_demand_meta_data.csv

The input_demand_meta data table is used to define the characteristics of demand data. Through a temporal demand profile table per record, users can define the proportion of demand in the network as a function of time, which is used to initiate trips in the simulation over the modeling horizon. This table can be used to supplement demand type information in an input

demand table, where DTALite will use the temporal demand profile information in place of other time information.

This meta-data file requires several entries, but the relevant entries are as explained below:

1. Specify the demand file name (e.g., sov_14_15.mtx) and format (e.g., column)
2. Specify the number of lines in the demand file to be skipped by NeXTA (8 for MTX file)
3. Indicate whether subtotals are present in the last column (zero for none)
4. Specify the loading start time and end time for the demand file (840 to 900, or 2PM to 3PM)
5. Specify the demand types associated with the demand file (only demand_type1)

Field Name	**Acceptable Values**	**Description**	**Sample Value**
scenario_no	Value ≥ 0	Scenario identification number	1
file_sequence_no	Value > 0	File identification number	1
file_name	demand.dat	Name of demand file	
format_type	column, matrix, dynasmart, agent_csv, agent_bin	Input file format type	
number_of_lines_ to_be_skipped	Value ≥ 0	The number of lines to be skipped at the beginning of demand file	0
loading_multiplier	Value > 0	Local multiplication factor applied to the number of trips in the demand file	
start_time_in_min	0 to 1440	Demand loading start time, which is the time gap in min from 0:00	
end_time_in_min	0 to 1440	Demand loading end time, which is the time gap in min from 0:00	

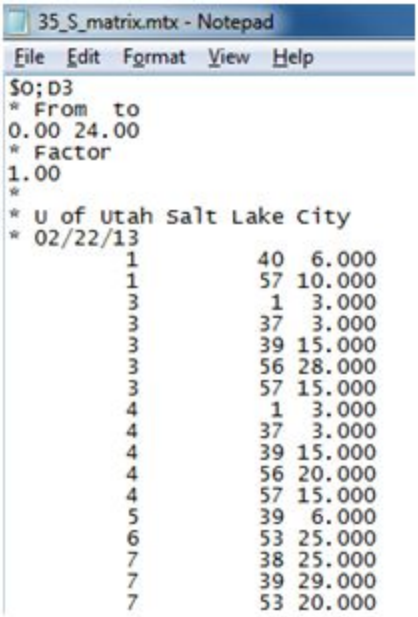
apply_additional_ time_dependent_ profile	0 or 1	0: not use the time dependent profile in this table, that is, a flat demand pattern is used between time period [start_time_in_min, end_time_in_min] 1: use the time dependent profile in this table	
subtotal_in_last_ column	0 or 1	flag used for subtotal in last column of matrix demand file	
number_of_ demand_types	Value ≥ 1	Number of demand types stored in demand file	
demand_type_in_3rd_c olumn	default value 0 1: a demand type is specified for each record	demand type for format: origin, destination, demand value, value	2,100,1, 10 2,100,2, 20 zone 2 to zone 100, 10 vehicles for demand type 1 zone 2 to zone 100, 20 vehicles for demand type 2
demand_type_1	Value ≥ 1		
demand_type_2	Value ≥ 1		
demand_type_3	Value ≥ 1		
demand_type_4	Value ≥ 1		
'00:00	0 to 1	Proportion of demand in specified time interval compared to 24-hour time period	
'00:15	0 to 1	Proportion of demand in specified time interval compared to 24-hour time period	
…			

'23:45	0 to 1	Proportion of demand in specified time interval compared to 24-hour time period	

Remarks:
We first show a few examples to show how to configure the following key parameters for different demand files.
format_type: subtotal_in_last_column; number_of_demand_types

Example 1: column format

Below is an OD demand file titled "35_S_matrix.mtx" from VISUM for both LOV and HOV demand types during 6AM and 11AM. This combined demand matrix has 90% LOV and 10% HOV.



The corresponding meta data configuration is shown at the following table. The main data block has a three-column format, and we need to skip the first 8 lines of comments. The first line after the comments reads combined demand 6.0 from zone 1 to zone 40.
As there are two demand types, we use two records with the same file name of "35_S_matrix.mtx", but with different loading_multiplier = 0.9 and 0.1 for different demand_type_1 = 1 (LOV for the first record) and demand_type_1 = 2 (HOV for the second record). The start_time_in_min = 360 corresponds to 6AM and the end time of 640 refers to 11AM.
demand_type_x means the x th field after the row and column indices in the three-column or multi-column format. DTAlite will have 0.9*6.0 = 5.4 LOV vehicles, and 0.1*6.0 for 0.6 HOV vehicles from zone 1 to zone 2. The 5.4 vehicles will be randomly rounded up or down according to a uniform

distribution to convert to an integer number (5 or 6 vehicles) in the final simulation process.

file_name	format_type	number_of_lines_to_be_skipped	loading_multiplier	start_time_in_min	end_time_in_min	number_of_demand_types	demand_type_1
35_S_matrix.mtx	matrix	8	0.9	360	540	1	1
35_S_matrix.mtx	matrix	8	0.1	360	540	1	2

Example 2.1: matrix format

Below is an OD demand matrix file titled "demand_matrix.csv" for 4 zones (1,2,3 and 40), and the demand horizon is 7AM to 9AM. A users wants to apply time-dependent departure time profile with a relatively lower demand volume between 7AM and 8AM.

```
TOT   1     2     3     40
1     0     1000  10    10    1020
2     10    0     10    10    30
3     10    10    0     10    30
40    10    10    10    0     30
```

The corresponding input_demand_meta_data.csv is shown below.

file_name	format_type	number_of_lines_to_be_skipped	loading_multiplier	start_time_in_min	end_time_in_min	apply_additional_time_dependent_profile	subtotal_in_last_column	number_of_demand_types	demand_type_1
demand_matrix.csv	matrix	0	2	420	540	1	1	1	1

'07:00	'07:15	'07:30	'07:45	'08:00	'08:15	'08:30	'08:45
0.2	0.2	0.2	0.2	0.3	0.3	0.3	0.3

We offer the following interpretation.

1. file_name: "demand_matrix.csv" is the OD table for the AM period being brought into DTALite/NeXTA.

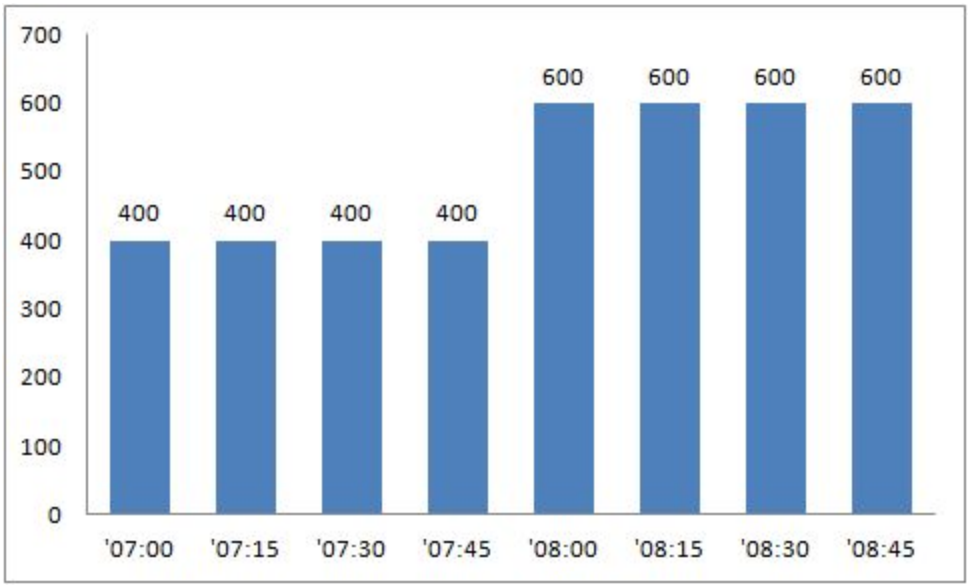
2. format_type: "matrix" is the type of OD table format for this small test application. DTALite

will skip the text string 'TOT' as it reads only numerical values.

3. Start_time is minute 420 or 7 AM and the End_time is minute 540 or 9 AM. The OD table stores number of trips between 7 and 9 AM. loading_multiplier is 2, so a multiplier of 2 is applied to each row. That is, the total demand from zone 1 to zone 2 is 1000*2 =2000.

4. subtotal_in_last_column = 1 as there an origin-based summation for each zone in the matrix.

5. As field apply_additional_time_dependent_profile is set to 1, then a time-dependent departure time profile will be read and result in the following demand loading pattern. That is, the demand from 7:00AM to 7:15 AM from zone 1 to zone 2 has a demand volume of 1000*2*0.2 = 400. Those 400 vehicles will be uniformly assigned a departure time between 7:00AM to 7:15 AM, which leads to an average departure time interval of 15 min /401 intervals = 2.24 seconds. As a result, the first three vehicles depart from zone at 7:00:00, 7:00:02, and 7:00:04, respectively.



Example 2.2: full matrix format

Below is an OD demand matrix file titled "demand_matrix.csv" for 4 valid zones (1,2,3 and 10). Note that, zones 4,5,6..,9 are not defined in input_zone.csv. A full matrix with sequential consecutive zone numbers (starting from zone 1 to the largest zone number 10) are listed for the first column and the first row, and all the OD pairs with invalid zone numbers (e.g. 1 to 4, 5, 6, 7,8,9; or zone 4 to all the other zones) have only values of zero.

trip	1	2	3	4	5	6	7	8	9	10
1	100	20	40	0	0	0	0	0	0	20
2	20	100	40	0	0	0	0	0	0	40

3	40	20	100	0	0	0	0	0	0	40
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0
10	20	40	20	0	0	0	0	0	0	100

The corresponding input_demand_meta_data.csv is shown below, assuming the demand horizon is 7AM to 9AM.

file_name	format_type	number_of_lines_to_be_skipped	loading_multiplier	start_time_in_min	end_time_in_min	apply_additional_time_dependent_profile	subtotal_in_last_column	number_of_demand_types	demand_type_1
demand_matrix.csv	full_matrix	0	2	420	540	0	0	1	1

Example 3: Multiple demand files for different demand types and hours

file_sequence_no	file_name	format_type	start_time_in_min	end_time_in_min	number_of_demand_types	demand_type_1
1	Demand_Data\\1sov-14.15.mtx	column	840	900	1	1
2	Demand_Data\\2hov-14.15.mtx	column	840	900	1	2
3	Demand_Data\\3heavytruck-14.15.mtx	column	840	900	1	3
4	Demand_Data\\4mediumtruck-14.15.mtx	column	840	900	1	3
5	Demand_Data\\5sov-15.16.mtx	column	900	960	1	1
6	Demand_Data\\6hov-15.16.mtx	column	900	960	1	2
7	Demand_Data\\7heavytruck-15.16.mtx	column	900	960	1	3
8	Demand_Data\\8mediumtruck-15.16.mtx	column	900	960	1	3
9	Demand_Data\\9sov-16.17.mtx	column	960	1020	1	1
10	Demand_Data\\10hov-16.17.mtx	column	960	1020	1	2
11	Demand_Data\\11heavytruck-16.17.mtx	column	960	1020	1	3
12	Demand_Data\\12mediumtruck-16.17.mtx	column	960	1020	1	3
13	Demand_Data\\13sov-17.18.mtx	column	1020	1080	1	1
14	Demand_Data\\14hov-17.18.mtx	column	1020	1080	1	2
15	Demand_Data\\15heavytruck-17.18.mtx	column	1020	1080	1	3
16	Demand_Data\\16mediumtruck-17.18.mtx	column	1020	1080	1	3
17	Demand_Data\\17sov-18.19.mtx	column	1080	1140	1	1
18	Demand_Data\\18hov-18.19.mtx	column	1080	1140	1	2
19	Demand_Data\\19heavytruck-18.19.mtx	column	1080	1140	1	3
20	Demand_Data\\20mediumtruck-18.19.mtx	column	1080	1140	1	3

The screenshot above shows an example of working with existing demand files.
There are 20 hourly demand files exported from VISUM, and all those MTX files are put under a subfolder of Demand_Data. There are 20 records in the above input_demand_meta_data.csv, and each record aims to map the data content to the overall time-dependent demand matrices used in DTALite. The first record specifies the following information:
(a) the demand file name (e.g., sov_14_15.mtx)
(b) format (e.g., column)
(c) the loading start time and end time for the demand file (e.g. 840 to 900, or 2PM to 3PM)

(d) the demand types associated with the demand file (only demand_type1), and demand types 1, 2 and 3 are SOV, HOV and truck, respectively.

Example 4: Multiple demand types as different columns per record from a TransCAD data set

A TransCAD user has the following demand file titled GH10PM_3Hrs.csv.

GH10PM_3Hrs.csv

```
1,1,0.3069,0.0484,0.0179,0.1847,0.0741,0.1242
1,2,1.9001,0.3872,0.1371,0.0782,0.0174,0.0165
1,3,0.1881,0.0172,0.0040,0.1689,0.0719,0.1736
1,4,1.7483,0.3281,0.1158,0.0752,0.0157,0.0148
1,5,0.6210,0.0707,0.0203,0.2744,0.1008,0.2907
1,6,0.9274,0.1720,0.0606,0.0417,0.0088,0.0085
1,7,0.0888,0.0114,0.0036,0.0525,0.0255,0.0439
1,8,0.5044,0.0932,0.0324,0.0576,0.0218,0.0421
1,9,1.7396,0.3067,0.1149,0.1976,0.0427,0.0533
1,10,1.0483,0.1873,0.0673,0.0681,0.0120,0.0107
1,11,0.6672,0.1230,0.0433,0.0307,0.0068,0.0063
1,12,0.7549,0.1280,0.0436,0.0533,0.0206,0.0309
```

Each record includes demand values for 6 demand types, shown in the following data structure description.

Vehicle Trip File Structure	
Field_Name	Type
Row index	Integer (4 bytes)
Column index	Integer (4 bytes)
SOV	Real (4 bytes)
HOV2	Real (4 bytes)
HOV3	Real (4 bytes)
Light Truck	Real (4 bytes)
Medium Truck	Real (4 bytes)
Heavy Truck	Real (4 bytes)

They want to use the following departure time profile from 3PM to 6PM.

Time	PM Period Proportion 15 minutes
3:00-3:15	0.07
3:15-3:30	0.07
3:30-3:45	0.08
3:45-4:00	0.08
4:00-4:15	0.08
4:15-4:30	0.08
4:30-4:45	0.09
4:45-5:00	0.09
5:00-5:15	0.10
5:15-5:30	0.10
5:30-5:45	0.09
5:45-6:00	0.09
	1.0000

The corresponding input_demand_meta_data.csv is configured below.

The first part of the file is shown below, which indicates we use a “column” format, with demand loading period from 3PM (900 min) to 6PM (1089 min) with additional time-dependent demand profile (= 1).

file_name	format_type	number_of_lines_to	loading_multiplier	start_time_in_min	end_time_in_min	apply_additional_time_dependent_profile
GH10PM_3Hrs.csv	column	0	1	900	1080	1

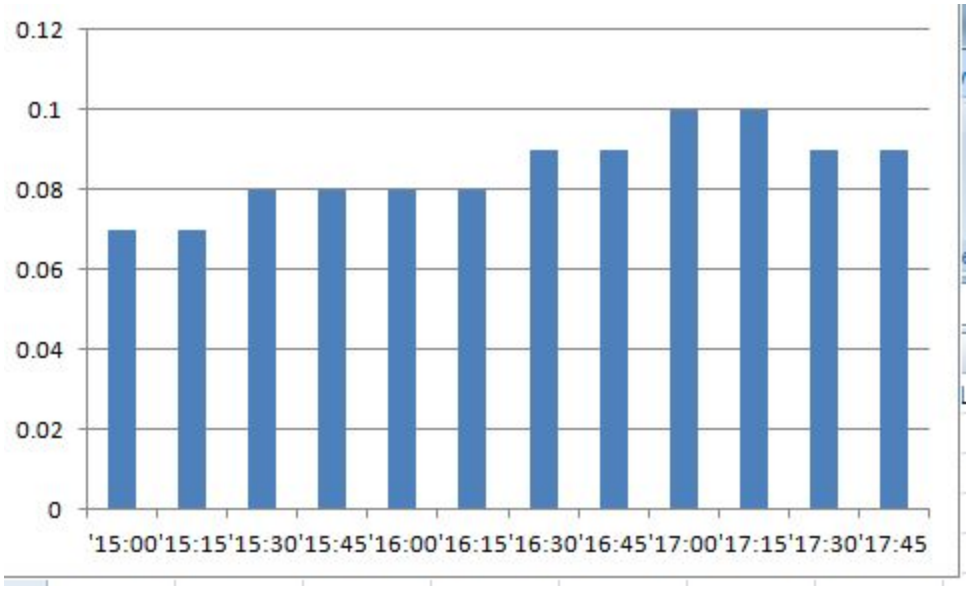
The second part of file specifies 6 demand types to be loaded.
number_of_demand_types = 6 for 6 demand types.
Field names demand_type_1, demand_type_2, demand_type_3, …, demand_type_6 requires demand type for each column after the row index and column index in the original demand file.
demand_type_1 = 1 for SOV type, demand_type_2 = for HOV2 and demand_type_3 = for HOV3. demand_type_4= demand_type_5 = demand_type_6 = 3 for different types of trucks.

me_dependent_profile	number_of_demand_types	demand_type_1	demand_type_2	demand_type_3	demand_type_4	demand_type_5	demand_type_6	'00:00
1	6	1	2	2	3	3	3	

The departure time profile is specified in the third part of the meta data file. Field ‘15:00 has a value of 0.07 for the time period between 15:00 and 15:15.

'15:00	'15:15	'15:30	'15:45	'16:00	'16:15	'16:30	'16:45	'17:00	'17:15	'17:30	'17:45	'18:00	'
0.07	0.07	0.08	0.08	0.08	0.08	0.09	0.09	0.1	0.1	0.09	0.09		

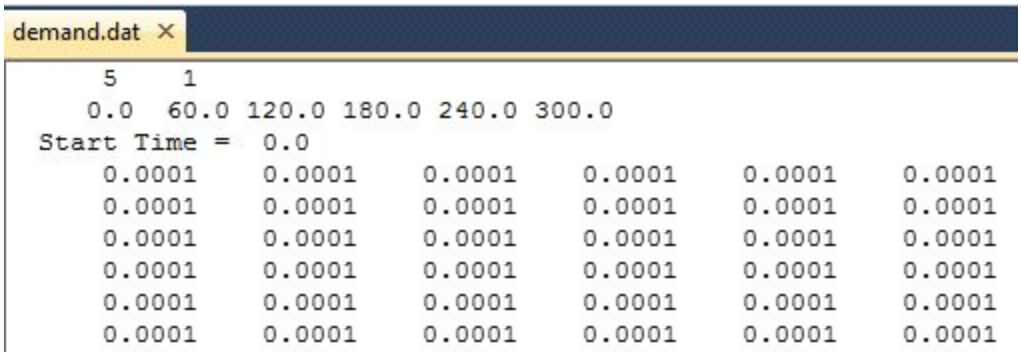
leads to the following demand loading pattern.



Example 5: Multiple time-dependent demand files from a DYNASMART data set

A typical DYNASMART demand data includes three file: demand.dat, demand_HOV.dat and demand_truck.dat. The following screenshot shows a demand.data file for a 6-hour horizon from

6AM to 11AM.



We can construct the following input_demand_meta_data.csv.

file_name	format_type	number_of_lines_to_be_skipped	loading_multiplier	start_time_in_min	end_time_in_min	demand_type_1
demand.dat	dynasmart	0	1	360	660	1
demand_hov.dat	dynasmart	0	1	360	660	2
demand_truck.dat	dynasmart	0	1	360	660	3

Each record corresponds a file for a certain demand type, using the same format_type = dynasmart but different demand_type_1 = 1,2,3 for LOV, HOV and trucks. We do not need to skip the first line so number_of_lines_to_be_skipped = 0, As DTALite uses a 24-hour time clock for both input and output format, we map the demand loading period to 6AM (start_time_in_min = 360 min) and 11AM (end_time_in_min = 660.).

Please note that, the existing Dynus-T package uses the same demand format as the DYNASMART-P package, so if you have a Dynus-T data set, please use the above demand meta data settings directly.

Example 6: Agent file after OD demand estimation run

The OD estimation process will calibrate path flow pattern accoding to the observed link counts and density measurments, and the resulting calibrated results (at the last iteration of ODME) are saved as a binary file as agent.bin file, and an ASICII file called output_agent.csv.

To evaluate traffic measurement strategies, after the ODME run, one can rename agent.bin to input_agent.bin, and set format_type=agent_bin in input_demand_meta.csv and run regular traffic assignment again (e.g. using traffic assignment method = MSA or fixed switching rate).

input_scenario_settings.csv

The scenario settings file allows the user to alter the characteristics of the scenarios being run, as well as create various traffic scenarios that can be run simultaneously. Scenario attributes such as demand multiplier, traffic flow model, and number of days a scenario will be run can all be changed in this file. Further, each row can contain data for a separate scenario, allowing the user to simultaneously run models with differing model attributes. The scenario settings file allows the user to alter different attributes for each scenario. Starting from the far-right column, these attributes are:

Variable	Description	Example Usage
scenario_no	This is a discrete integer value assigned to a given scenario, and will be used as the scenario's unique identifier when the simulation is running in DTALite.	scenario_no =1
scenario_name	This is the identifier by which the scenario will be displayed to the end user. This identifier, unlike the scenario_no, need not be an integer.	scenario_name = test1
number_of_assignment_days	This value, an integer, is the number of days the scenario will be run. If the user is employing Origin-Destination Matrix Estimation, the scenario should run for at least 15 assignment days.	number_of_assignment_days = 95
demand_multiplier	This value is the number by which the demand given in the imput_demand.csv file will be multiplied for a given scenario, e.g. if the demand for a given OD pair is 1000, and a demand multiplier of	demand_multiplier =0.7

	1.8 is used for a given scenario, then for that scenario DTALite will use a value of 1800 for the demand on that OD pair.	
random_seed	This value is the seed number used for the pseudorandom number generator, used to create a level of randomness in certain aspects of the simulation.	random_seed =100
traffic_flow_model	This value must be one of four possible values: 0: Bureau of Public Roads (BPR) Model. This is a simple model which relates flow rate on a link to its volume-to-capacity ratio. 1: Point Queue Model. This model assumes vehicles “stack up” at nodes, rather than filling up the link. 2: Spatial Queue Model. This model represents queues as they actually exist, filling up links as they form. 3: Newell’s N-Curve Model. The most thorough of the four models, which takes into account features such as wave propagation through traffic.	traffic_flow_model = 3
freeway_bias_factor	This value dictates the degree to which agents modeled in the simulation will choose routes. The default value is sufficient for most simulations.	freeway_bias_factor=1

traffic_assignment_method	Like traffic_flow_model, this field allows integer values from zero to four. For assignment method, these values are pre-defined: 0: Method of Successful Averages (MSA) 1: Day-to-day learning 2: Gap-based switching rule for user equilibrium 3: Gap-based switching rule and MSA step-size for user equilibrium	traffic_assignment_method=1

import_GIS_settings.csv

Related learning document:

File import_GIS_settings.csv is a user-defined configuration file, and a user can use Excel to edit it. This file can help users to associate/map fields in shapefile DBF files to the AMS Data Hub schema data format, and NeXTA can use this file to read the network geometry from shapefiles and create an Analysis Modeling Simulation (AMS) Data Hub compatible transportation network project file (which is readable by both DTALite and NeXTA). It should be remarked that, this file is not used by DTALite, as DTALite reads directly the converted files of input_node.csv and input_link.csv.

Depending on the type of data to be imported. the configuration file consists of several **sections**. As defined in Column A, there are general sections of file, configuration, and sections for different network objects such as node, link, zone, and optional centroid and connector layers.

The "**key**" settings given in Column B are used by DTALite/NeXTA to match and convert the shapefiles from the DBF files.

"**value**" in Column C is the matching name between NeXTA's "key" values, and those given in the shapefiles.

Column "**allowed_values**" show which definitions are related to "value" for binary values (such as km vs. mile, lane vs. link, etc). Column "**required_or_optional**" shows which values must be defined for a successful import.

Remarks:

- Although this configuration contains many columns, **users only need to prepare or change the data in column "value".** When the converted shapefiles can be opened in any GIS software (we recommend open-source Q-GIS) and accessed through attribute tables.
- Once a configuration file is established for a regional travel demand format, it can be **replicated as a template** for future imports from that format.
- The open-source GIS library GDAL is used by NeXTA to import the geometry and field data from GIS shape files.
- Sample GIS importing files for can be found at the online Google Drive or under Internal_release\importing_sample_data_sets\GIS_files\West_Jordan_from_CUBE.
- The user needs to be careful with **attribute names that have more than 10 characters** (such as ControlType in this case). For example, an exported SHP/DBF file will convert the file name to the 10-character length (Controlt~1). To see the correct attribute name definitions, the user can use any GIS software to open the corresponding shapefile and read the name from the attribute table.

The first **file section defines the shape file names for GIS layers of nodes, links, zones, centroids, and connectors. In this example, the file name table looks like the following table. The reference file names should correspond to the layers exported from planning packages such as VISUM, CUBE and TransCAD. Node, link and zone are the required layers for a successful GIS network import. DTALite does not require connectors in its network representation, so users can leave empty values for keys "centroid" and "connector".**

Remarks for adding zones and activity locations based on node number:

For a typical Cube planning data set, layers centroid and connector are not required. One can set key node_number_threshold_as_centroid in section configuration to a user-defined value, say 3000, so that all the nodes below this number will be used as centroids and a corresponding zone will be created for each centroid node.

section	key	value	required_or_optional
file_name	node	SLC_90_Work_Zone_Network_Node.shp	optional
file_name	link	SLC_90_Work_Zone_Network_Link.shp	required
file_name	zone	SLC_90thSouth_Zone.shp	optional
file_name	centroid		optional
file_name	connector		optional

Remarks:
(1) Only the link layer is required.
(2) If the node layer is not provided, then please set use_boundary_nodes_as_activity_locations to yes. In this case, node ids will be automatically generated from starting and ending points of a link.
(3) If the zone layer is not provided, then please specify node_number_threshold_as_centroid to define nodes as zone centroids. Running DTALite does not require zone boundary information.
(4) The additional centroid layer can help us build the activity location file.
(5) The additional connector layer can help us build the activity location file that links centroid to physical nodes.

The next **configuration section describes general model attributes and import options that can accommodate different network coding conventions. These settings include the latitude/longitude coordinate system definition, length units, one way vs. two way links, and whether the capacity is given per lane, or per link. Its advanced settings also allow users to import link type-specific speed limit and capacity values, and to estimate the possible locations of signalized intersections, and link-based effective green time. In the following example, the configuration settings are given with sample values.**

section configuration/ key ID	**key**	**value**	**allowed_values**
1	with_decimal_long_lat	yes	yes;no
2	length_unit	mile	km;mile
3	number_of_lanes_oneway_vs_twoway	oneway	oneway;twoway
4	lane_capacity_vs_link_capacity	lane	lane;link
5	direction_0_as_oneway_vs_twoway	oneway	oneway;twoway
5.1	default_link_direction	oneway	oneway;twoway
6	node_number_threshold_as_centroid	0	default 0
7	use_default_speed_limit_from_link_type	no	yes;no
8	use_default_lane_capacity_from_link_type	no	yes;no
9	use_default_number_of_lanes_from_link_type	no	yes;no

10	identify_signal_intersection	yes	yes;no
11	minimum_speed_limit_for_signals	28	
12	maximum_speed_limit_for_signals	60	
13	default_cycle_length_in_second	110	
14	default_saturation_flow_rate_in_vhc_per_hour_per_lane	1800	
15	minimum_length_for_importing_links	0.00001	
16	identify_from_node_id_and_to_node_id_based_on_geometry	no	yes;no
17	create_connectors_for_isolated_nodes	no	
18	multiplier_for_obtaining_hourly_capacity	1.0	

Item-by-item descriptions:

1) Key **with_decimal_long_lat** indicates if a decimal long/lat format is used.
For example, longitude -111.943375 has a decimal point, while -111943375 has no decimal point.
If "value" is set to no, a multiplier of 0.00001 is used to convert the sample value of -111943375 to -111.943375.

2) Key **length_unit** indicates if the unit of link length is mile or km. If "value" = km, then the length value for link layer will be multiplied by a factor of 0.621371 to convert km to miles.

3) Key **number_of_lanes_oneway_vs_twoway** indicates if the the number of lanes field in link layer is refereed to an one-way link or an two-way link. If "value" = twoway, then the number of lanes will be divided by 2 as DTALite/NeXTA uses one-way links.

4) Key **lane_capacity_vs_link_capacity** indicates if the capacity filed in the link layer is referred to link capacity across all lanes or simple lane capacity. If "value" = twoway, then the number of lanes will be divided by the number of lanes as DTALite/NeXTA uses lane-based capacity.

5) Key **direction_0_as_oneway_vs_twoway** indicates if the direction field of 0 represents an one-way or two-way link. If "value" = twoway, NeXTA will duplicate the two links based on a single link record with different directions. Parameter default_link_direction specifies if a link record is an oneway link or twoway link. If a record is a twoway link, then two directional links will be created by NeXTA.

6) Key **node_number_threshold_as_centroid** specifies the value for using nodes directly as zone centriods. In VISUM or TransCAD data set, a centroid layer is provided, while Cube might assume a set of nodes below a certain number (say 3000) is used as zone centriods by default. This number typically is specified in a script for traffic assignment in Cube. For example there are 2000 nodes with node numbers < 3000. By reading this number threshold, NeXTA can automatically (a) add 2000 traffic analysis zones, (b) assign a node to as the activity location for each newly created zone. Thus, the zone and activity location layers (namely files input_activity.csv and input_zone.csv) are created

automatically.

7,8,9) Keys **use_default_speed_limit_from_link_type; use_default_lane_capacity_from_link_type; use_default_number_of_lanes_from_link_type** allow users to input their link-type specific lane capacity, number of lanes and speed limit fields in file input_link_type.csv. If "value" = yes, then users do not need to specify the field names in section link for these three important link attributes. For each link, based on the link type value (e.g. 1 in sample table) , NeXTA will fetch the default link attributes for the link type, and fill out the records with a capacity of 1800, a speed limit of 65 mph and 3 lanes.

10, 11, 12,13, 14) Keys **identify_signal_intersection, minimum_speed_limit_for_signals, maximum_speed_limit_for_signals, default_cycle_length_in_second** are a group of configuration settings if users need to estimate the signal locations based on speed limit and geometry connections. Using the sample value in the above table, if identify_signal_intersection = yes, then NeXTA will identify a node with the following two conditions as signalized intersections and put a default cycle length of 110 seconds.

(a) a node with more than 3 incoming directed links, and (b) at least one of links has a speed limit between 28 mph and 60 mph.

This process will set the control type and cycle length in file input_node.csv, and set a default effective green time for each link with its downstream node as a signalized intersection.

The link effective green time = lane capacity/default_saturation_flow_rate * default cycle length.

Tips: The locations of signalized intersections can be visualized through Google Earth through NeXTA menu -> export -> export signal node KML file.

15) Key **minimum_length_for_importing_links** specifies the minimum length threshold for importing a link. With a default value of 0.00001, NeXTA does not import a link with a length of 0. If a negative value of -0.00001 is given, then NeXTA will keep the links with zero distance in the final imported data set.

16) In the link layer, if the upstream node id and downstream node id of a link is not provided, for example, in a TransCAD dataset, then the flag of **identify_from_node_id_and_to_node_id_based_on_geometry** can allow NeXTA to use the geometry information of a link to find the nearby nodes from the node layer to construct the from_node_id and to_node_id fields for a link. If no such a node is available or even no node layer is provided, then NeXTA will create new nodes and use the corresponding new node numbers for the fields of from_node_id and to_node_id. This setting significantly relaxes the data requirements.

17) In a data set with the link layer as the road network and the node layer as the household locations (see data set in \\importing_sample_data_sets\RoadLinkLayer+HouseholdNodeLayer\\), setting **create_connectors_for_isolated_nodes** can allow NeXTA to create new connectors to connect household locations (as activity locations) to the road network.

18) In some data sets, the link capacity values are provided as a daily value or a period value for AM or PM periods. The parameter of **multiplier_for_obtaining_hourly_capacity** allows users to convert the given value to the hourly capacity value required by DTALite.

The next section defines node attributes. The import values are node ID, node name, the zone (TAZ) to which a certain node belongs to, and node control type. The sample node table is as follows, while fields node_id and TAZ are required. TAZ can be also set to the same as node_id.

section	**key**	**value**	**required_or_optional**
node	node_id	N	required only when the node layer file is provied
node	name		optional, for display only
node	TAZ	TAZID	optional, if not provided, then usersc can use node_number_threshold_as_centroid to specify the first n nodes directly as centroids of zones.
node	control_type		optional, if node control type has been specified.

The **link section defines the link shapefile attributes. In order to code the corresponding attributes correctly, the user can access the link shapefile through GIS software, and read the link attributes. The available link attributes are from and to node, name, link ID, link type, transportation modes that link is open for, direction definition, number of lanes, hourly capacity, speed limit, and number of lanes, capacity, speed limit and link type for reversible lanes, if links are defined as two way links. For this example, the link table is as follows:**

section/key ID	**key**	**value**	**required**	**Remark**
1	from_node_id	A	optional	Typical values include usn, if data is not provided, please

				set identify_from_node_id_and_to_node_id_based_on_geometry = yes
2	to_node_id	B	optional	Typical values include dsn. if data is not provided, please set identify_from_node_id_and_to_node_id_based_on_geometry = yes
3	name	STREET	optional	for display only
4	link_id		optional	for searching links by ids,
5	TMC		optional	for searching links by TMC, e.g. from Inrix data set
6	mode_code			reserved for transit, pedestrian modeling.
7	direction	ONEWAY		If no value is given, an one-way link is assumed by default.
8	length	DISTANCE	required	The unit should be miles or km.
9	number_of_lanes	LANES	optional	See further remarks if you want to input a link type-specific # of lanes.
10	hourly_capacity	CAP1HR1LN	optional	See further remarks if you want to input a link type-specific hourly capacity.
11	speed_limit	SFF	optional	See further remarks if you want to input a link type-specific speed limit.
12	r_number_of_lanes			
13	r_hourly_capacity			

14	r_speed_limit			
15	r_link_type			

Remarks

There is a separate CSV file for link types (input_link_type.csv) that needs to be updated with the correct link types. It length, speed and capacity data are indirectly defined through link type table, users can also set the following key to yes in the configuration section:

use_default_speed_limit_from_link_type; use_default_lane_capacity_from_link_type; use_default_number_of_lanes_from_link_type

- If field r_number_of_lanes has a user specified value, then the link attributes will be read for the link on the reversed direction. Typically, the capacity, speed limit and link type values of the link the forward direction will be used as the default, unless users provided values for fields 13-15.

The **zone section defines the zone shapefile attributes. Only a zone id field is needed, if users have specified a value for key centroid** in the file_name section. This zone id should be consistent with the zone id system used OD demand files. If the zone layer does not present, one can set a positive value (say 3000) for key **node_number_threshold_as_centroid** in section configuration to add zones (and the corresponding activity locations).

section	key	value	required_or_optional
zone	zone_id	Id	required

The **centroid section defines the centroid shapefile attributes. Fields node_id and TAZ can have the same or different names. The value of NO for both fields in the following example.**

The connector section defines the connector shapefile attributes. Fields zone_end and node_end are referred to the upstream and downstream node ends of a connector. A connector can have its own speed limit, number of lanes, link type per record or use the default values given in the configuration keys: default_speed_limit, default_link_type, defaut_direction and default_number_of_lanes.

section	key	value

centroid	name	
centroid	node_id	NO
centroid	TAZ	NO
connector	zone_end	ZONENO
connector	node_end	NODENO
connector	length	LENGTH
connector	number_of_lanes	
connector	hourly_capacity	
connector	direction	DIRECTION
connector	default_speed_limit	60
connector	default_link_type	99
connector	defaut_direction	0
connector	default_number_of_lanes	7

The **finel_output section** **allows users to define if a two-way link should be applied with the offset or not, as a two-way link is converted to two directed links in NeXTA but still have the same geometry data.**

final_output	offset_link	yes		

A complete sample imput_GIS_configuration.csv file is listed below. The related source code can be found at https://code.google.com/p/nexta/source/browse/trunk/Version1_01/NEXTA-GUI/AMS_Import.cpp. If you have any questions about this file, please send an email to the author at xzhou99@gmail.com.

section	key	value	required_or_optional	allowed_values
file_name	node	SLC_90_Work_Zone_ Network_Node.shp	required	
file_name	link	SLC_90_Work_Zone_	required	

		Network_Link.shp		
file_name	zone	SLC_90thSouth_Zone.shp	required	
file_name	centroid			
file_name	connector			
configuration	with_decimal_long_lat	yes		yes;no
configuration	length_unit	mile		km;mile
configuration	number_of_lanes_oneway_vs_twoway	oneway		oneway;twoway
configuration	lane_capacity_vs_link_capacity	lane		lane;link
configuration	direction_0_as_oneway_vs_twoway	oneway	oneway;twoway	
configuration	node_number_threshold_as_centroid	0		default 0
configuration	use_default_speed_limit_from_link_type	no		yes;no
configuration	use_default_lane_capacity_from_link_type	no		yes;no
configuration	use_default_number_of_lanes_from_link_type	no		yes;no
configuration	identify_signal_intersection	yes		yes;no
configuration	minimum_speed_limit_for_signals	28		
configuration	maximum_speed_limit_for_signals	60		
configuration	default_cycle_length_in_second	110		
configuration	default_saturation_flow_rate_in_vhc_per_hour_per_lane	1800		
configuration	minimum_length_for_importing_links	0.00001		
node	node_id	N	required	

node	name			
node	TAZ	TAZID	required	
node	control_type			
link	from_node_id	A	required	
link	to_node_id	B	required	
link	name	STREET		
link	link_id			
link	link_type	FT		
link	mode_code			
link	direction	ONEWAY		
link	length	DISTANCE	required	
link	number_of_lanes	LANES	required	
link	hourly_capacity	CAP1HR1LN	required	
link	speed_limit	SFF	required	
link	r_number_of_lanes			
link	r_hourly_capacity			
link	r_speed_limit			
link	r_link_type			
zone	zone_id	Id	required	
centroid	name			
centroid	node_id			
centroid	TAZ			
connector	zone_end			
connector	node_end			
connector	length			
connector	number_of_lanes			
connector	hourly_capacity			
connector	default_speed_limit			

connector	default_link_type			
connector	defaut_direction	0		
final_output	offset_link	yes		

Sample configuration file for importing Inrix data

section	key	value	remark
file_name	node		the node layer can be empty
file_name	link	INRIXroads.shp	only link layer is provided
configuration	length_unit	mile	
configuration	identify_from_node_id_and_to_node_id_based_on_geometry	yes	new nodes will be created based on geometry of link geometry data
configuration	default_link_direction	twoway	each link record corresponds a two way link by default
link	from_node_id		keep those two fields empty
link	to_node_id		
link	name	TMC	
link	link_id	objectid	
link	TMC	TMC	TMC coded is required to read the input_TMC_speed.csv data
link	link_type	FRC	road type
link	length	Miles	

input_MOE_settings.csv

The measure of effectiveness, or MOE, settings allow the user to test the effectiveness of the network as a whole, or smaller sections of a network, such as a single link, 3-point path, or

origin-destination pair. The MOE settings file also allows the user to identify links, paths, and origin-destination pairs that are above user-defined threshold values. The following are the possible values for the moe_type field (the first column in the input_MOE_settigns.csv file), as well as which fields must be filled in for each:

MOE_type	**Description**
network	Network MOE measures the effectiveness of the network at large. This network-wide measure can also be broken down based on attributes such as demand type and vehicle type.
od	This MOE type gauges the effectiveness of the network from one zone to another. The only field that must be populated are “origin_zone_id” and destination_zone_id It should be noted that it will only measure the effectiveness in the from-to direction. That is to say, if zone 1 is set as the origin zone, and zone 2 as the destination, effectiveness will only be measure from zone 1 to 2, not 2 to 1. In order to measure effectiveness in both directions, create two separate OD MOEs.
link	To measure the effectiveness of a link, only the “from_node_id” and “to_node_id” fields must be populated. As with OD MOE, the measure of effectiveness only goes in the from-to direction.
path_3point	Much the same as link MOE, 3-point path MOE measures the effectiveness of a path between three connected nodes. This MOE needs an entry in “from_node_id,” “mid_node_id,” and “to_node_id.”
network_time_dependent	This measures the effectiveness of the network on a minute-by-minute basis. The results from this MOE are displayed in the output_NetworkTDMOE.csv file.
od_critical, link_critical, and path_critical	Each of these only requires the user to enter a value in the “threshold_volume” field. For

	example, if link critical MOE is performed with a threshold volume of 1250, then, in the output summary file, NeXTA will print MOE results for all of the links with volume over 1250.

The “moe_group” column is used to break the MOE settings into discreet groups in the output summary. For example, to have all MOE critical values displayed together, assign them all the same group number, and they will be clustered together in the output summary. The “moe_cetegory_label” is a user-defined field used to give simpler names to each individual measure of effectiveness. Each MOE may also have an associated start and end time based on when vehicles enter or exit the network, OD pair, link, or path.

Group IV: Scenario files

The user may prepare scenarios by preparing the following input files, which describe different network conditions so that their effects on operations may be evaluated. Different scenarios available include tolling (distance-based and link-based tolls), dynamic message signs, incidents, and work zones.

Scenario_Link_Based_Toll.csv

The link-based toll scenario input table is used to define tolling conditions on a road segment in the simulation. Currently, there are three classes defined for different toll pricing – SOV, HOV, and trucks.

Variable Name	Type	Acceptable Values	Description
Link	Integer	[1,2]	Node pair [upstream, downstream] used to identify the link on which the toll is implemented
Day No	Integer	Value > 0	Day identification number in the simulation on which the tolling strategy is implemented
Start Time in Min	Integer	0 to 1440	Daily starting time for the link-based toll
End Time in min	Integer	0 to 1440	Daily ending time for the link-based toll

Charge for LOV ($)	Float	0 to 999	Charge for Single Occupancy Vehicles (SOV) to travel across the link
Charge for HOV ($)	Float	0 to 999	Charge for High Occupancy Vehicles (HOV) to travel across the link
Charge for Truck ($)	Float	0 to 999	Charge for Trucks to travel across the link

Scenario_Dynamic_Message_Sign.csv

The dynamic message sign scenario input file is used to define the location and characteristics of variable message signs in the simulation, which influences driver route choice by the response percentage defined in the table.

Variable Name	Type	Acceptable Values	Description
Link	Integer	[1,2]	Node pair [upstream, downstream] used to identify the link on which the sign is installed
Start Time in Min	Integer	0 to 1440	Starting time for the dynamic message sign display
End Time in min	Integer	0 to 1440	Ending time for the dynamic message sign display
Responce Percentage ()	Float	Value ≥ 0	Percentage of drivers on the link which respond to the real time information displayed on the sign.

Scenario_Incident.csv

The incident scenario input file is used to define the location and characteristics of incidents in the simulation, which may include any general capacity reduction and can be applied for general incidents (e.g., debris) weather, and crashes.

Variable Name	Type	Acceptable Values	Description

Link	Integer	[1,2]	Node pair [upstream, downstream] used to identify the link on which incident occurs
Day No	Integer	Value > 0	Day identification number in the simulation on which the incident occurs
Start Time in Min	Integer	0 to 1440	Starting time for the capacity reduction due to incident
End Time in min	Integer	0 to 1440	Ending time for the capacity reduction due to incident
Capacity Reduction Percentage (%)	Float	Value ≥ 0	Capacity reduction percentage (1 – percent remaining capacity) due to incident
Speed Limit (mph)	Integer	Value ≥ 0	Speed limit on link due to incident

Scenario_Work_Zone.csv

The work zone scenario input file is used to define the location and characteristics of work zones in the simulation, which is described in terms of capacity reduction, project duration, and speed reduction.

Variable Name	**Type**	**Acceptable Values**	**Description**
Link	Integer	[1,2]	Node pair [upstream, downstream] used to identify the link on which work zone is located
Day No	Integer	Value > 0	Day identification number in the simulation on which the work zone causes capacity reductions
Start Time in Min	Integer	0 to 1440	Starting time for capacity reduction due to work zone

End Time in min	Integer	0 to 1440	Ending time for capacity reduction due to work zone
Capacity Reduction Percentage (%)	Float	Value ≥ 0	Capacity reduction percentage (1 – percent remaining capacity) due to work zone
Speed Limit (mph)	Integer	Value ≥ 0	Speed limit on link posted during work zone

Group V: Application support files

Demand files

: required for simulation, but typically provided by users.

Signal control file

AMS_movement.csv

Sensor data file

sensor_data
Subarea..

Group VI: Simulation output files from DTALite

The following table summarizes the output files from DTALite traffic assignment.

File Name	Type	Data Description
1.Output_summary.csv	Scenario statistics	Traffic assignment results for each iterations, such as **computational time, average travel distance and travel time, number of travelers and user equilibrium gaps.**
2.output_NetworkTDMOE.csv	Network	Time-dependent network-level information about assignment results over the modeling horizon, such as, **cumulative inflow count, culmulative outflow count, number of vehicles existing in the network, average trip time**.

3.Output_ODMOE.csv	OD	ODMOE simulation results, such as, **number of vehicles completing trips, trip time, trip distance for each OD pair.**
4.Output_linkMOE.csv	Overall link statistics for all links in the entire network	Detailed simulation results aggregated at each link, such as**, link volume, link speed, level of service, emissions, volume of different types of vehicles.**
5.Output_linkTDMOE.csv	Time-dependent link statistics for for all links in the entire network	Time-dependent detailed simulation results aggregated at each link at each minute, such as, **density, volume, speed, queue length, emissions.** Output_linkTDMOE.bin is read for the time-dependent network-level visualization.
6.Output_agent.csv	Vehicle/ agent	Specific information of each agent in the simulation network, such as, **origin, destination, departure time, node sequence of its path, time sequence of each node in its path, emissions;** NEXTA uses binary file for fast data processing

The following table summarizes the output files from NeXTA data processing.

File Name	Type	Description	Visualization
1.AMS_OD_table.csv	Subarea cut	outputs the OD time span volume.	Excel
2.AMS_path_flow.csv	Subarea cut	Outputs the path flow	NEXTA
3.AMS_movement.csv	Subarea cut	outputs the number of vehicles making	Excel

		movements in the intersections	
4.AMS_link shape files (dbf, shp)	network	For GIS visualization	GIS Editor
5.AMS_link.kml	exporting	For Google Earth visualization	Google Earth & Google Fusion tables
6.output_travel_time_matrix.csv	network	Outputs the zone-to-zone travel time matrix	Excel
7.UTDF files	exporting	Files generated during Synchro exports	Synchro
8.VISSIM ANM files	exporting	Files generated during VISSIM exports	VISSIM

3. Preparing a Dynamic Traffic Assignment Data Set: A Streamline Workflow

3.1. Preparing a Network Data Set

The following sections describe processes for preparing network data using NeXTA's network conversion utilities. A network data set for the DTALite/NeXTA package includes four major CSV files: input_node.csv, input_link.csv, input_zone.csv, input_activity_location.csv, as well as a number of definitional files such as input_node_control_type.csv and input_link_type.csv.

There are multiple ways of constructing a network data set for the DTAlite/NeXTA package:

(1) Importing **shape files from regional planning packages such as VISUM, TransCAD, Cube, which typically consist of node, link and traffic analysis zone layers, with or without centriod and connector layers. Because network data coding is very flexible in different network modeling software packages, with many applications allowing users to define custom formats and data fields for use within a software package. To support various transportation planning applications, NeXTA uses a configuration CSV file for** importing GIS settings, to identify and connect the fields in the input DBF files to their corresponding fields in the NeXTA/DTALite data format. Related learning document on

importing VISUM and Cube sample data sets.

(2) Import shape files from **GIS shape files with a link layer only**, **such as an openstreet map layer, a Traffic Message Channel (TMC) coded network file. First, those GIS sources might not have a separate node layer, so users need to use NeXTA to automatically create the node layer based on the end feature points of a link curve. In order to support the traffic simulation, users also need to create zone layers and define the additional activity layer to link the node layer to the traffic analysis zones.** Related learning document on importing a typical GIS line layer.

(3) Import a simple geo-coded network data set through **a single Excel file as a template**. **It might be inconvenient for a user to directly prepare the csv files, so NeXTA 32-bit version provides an Excel-based interface to import the essential node, link and zone attributes from an Excel file with multiple sheets. Users can find the sample Excel file from the standard release of NeXTA package, use it as a template to input their own network data. By doing so, users can avoid unnecessary data conversion errors.** Related learning document on preparing and importing a simple network in Excel.

(4) NeXTA provides user-friendly interface for **creating a transportation network from scratch, using a background image as a reference. The network to be created includes a number of links, nodes and zones.** Related learning document.

(5) As **Synchro** is a widely used software package for many traffic engineers to manage urban street networks with signal timing data and traffic turning counts. NeXTA also provides an interface for user to import Universal Traffic Data Format (UTDF) into DTALite's network format with node/link layers. The traffic signal phasing and timing data can be also imported, which can provide useful signal control information for dynamic traffic assignment and simulation.

Below is a simplified step-by-step instruction for different network data preparation methods.
(1) Importing **shape files from regional planning packages**

The user needs to prepare three files first:
(a) input_node_control_type.csv
(b) input_link_type.csv, and
(c) import_GIS_settings.csv,
in addition to the (d) shape files exported from the GIS planning package.

1. Load the network in originating software application
2. Export network as shapefiles, and ensure the coordinate system is WGS_1984.
    a. sample steps for exporting data set from VISUM
    b. sample steps for exporting data set from Cube
    c. sample steps for exporting data set from TransCAD
3. Prepare input_node_control_type file, according to document here
4. Prepare input_link_type.csv file, according to document here.
5. Prepare import_GIS_settings.csv file, according to data structure description here and

sample import_GIS_settings.csv for VISUM, Cube and TransCAD

6. Use shape file importing utility in NeXTA through File -> Import -> GIS Planning Data Set.
7. Save the new network data set as a new project

Suggested map projection: **WGS84.** The open-source GIS library (Geospatial Data Abstraction Library) allows NeXTA to directly import shape files from GIS packages. Although NeXTA can work with arbitrary coordinate systems, we highly suggest users to export their shape files according to map projection – WGS84, the default projection for Google Earth and the KML data standard. Map projection data is stored in a projection (PRJ) file within the shapefile standard. For other users-define coordinate systems, users can also follow the instruction here to look up the long/lat coordinates of key nodes and use NeXTA to convert the network coordinate system to long/lat format.

3.2. Preparing a Demand Data Set

There are multiple ways of constructing a demand data set for DTALite, while all demand files should be defined clearly through the demand meta database file input_demand_meta_data.csv. This section assumes readers have understood the basic structure and examples of input_demand_meta_data.csv. The following instruction will use a few sample files to illustrate the steps of preparing a demand data set.

3.2.1. Start from the Scratch

Please first read document here on creating a network by clicking-and-drawing nodes/links. The next task is to create zones and associated activity locations. A simple way is to manually define the boundary locations of zones through NeXTA, and the nodes inside each boundary will be automatically identified as the activity locations of the zone. For a user who is familiar with the data structure in files input_zone.csv and input_activity_location.csv, you also can quickly prepare the files input_zone.csv and input_activity_location.csv using Excel. For example, there are four nodes that serve as an activity location, namely 1, 2, 3, and 10. The last node number of 10 is used to illustrate that DTALite does not require sequential node/zone numbers.

Prepare input_activity_location.csv as the following table.

zone_id	node_id	external_OD_flag
1	1	0
2	2	0
3	3	0
10	10	0

Prepare input_zone.sv as the following table

zone_id	geometry
1	
2	
3	
10	

Please recall that the zone boundary is used by NeXTA for visualization purposes only, and it is not used in the DTALite simulation engine. Thus, as long as four rows are defined with different zone ids, it is fine to leave the field geometry as blank.

To define a demand table, one can use the following 3-column format (file input_demand.csv)

from_zone_id	to_zone_id	number_of_trips_demand_type1
1	1	1000
1	2	1000
1	3	1000
1	10	2000

with the following input_demand_meta_data.csv.

file_name	format_type	number_of_lines_to_be_skipped	loading_multiplier	start_time_in_min	end_time_in_min	apply_additional_time_dependent_profile	subtotal_in_last_column	number_of_demand_types	demand_type_1
input_demand.csv	column	0	2	420	540	1	0	1	1

'07:00	'07:15	'07:30	'07:45	'08:00	'08:15	'08:30	'08:45
0.2	0.2	0.2	0.2	0.3	0.3	0.3	0.3

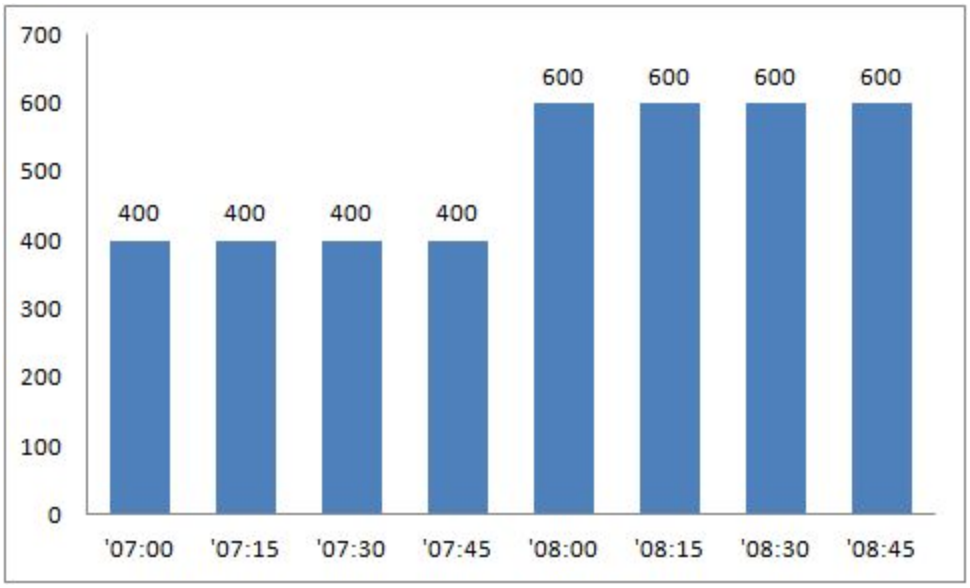
The meta-data file requires several other entries, but the relevant entries are as listed and explained below.

1. file_name: “input_demand.csv” is the OD table for the AM period being brought into DTALite/NeXTA.

2. format_type: “column” is the type of OD table format for this small test application.

3. Start_time is minute 420 or 7 AM and the End_time is minute 540 or 9 AM. The OD table stores number of trips between 7 and 9 AM. loading_multiplier is 2, so a multiplier of 2 is applied to each row. That is, the total demand from zone 1 to zone 2 is 1000*2 =2000.

4. As field apply_additional_time_dependent_profile is set to 1, then a time-dependent departure time profile will be read and result in the following demand loading pattern. That is, the demand from 7:00AM to 7:15 AM from zone 1 to zone 2 has a demand volume of 1000*2*0.2 = 400. Those 400 vehicles will be uniformly assigned a departure time between 7:00AM to 7:15 AM, which leads to an average departure time interval of 15 min /401 intervals = 2.24 seconds. As a result, the first three vehicles depart from zone at 7:00:00, 7:00:02, and 7:00:04, respectively.



3.2.2. Integrate Existing Demand Files with Different Demand Types

file_sequence_no	file_name	format_type	start_time_in_min	end_time_in_min	number_of_demand_types	demand_type_1
1	Demand_Data\\1sov-14.15.mtx	column	840	900	1	1
2	Demand_Data\\2hov-14.15.mtx	column	840	900	1	2
3	Demand_Data\\3heavytruck-14.15.mtx	column	840	900	1	3
4	Demand_Data\\4mediumtruck-14.15.mtx	column	840	900	1	3
5	Demand_Data\\5sov-15.16.mtx	column	900	960	1	1
6	Demand_Data\\6hov-15.16.mtx	column	900	960	1	2
7	Demand_Data\\7heavytruck-15.16.mtx	column	900	960	1	3
8	Demand_Data\\8mediumtruck-15.16.mtx	column	900	960	1	3
9	Demand_Data\\9sov-16.17.mtx	column	960	1020	1	1
10	Demand_Data\\10hov-16.17.mtx	column	960	1020	1	2
11	Demand_Data\\11heavytruck-16.17.mtx	column	960	1020	1	3
12	Demand_Data\\12mediumtruck-16.17.mtx	column	960	1020	1	3
13	Demand_Data\\13sov-17.18.mtx	column	1020	1080	1	1
14	Demand_Data\\14hov-17.18.mtx	column	1020	1080	1	2
15	Demand_Data\\15heavytruck-17.18.mtx	column	1020	1080	1	3
16	Demand_Data\\16mediumtruck-17.18.mtx	column	1020	1080	1	3
17	Demand_Data\\17sov-18.19.mtx	column	1080	1140	1	1
18	Demand_Data\\18hov-18.19.mtx	column	1080	1140	1	2
19	Demand_Data\\19heavytruck-18.19.mtx	column	1080	1140	1	3
20	Demand_Data\\20mediumtruck-18.19.mtx	column	1080	1140	1	3

The screenshot above shows an example of working with existing demand files.
There are 20 hourly demand files exported from VISUM, and all those MTX files are put under a subfolder of Demand_Data. There are 20 records in the above input_demand_meta_data.csv, and each record aims to map the data content to the overall time-dependent demand matrices used in DTALite. The first record specifies the following information:
(a) the demand file name (e.g., sov_14_15.mtx)
(b) format (e.g., column)
(c) the loading start time and end time for the demand file (e.g. 840 to 900, or 2PM to 3PM)

(d) the demand types associated with the demand file (only demand_type1), and demand types 1, 2 and 3 are SOV, HOV and truck, respectively.

3.3. Preparing a Traffic Signal Control Representation

DTA simulation needs a realistic signal timing data. We suggest a step-by-step process to construct the signal timing data as follows.

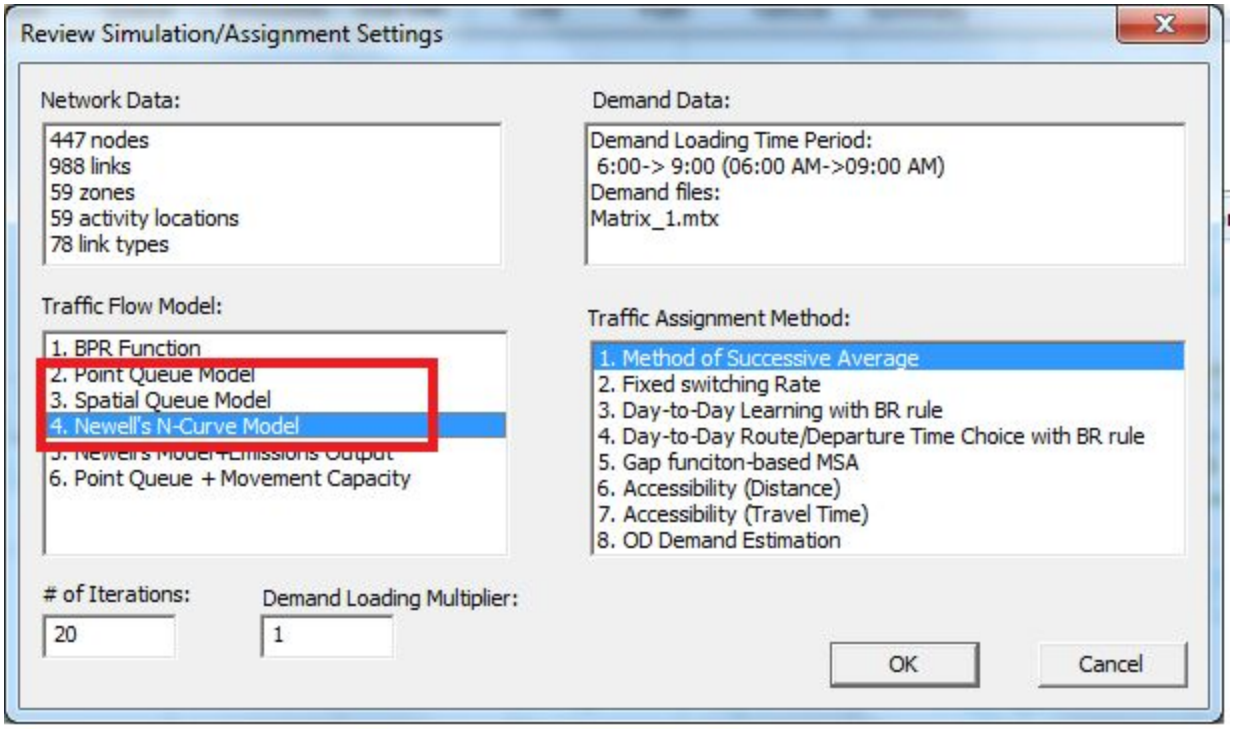
1) Use the capacity value from static traffic assignment, use the point queue model to perform iterative DTA using MSA. We do not identify signal locations at this step. Check if all vehicles can be discharged from the network. Check link volume of incompleted trips.

3.3.1. Start from the Scratch

Simple descriptions:
1) Identify the locations of signalized intersections.
2) In file input_node.csv, change control_type of a signalized node to pretimed signal (typical value = 5). Assign a cycle time in second for this type (a typical value = 110 seconds).
3) In file input_link.csv, update the links with this node as the downstream node with a positive value of effective green time, for example 60 second.
Related learning document can be found here (task 8:).

3.3.2. Use a Capacity-Constrained Queue Model by Extending Static Traffic Assignment

One can simply use the capacity value from the static traffic assignment model, without identifying the signal locations and specifying the link-based signal timing data. The user can select one of queue-based traffic assignment models, highlighted in red below to capture capacity-constrained dynamic congestion effects.



3.3.3. Approximate Signal Timing through Existing Settings for Static Traffic Assignment

Simple descriptions:
1) configure import_GIS_settings.csv when importing existing GIS, and use the following settings:

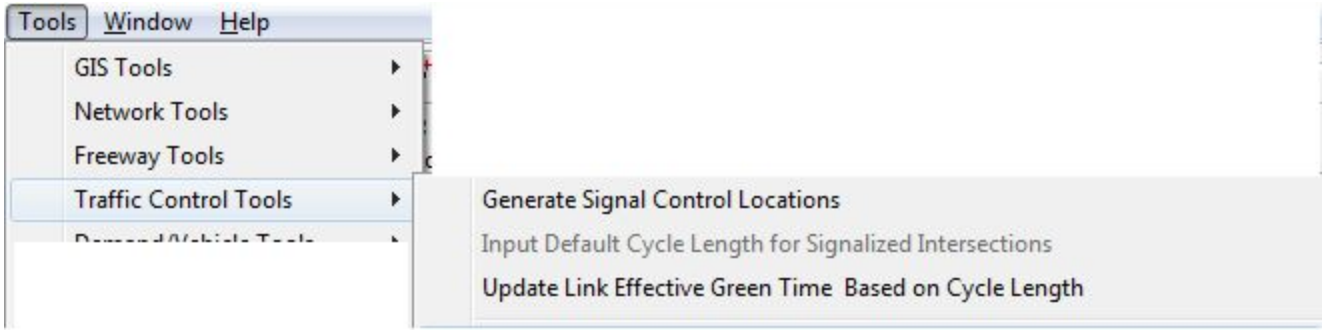
identify_signal_intersection = yes;
minimum_speed_limit_for_signals = 28 (seconds)
maximum_speed_limit_for_signals = 60 (seconds)
default_cycle_length_in_second = 110 (seconds)
default_saturation_flow_rate_in_vhc_per_hour_per_lane = 1800 per hour per lane.
2) When importing the GIS shape file data, NeXTA will identify a node with the following two conditions as signalized intersections and put a default cycle length of 110 seconds.

(a) a node with more than 3 incoming directed links, and (b) at least one of links has a speed limit between 28 mph and 60 mph.

This process will set the control type and cycle length in file input_node.csv, and set a default effective green time for each link with its downstream node as a signalized intersection.

The link effective green time = lane capacity/default_saturation_flow_rate * default cycle length.

THe above functions can be also executed through NeXTA menu -> tools -> traffic control tools.

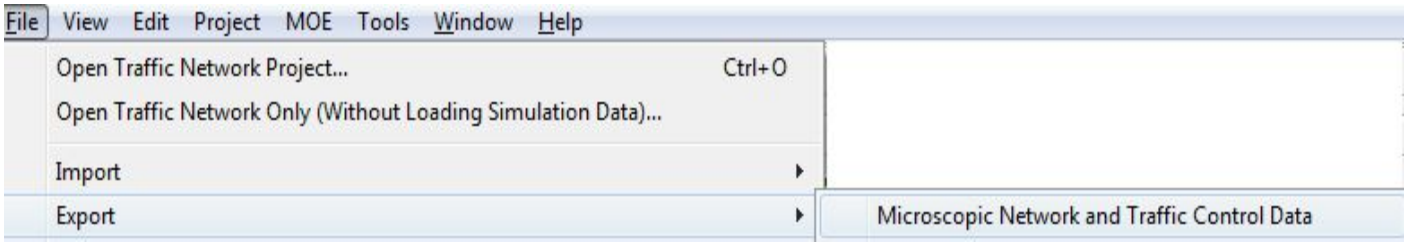


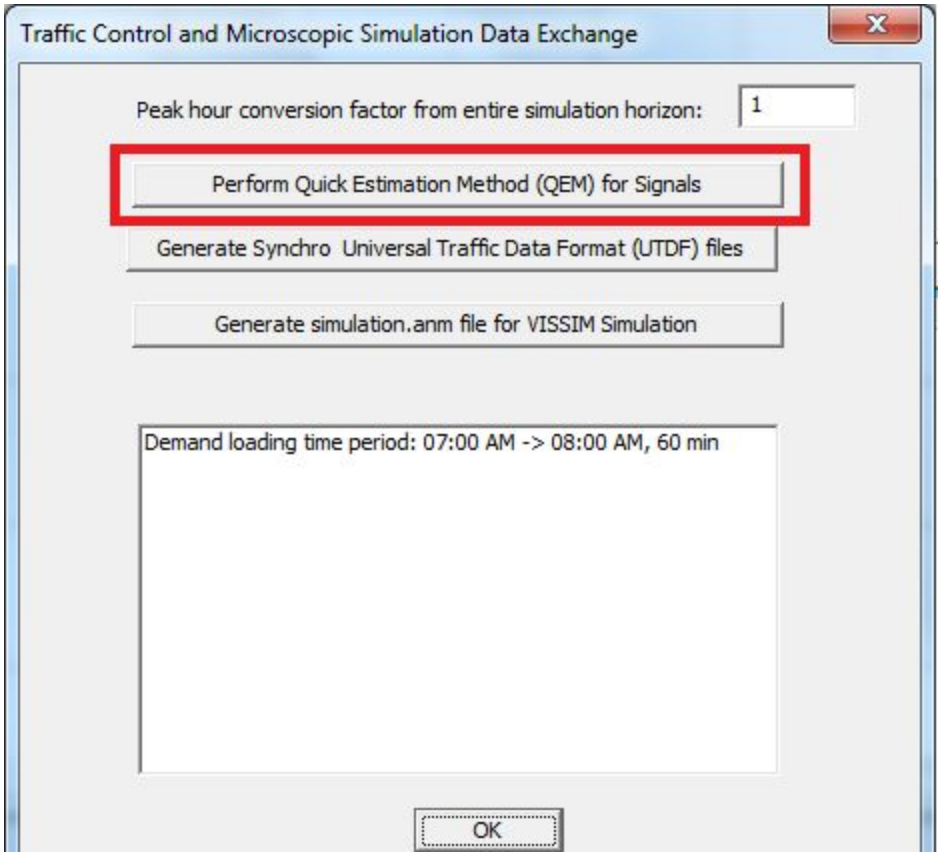
3) Check the locations of signalized intersections can be visualized through Google Earth through NeXTA menu -> export -> export signal node KML file.

3.3.4. Generate movement-specific capacity through Quick Estimation Method

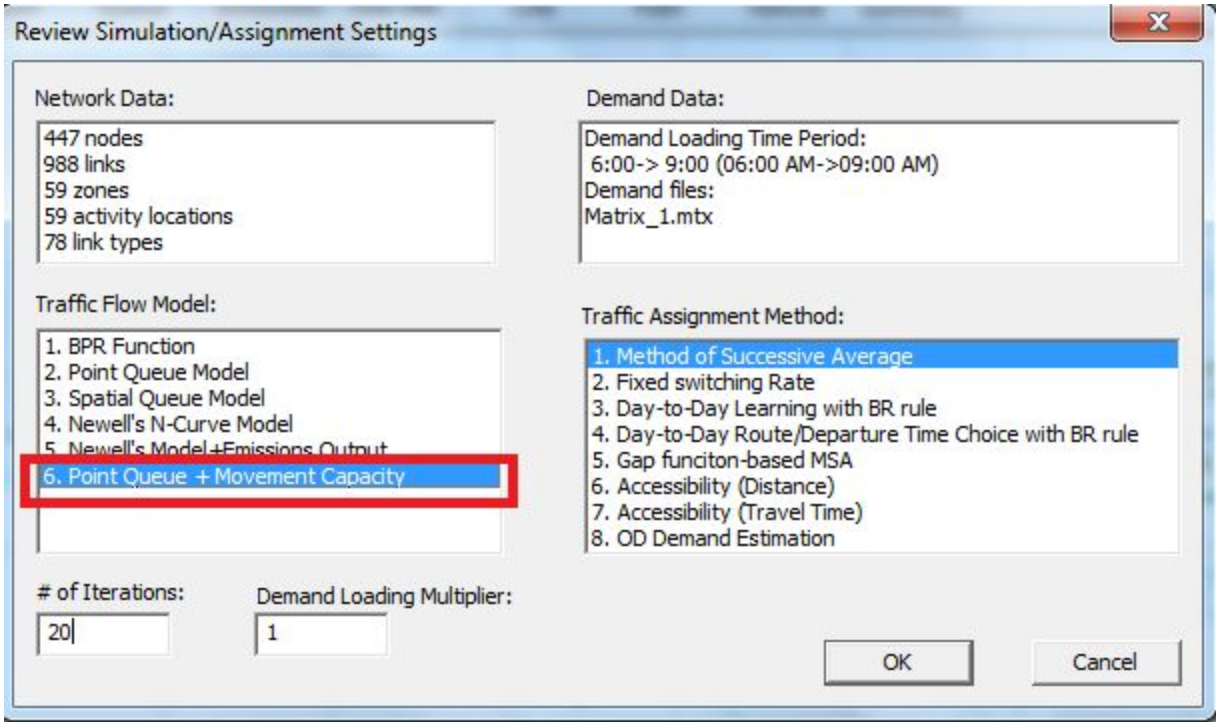
Simple descriptions:

1) Run traffic assignment to generate turning movement volume
2) Run the QEM tool through NeXTA menu -> export



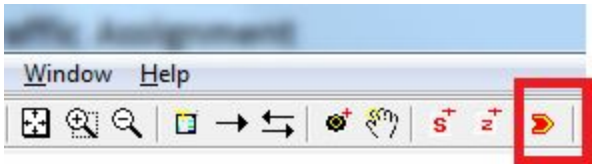


3) Run traffic assignment using the following Point queue+ movement capacity settings.

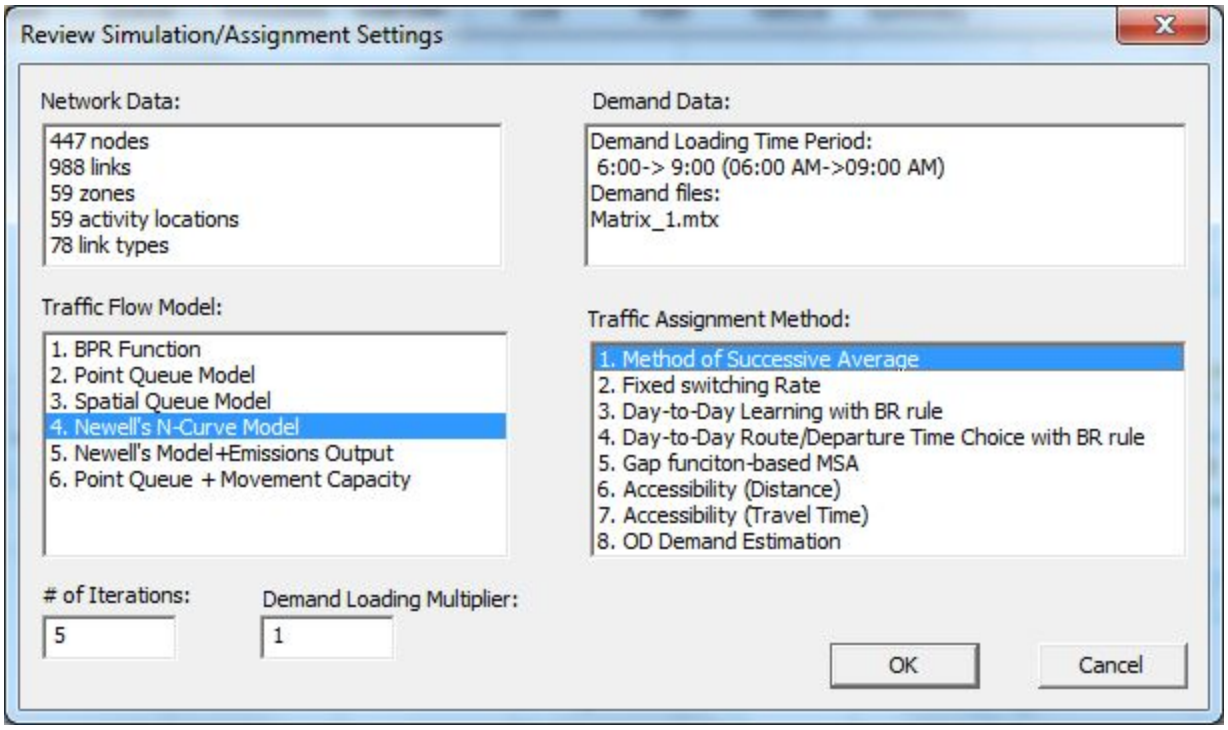


3.4. Running Traffic Assignment

After all network and demand data are ready,, a user can run the simulation by pressing the Start simulation button located in the toolbar menu.



The popup window that appears shows the defined settings, and allows a selection of the traffic flow model, traffic assignment method, the number of iterations, and the demand loading multiplier, as shown below. When the selections are made, pressing OK starts the DTA simulation.



Simulation/Assignment settings window

To understand the basic modeling principle of DTALite, please read the presentation here.

The following table describes the basic data input and output for each traffic flow model in DTALite.

Traffic Flow Model	Data Requirements	Typical output	Remark
1. BPR function	capacity, alpha and beta values from BRP function	time-invariant link and path travel time	Please try Braess Network and learning document
2. Point queue model	flow capacity	time-dependent link volume and density, but jam density constraint is not imposed	
3. Spatial queue model	flow capacity, jam density on links	time-dependent link volume, density and queue length, queue spillback is due to density>=jam density on downstream link	
4. Newell's N-Curve Model	flow capacity, jam density and backward wave speed on links	time-dependent link volume, density and queue length queue spillback on arterial streets is due to density>=jam density on downstream link queue spillback on freeway links is due to backward wave propagation	Newell's kinematic wave model is a link-based equivalent of cell transmission model. Newell's model can accurately capture shock wave propagation. Please read white paper here for modeling details.
5. Newell's N-Curve Model + Emissions Output	flow capacity, jam density and backward wave speed on links Data input for MOVES Lite	similar to model 4 with additional vehicle-specific emission statistics.	Please read the presentation here for modeling details about integration of DTALite and MOVES Lite

6. Point Queue+ Movement Capacity	Correct # of lanes and capacity values per movement at AMS_movement.csv	Movement-specific delay consistent to HCM	Please read learning document here for step-by-step instructions.

DTA was performed with DTALite assignment engine (directly accessed in NeXTA). Simulation settings should be edited in the *input_scenario_settings.csv* file prior to initiating the assignment engine (e.g., selecting Simulation from one of the NeXTA Toolbars).
DTALite is fairly efficient. For the Portland regional network with 15 simulation runs for about 1,100,000 vehicles, DTALite took 1 hour 13 minutes computational time on an Intel Core I7 2760QM (2.4-3.5 GHz quad core) with 32GB RAM. Assignment resulted in an average travel time of 26.18 minutes with an average trip length of 7.72 miles.

4. Task-specific Workflow

Required file and essential steps for different applications

4.1. OD demand estimation
    a) static OD demand (learning document here)
    b) dynamic OD demand

4.2: synchro data export
    layout, phasing data

4.3: path-based travel time comparison
input_path, TMC_speed

5. Analyzing Traffic Assignment Results

5.1. Learning Documents on NeXTA's Visualization Capabilities

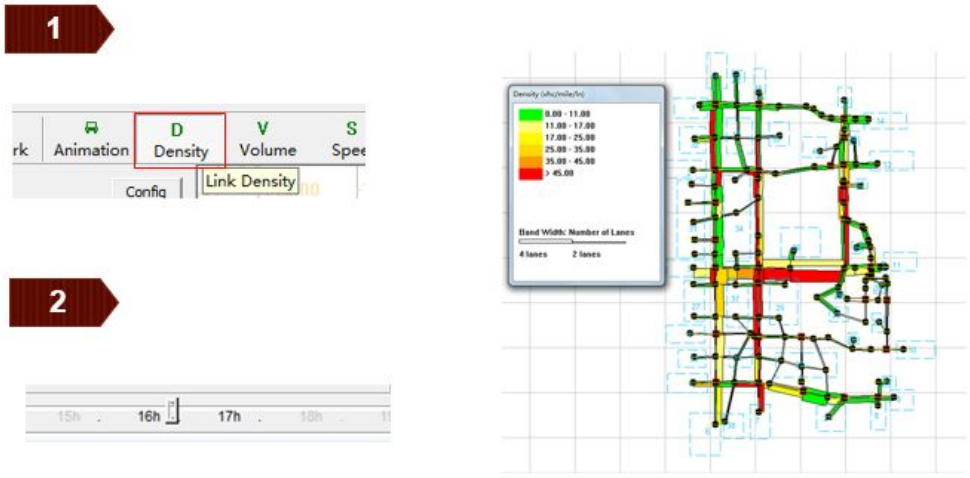
1. Basic control and interfaces of NeXTA
2. Understanding demand-supply Interaction by Reconstructing Braess's Paradox
3. Visualizing space-time congestion contour on a 3-corridor network
4. Visualizing Travel Time Reliability through Vehicle Data Analysis

5.2. Visualization for network-level time-dependent simulation results

The data source of visualization of Density, Volume, Speed, Queue, Impact, Bottleneck and Emissions in NeXTA is **output_linkTDMOE.csv.** In this file, all simulation results are aggregated at each link at each minute.

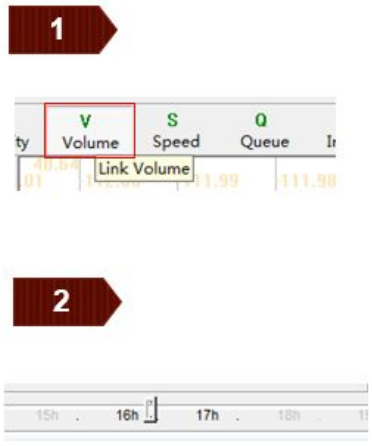
Density:

(1)Click on the “Density” button in the toolbar to show link volume; (2) Use the slider in the toolbar to adapt timestamp.



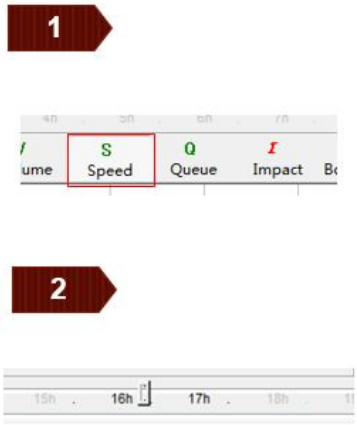
Volume

(1) Click on the “Volume” button in the toolbar to show link volume; (2) Use the slider in the toolbar to adapt timestamp.



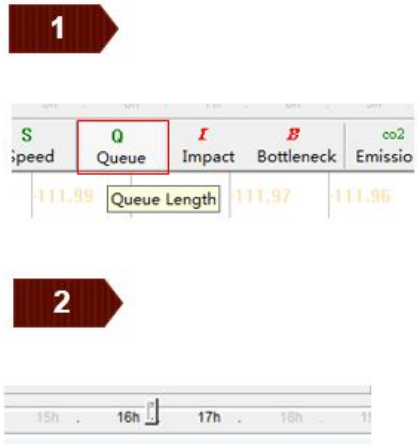
Speed

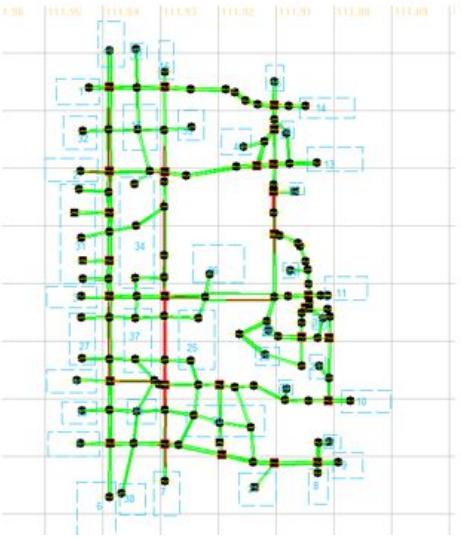
(1) Click on the “Speed” button in the toolbar to show link volume; (2) Use the slider in the toolbar to adapt timestamp.



Queue

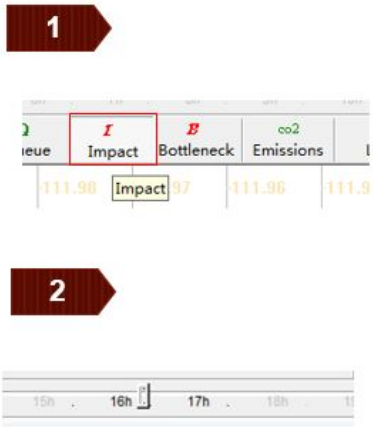
(1) Click on the “Queue” button in the toolbar to show link volume; (2) Use the slider in the toolbar to adapt timestamp.

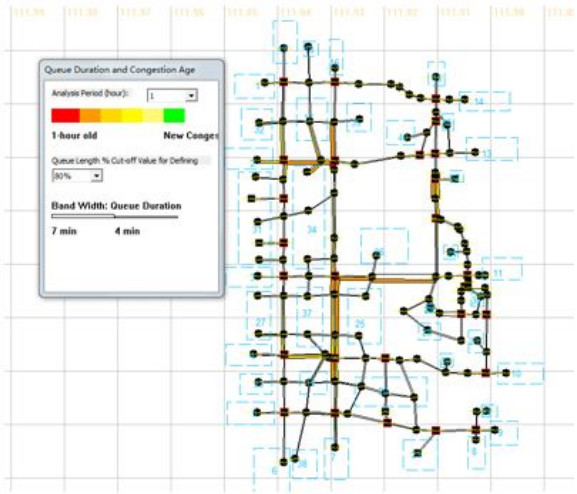




Impact

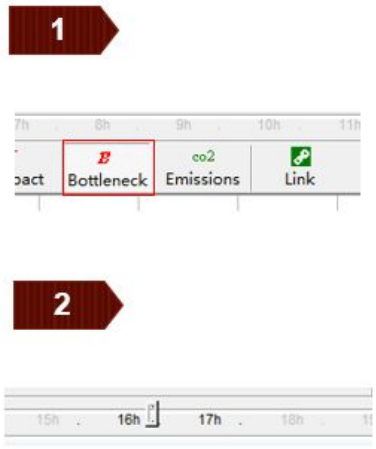
(1) Click on the “Impact” button in the toolbar to show link volume; (2) Use the slider in the toolbar to adapt timestamp.

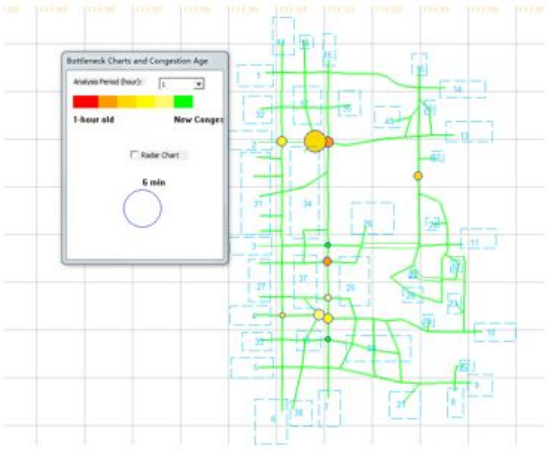




Bottleneck

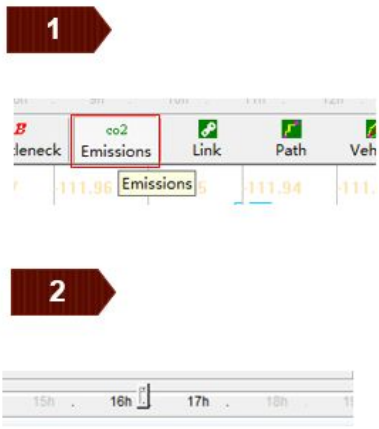
(1)Click on the “Bottleneck” button in the toolbar to show link volume; (2) Use the slider in the toolbar to adapt timestamp.

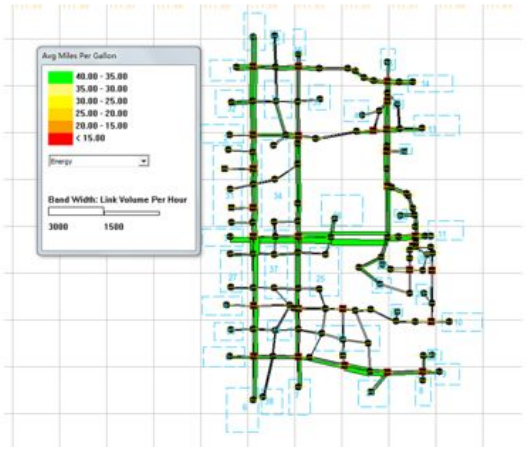




Emissions

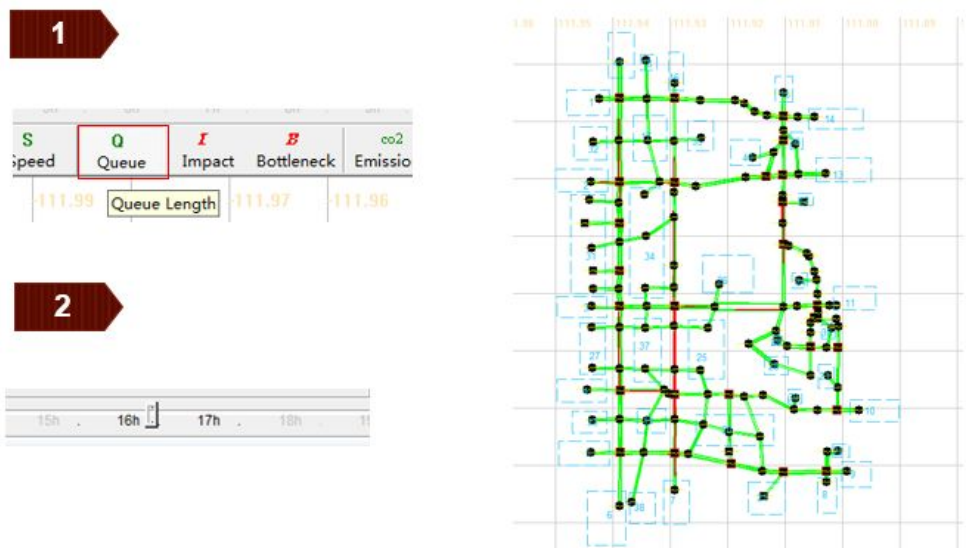
(1)Click on the “Emissions” button in the toolbar to show link volume; (2) Use the slider in the toolbar to adapt timestamp.





The data source of visualization of Animation in NeXTA is **output_agent.bin.**

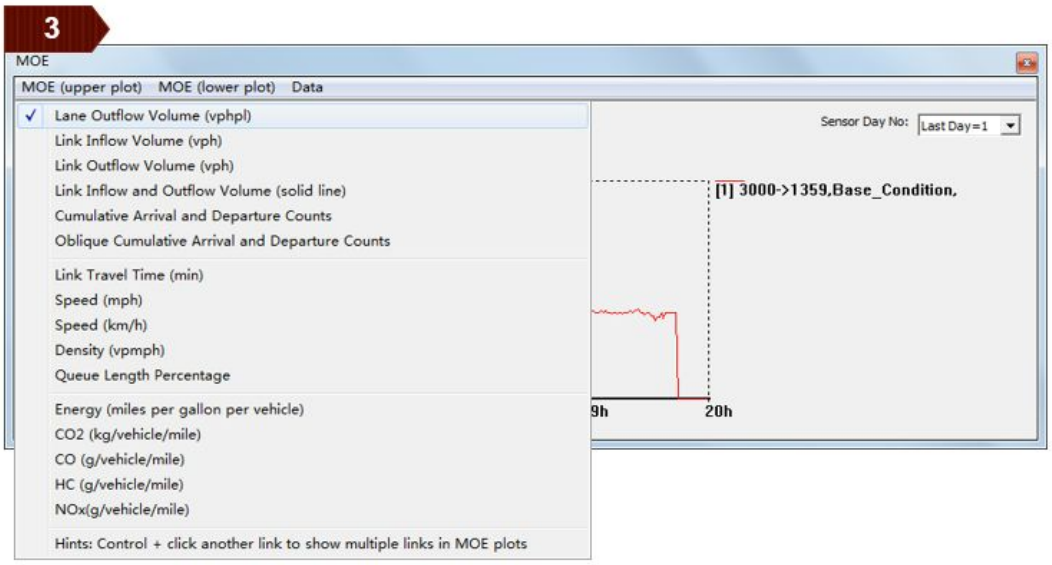
(1)Click on the “Animation ” button in the toolbar to show link volume; (2) Use the slider in the toolbar to adapt timestamp.



5.3 Visualization for link-level/path-level time-dependent simulation results

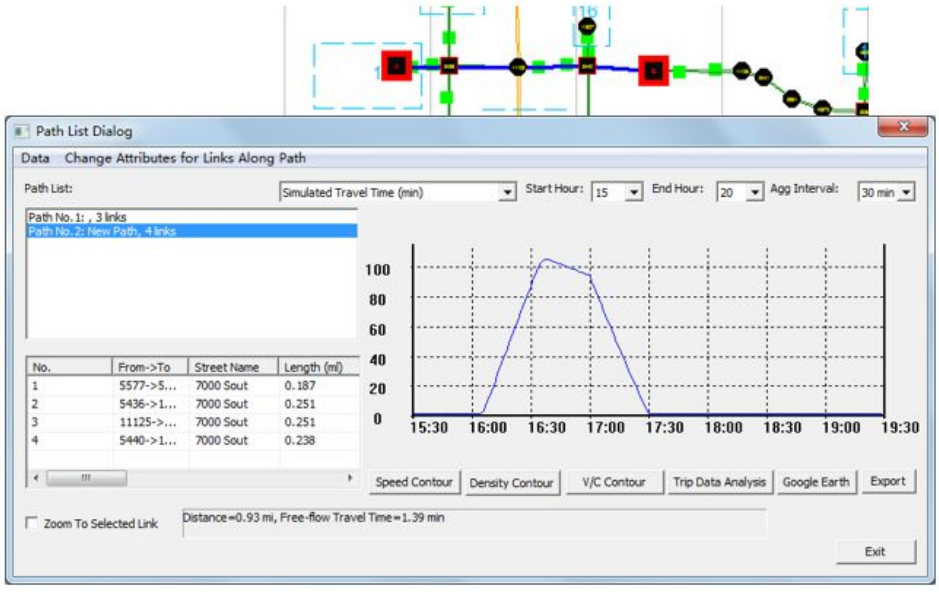
Link MOE (link inflow volume, link outflow volume, link travel time, speed, density, queue length percentage, emissions). The data source is from **output_LinkTDMOE.csv.**

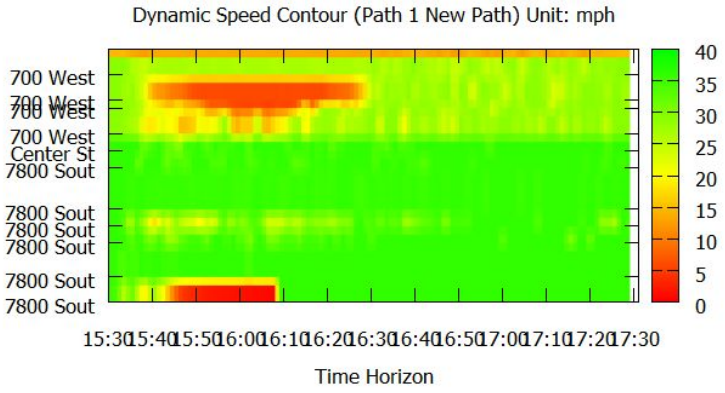
(1)Choose Link MOE layer in GIS layer panel; (2) Choose a link on the display view; (3)Choose different MOEs from drop-down menu.



Path-level MOE (travel time, speed, density, V/C) .The data is still extracted from **output_LinkTDMOE.csv.**

(1)Click the checkbox of “Path” in GIS layer panel;(2) Choose the from_node of your path and right click the "Direction from here";(3) Choose the to_node of your path and right click the "Direction to here";(4) Click on the path in the “Path List” box in the “Path List Dialog” window, and the corresponding path is selected in the display view. (5) you can also click the “speed contour”, “density contour” and “V/C contour” to show them in Gnuplot.





5.4 Visualization for Network-level/link-level/OD-level statistics

The data source is from output_summary.csv, output_trip.csv, output_linkTDMOE.csv, output_agent.csv and so on.

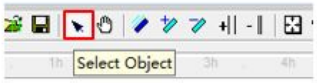
(1) network-level: (1) Click on the “Summary” button in the toolbar to open “Data Summary Dialog”; (2) Choose x and y axis to plot various summary chart.

(2) Link-level: (1)Choose Link layer in GIS layer panel; (2) Choose “Select Object” button in the toolbar; (3) Select the link you want to analyze; (4) Right-click on the display view and click “Perform Vehicle Statistic Analysis on Selected Link”; (5) Choose the data for x axis and y axis from the drop-down list.

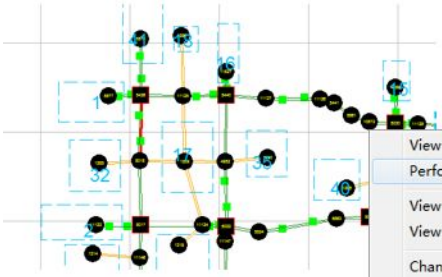
1


Config
Layer
Node
Link
Movement/Signal
Zone
Connector/Activity Location
OD Matrix
Link MOE
Path
Subarea
Sensor
Workzone
Incident
VMS
Toll
Ramp
Vehicle Trajectory
Transit & Walking
Grid/Coordinate
Background Image

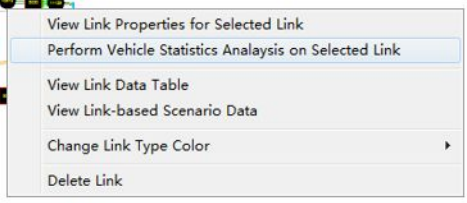
2

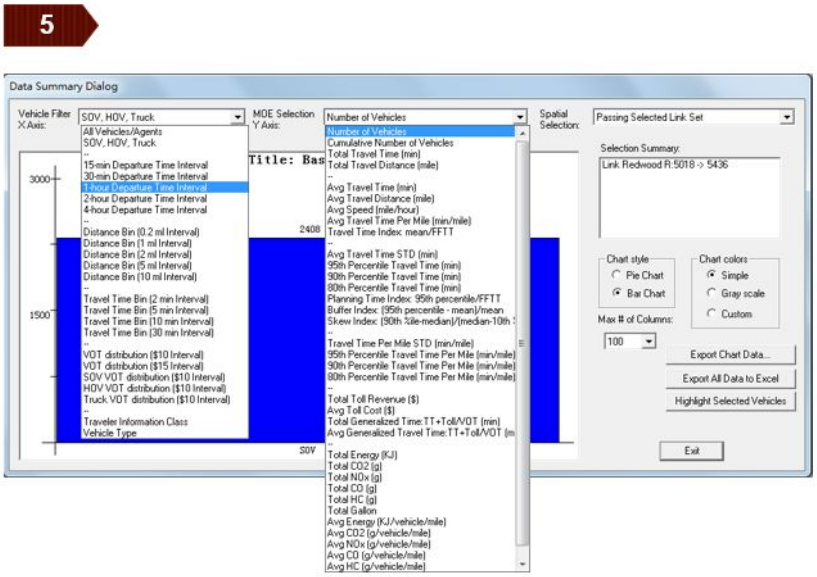

Select Object

3

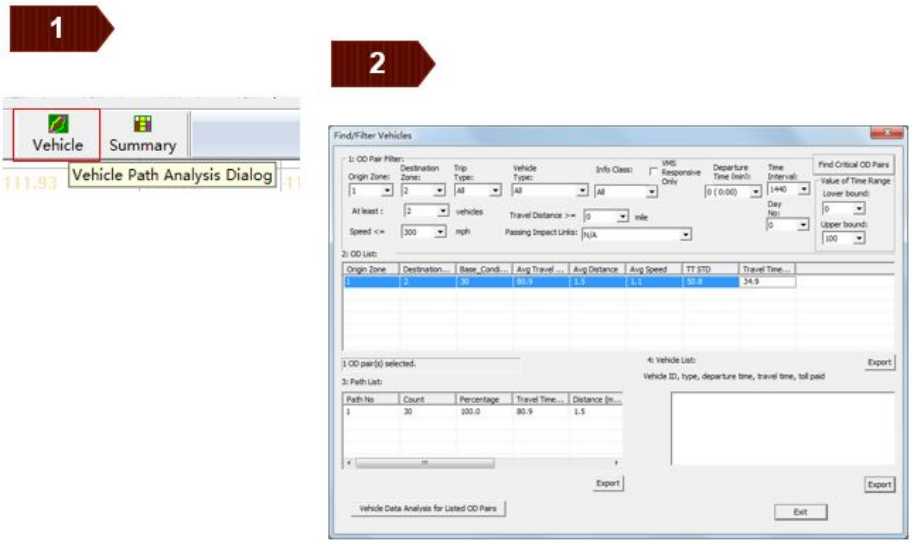


4

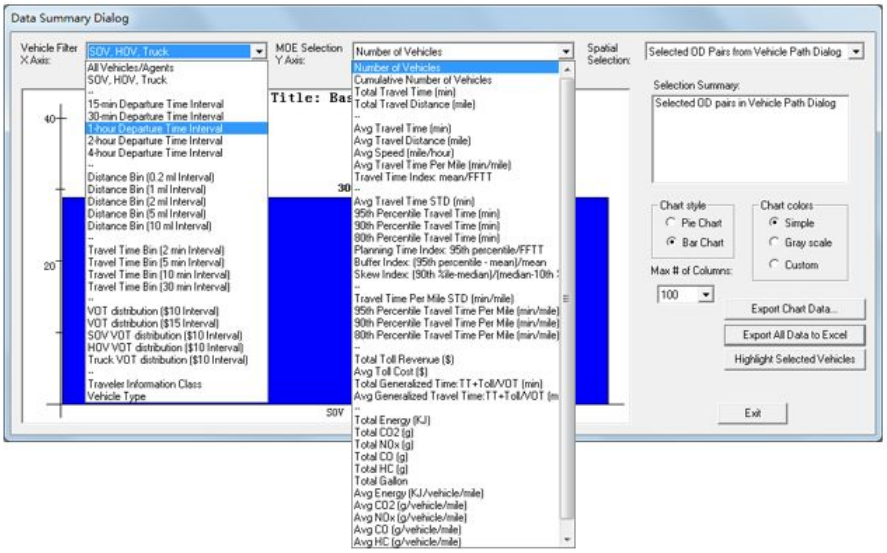

View Link Properties for Selected Link
Perform Vehicle Statistics Analaysis on Selected Link
View Link Data Table
View Link-based Scenario Data
Change Link Type Color
Delete Link



(3). OD-level: (1) Click on the “Vehicle” button in the toolbar; (2) In the “Find/Filter” window, choose the OD information in “1: OD Pair Filter”, click the OD record in the “2: OD List” and click on “Vehicle Data Analysis for Listed OD Pairs”; (3)Choose the data for x axis and y axis from the drop-down list.



3


Data Summary Dialog
Vehicle Filter
X Axis:
SOV, HOV, Truck
All Vehicles/Agents
SOV, HOV, Truck
15-min Departure Time Interval
30-min Departure Time Interval
1-hour Departure Time Interval
2-hour Departure Time Interval
4-hour Departure Time Interval
Distance Bin (0.2 mi Interval)
Distance Bin (1 mi Interval)
Distance Bin (2 mi Interval)
Distance Bin (5 mi Interval)
Distance Bin (10 mi Interval)
Travel Time Bin (2 min Interval)
Travel Time Bin (5 min Interval)
Travel Time Bin (10 min Interval)
Travel Time Bin (30 min Interval)
VOT distribution ($10 Interval)
VOT distribution ($15 Interval)
SOV VOT distribution ($10 Interval)
HOV VOT distribution ($10 Interval)
Truck VOT distribution ($10 Interval)
Traveler Information Class
Vehicle Type
MOE Selection
Y Axis:
Number of Vehicles
Cumulative Number of Vehicles
Total Travel Time (min)
Total Travel Distance (mile)
Avg Travel Time (min)
Avg Travel Distance (mile)
Avg Speed (mile/hour)
Avg Travel Time Per Mile (min/mile)
Travel Time Index: mean/FFTT
Avg Travel Time STD (min)
95th Percentile Travel Time (min)
90th Percentile Travel Time (min)
80th Percentile Travel Time (min)
Planning Time Index: 95th percentile/FFTT
Buffer Index: (95th percentile - mean)/mean
Travel Time Per Mile STD (min/mile)
95th Percentile Travel Time Per Mile (min/mile)
90th Percentile Travel Time Per Mile (min/mile)
80th Percentile Travel Time Per Mile (min/mile)
Total Toll Revenue ($)
Avg Toll Cost ($)
Total Energy (KJ)
Total CO2 (g)
Total NOx (g)
Total CO (g)
Total HC (g)
Total Gallon
Avg Energy (KJ/vehicle/mile)
Avg CO2 (g/vehicle/mile)
Avg NOx (g/vehicle/mile)
Avg CO (g/vehicle/mile)
Avg HC (g/vehicle/mile)
Title: Bas
40
20
30
SOV
Spatial
Selection:
Selected OD Pairs from Vehicle Path Dialog
Selection Summary:
Selected OD pairs in Vehicle Path Dialog
Chart style
Pie Chart
Bar Chart
Chart colors
Simple
Gray scale
Custom
Max # of Columns:
100
Export Chart Data...
Export All Data to Excel
Highlight Selected Vehicles
Exit

6. References

Meso-scopic simulation-assignment methodology

Hani S. Mahmassani, Ta-Yin Hu, Srinivas Peeta, and Athanasios Ziliaskopoulos. Development and testing of dynamic traffic assignment and simulation procedures for ATIS/ATMS applications. Report DTFH61-90-R-00074-FG, U.S. DOT, Federal Highway Administration, McLean, Virgina, 1994.

Mahmassani, H. S.Dynamic Network Traffic Assignment and Simulation Methodology for Advanced System Management Applications, Networks and Spatial Economics, Vol. 12, 2001, pp. 267-292.

Jayakrishnan, R., H. S. Mahmassani, and T.-Y. Hu. An Evaluation Tool for Advanced Traffic Information and Management Systems in Urban Network, Transportation Research C, Vol. 2C, 1994, pp. 129-147.

Newell's traffic flow model

Newell, G. F., 1993a. A simplified theory on kinematic waves in highway traffic, part I: general theory. Transportation Research Part B, Vol. 27(4), pp. 281-287.

Newell, G. F., 1993b. A simplified theory on kinematic waves in highway traffic, part II: queueing at freeway bottlenecks. Transportation Research Part B, Vol. 27(4), pp. 289-303.

Newell, G. F., 1993c. A simplified theory on kinematic waves in highway traffic, part III: multi-destination flows. Transportation Research Part B, Vol. 27(4), pp. 305-313.

Time-dependent shortest path algorithm

Ziliaskopoulos, A. K. and Mahmassani, H. S., 1993. Time dependent shortest-path algorithm for real-time intelligent vehicle highway system applications. Transportation Research Record, 1408, pp. 94-100.

Dynamic user equilibrium solution algorithm

Lu, C-C., Mahmassani, H.S. and Zhou, X. (2009) Equivalent Gap Function-Based Reformulation and Solution Algorithm for the Dynamic User Equilibrium Problem. Transportation Research Part B. Vol. 43(3), pp. 345-364

DTA user surveys and DTA primer from TRB Network Modeling Committee.

ADB30 website (check special topics section):

http://www.nextrans.org/ADB30/index.php?option=com_content&view=article&id=28&Itemid=33