Chapter 6

MicroSimulator

MICROSIMULATOR

Chapter Outline

The TRANSIMS Microsimulator module executes individuals' travel plans, link by link, as provided by the Route Planner module at the start-time specified by the plan. Plans overlapping in time are executed simultaneously by the Microsimulator, resulting in overall transportation dynamics along the intermodal networks.

The Microsimulator uses the TRANSIMS network, the vehicle data and the intermodal plans in executing the simulation process over the intermodal networks. Figure-6.1 shows the inputs and the outputs to this module, along with its interactions with other relevant modules of TRANSIMS.

This chapter begins with an overview, then discusses the terminology used for the Microsimulator and provides the key concepts used in the process. It also highlights the major inputs and outputs to the module, including the various module interfaces, the output visualizer and the values for relevant configuration files.

The chapter ends with a detailed discussion of the set of rules governing the movement of vehicles and travelers in the network, including the inner workings of the Microsimulator in terms of preparing for a time update and for conducting parallel computation.

6.1 Overview

The TRANSIMS Microsimulator module simulates the movement and the interactions of travelers in the transportation system of the study area. In this module every traveler tries to execute his/her travel movements according to plans generated from the Route Planner Module. These movements and interactions produce key data that is output from the Microsimulator.

The TRANSIMS Microsimulator is capable of handling intermodal travel plans, multiple travelers per vehicle, multiple trips per traveler and vehicles with different operating characteristics. Realistic traffic behavior is simulated in the Microsimulator by making decisions about lane changes, passing slow vehicles and evaluating interactions with other vehicles.

The amount of computation necessary for simulating a large transportation network at a level of detail down to each individual traveler and vehicle is huge. Hence, a coarse simulation approach referred to as "Cellular Automata" (CA) is used to keep up with a fast computational speed necessary to simulate a whole region. The TRANSIMS Microsimulator also provides the capability of using multiple computer processors (CPUs) to support a large number of travelers and a considerably sized transportation network.

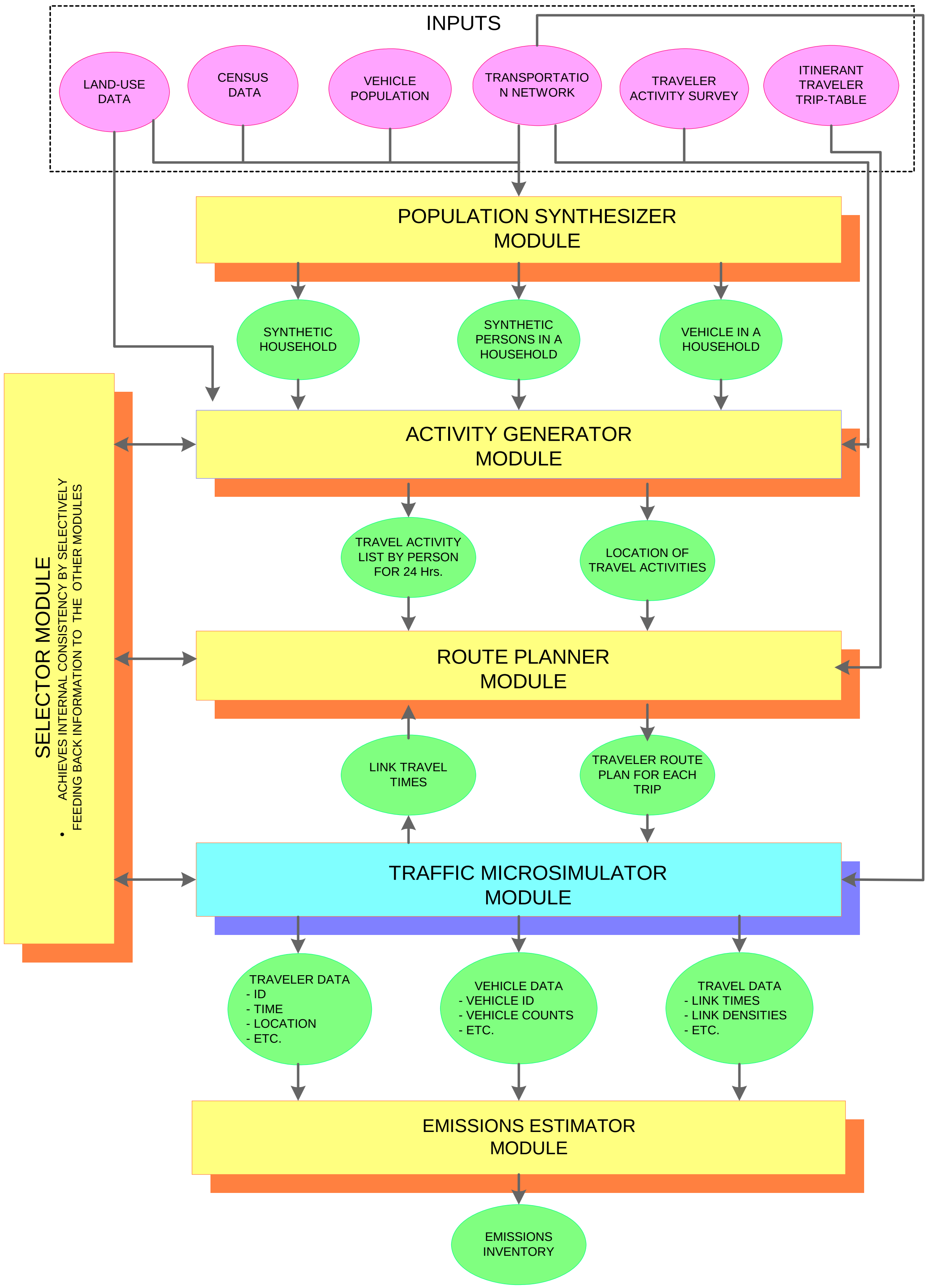


Figure-6.1: TRANSIMS Framework

The Cellular Automata approach essentially divides every link on the network into a finite number of cells. At every time step each of these “cells” is scanned for a vehicle presence. If a vehicle is present, the vehicle position is advanced to another cell using a simple rule set. The rule set is made simple to increase the computational speed necessary for a large simulation.

Reducing the size of the “cell”, expanding the rule set and adding vehicle attributes increases the fidelity of the Microsimulator but would greatly affect the computational speed. The size of 7.5 meters length and a traffic lane in width is chosen as a default value for the “cell”.

The input to the Microsimulator includes data about the transportation network, vehicles, transit and traveler plans. The outputs from the Microsimulator include spatial, temporal summary data (e.g., densities, travel times etc.), snapshot data and traveler event data as shown in Figure-6.2.

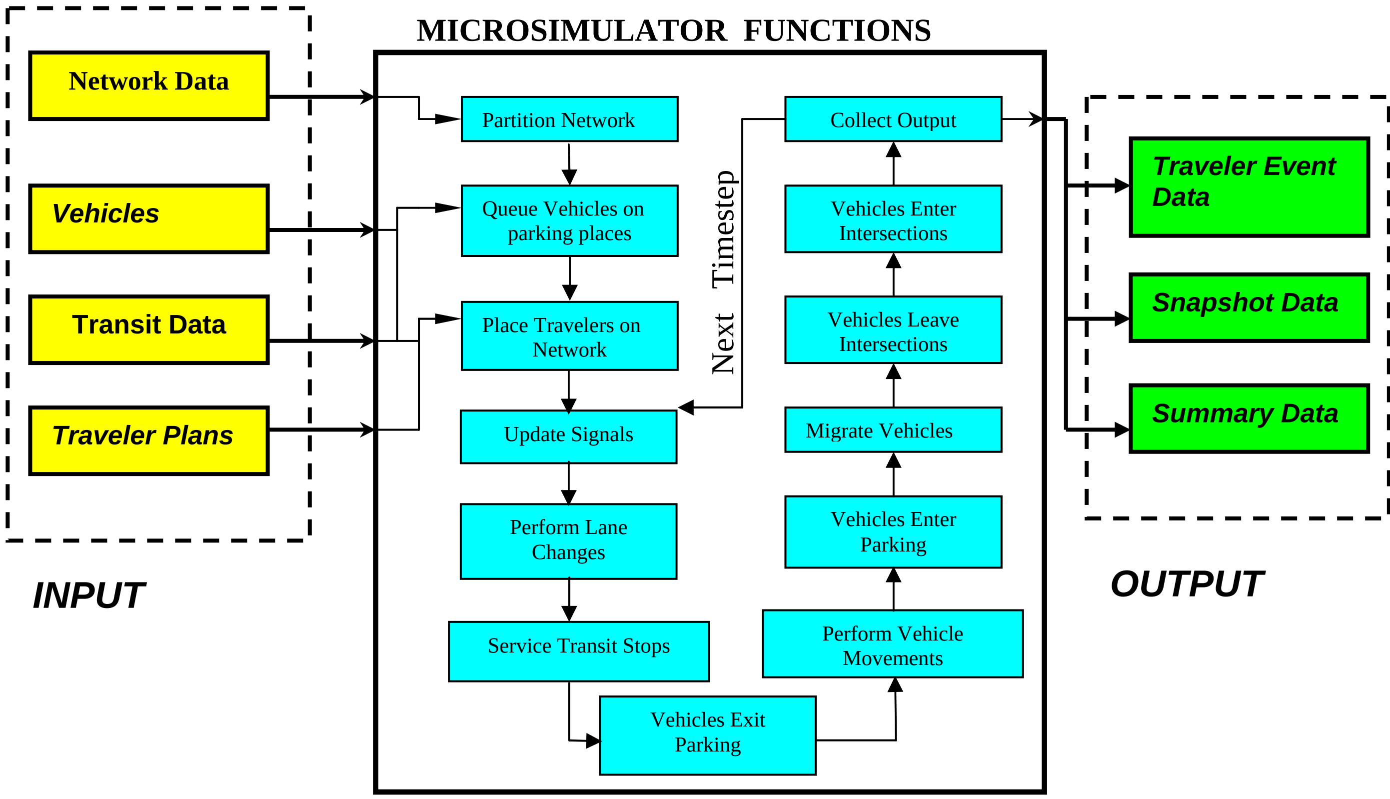


Figure-6.2: Overview of Microsimulator

A brief explanation of the Microsimulator is also presented in Figure-6.2 and is explained in detail in the following sections. The Microsimulator first reads in the network data, which includes details such as links, nodes, lane connectivity, parking locations etc. In the next step the vehicle type, location and traveler plans are read. Then, travelers and vehicles are placed on the network. Step three includes the movement of the travelers and vehicles on the network in tandem with their travel plans using the Cellular Automata approach.

6.2. Terminology

- **Traveler:** A person making or who has to make a trip on the transportation network to accomplish activities.

- **Accessory:** A buffer place to store travelers and vehicles when not active on the transportation network. (E.g., parking locations, activity locations, transit stops)

- **Trip:** A set of contiguous legs of travel constitutes a trip. The different legs of a trip may have different modes. Every trip starts and ends at an activity location.

- **Traveler Plan:** A sequence of trips by a traveler.

- **Vehicle:** A motorized means by which a traveler moves in the transportation network. Non-Motorized means of travel such as by foot or by bicycle do not require a vehicle.

- **Vehicle Type:** A particular category of vehicles classified by some common basis that affects the simulation, such as performance characteristics (e.g., length, acceleration profile); and emission type (e.g., power/weight ratio). They may also be classified by network type: (e.g., definitions used in imposing lane use or turn prohibition restrictions) or by usage (e.g., transit, private auto, carpool, jitney).

- **Vehicle ID:** A unique identifier for every vehicle created during the simulation.

- **Lane:** A place where traffic flows on a link. Lanes on each side/direction of the link are numbered separately, starting with lane number one as the leftmost lane (relative to the direction of the travel). Each successive lane to the right is numbered one greater than its predecessor. Pocket lanes (i.e., turn pockets, merges, and pull-outs) are numbered in sequence, even if they do not exist for the full length of the link. A two-way left turn lane, if present, is considered lane number zero.

- **Cellular Automata:** The theory behind dividing the transportation network into individual cells.

- **Grid:** The division of the link into cells forms a grid as shown in Figure-6.3. The Microsimulator uses a separate grid for each lane on the roadway.

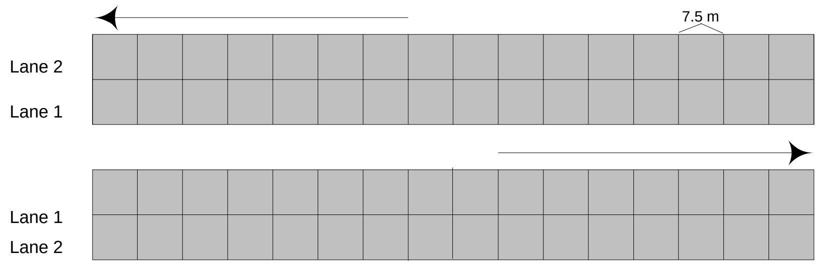


Figure-6.3: Link with Grid Cells

- **Cell:** The smallest unit of division on a grid. The Microsimulator uses a cell size of 7.5 meters long. Vehicles may occupy more than one cell of the grid (e.g., transit vehicles).

- **Time step:** One microsimulation update cycle in which all movements and lane changes are executed for each vehicle. Typically each time step represents one second of simulation time.

- **Gap:** The number of empty cells between a vehicle and the next vehicle on the grid as shown in Figure-6.4. There is a gap of 3 cells between Vehicle A and Vehicle C in lane 1 and it is referred as the current lane gap (G_c).

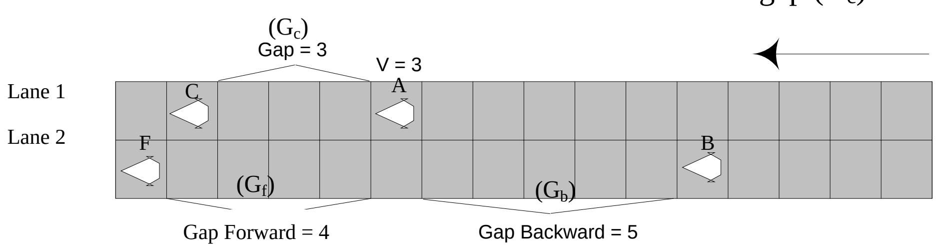


Figure-6.4: Gaps

In case vehicle A needs to make a lane change to lane 2 the gap between it and vehicle B, which is 5 cells, becomes an important consideration. This gap is referred to as gap backward (G_b). Similarly, the lane change rule considers the gap forward (G_f), which is the gap between vehicle A and vehicle F. In this case, the gap is 4 cells.

- **Maximum Speed:** It is the simulation speed limit on the link in cells/time step (V_{max}).

- **Global Maximum Speed:** It is the maximum speed on any link in cells/time step ($V_{GlobalMax}$). It is set in the Microsimulator at 5-cells/time step, which is 135 km/hr (85 m/hr?).

- **Deceleration Probability:** The probability that a vehicle will decelerate during a time step (P_D). It is set using the configuration key CA_DECELERATION_PROBABILITY.
- **Lane Changing Probability:** The probability that a vehicle will change lanes during a time step for reasons other than plan following (P_L). It is set using the configuration key CA_LANE_CHANGE_PROBABILITY.
- **Distance From the Intersection:** The distance where a vehicle starts to consider changing lanes in order to follow its plan (D_{pf}). It is set using the configuration key CA_PLAN_FOLLOWING_CELLS.
- **Random Number:** A number randomly selected between 0.0 and 1.0 (N_{rand}). The random number sequence can be controlled using the configuration keys CA_RANDOM_SEED1, CA_RANDOM_SEED2 and CA_RANDOM_SEED3, which are all combined into a single seed.
- **Transit:** Vehicles traveling on pre-specified routes, stopping at specific accessory locations listed in the Transit Stop network data table, and following a predetermined schedule.
- **Transit Route:** A sequential set of transit stops visited by the transit vehicle. Every route has an integer ID. No transit route may include the same transit stop more than once. The inbound and the outbound portions of a round trip must be assigned different route ID's. Two transit vehicles following the same path through the network but stopping at different places along the path must have different route IDs.
- **Boundary Parking Area:** Places where vehicles leave the network within the study area.
- **Hibernating Travelers:** Travelers who have executed one leg of their plan and are waiting to depart on another.
- **Local Plan:** The origin of the traveler plan must be in an accessory that is a part of the network under the control of that central processing unit (CPU).
- **Active Plan:** A plan in which the expected arrival time is after the simulation start time and the departure time is before the simulation end time.
- **Non-Simulated mode of travel:** Modes of travel that are not explicitly simulated in the Microsimulator (e.g., walk and bicycle).
- **Migrating travelers:** Travelers that are passed on from one CPU dealing with one part of the network to another CPU dealing with another part of the network as the travelers' plans are executed.

- **Traffic Control:** Data at every node represents how lanes are connected across the node and the type of sign, if any, or signalized control that determines who has the right of way.

- **Signal Coordinator:** A device that controls the operation of two or more traffic controls.

- **Unsignalized Node:** Represents a type of sign control, either stop or yield sign.

- **Signalized Node:** Represents a traffic light at an intersection. Each signal has a timing plan and a phasing plan.

- **Phasing Plan:** Specifies the turn protection in effect for transitioning from an incoming link to an outgoing link during a particular phase of a specific timing plan.

- **Timing Plan:** Specifies the duration of the time intervals during the specific phases of a traffic light at a signalized intersection. The timing plan could change from one period to another.

- **Offset:** The time relationship determined by the difference between the beginning of coordinated phase green and a system wide reference point.

- **Pretimed Signal:** A traffic control signal whose display repeats, cycle after cycle, on one or more preset timing plans with no regard to the actual demand on the street.

- **Actuated Signal:** A traffic control signal wherein the timing plan is varied for some or all controlled conflicting movements depending on the vehicular or pedestrian demand as determined by detectors on roadways or near pedestrian crossings.

- **Single Ring Controller:** A signal controller where two or more sequentially timed and individually selected phases are arranged to occur in an established order.

- **Dual Ring Controller**: A Dual Ring Controller is a real time signal control strategy that allows efficient phasing of traffic movements depending on the current traffic conditions.

- **Detector:** A device that identifies the presence or passage of vehicle over an area of the lanes on a link.

- **Study Area and Buffer Links:** The microsimulation distinguishes two types of links in its calculations: Study area links and buffer links. Study area links are the links within the study area boundary. Buffer links typically form a fringe about two links thick around the study area. A simulation includes buffer links in order to avoid edge effects such as when vehicles enter the study area on its boundary. The buffer gives these vehicles time to interact with other traffic and achieve realistic behavior before entering or leaving the study area.

- **Event Data:** Microsimulator output that reports when an event, as specified by the user, occurs for a traveler. Events are recorded as they occur, at irregular time intervals.

- **Evolution Data:** Also known as snapshot data, provides detailed information about how the state of the simulation evolves in time. Evolution data may be recorded on every time step or as desired.

- **Summary Data:** Reports aggregate data about the simulation such as link velocities and link densities. Summary data is sampled, accumulated, and reported periodically throughout the simulation.

6.3. Key Concepts

This section briefly discusses the various concepts that are key to understanding the Microsimulator. Selected ideas and terms listed below give the necessary background to better understand the workings of this module.

6.3.1 Simulation

Simulation is essentially the use of a computer to evaluate a real world system numerically, and to gather data in order to estimate the desired true characteristics of the system. This technique is generally used over other analytical techniques when treating complex systems having more than simple relationships. There are many kinds of simulations but all of them can be broadly classified into Continuous or Discrete simulations.

6.3.1.1 Continuous simulation

Continuous simulation considers the modeling of the system state variables, which change continuously with respect to time. Typically this would involve differential equations that give relationships for the rates of change of the state variables with time.

6.3.1.2 Discrete Event simulation

Discrete Event simulation considers the modeling of the system as it evolves over time by a representation in which the state variables change instantaneously at separate points of time. These points in time are the ones at which an event occurs, which is defined as an instantaneous occurrence that may change the state of the system. Traditionally two methods to advance the simulation time clock have been known as “next-event time advance” and “fixed increment time advance.”

6.3.1.2.1 Next-event time advance

In this approach, the simulation clock is initialized to zero and the times of occurrence of future events are determined. The simulation clock is then advanced to the time of the occurrence of the most imminent of the future events at which the state of the system is updated to account for the fact that an event has occurred, and the times of occurrence of future events is also updated.

6.3.1.2.2 Fixed increment time advance

In this approach, the simulation clock is advanced in increments of exactly Δt time units for some appropriate choice of Δt. After each update of the clock, a check is made to determine if any events should have occurred during the previous interval of time length Δt. If one or more events were scheduled to have occurred during this interval, these events are considered to occur at the end of the interval and the system state is updated accordingly. The primary use of this approach is for systems where it can reasonably be assumed that all events actually occur at the beginning of one of the time steps. The Microsimulator is a fixed increment time advance type of a simulation where one time step equals one second of real time.

6.3.2 Cellular Automata

The Microsimulator uses the concept of cellular automata; where each section of the roadway is divided into cells of one grid lane wide by 7.5 m long. Each cell contains a vehicle, a part of one or is empty. The simulation is carried out in discrete time steps where each simulation time step is one second of real time. In each time step, a vehicle on the network decides to accelerate, brake, or change lanes in response to the occupancy of the nearby grid cells. After every vehicle is allowed to make these decisions, they are all moved to new grid cells in accordance with their current velocity.

The simulation also guarantees that each vehicle makes decisions based on the state of every other vehicle in its local vicinity at the same time. This implies that every vehicle on the network makes its acceleration decision based only on information available at time t. This does not include the time *t+1* positions of vehicles that already have made their acceleration decision. This parallel update scheme ensures that the simulation results do not depend on the order in which streets in the network are updated.

TRANSIMS simulation model is microscopic. However, to get high computational speed the traffic dynamic rules are kept to a minimum. This is an explicit trade-off between realism and simulation speed or between fidelity and resolution. While the Microsimulator is a low fidelity, microscopic traffic simulation model based on very simple rules describing driver behavior, empirical studies have shown that the Microsimulator yields realistic traffic behavior.

6.3.3 Parallel Computation

Parallel Computation is the effective use of more than one central (or computer) processing unit (CPU) to assist in solving a specific problem. The theory behind Parallel Computation and how to handle it is exhaustive and is beyond the scope of this text. This chapter will highlight some main concepts of parallel computation and how the Microsimulator uses these concepts.

There are many paradigms/architectures for Parallel Computation. The most common of them is the Master/Slave paradigm. This model allows the Master CPU to take charge and to control the flow of the algorithm. It also initiates other CPUs, distributes work among them, looks after the overall flow, and at the end sends termination directions to the other CPUs. Due to the heavy role of this CPU it is aptly named the Master and the other CPUs that receive instructions and perform them are referred to as Slaves.

The TRANSIMS microsimulation uses multiple CPUs where available, in order to deal with the large number of travelers, vehicles and activities. The algorithms in the Microsimulator are also designed to incorporate the use of multiple CPUs. A detailed explanation of the information flow and structure required to support parallel computation is presented later.

The use of multiple CPUs in the Microsimulator employs a divide-and-conquer strategy where the transportation network is split up into different parts and each part is assigned to a specific CPU. The part of the transportation network that is handled by a particular CPU is termed as local to that CPU. The roadway segment that is split up and shared between two CPUs is called the distributed link. Once the transportation network is split each CPU performs the analysis separately, but keeps track of the information at the distributed links.

The Master CPU does the network partitioning and is responsible for the termination of the simulation. The network partitioning is done using the Orthogonal Bisection Algorithm (OBA) or by METIS (a public domain package). The idea behind the partitioning of the network is to distribute the "load" to different processors in ratio to their capacities. These algorithms use a cost function to estimate where the network has to be partitioned. The cost functions typically reflect how much processor time is necessary for computations, and the partition occurs in such a way that there is more or less uniformity in the costs associated with each partitioned network. The cost function typically depends on a lot of factors and varies for different situations. The cost function for a signalized intersection may be higher than an unsignalized one. The cost function also depends on the amount of processing required for vehicles on that part of the network, which can only be obtained after a simulation. When no other information is available the costs could be based on the number of cells in a link.

An analysis of how the partitioning occurs is presented in figures-6.5 and 6.6. It can be seen from figure-6.5 that no information other than the length of the links on the network is known. Hence, the cost function for each link is assumed to be proportional to the number of cells on the link. Considering that the network has to be partitioned into two sub-networks a suitable place for the partition is sought.

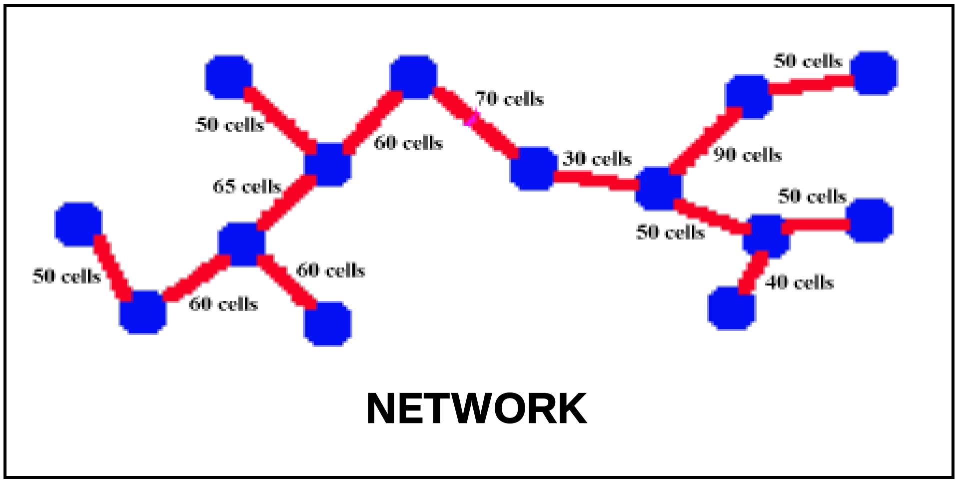


Figure-6.5: An Example Network

By looking at the network and by some trial and error it can be seen that the network can be split at the link containing 70 cells (shown in Figure-6.6) in order to create approximately equal number of cells on each subdivided network. This split gives the least difference in the number of cells between both networks. A break in any other link would only increase the imbalance in the number of cells between the two networks. Hence, network partitioning occurs at the middle of the link containing the 70 cells. It should be noted that the split only occurs in the middle of a link and cannot occur at nodes or any other place on a link. This requires a minimum information transfer from one CPU to another and the only information to be taken care of are vehicle movements in the same lane and lane change procedures.

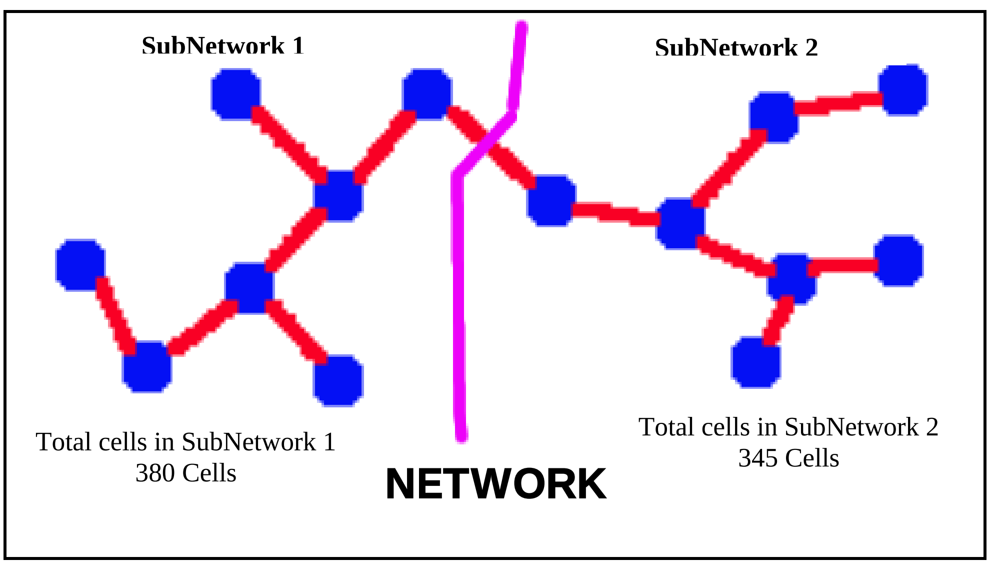


Figure-6.6: Example Network after Partitioning Between Two CPUs

6.4. Major Data Inputs

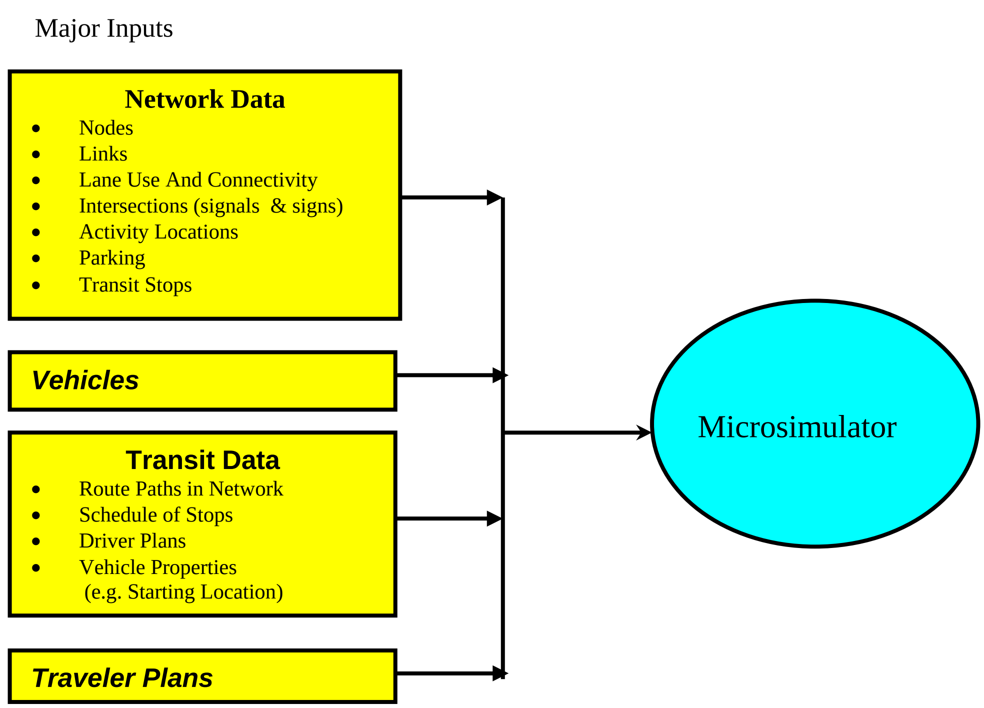


Figure-6.7: Major Inputs to the Microsimulator

This section explores the various inputs to the Microsimulator. These inputs can be grouped into four different categories, each providing information critical to the operation of the Microsimulator.

As shown in Figure-6.7 the inputs to the Microsimulator include the *Network Data files*, the *Vehicle files*, the *Transit Data files* and *the Traveler Plans*. These data files provide sufficient information for the Microsimulator to simulate the movements and the interactions of the travelers in the transportation study area. The detail of the information input also influences the coarseness of the Microsimulator output. Some studies may require highly detailed information about the network while some others may require just the minimum to construct a network.

6.4.1 Network Data Files

Microsimulator uses the network data files as the major data input. Since the network data files have been explained in detail in chapter 2, they are not repeated here.

6.4.2 Vehicle Data files

The Microsimulator also reads in the vehicle data files, which affects vehicle usage (e.g., transit, private auto, taxi etc.) referred to as vehicle specification file, and vehicle characteristics (length, acceleration, profile, etc.) referred to as the vehicle prototype file.
The TRANSIMS Population Synthesizer generates and assigns private vehicles to households and the Activity Generator assigns a possible set of vehicles to each member of a household to execute their activities. Freight and transit vehicles, though generated by separate utilities, are included in the vehicle database. Every freight and transit vehicle has a unique ID.

6.4.2.1 Vehicle Specification File

The vehicle specification file (Table-6.1) contains four primary fields (Household ID, Vehicle ID, ID of vehicle's starting location and the TRANSIMS vehicle type).

Table-6.1: Vehicle Specification File Format

COLUMN NAME	DESCRIPTION
HHID	Household ID
VEHICLE	Vehicle ID
LOCATION	ID of vehicle's starting location
VEHTYPE	Vehicle type, which is one of the following: 1= auto, 2= truck, 4= taxi, 5= bus, 6= trolley, 7= streetcar, 8= light rail, 9= rapid rail, 10= regional rail
VSUBTYPE	Vehicle subtype, used for emissions. The vehicle subtype must correspond to a vehicle subtype specified in the vehicle prototype file.

The user has the option to define some additional fields that may vary in this file for every simulation, but should essentially be the same in every line within a vehicle file.
An example of a vehicle data file is shown below in figure-6.8. It shows that Household 1460 would have 2 vehicles. Both vehicles start at the same home location (78) and are of

the vehicle type 1 or Auto. Two optional user defined fields are present in the file, which are used by the emissions estimator.

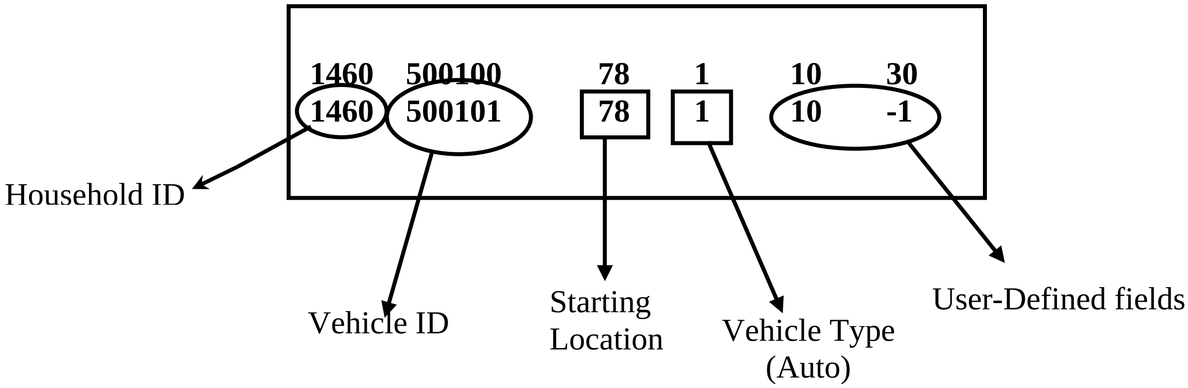


Figure-6.8 Example of a vehicle Data File

6.4.2.2 Vehicle Prototype File

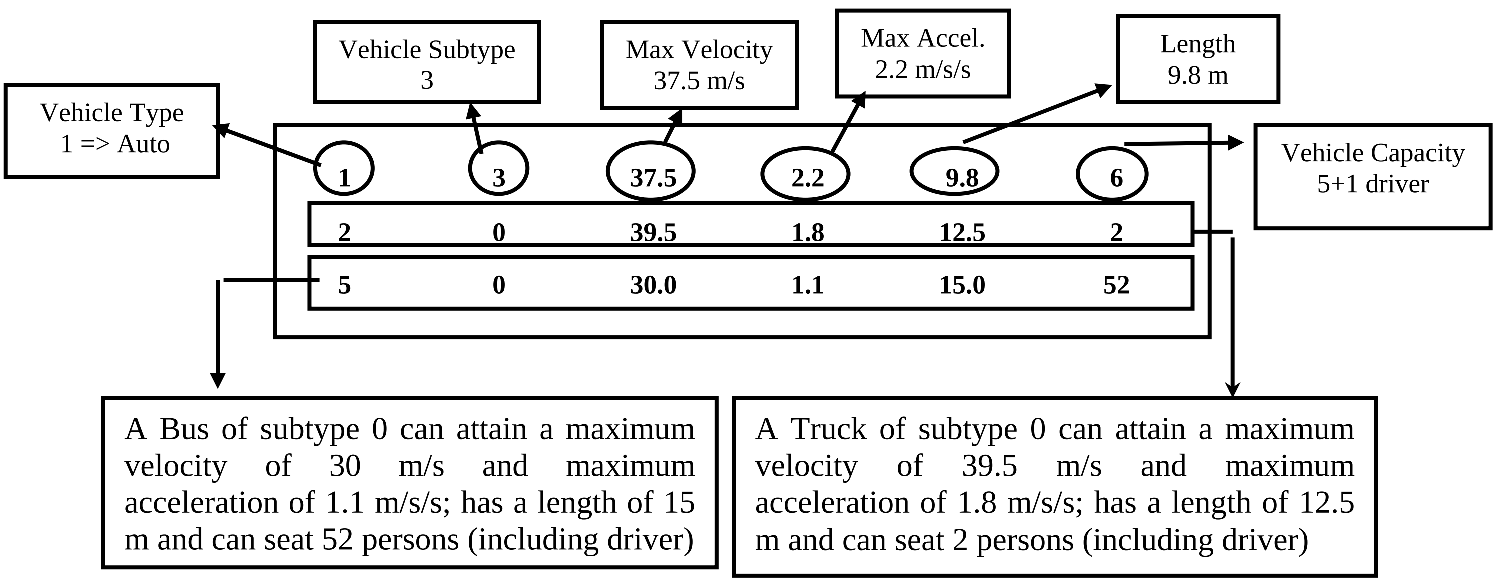
This file, as shown in Table-6.2, defines the characteristics common to whole categories of simulated vehicles. These characteristics include vehicle type, subtype, maximum velocity and acceleration, vehicle length and occupant capacity. All vehicles simulated by the Microsimulator must belong to a defined prototype.

Table-6.2: Vehicle Prototype File Format

COLUMN NAME	DESCRIPTION
VEHTYPE	Vehicle type, which is one of the following: 1= auto, 2= truck, 4= taxi, 5= bus, 6= trolley, 7= streetcar, 8= light rail, 9= rapid rail, 10= regional rail
VSUBTYPE	Vehicle subtype, used for emissions.
MAXVEL	Maximum velocity (meters/second)
MAXACCEL	Maximum acceleration (meters/second/second)
LENGTH	Vehicle length (meters)
CAPACITY	Vehicle capacity (driver + number of possible passengers)

An example of the vehicle prototype file is shown below with an example explaining the fields.

VEHTYPE	VSUBTYPE	MAXVEL	MAXACCEL	LENGTH	CAPACITY
1	1	44.0	4.5	5.5	2
1	2	40.7	3.0	4.4	5
1	3	37.5	2.2	9.8	8
2	0	39.5	1.8	12.5	2
5	0	30.0	1.1	15.0	52



6.4.3 Traveler Plans

Another input to the Microsimulator comes from the Route Planner in the form of the travelers' plans. These files have already been discussed in the output section of the Route Planner Module. In this section, a typical example file of traveler plans is provided in Table-6.3 with a brief description of what the coding in the plan contains.

Table-6.3: Example Plan file

Trip/Leg	Plan	Description
Traveler 101, Trip 1, Leg 1	101 3 1 1 1 0 24600 1002 2 1003 2 400 24600 1 1 0 1 6 300 0 8520 14141 8522 8521	Traveler 101 drives auto 300 from parking 1002 to parking 1003 via nodes 8520, 14141, 8522, 8521.
Traveler 101, Trip 1, Leg 2	101 3 1 2 0 0 25000 1003 2 3002 3 120 25000 1 0 2 0 0	Traveler 101 walks from parking 1003 to transit stop 3002.
Traveler 1, Trip 1, Leg 1	1 10 1 1 1 0 25200 3002 2 3005 2 300 25200 1 1 1 5 6 100 20 8525 8603 14340 8608	Traveler 1 drives bus 100 along bus route 20 from transit stop 3002 to transit stop 3005 via nodes 8525, 8603, 14340, 8608.
Traveler 101, Trip 1, Leg 3	101 3 1 3 0 0 25200 3002 3 3005 3 300 25300 1 0 1 5 1 20	Traveler 101 rides bus from transit stop 3002 to transit stop 3005 along bus route 20.
Traveler 1, Trip 1, Leg 2	1 10 1 2 0 0 25500 3005 2 3005 2 0 25800 1 0 4 0 0	Traveler 1 has a layover activity at transit stop 3005 from the time of arrival until time 25800 seconds past midnight.
Traveler 101, Trip 1, Leg 4	101 3 1 4 0 1 25500 3005 3 1006 2 30 25500 1 0 2 0 0	Traveler 101 walks from transit stop 3005 to parking 1006.
Traveler 1, Trip 1, Leg 3	1 10 1 3 0 1 25800 3005 2 3002 2 200 25800 1 1 1 5 6 100 21 8608 14340 8603 8525	Traveler 1 drives bus 100 along bus route 21 from transit stop 3005 to transit stop 3002 via nodes 8608, 14340, 8603, 8525.

This example describes the plans of two travelers with IDs 101 and 1, executing their trips on the Test Network. Each Traveler performs one trip with several legs as described in Table-6.3. Traveler 1 is a bus driver, while traveler 101 is an auto driver and a bus passenger. The layout of the two traveler plans in the Test Network is shown in Figure-6.9.

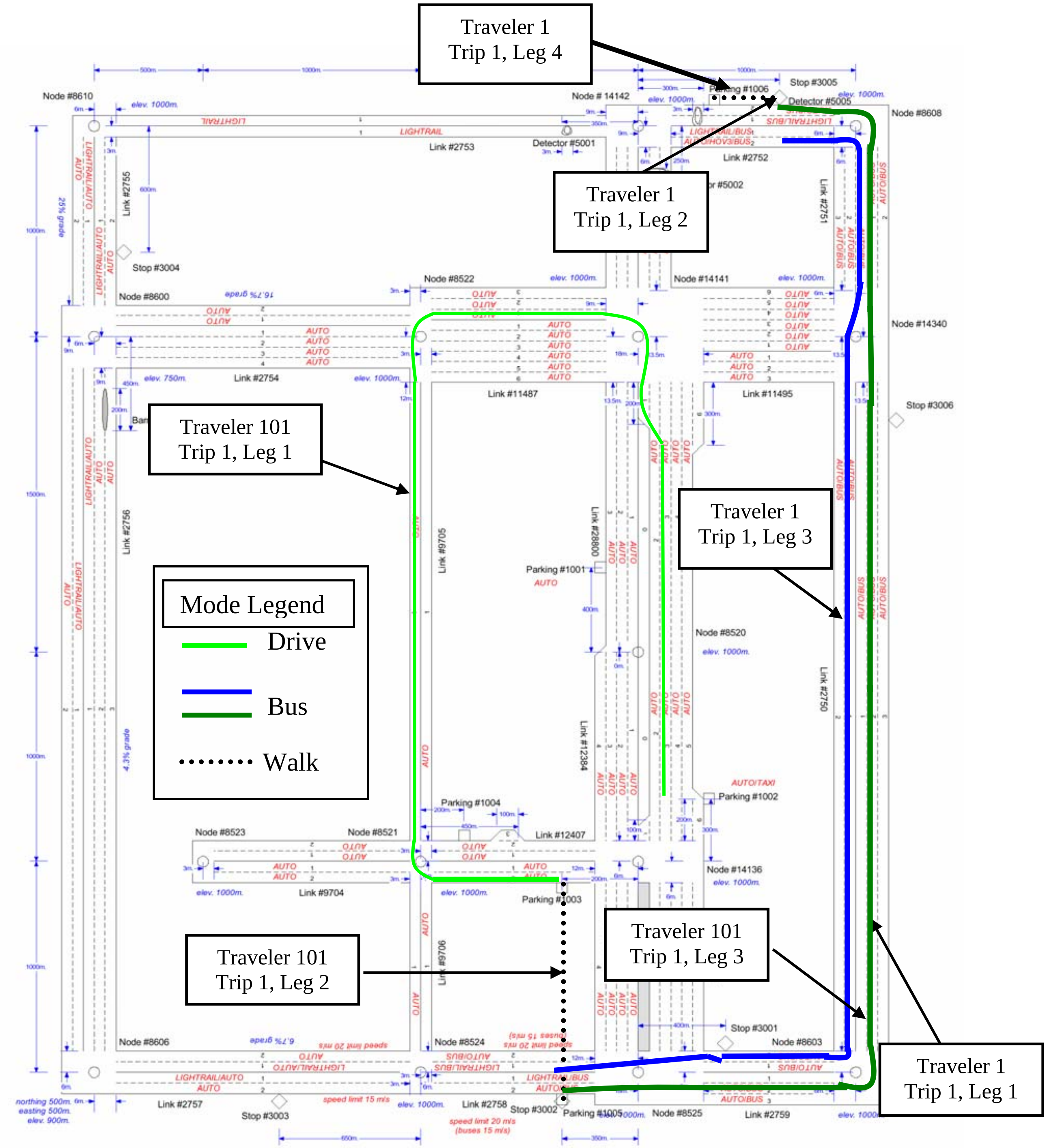


Figure-6.9: Layout of Travel Plans of Traveler 101 and Traveler 1 on the Test Network

6.5. Major Data Outputs

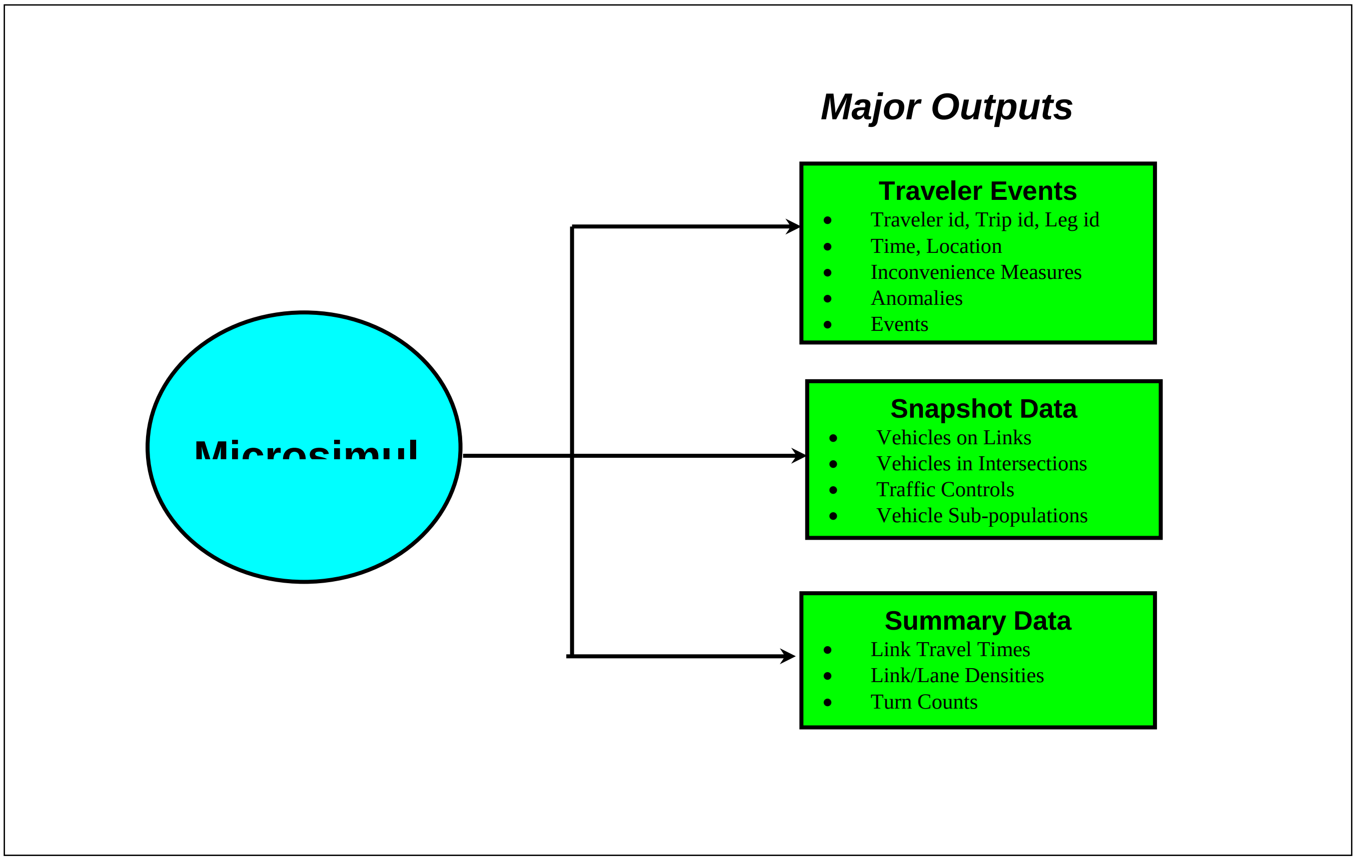


Figure-6.10: Major Outputs of the Microsimulator

Outputs from the Microsimulator are shown in Figure-6.10. There are three major categories of output. The *Traveler Events* provides information whenever an event occurs for a traveler, such as trip ID, leg ID, time, location, or anomalies. The *Vehicle Snapshot data* gives the positions of vehicles on links, at intersections etc., recorded by every time step or less frequently as desired. *Summary data* includes *Spatial* – data collected over sections of roadway such as flow, density etc. and *Temporal* – data such as travel time over links. Summary data is sampled, accumulated and reported periodically throughout the simulation. These three broad data types capture most kinds of output a user might find necessary for analysis.

TRANSIMS Microsimulator also allows the user to record information tailored to his/her specific needs. For example, a user may want to study how well the traffic flows in a specific area. He/she can define the area boundary and specify the parameters he/she is interested in collecting for analysis, as shown in Figure-6.11. These output parameters could be read off a file and/or be viewed using the Output Visualizer. The output visualizer with its graphical user interface is a very helpful tool in the analysis of output.

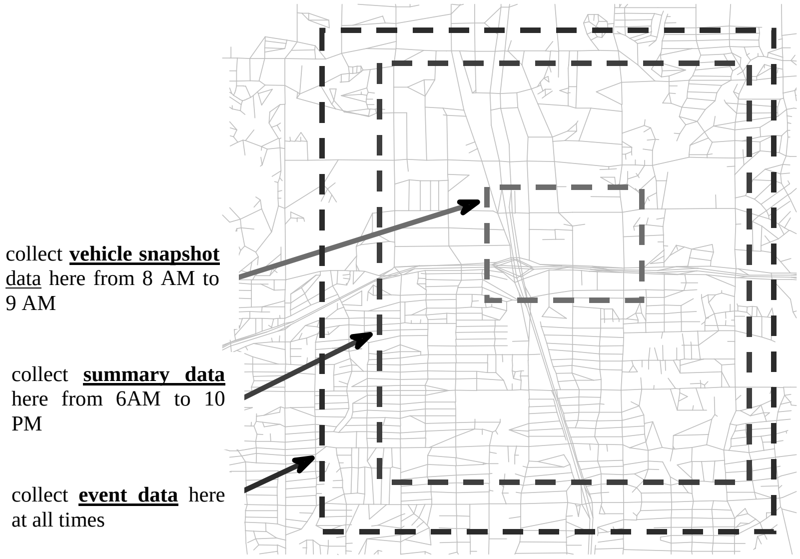


Figure-6.11: Typical Output Collection for a Network

The following section looks closely at the major outputs from the Microsimulator and also briefly explains the Output Visualizer. In the discussion of the outputs from the Microsimulator, the files generated from the plan set of the two travelers given in Table-6.3 are used in explaining the various types of outputs.

6.5.1 Link Temporal and Spatial Summary Data

The link summary report includes data aggregated over sections of roadway defined by the user along street networks. Link traffic characteristics like density, velocity, flow and travel times are usually measured. The densities and velocities are gathered over 150-meter long sections. How frequently this data is collected or over what part of the network this data is summarized has to be specified by the user. The data is collected as frequently as the analyst indicates in the configuration files.

6.5.1.1 Link Densities Summary

Link density summary data, as shown in Table-6.4, reports the vehicle counts and velocities within boxes that partition the link. The box length is specified in the input configuration file. There are separate data records for each link. This data is collected as frequently as the analyst indicates in the input configuration file for the specified links.

Table-6.4: Link Densities Summary Record Data

Field	Interpretation
LINK	Link ID being reported.
NODE	Node ID from which the vehicles were traveling away.
DISTANCE	Ending distance of the box (in meters) from the setback of the node from which the vehicles were traveling away.
TIME	Current time (seconds from midnight).
COUNT	Number of vehicles in the box.
SUM	Sum of the vehicle velocities (in meters per second) in the box.
SUMSQUARES	Sum of the squares of the vehicle velocities (in meters squared per second squared).
LANE	Lane number.

A sample output file for link density summary data is shown below for leg 1 of trip 1 of bus driver 1 in traveling from transit stop #3002 to transit stop #3005 as illustrated in Figure-6.9. On link #2759, on lane 2 traveling away from node #8525, at simulation time of 25510 seconds from midnight in the defined box with its ending distance at 600 m from the node, with only one vehicle in the box has the sum of the velocities of the vehicles in the box as 7.5 m/sec and the sum of squares as 56.25 m^2/sec^2. In this case, since there is only one bus in the box at that time, the sum of the squares of velocities ended up to be equal to the square of velocity.

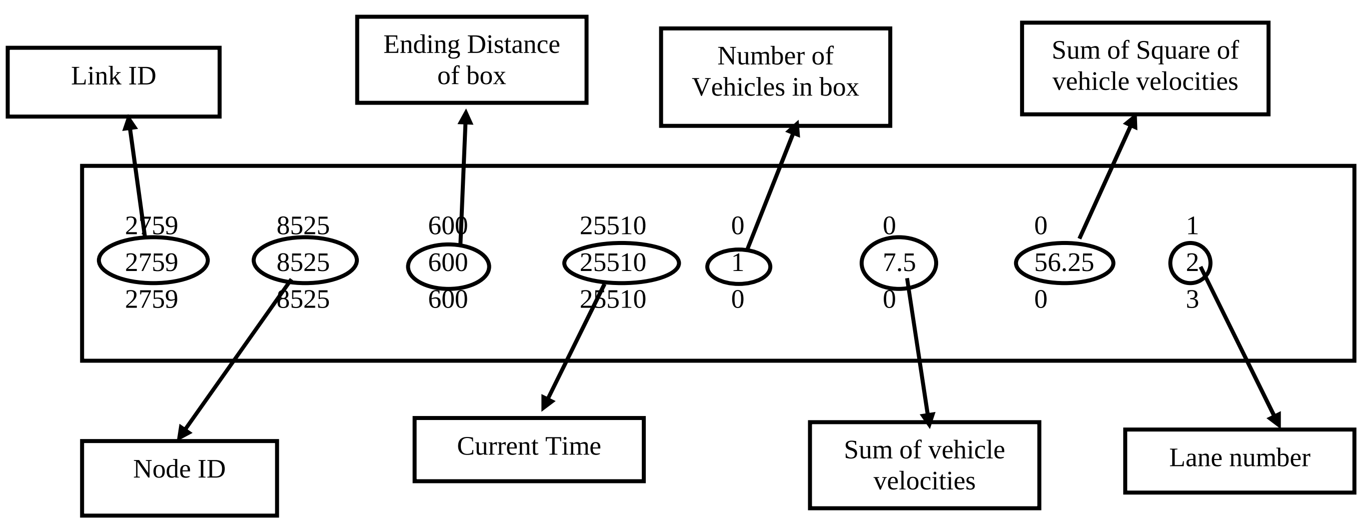


Table-6.5 shows the link density summary table that was collected for one link at the first summary collection time.

Table-6.5: Link Density Summary Table

LINK	NODE	DISTANCE	TIME	COUNT	SUM	SUMSQUARES	LANE
2759	8525	150	25510	0	0	0	1
2759	8525	150	25510	0	0	0	2
2759	8525	150	25510	0	0	0	3
2759	8525	300	25510	0	0	0	1
2759	8525	300	25510	0	0	0	2
2759	8525	300	25510	0	0	0	3
2759	8525	450	25510	0	0	0	1
2759	8525	450	25510	0	0	0	2
2759	8525	450	25510	0	0	0	3
2759	8525	600	25510	0	0	0	1
2759	8525	600	25510	1	7.5	56.25	2
2759	8525	600	25510	0	0	0	3
2759	8525	750	25510	0	0	0	1
2759	8525	750	25510	0	0	0	2
2759	8525	750	25510	0	0	0	3
2759	8525	900	25510	0	0	0	1
2759	8525	900	25510	0	0	0	2
2759	8525	900	25510	0	0	0	3
2759	8525	975	25510	0	0	0	1
2759	8525	975	25510	0	0	0	2
2759	8525	975	25510	0	0	0	3
2759	8603	150	25510	0	0	0	1
2759	8603	150	25510	0	0	0	2
2759	8603	300	25510	0	0	0	1
2759	8603	300	25510	0	0	0	2
2759	8603	450	25510	0	0	0	1
2759	8603	450	25510	0	0	0	2
2759	8603	600	25510	0	0	0	1
2759	8603	600	25510	0	0	0	2
2759	8603	750	25510	0	0	0	1
2759	8603	750	25510	0	0	0	2
2759	8603	900	25510	0	0	0	1
2759	8603	900	25510	0	0	0	2
2759	8603	975	25510	0	0	0	1
2759	8603	975	25510	0	0	0	2

6.5.1.2 Link Velocities Summary

The link velocity summary data report histograms of vehicle velocities within “boxes” that partition the link as shown in Table-6.6. This data is summarized as the user indicates in the input configuration file for the specified link. The configuration file specifies the box length, number of histogram bins, and the maximum velocity. The maximum velocity is typically 37.5m/s and the velocity range is divided into six bins with the last one being the overflow one that extends to infinity. The intervals are defined to be closed at the lower end of the bin and open at the upper end.

Table-6.6: Link Velocities Summary Record Fields

Field	Interpretation
LINK	Link ID being reported.
NODE	Node ID from which the vehicles were traveling away.
DISTANCE	Ending distance of the box (in meters) from the setback of the node from which the vehicles were traveling away.
TIME	Current time (seconds from midnight).
COUNT0	Number of vehicles with velocities in the range [0, 7.5).
COUNT1	Number of vehicles with velocities in the range [7.5, 15).
COUNT2	Number of vehicles with velocities in the range [15, 22.5).
COUNT3	Number of vehicles with velocities in the range [22.5, 30).
COUNT4	Number of vehicles with velocities in the range [30, 37.5).
COUNT5	Number of vehicles with velocities in the range [37.5, infinity).

For the same bus example cited earlier and shown in Figure-6.9, a count of vehicles in the velocity ranges of 0-7.5 m/s, 7.5-15m/s, 15-22.5 m/s, 22.5-30m/s, 30-37.5 m/s and beyond for vehicles traveling away from node #8525 on link #2759 at the simulation time of 25510 seconds from midnight in a box ending at 600m from node, is captured by the first line of the sample output link velocity summary file shown below. The data shows that there is one vehicle (bus driver1) in COUNT1, whose velocity falls within the range of 7.5 and 15 m/s.

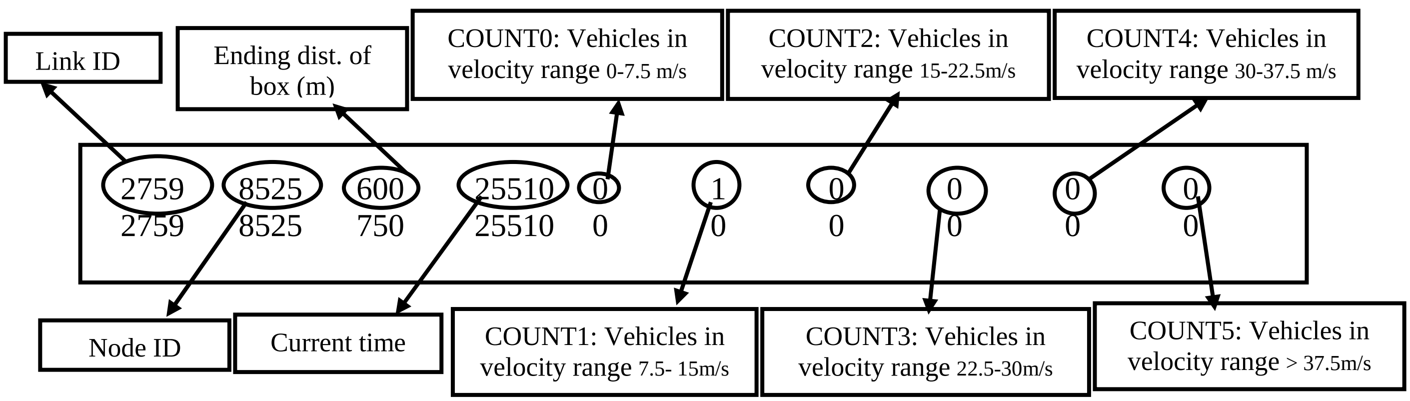


Table-6.7 shows the link velocity summary data that were collected for one link at the first summary collection time.

Table-6.7: Link Velocity Summary Data

LINK	NODE	DISTANCE	TIME	COUNT0	COUNT1	COUNT2	COUNT3	COUNT4	COUNT5
2759	8525	150	25510	0	0	0	0	0	0
2759	8525	300	25510	0	0	0	0	0	0
2759	8525	450	25510	0	0	0	0	0	0
2759	8525	600	25510	0	1	0	0	0	0
2759	8525	750	25510	0	0	0	0	0	0
2759	8525	900	25510	0	0	0	0	0	0
2759	8525	975	25510	0	0	0	0	0	0
2759	8603	150	25510	0	0	0	0	0	0
2759	8603	300	25510	0	0	0	0	0	0
2759	8603	450	25510	0	0	0	0	0	0
2759	8603	600	25510	0	0	0	0	0	0
2759	8603	750	25510	0	0	0	0	0	0
2759	8603	900	25510	0	0	0	0	0	0
2759	8603	975	25510	0	0	0	0	0	0

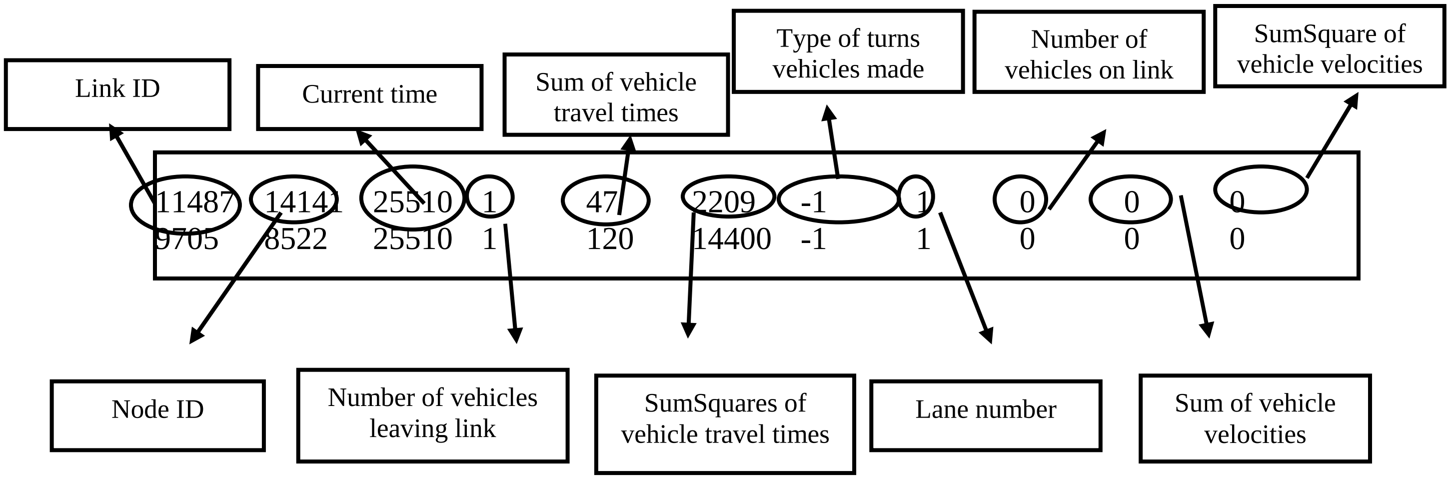
6.5.1.3 Link Travel Times Summary

Link Travel times summary data, as shown in Table-6.8, report vehicle counts and travel times on links accumulated as the vehicles exit the links. There are also separate records for each turning movement leaving each lane on the link.

Table-6.8: Link Travel Times Summary Record Fields

Field	Interpretation
LINK	Link ID being reported.
NODE	Node ID from which the vehicles were traveling away.
TIME	Current time (seconds from midnight).
COUNT	Number of vehicles leaving the link.
SUM	Sum of the vehicle travel times (in seconds) for vehicles leaving the link. (The time spent in the previous intersection is included in this value.)
SUMSQUARES	Sum of the vehicle travel time squares (in seconds squared) for vehicles leaving the link. (The time spent in the previous intersection is included in this value.)
TURN	Type of turn the vehicle made leaving the link. For detailed explanation see Traveler Event Data (Table 55)
LANE	Lane number.
VCOUNT	Number of vehicles on the link.
VSUM	Sum of vehicle velocities (in meters per second) on the link.
VSUMSQUARES	Sum of the squares of the vehicle velocities (in meters squared per second squared).

A typical link travel time summary record file is shown and explained below.



The first line of the sample output file states the condition of link #11487 for vehicles traveling away from the node #14141 at the simulation time of 25510 seconds from midnight. It states that there is one vehicle leaving the link on lane 1 making a left turn and there are no other vehicles at that time on the link. This can be identified as traveler 101 on leg 1 of trip 1 in the drive mode turning left onto link #9705, and going towards parking #1003. The sum of travel times for the vehicles leaving the link is 47 seconds. In this case, it is the same travel time as that of traveler 101 since there are no other vehicles leaving the link.

Table-6.9 shows the travel time summary data collected every 15 minutes for the simulation run on the Test Network.

Table-6.9: The Travel Time Summary Data Collected Every 15 Minutes

LINK	NODE	TIME	COUNT	SUM	SUMSQUARES	TURN	LANE	VCOUNT	VSUM	VSUMSQUARES
11487	14141	25510	1	47	2209	-1	1	0	0	0
9705	8522	25510	1	120	14400	-1	1	0	0	0
28800	8520	25510	1	142	20164	-1	2	0	0	0
2759	8525	25510	1	73	5329	0	2	0	0	0
2759	8525	26410	0	0	0	0	2	0	0	0
2759	8603	26410	1	72	5184	0	1	0	0	0
2751	14340	26410	1	72	5184	0	2	0	0	0
2751	8608	26410	1	74	5476	0	1	0	0	0
11487	14141	26410	0	0	0	-1	1	0	0	0
2750	8603	26410	1	308	94864	0	2	0	0	0
2750	14340	26410	1	256	65536	1	1	0	0	0
9705	8522	26410	0	0	0	-1	1	0	0	0
28800	8520	26410	0	0	0	-1	2	0	0	0

6.5.1.4 Link Energy Summary

Link energy summary data report histograms of vehicle energies (integrated power) accumulated as vehicles enter the link. Energy is defined as the sum of the vehicle's power over each time step, where power is defined as the velocity times the acceleration when the acceleration is greater than zero.

The metadata for the link energy summary file are the `TIME_STEP`, `ENERGY_SOAK`, `ENERGY_MAX`, `ENERGY_BINS`, `SHORT_SOAK_TIME`, `MEDIUM_SOAK_TIME`, and `LONG_SOAK_TIME` configuration file keys.

Vehicles are assumed to have zero power while they are at intersections. The units for energy are cells-squared per second-squared.

These data are collected as frequently as the analyst indicates in the input configuration file for the specified links. The number of histogram bins and maximum energy is specified in the input configuration file. Histogram intervals are defined to be closed at the lower end of the bin and open at the upper end. Table-6.10 lists link energy summary record fields.

Table-6.10: Link energy summary data record fields

Field	**Interpretation**
LINK	The link ID being reported.
NODE	The node ID from which the vehicles were traveling away.
TIME	The current time (seconds from midnight).
ENERGY0	The number of vehicles with integrated power in the range [0, *energy_maximum* / *number_bins*).
ENERGY1	The number of vehicles with integrated power in the second bin.
ENERGY2	The number of vehicles with integrated power in the third bin.
ENERGYn	The number of vehicles with integrated power in the range [energy_maximum, infinity).

6.5.2 Snapshot Data

The snapshot data provide information about vehicles traveling on a link at every time step in the simulation. The snapshot data files generally contain time, position and velocity information for each vehicle in the simulation. They also provide information about the movement of vehicles at an intersection and the current status of the traffic signal at a node.

These snapshot data are collected as frequently as indicated by the analyst in the configuration files. Dumping these files for a long period of time creates extremely large data files. Users are allowed to restrict the output to smaller portions of the network and for specific times during the simulation, and also may only select certain records that meet a specific criterion.

6.5.2.1 Vehicle Snapshot file

Vehicle snapshot data files provide information about vehicles traveling on a link. When collected over all links and over the whole simulation time, a complete trajectory of each vehicle in the simulation can be obtained. The format of the vehicle snapshot record is shown in Table-6.11.

Table-6.11: Vehicle Snapshot Record Fields

Field	**Interpretation**
VEHICLE	Vehicle ID.
TIME	Current time (seconds from midnight).
LINK	Link ID on which the vehicle was traveling.
NODE	Node ID vehicle was traveling away from.
LANE	Number of the lane on which the vehicle is traveling.
DISTANCE	Distance (in meters) the vehicle is away from the node from which it is traveling.
VELOCITY	Velocity (in meters per second) of the vehicle.
VEHTYPE	Vehicle type: 0 = walk 1 = auto 2 = truck 3 = bicycle 4 = taxi 5 = bus 6 = trolley 7 = streetcar 8 = light rail 9 = rapid rail 10 = regional rail
ACCELER	Acceleration (in meters square per second) the vehicle has in the current time step.
DRIVER	Driver ID.
PASSENGERS	Count of passengers in vehicle.
EASTING	Vehicle's *x*-coordinate (in meters).
NORTHING	Vehicle's *y*-coordinate (in meters).
ELEVATION	Vehicle's *z*-coordinate (in meters).
AZIMUTH	Vehicle's orientation angle (degrees from east in the counterclockwise direction).
USER	User-defined field that can be set on a per-vehicle basis.

A typical snapshot record file for traveler 101 driving from parking #1002 towards parking #1003 is shown below with an explanation of each field. The first line indicates an instant when the vehicle with an ID 300 at the simulation time of 24610 seconds from midnight is traveling on lane 2 of link #12384 and away from node #14136. It is at a distance of 442.5 m from the node. Further, this vehicle is an auto traveling at a speed of 15 m/s and with no acceleration. The driver of the vehicle has an ID of 101 and there are no passengers in the vehicle. The vehicle has an easting of 3005.5, a northing of 1948.5, an elevation of 1000 m and an azimuth of 90 degrees. This snapshot is for the initial part of leg 1 of trip 1 of traveler ID 101 according to the travel plans shown in Table-6.3.

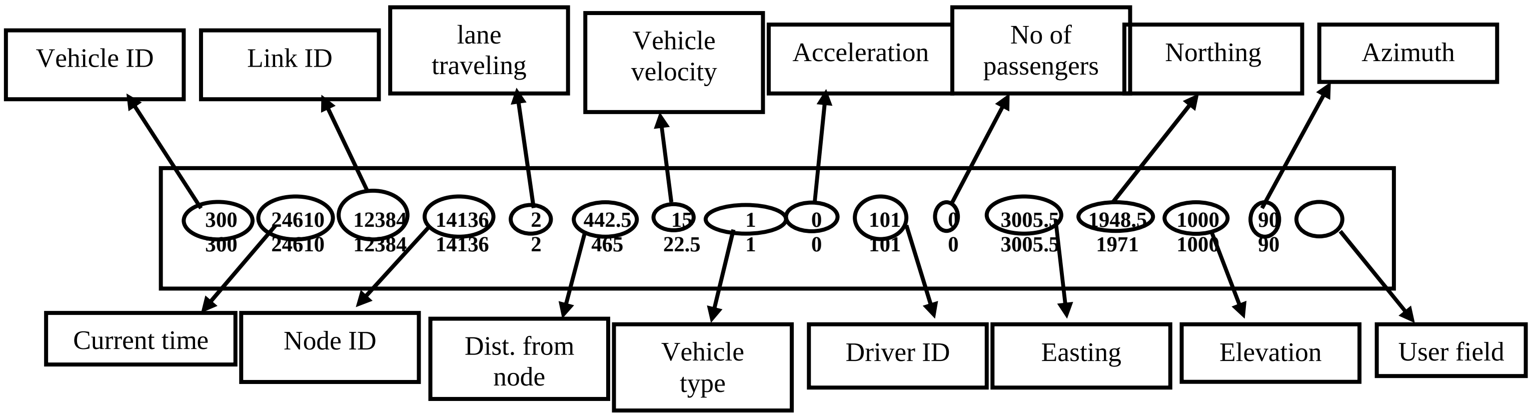


Table-6.12 (parts a and b) shows the first 30 seconds of vehicle snapshot data collected for the simulation run on the Test Network with the Traveler Plans and the Input files discussed earlier.

Table-6.12a: First 30 Seconds of Vehicle Snapshot Data

	VEHICLE	TIME	LINK	NODE	LANE	DISTANCE	VELOCITY	VEHTYPE	DRIVER	PASSENGERS	EASTING	NORTHING	ELEVATION	AZIMUTH
A	300	24610	12384	14136	2	443	15	1	101	0	3005.25	1949	1000	90
B	300	24611	12384	14136	2	465	22.5	1	101	0	3005.25	1971	1000	90
C	300	24612	12384	14136	2	480	15	1	101	0	3005.25	1986	1000	90
D	300	24613	12384	14136	2	503	22.5	1	101	0	3005.25	2009	1000	90
E	300	24614	12384	14136	2	518	15	1	101	0	3005.25	2024	1000	90
F	300	24615	12384	14136	2	540	22.5	1	101	0	3005.25	2046	1000	90
G	300	24616	12384	14136	2	563	22.5	1	101	0	3005.25	2069	1000	90
H	300	24617	12384	14136	2	578	15	1	101	0	3005.25	2084	1000	90
I	300	24618	12384	14136	2	600	22.5	1	101	0	3005.25	2106	1000	90
J	300	24619	12384	14136	2	623	22.5	1	101	0	3005.25	2129	1000	90
K	300	24620	12384	14136	2	645	22.5	1	101	0	3005.25	2151	1000	90
L	300	24621	12384	14136	2	668	22.5	1	101	0	3005.25	2174	1000	90
M	300	24622	12384	14136	2	690	22.5	1	101	0	3005.25	2196	1000	90
N	300	24623	12384	14136	2	713	22.5	1	101	0	3005.25	2219	1000	90
O	300	24624	12384	14136	2	735	22.5	1	101	0	3005.25	2241	1000	90

Table-6.12b: First 30 Seconds of Vehicle Snapshot Data

	VEHICLE	TIME	LINK	NODE	LANE	DISTANCE	VELOCITY	VEHTYPE	DRIVER	PASSENGERS	EASTING	NORTHING	ELEVATION	AZIMUTH
P	300	24625	12384	14136	2	750	15	1	101	0	3005.25	2256	1000	90
Q	300	24626	12384	14136	2	765	15	1	101	0	3005.25	2271	1000	90
R	300	24627	12384	14136	2	788	22.5	1	101	0	3005.25	2294	1000	90
S	300	24628	12384	14136	2	803	15	1	101	0	3005.25	2309	1000	90
T	300	24629	12384	14136	2	825	22.5	1	101	0	3005.25	2331	1000	90
U	300	24630	12384	14136	2	848	22.5	1	101	0	3005.25	2354	1000	90
V	300	24631	12384	14136	2	863	15	1	101	0	3005.25	2369	1000	90
W	300	24632	12384	14136	2	885	22.5	1	101	0	3005.25	2391	1000	90
X	300	24633	12384	14136	2	900	15	1	101	0	3005.25	2406	1000	90
Y	300	24634	12384	14136	2	923	22.5	1	101	0	3005.25	2429	1000	90
Z	300	24635	12384	14136	2	938	15	1	101	0	3005.25	2444	1000	90
AA	300	24636	12384	14136	2	960	22.5	1	101	0	3005.25	2466	1000	90
BB	300	24637	12384	14136	2	983	22.5	1	101	0	3005.25	2489	1000	90
CC	300	24638	28800	8520	2	15	22.5	1	101	0	3005.25	2515	1000	90
DD	300	24639	28800	8520	2	30	15	1	101	0	3005.25	2530	1000	90
EE	300	24640	28800	8520	2	52.5	22.5	1	101	0	3005.25	2553	1000	90

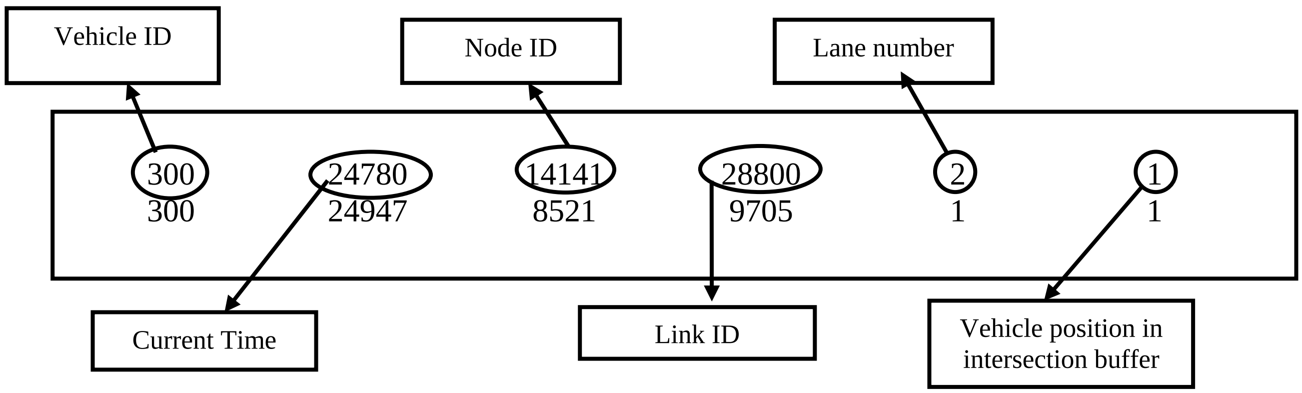
6.5.2.2 Intersection Snapshot file

Intersection Snapshot data files provide information about a vehicle as it passes through an intersection. The intersection snapshot record file is shown in Table-6.13.

Table-6.13: Intersection Snapshot Record Fields

Field	Interpretation
VEHICLE	Vehicle ID.
TIME	Current time (seconds from the midnight).
NODE	Node ID where the vehicle is located.
LINK	Link ID from which the vehicle entered.
LANE	Number of the lane from which the vehicle entered.
QINDEX	Vehicle position in the intersection buffer.

A typical Intersection Snapshot Record file driven by Traveler 101 is shown below, including the interpretation of the various fields. The vehicle with an ID 300 on lane 2 of link #28800 traveling away from node #14141 at simulation time 24780 seconds from midnight is the first vehicle on the intersection buffer as depicted by the first line of the intersection snapshot record file. This snapshot of the simulation shows the position of traveler 101 at signalized intersections (node #14141 and node #8521) in executing his/her travel plans of leg 1 of trip 1.



The Intersection Snapshot data for the simulation run on the Test network is shown in Table-6.14.

Table-6.14: Intersection Snapshot Data

VEHICLE	TIME	NODE	LINK	LANE	QINDEX
300	24780	14141	28800	2	1
300	24947	8521	9705	1	1

6.5.2.3 Traffic Control Snapshot File

The Traffic Control Snapshot File reports the current state of the traffic signal at a node. Table-6.15 gives what each of the fields of the Traffic Control Snapshot file stand for.

Table-6.15: Traffic Control Snapshot File Format

Field	**Interpretation**
NODE	Node ID where the signal is located.
TIME	Current time (seconds from midnight).
LINK	Link ID entering the signal.
LANE	Number of the lane entering the signal.
SIGNAL	Type of control present: 0: None 1: Stop 2: Yield 3: Wait 4: Caution 5: Permitted 6: Protected

An example of a traffic control snapshot file is as follows. The first line of the sample output depicts the condition at simulation time 24610 seconds from midnight at node #14141 for lane 1 on link #11487. The type of control for this lane at this time is 3 or "wait."

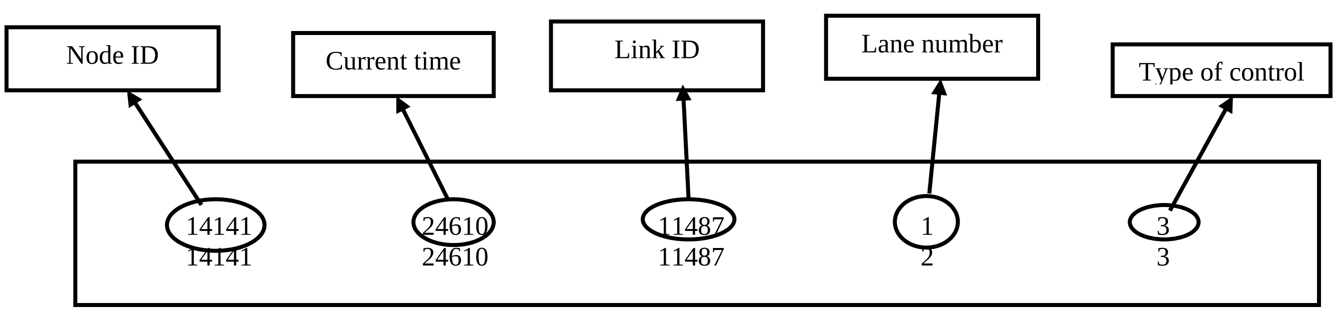


Table-6.16 below shows the signal snapshot data collected during the first second of the simulation for the simulation run on the Test Network.

Table-6.16: Signal Snapshot Data for a Simulation Run on Test Network

NODE	TIME	LINK	LANE	SIGNAL
8521	24610	9704	1	5
8521	24610	9704	2	5
8521	24610	9705	1	3
8521	24610	12407	1	5
8521	24610	12407	2	5
8521	24610	12407	3	3
8521	24610	9706	1	3
14141	24610	11487	1	3
14141	24610	11487	2	3
14141	24610	11487	3	3
14141	24610	11487	4	3
14141	24610	11487	5	3
14141	24610	11487	6	6
14141	24610	11486	1	5
14141	24610	11486	2	5
14141	24610	11486	3	5
14141	24610	11495	1	3
14141	24610	11495	2	3
14141	24610	11495	3	3
14141	24610	11495	4	3
14141	24610	11495	5	3
14141	24610	11495	6	7
14141	24610	28800	1	3
14141	24610	28800	2	3
14141	24610	28800	3	5
14141	24610	28800	4	5
14141	24610	28800	5	5
14141	24610	28800	6	6

6.5.3 Traveler Event Data

A traveler event is recorded each time an event of interest (defined by the user) takes place. The input configuration file specifies the simulation time interval in which the event is recorded. Filtering capabilities are provided so that the analyst may choose which of the many potentially interesting events should be recorded. The events of interest are specified in the STATUS and ANOMALY output fields as shown in Table-6.17.

Table-6.17: Traveler Event Data

Field	Description
TIME	Current time (seconds from midnight).
TRAVELER	Traveler ID.
TRIP	Traveler's trip ID.
LEG	Traveler's plan leg ID.
VEHICLE	Vehicle ID; value = 0 if not in a vehicle.
VEHTYPE	Vehicle type: 0 = walk 1 = auto 2 = truck 3 = bicycle 4 = taxi 5 = bus 6 = trolley 7 = streetcar 8 = light rail 9 = rapid rail 10 = regional rail
VSUBTYPE	Vehicle subtype may be unused; value = 0 if not applicable.
ROUTE	Transit route ID; value = -1 if not in a transit vehicle.
STOPS	Count of number of stop signs encountered on current plan leg.
YIELDS	Count of number of yield signs encountered on current plan leg.
SIGNALS	Number of traffic signals encountered on current plan leg.
TURN	Type of last turn made: 0 = straight direction (no turn) 1 = right turn -1 = left turn 2 = hard right turn -2 = hard left turn values 3 to 6 represent increasingly more extreme right turns values –3 to –6 represent increasingly more extreme left turns -7 = reverse direction (U-turn)
STOPPED	Time (seconds) spent stopped on current plan leg.
ACCELS	Time (seconds) spent accelerating from 0 on current plan leg.
TIMESUM	Total time (seconds) spent on current plan leg.

Field	Description
DISTANCESUM	Total distance (meters) traveled on current plan leg (see accompanying text for more information).
USER	Analyst-defined field: any integer value is acceptable, and definition may vary with each case study.
ANOMALY	Type of anomaly: 0 = no anomaly occurred 1 = traveler is off plan 2 = traveler cannot find next link in plan 3 = traveler cannot find next parking place in plan 4 = traveler cannot find next vehicle in plan 5 = traveler cannot find next transit stop in plan 6 = traveler cannot board full transit vehicle 7 = driver of transit vehicle skipped stop that had passengers waiting to board 8 = driver of vehicle cannot change lanes because of congestion
STATUS	Traveler's current status bits: (see accompanying text for a detailed explanation of status bit interpretation). 0x1 = traveler is on a link (persistent) 0x2 = change in traveler's on-link status 0x4 = traveler is on a leg (persistent) 0x8 = change in traveler's on-leg status 0x10 = change in traveler's on-trip status 0x20 = traveler is non-motorized, i.e., walking, bicycling (persistent) 0x40 = traveler is not in the study area (persistent) 0x80 = change in traveler's in-study area status 0x100 = traveler is in a vehicle (persistent) 0x200 = change in traveler's vehicle occupancy status 0x400 = traveler is the driver (persistent) 0x800 = change in traveler's driver status 0x1000 = traveler is waiting at some location (persistent) 0x2000 = change in traveler's waiting status 0x4000 = location is a parking place (persistent) 0x8000 = location is a transit stop (persistent) 0x10000 = driver of transit vehicle is at a transit stop (persistent) 0x20000 = change in driver's transit vehicle at stop status 0x40000 = driver of transit vehicle is on a layover (persistent) 0x80000 = change in driver's transit vehicle on layover status 0x100000 = driver's transit vehicle is full (persistent) 0x200000 = change in driver's transit vehicle full status 0x400000 = traveler is off plan (persistent) 0x800000 = change in traveler's off-plan status 0x1000000 = beginning of simulation 0x2000000 = end of simulation 0x4000000 = location is an activity location (persistent) 0x8000000 = undefined 0x10000000 = undefined 0x20000000 = undefined 0x40000000 = undefined 0x80000000 = undefined

Field	Description
LOCATION	Where traveler is located: link ID, parking place ID, transit stop ID, or activity location ID, depending on the event as defined here EVENT — LOCATION value Enter/Exit/On link — link ID Begin/End plan leg — parking place ID or transit stop ID Begin/End trip — parking place ID or transit stop ID Enter/Exit study area — link ID Enter/Exit vehicle — parking place ID or transit stop ID Begin/End driving — parking place ID or transit stop ID Waiting for transit — transit stop ID Waiting at parking — parking place ID Begin/End activity — activity location ID Transit vehicle at stop — transit stop ID Transit vehicle on layover — transit stop ID Transit vehicle full — transit stop ID Off plan — link ID Begin/End Simulation — link ID Can’t find link — link ID Can’t find parking — parking place ID Can’t find vehicle — parking place ID Can’t find transit stop — transit stop ID Can’t board transit — transit stop ID Skipped transit stop — transit stop ID Can’t change lanes — link ID

An excerpt from traveler 101-travel event file is shown below. Each entry in the table is explained on the next page.

```
24610 101 1 1     0 0 0 –1 0 0 0 0 0 0 10       0 3 0 16412 1002 Begin Leg, non motorized, at parking 1002
24610 101 1 1 300 1 0 –1 0 0 0 0 0 0 10 307.5 3 0 17156 1002 Begin Leg, at parking at 1002
```

The first line of the excerpt indicates an event at a simulation time of 24610 from midnight. This data is collected for the traveler ID 101 on leg 1 of trip 1. Information about the driven vehicle being auto is provided along with other information stating that the traveler has not encountered any stops, yields or signals in the current leg of the plan. The traveler has been moving straight and has not stopped on the current leg, neither has he/she spent any time in accelerating from 0 m/s on the leg which he/she has been executing for 10 seconds now. These conditions exist because traveler 101 is executing the non-motorized leg of the trip, which is at the parking location #1002.

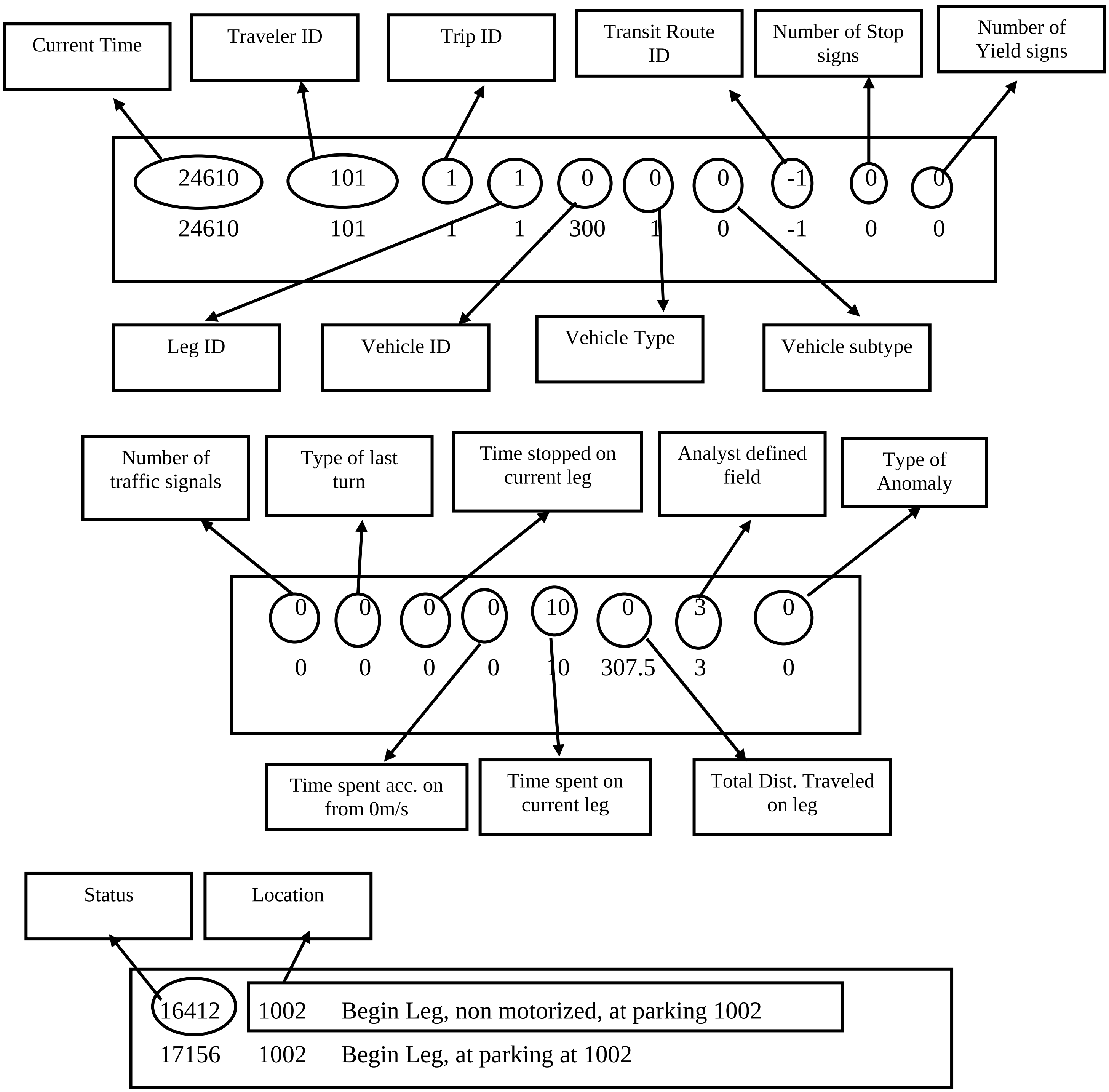


Table-6.18 (parts a and b) shows the traveler event output that was collected for an 1800-second simulation of the Test Network, with the plan set data and the configuration keys as specified earlier.

Table-6.18a: Traveler Event Output Data

TIME	TRAVELER	TRIP	LEG	VEHICLE	VEH TYPE	VSUB TYPE	ROUTE	STOPS	YIELDS	SIGNALS	TURN	STOPPED	ACCELS	TIME SUM	DISTANCE SUM	USER	ANOMALY	STATUS	LOCATION
24610	101	1	1	0	0	0	-1	0	0	0	0	0	0	10	0	3	0	16412	1002
24610	101	1	1	300	1	0	-1	0	0	0	0	0	0	10	307.5	3	0	17156	1002
24610	101	1	1	300	1	0	-1	0	0	0	0	0	0	10	307.5	3	0	19716	1002
24610	101	1	1	300	1	0	-1	0	0	0	0	0	0	0	135	3	0	16778501	12384
24638	101	1	1	300	1	0	-1	0	0	0	0	0	0	28	694	3	0	1286	12384
24780	101	1	1	300	1	0	-1	0	1	1	0	59	0	170	2194	3	0	1286	28800
24827	101	1	1	300	1	0	-1	0	1	1	-1	59	0	217	3194	3	0	1286	11487
24947	101	1	1	300	1	0	-1	0	1	2	-1	63	0	337	5694	3	0	1286	9705
24986	101	1	1	300	1	0	-1	0	1	2	-1	63	0	376	6499.5	3	0	1286	12407
24986	101	1	1	300	1	0	-1	0	1	2	-1	63	0	376	6499.5	3	0	18692	1003
24986	101	1	1	300	1	0	-1	0	1	2	-1	63	0	376	6499.5	3	0	16900	1003
24986	101	1	1	0	0	0	-1	0	1	2	-1	63	0	376	6499.5	3	0	16392	1003
24986	101	1	2	0	0	0	-1	0	0	0	0	0	0	0	0	3	0	16428	1003
25106	101	1	2	0	0	0	20	0	0	0	0	0	0	0	0	3	0	32808	3002
25106	101	1	3	0	0	0	20	0	0	0	0	0	0	0	0	3	0	32780	3002
25106	101	1	3	0	0	0	20	0	0	0	0	0	0	0	0	3	0	45060	3002
25200	1	1	1	0	0	0	20	0	0	0	0	0	0	0	0	10	0	16412	1005
25201	1	1	1	0	0	0	20	0	0	0	0	0	0	0	0	10	0	28676	1005
25201	1	1	1	100	5	0	20	0	0	0	0	0	0	0	0	10	0	21252	1005
25201	1	1	1	100	5	0	20	0	0	0	0	0	0	0	0	10	0	23812	1005
25201	1	1	1	100	5	0	20	0	0	0	0	0	0	0	0	10	0	25860	1005
25203	1	1	1	100	5	0	20	0	0	0	0	0	1	2	15	10	0	230661	3002
25209	101	1	3	100	5	0	20	0	0	0	0	0	0	103	7.5	3	0	37636	3002
25209	101	1	3	100	5	0	20	0	0	0	0	0	0	103	7.5	3	0	41220	3002
25209	1	1	1	100	5	0	20	0	0	0	0	0	1	8	37.5	10	0	132357	3002
25231	1	1	1	100	5	0	20	0	0	0	0	0	2	30	374.5	10	0	1286	2758
25231	101	1	3	100	5	0	20	0	0	0	0	0	1	125	367	3	0	262	2758
25304	1	1	1	100	5	0	20	0	0	0	0	0	2	103	1374.5	10	0	1286	2759
25304	101	1	3	100	5	0	20	0	0	0	0	0	1	198	1367	3	0	262	2759

Table-6.18b: Traveler Event Output Data

TIME	TRAVELER	TRIP	LEG	VEHICLE	VEH TYPE	VSUB TYPE	ROUTE	STOPS	YIELDS	SIGNALS	TURN	STOPPED	ACCELS	TIME SUM	DISTANCE SUM	USER	ANOMALY	STATUS	LOCATION
25612	1	1	1	100	5	0	20	0	0	0	0	1	21	411	4874.5	10	0	1286	2750
25612	101	1	3	100	5	0	20	0	0	0	0	1	20	506	4867	3	0	262	2750
25684	1	1	1	100	5	0	20	1	0	0	0	1	21	483	5874.5	10	0	1286	2751
25684	101	1	3	100	5	0	20	1	0	0	0	1	20	578	5867	3	0	262	2751
25708	1	1	1	100	5	0	20	1	0	0	0	1	21	507	6203	10	0	230661	3005
25712	101	1	3	100	5	0	20	1	0	0	0	1	20	606	6195.5	3	0	33284	3005
25712	101	1	3	0	0	0	-1	1	0	0	0	1	20	606	6195.5	3	0	32776	3005
25712	101	1	4	0	0	0	-1	0	0	0	0	0	0	0	0	3	0	32812	3005
25712	1	1	1	100	5	0	20	1	0	0	0	1	21	511	6233	10	0	132357	3005
25712	1	1	1	100	5	0	20	1	0	0	0	1	21	511	6233	10	0	1286	2752
25712	1	1	1	100	5	0	20	1	0	0	0	1	21	511	6225.5	10	0	18692	1006
25712	1	1	1	100	5	0	20	1	0	0	0	1	21	511	6225.5	10	0	16900	1006
25712	1	1	1	0	0	0	-1	1	0	0	0	1	21	511	6225.5	10	0	16392	1006
25712	1	1	2	0	0	0	-1	0	0	0	0	0	0	0	0	10	0	16428	1006
25712	1	1	2	0	0	0	-1	0	0	0	0	0	0	0	0	10	0	802852	1006
25712	1	1	2	0	0	0	21	0	0	0	0	0	0	0	0	10	0	540708	1006
25712	1	1	2	0	0	0	21	0	0	0	0	0	0	0	0	10	0	16424	1006
25712	1	1	3	0	0	0	21	0	0	0	0	0	0	0	0	10	0	16396	1006
25800	1	1	3	0	0	0	21	0	0	0	0	0	0	0	0	10	0	28676	1006
25800	1	1	3	100	5	0	21	0	0	0	0	0	0	0	0	10	0	21252	1006
25800	1	1	3	100	5	0	21	0	0	0	0	0	0	0	0	10	0	23812	1006
25800	1	1	3	100	5	0	21	0	0	0	0	0	0	0	0	10	0	25860	1006
25823	1	1	3	100	5	0	21	0	0	0	0	0	1	23	353.5	10	0	1286	2752
25897	1	1	3	100	5	0	21	0	0	0	0	2	1	97	1353.5	10	0	1286	2751
26153	1	1	3	100	5	0	21	1	0	0	0	3	1	353	4853.5	10	0	1286	2750
26225	1	1	3	100	5	0	21	1	0	0	0	4	1	425	5853.5	10	0	1286	2759
26250	1	1	3	100	5	0	21	1	0	0	0	4	1	450	6225.5	10	0	1286	2758
26250	1	1	3	100	5	0	21	1	0	0	0	4	1	450	6495.5	10	0	18692	1005
26250	1	1	3	100	5	0	21	1	0	0	0	4	1	450	6495.5	10	0	16900	1005
26250	1	1	3	0	0	0	21	1	0	0	0	4	1	450	6495.5	10	0	16408	1005

6.6. Output Visualizer

The output from TRANSIMS can be pictorially represented and viewed by the Output Visualizer. The Output Visualizer enables the user to display various input and output sets. It also provides tools to facilitate the analysis of the data sets. The tools encompass plotting, GIS, statistics, and animation. The Output Visualizer displays include those of travel plans, vehicles, signals and intersection queues, among other data types.

The graphical user interface (GUI) of the Output Visualizer both allows the user to manipulate the three-dimensional objects and presents the output in a graphical way. A user could be interested in the travelers' plans (single, aggregated or filtered) that could be overlaid on a network as shown in Figure-6.14. As can be seen from the figure, such a representation would clearly help in understanding of the plan data characteristics as overlaid over the network.

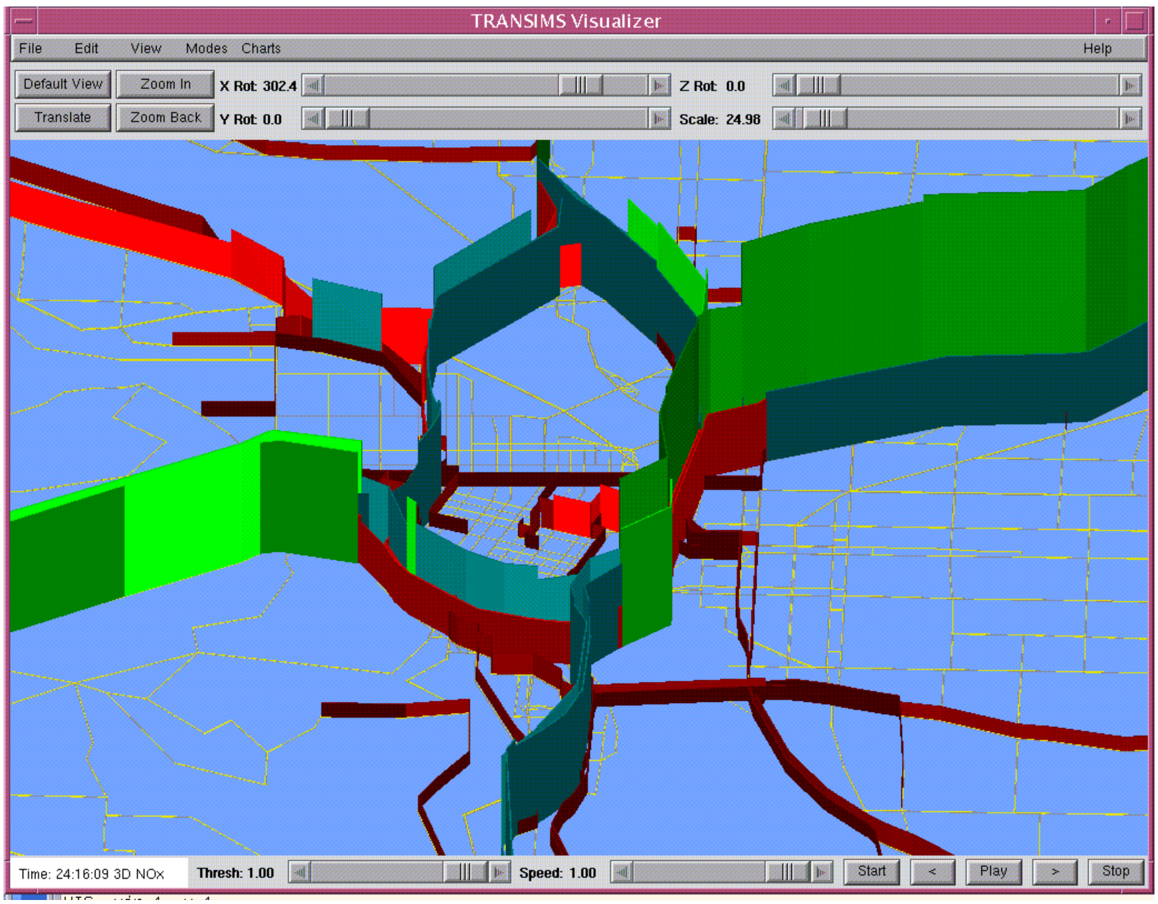


Figure-6.14: The Output Visualizer Displaying Cumulative Travelers' Plan Data

Data pertaining to vehicles' characteristics could also be examined by using the output visualizer. A specific study might involve a close following of a certain type of vehicle such as trucks and buses. This can be easily achieved by using a color scheme for different types of vehicles. The graphical nature of the display presented by the visualizer helps in achieving this objective.

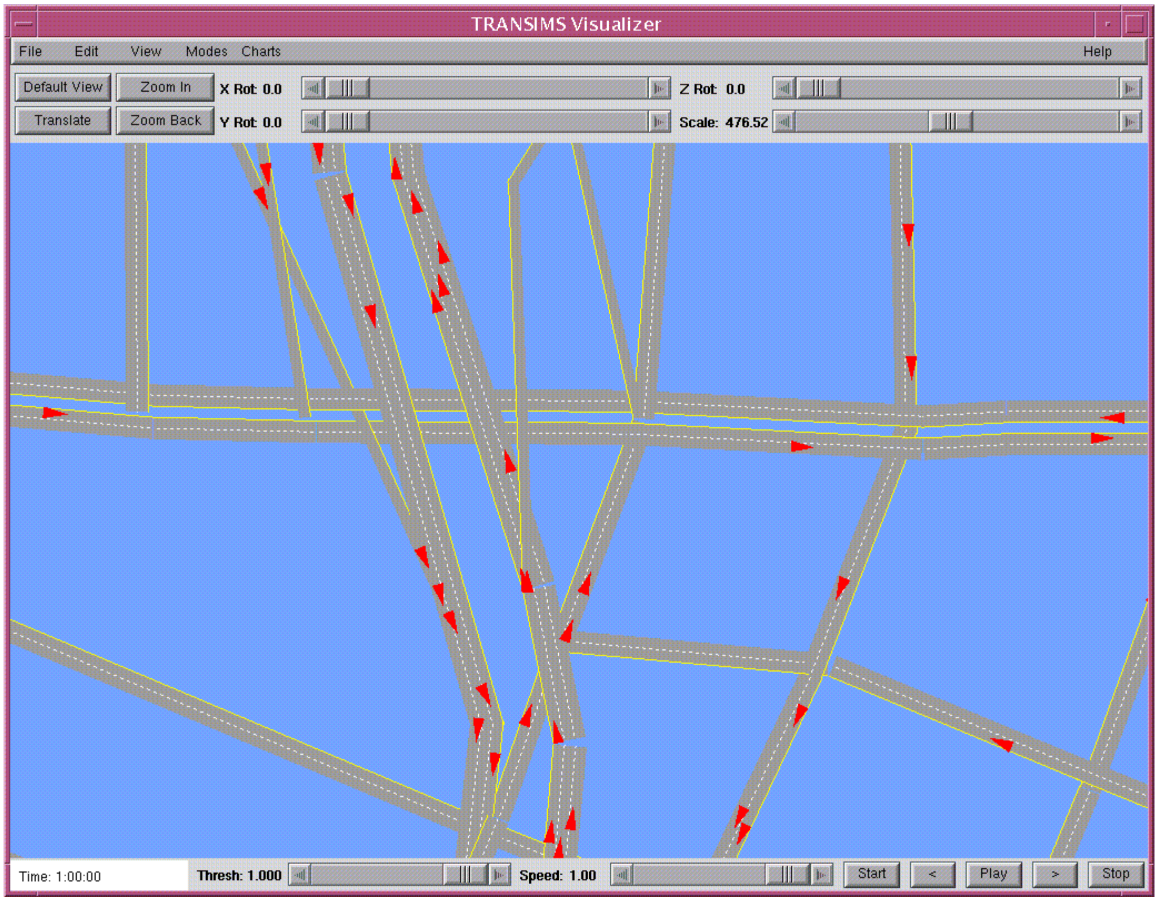


Figure-6.14: The Output Visualizer Displaying Vehicles on the Portland EMME/2 Network in a Zoomed-in View

Features such as zoom, pan, rotate, etc. facilitate the analysis of the congested areas of the network. The summary data from the Microsimulator such as the link densities or the link average velocities can be drawn or even animated as shown in Figure 6-1523. Selected parameters could be also displayed for a specific geographic area of interest. Other parameters could include anything from plans to vehicle emissions

Apart from the above-mentioned features, the Output Visualizer is capable of animating the simulation as it progresses. Essentially, the snapshot data (that is output from the Microsimulator) feeds the frames for animation. It should be noted that an animated simulation for the whole network would include a lot of data and would result in very huge files which require a lot of memory and disk space.

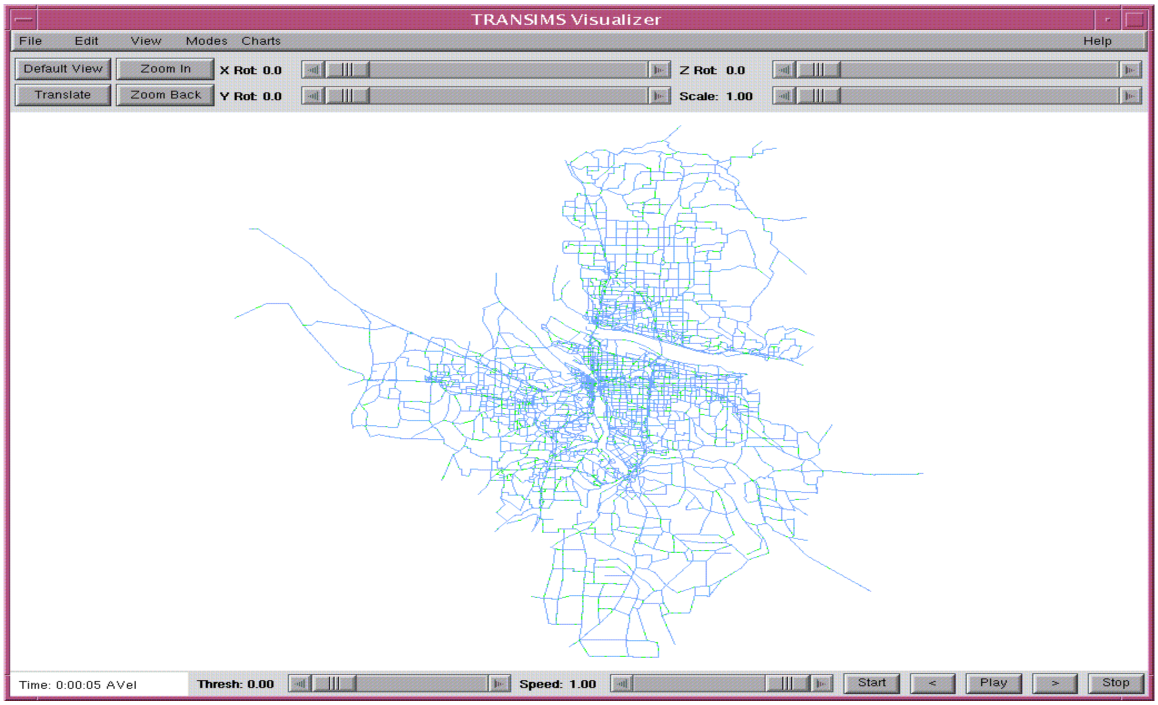


Figure-6.15: The Output Visualizer Displaying Average Velocity Summary Data

6.7. Module Interfaces

An outline for the module interfaces involving the Microsimulator Module is displayed in Figure-6.17.

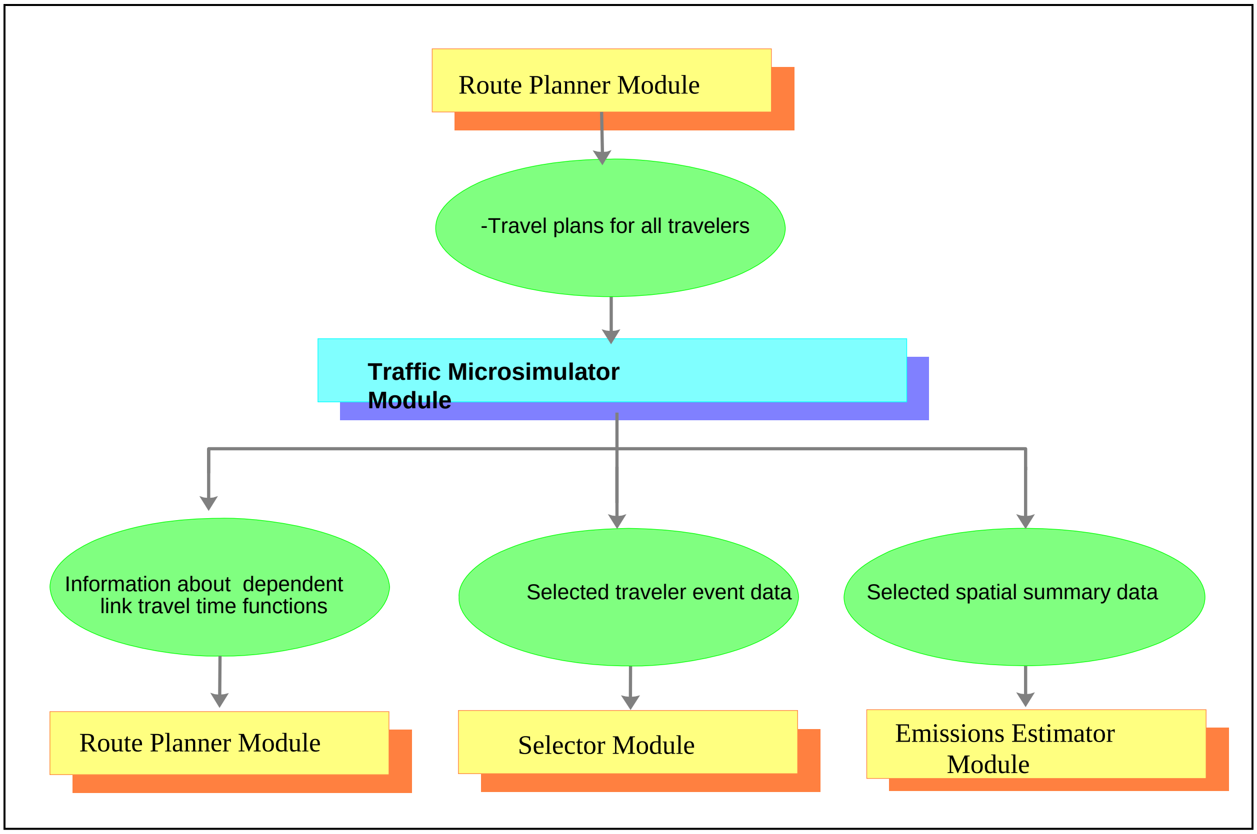


Figure-6.17: Flowchart of the Module Interfaces for the Microsimulator Module

6.7.1 Inputs from the Route Planner Module

Once the shortest travel plans are generated for all the travelers by the Route Planner, they are fed into the Traffic Microsimulator Module as shown in Table-6.5. Tables 6.19 to 6.23 list the additional files sent from the Route Planner Module to the Traffic Microsimulator Module differentiated by travel-modes and whether the traveler is a car driver or a car passenger.

Table-6.19: The List of Files for a Car Driver

File	**Description**
Vehicle ID	Each vehicle (with its ID) available in the simulation is listed in the vehicle database.
Number of Passengers	The number of passengers, not including the driver, on this leg.
List of Node Ids	The nodes (in order) through which the driver's route will pass.

Table-6.20: The List of Files for a Car Passenger

File	**Description**
Vehicle ID	Vehicle ID of the car in which the Car Passenger will travel on the leg of his/her plan.

Table-6.21: The List of Files for a Transit Driver

File	**Description**
Schedule Pairs	Number of (stop ID, depart time) pairs.
Vehicle ID	Each vehicle (with its ID) available in the simulation is listed in the vehicle database.
Route ID	Route Ids are specified in the transit route file. Only one route ID is allowed per leg.
List of Node Ids	The nodes (in order) through which the driver's route will pass.
List of Schedule Pairs	Each pair consists of a stop ID and a depart time. When a transit driver arrives at a transit stop who's ID is given in this list, the driver will remain at that stop until the departure time.

Table-6.22: The List of Files for a Transit Passenger

File	**Description**
Route ID	Traveler will board any transit vehicle whose driver's plan matches this Route ID.

Table-6.23: The List of Files for a Pedestrian

File	**Description**
List of Node Ids	The nodes (in order) through which the traveler's route will pass.

For the non-transportation activities, the Route Planner sends no files to the Traffic Microsimulator Module.

6.7.2 Outputs to the Route Planner Module

The output from the Traffic Microsimulator Module going back into the Route Planner comprises the information about time-dependent travel time functions for all the links in the Internal Network. Table-6.24 lists all the Configuration Keys that are sent to the Route Planner Module that contain information about travel time functions.

Table-6.24: The List of Configuration Keys Sent to the Route Planner Module that Contains Information about Travel Time Functions

File	Description
ROUTER_LINK_DELAY_FILE	Feedback file from which to read link delays/travel times. If the key is not present or the file does not exist, the free speed delays are used.
ROUTER_WALKING_SPEED	Speed to use when computing travel times for walk links (meters/second). **Default 1.0**.
ROUTER_BIKING_SPEED	Speed to use when computing travel times for walk links traversed by bicycle (meters/second). **Default 4.0**.
ROUTER_GET_ON_TRANSIT_DELAY	Delay/travel time encountered when boarding a transit vehicle. **Default 3 seconds**.
ROUTER_GET_OFF_TRANSIT_DELAY	Delay/travel time encountered when exiting a transit vehicle. **Default 4 seconds**.

6.7.3 Inputs to the Selector Module from the Microsimulator

One of the modules that use the output from the Microsimulator is the Selector Module. The Selector is the key module for TRANSIMS to reach an equilibrium condition quickly after starting the simulation. Table-6.25 summarizes the contents of the iteration database used by the Selector Module that is supplied by the traveler event file from the Microsimulator.

Table-6.25: Description of Iteration Database Fields

Field Description	Description
ACTUAL_ARRIVAL_TIME	the actual arrival time (measured in seconds past midnight) at the activity
NUM_STOPS	the number of stops signs the traveler encountered on this leg
TIME_STOPPED	the number of seconds the traveler was stopped in traffic on this leg
TOTAL_DISTANCE	the total distance (measured in meters) traveled on this leg
TOTAL_TIME	the total time (measured in seconds) traveled on this leg

The fields in Table-6.25, which are mainly obtained from the Microsimulator Travel Event File, are used by the selector module in evaluating the scenario at the current position and aid in the decision to reach a faster or smoother equilibrium. After updating the iteration database, the Selector builds a variety of cost functions that are used in the feedback process to other modules: namely,

- *Duration cost:* $c_{duration}(i_{traveler,leg}) = \frac{T_{actual} - T_{expected}}{T_{expected}}$, where T_{actual} is the actual travel time for the trip as realized by the Traffic Microsimulator and $T_{expected}$ is the expected travel time for the leg as estimated by the Route Planner. This measures travel time "frustration."
- *Distance cost:* $c_{distance}(i_{traveler,leg}) = \frac{D_{actual} - D_{geometric}}{D_{geometric}}$, where D_{actual} is the actual distance traveled in the Traffic Microsimulator and $D_{geometric}$ is the point-to-point Euclidean distance between the leg's endpoints. This measures how far out of his or her way the traveler goes.
- *Stopped cost:* $c_{stopped}(i_{traveler,leg}) = \frac{T_{stopped}}{T_{actual}}$, where $T_{stopped}$ is the time stopped in traffic and T_{actual} is the total travel time. This measures the fraction of the time a traveler spends waiting.
- *Late cost:* $c_{late}(i_{traveler,leg}) = A_{desired} - A_{actual}$, where $A_{desired}$ is the arrival time desired by the Activity Generator and A_{actual} is the actual arrival time realized by the Traffic Microsimulator. This measures how late the traveler is for his or her activity.
- *Effective speed cost:* $c_{speed}(i_{traveler,leg}) = \frac{D_{geometric}}{T_{actual}}$, where $D_{geometric}$ is the point-to-point Euclidean distance between the leg's endpoints and T_{actual} is the total travel time. This measures the traveler's effective speed through the network.

6.7.4 Inputs to the Emissions Estimator from the Microsimulator

The outputs from the Microsimulator also produce most of the information required by the Emissions Estimator Module. The necessary information from the Microsimulator is:

- spatial summaries of vehicle velocities over 30-meter sections of roadway,
- histograms of the number of vehicles entering a link grouped by velocity-acceleration product summed over time since the vehicles were parked. (This input is discussed earlier in section 6.5.1.4 titled Link Energy Summary.)

One of the important files from the Microsimulator that is used by the Emissions Estimator is the *velocity.out* file that contains the *link velocity summary data*. However, this file is reformatted for input into the Emissions Estimator by using the utility *ConvertVELfile* program. Listed in Table-6.26 are the various fields in the reformatted file with a brief explanation of what each field stands for. The file format of the raw file containing the *link velocity summary data* was shown earlier in Table-6.8.

Table-6.26: Velocity.out File Data Format (assuming the microsimulation was running with OUT_SUMMARY_VELOCITY_BINS set to 6)

Field	Interpretation
NV	Number of velocity records for this link, equivalent to the number of boxes that partition the link.
TIME	Current time (seconds from midnight).
LINK	Link ID being reported.
NODE	Node ID from which the vehicles were traveling away.
LENGTH	Length of box.
COUNT0	Number of vehicles with velocities in the range [0, 7.5).
COUNT1	Number of vehicles with velocities in the range [7.5, 15).
COUNT2	Number of vehicles with velocities in the range [15, 22.5).
COUNT3	Number of vehicles with velocities in the range [22.5, 30).
COUNT4	Number of vehicles with velocities in the range [30, 37.5).
COUNT5	Number of vehicles with velocities in the range [37.5, infinity).

A more detailed explanation of the files that are used by the Emissions Estimator Module and those supplied by the Microsimulator Module are presented in the Chapter on the Emissions Estimator Module.

6.8 Algorithms

6.8.1 Introduction

As stated in earlier sections, the Microsimulator first reads in the transportation network. Next it reads the type and location of every vehicle. Then the travelers' plans are read in as needed, according to the starting time of the simulation. The travelers are placed on the network and allowed to move from their origins to their destinations. For non-simulated modes, such as movement from a transit stop to a parking place, a traveler is removed from the buffer for one activity (transit stop) and placed in the buffer for another activity (parking place), with a new departure time reflecting the estimated duration of the trip in the process link. Vehicles are moved from one grid cell to another using the rules embedded in the CA approach, to be described next, with modifications to support lane changing and plan following until they reach the end of the grid. There they wait for an acceptable gap in the traffic or for protection at a signal before moving through the intersection onto the next grid. This continues until each vehicle reaches its destination, where it is removed from the grid.

To illustrate the overall working of the microsimulation module, a single trip example is presented next. This example considers a six-leg multimodal work-to-home trip that begins and ends at activity locations that are coded in the TRANSIMS network. The six different legs of the trip are detailed below.

Leg 1: walk from activity location *W* to bus stop X, where *W* is the work activity location and *X* is a bus stop in the network description.
Leg 2: take route *Y* to bus stop Z.
Leg 3: walk to parking lot P.
Leg 4: drive to day care at activity location D.
Leg 5: drive (with one passenger) to parking location *P*2.
Leg 6: walk to activity location *H* (home).

Walking Legs
For walking legs, TRANSIMS does not explicitly simulate the second-to-second locations of pedestrians. The walking trip begins at the time specified by the intermodal planner at the point of origin for the leg. The traveler arrives at the destination point of the leg at a simulation time computed by adding the delay time (contained in the plan) to the start time for the walk mode leg. No additional information is required or generated for walk-mode legs.

Bus Legs
Bus-leg plans require information on the acceptable bus routes. The precise itinerary and schedule of the bus the traveler boards is determined by the bus driver's plan, which is similar to all other travelers' plans. The traveler simply boards the bus at a bus stop and rides it until his or her desired stop is reached, at which point he or she exits the bus.

The microsimulation explicitly represents bus loading and unloading. Resource constraints such as vehicle capacity and transit stop capacity are observed. If a bus is full when it reaches the bus stop, a traveler is not permitted to board and will wait for the next bus on the same route. This level of detail makes it possible to determine the number of passengers who cannot find space on the bus or how many minutes a traveler waits for a bus.

Parking Lot
After getting off the bus, the traveler must walk to the parking lot. In this instance, the parking lot is where the traveler left his or her private vehicle. This walking leg is handled as previously described.

Driving Legs
Upon arriving at the parking lot, the traveler is associated with a specific vehicle, which either must have been left in the parking lot earlier in the simulation or placed there during initialization.

The traveler and car exit the parking lot and enter the traffic network. The traveler's plan specifies exactly which turns he or she will take until he or she arrives at the daycare center. At this point, the traveler waits until the passenger enters the vehicle. The passenger's plan will specify what vehicle to ride in, and the passenger will be waiting for this vehicle to arrive. The driver's plan specifies how many passengers to pick up. Once again, the driver re-enters the transportation network, following his/her plan, parks, and walks home.

In carrying out the movements of travelers and vehicles on the transportation network, the Microsimulator invokes several procedures, which can be categorized as follows:

1) Placing Travelers and Vehicles on the Network,
2) Updating the location of each Traveler and Vehicle,
3) Preparing for a Timestep,
4) Cleaning up after a Timestep, and
5) Supporting Parallel Computation.

The following sections address these five procedures.

6.8.2 Placing Travelers and Vehicles on Network

The placement of the travelers and the vehicles on the network takes place at the start of the simulation. In this initialization step, all the input information required to run the Microsimulator is read from the vehicle, the plan, the transit route, and the network files. The vehicle and the plan files are accessed through an index, which are generated from an appropriate file. A list of Vehicle IDs located at each parking accessory is initialized from the vehicle file.

The Microsimulator reads in the traveler plans (i.e., legs of a plan) using the index sorted by expected departure time, until all the plans departing before or on the current simulation time are read. In addition, the IDs of the hibernating travelers (those who have already executed one leg of their plan and are waiting to depart on another) are popped off the queue of Arrived Traveler list. The next leg of plan for each of the arrived travelers is read using an index that is sorted according to the traveler ID.

For a traveler to get onto the transportation network, the corresponding plan needs to be: a) local; originating in an accessory (transit stop or parking place) that is a part of the network under the control of the CPU and b) should be active; expected arrival time before the simulation start time and departure time before the simulation end time. The transportation network is partitioned into several CPUs to facilitate the running of the simulation.

At the start of the simulation, if the read plan is active, local, and calls for a non-simulated mode of travel (walk, bicycle, and activity), the traveler is placed into the arrived traveler queue at the destination accessory with a new departure time reflecting the time taken to reach the destination accessory (transit stop or parking place) as specified by the plan. If the destination accessory is not local, the traveler must migrate to another CPU, where he/she will be placed into the arrived traveler queue for that CPU as explained in the flowchart shown in Figure-6.18. However, if the traveler uses a simulated mode of travel involving a vehicle and his/her plan is not in progress (i.e., the departure time is after the simulation start time), he/she is placed in a queue at the origin accessory.

All the vehicles whose drivers' plans are in progress (departure time before the simulation start time and the arrival time after the simulation start time) are placed on the roadway links based on the prediction of their locations at the start of the simulation time. This is made possible by estimating the plan's geometric length and by selecting the link along the leg path using interpolation based on the duration of the leg in comparison to the start time of the simulation. However, if the whole leg of the plan is not local to one CPU, the determination of the length is difficult. Hence the initial conditions vary depending on the number of CPUs.

The traveler is placed randomly on a selected cell of the link. However, if the selected cell is occupied, a new cell is sought upstream of the initially selected cell. In the case that all the upstream cells are occupied, a cell is sought downstream. If all the downstream cells on the link are occupied, a warning message is generated and the vehicle is deleted. No attempt is made to find an available cell on an adjacent link.

Transit vehicles are placed on the network by interpolating their location at the beginning of the simulation start time. Transit passengers at the start of the simulation are placed directly at their destinations.

If the interpolation scheme does not run satisfactorily, the user should start the simulation at an earlier time.

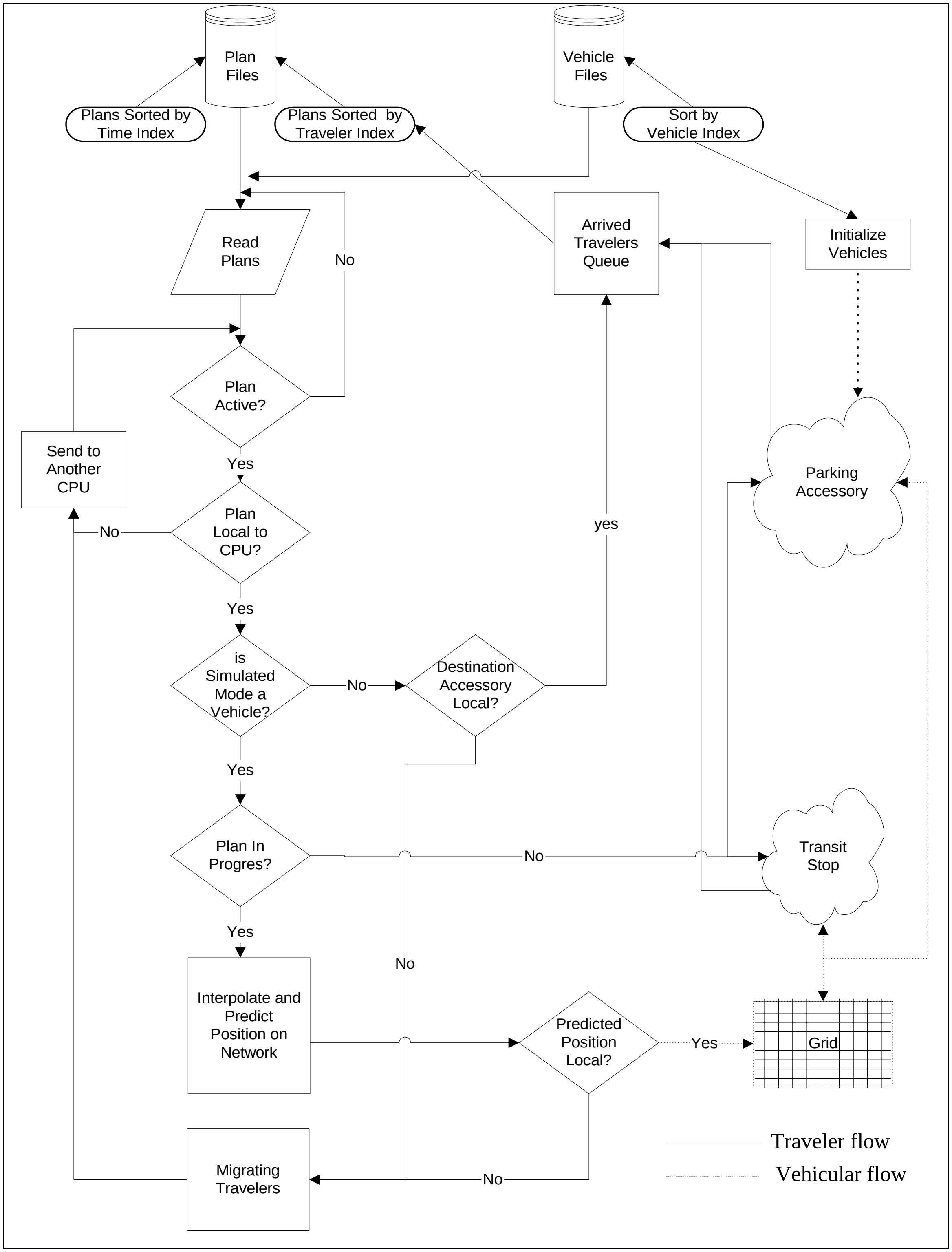


Figure-6.18: Flowchart Explaining the Placement of Travelers and Vehicles on Network

6.8.3 Update Travelers Locations

After reading and placing travelers, the simulation executes their plans one step at a time. A single time step is broken down into several events as shown in Figure-6.19.

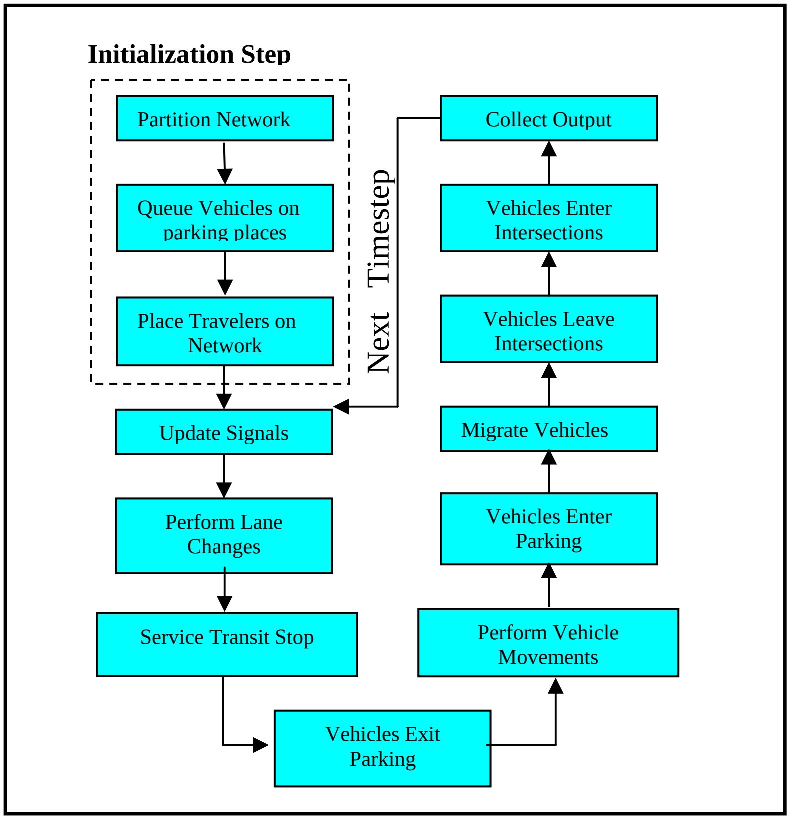


Figure-6.19: Microsimulator Steps in Each Timestep Update

To accomplish a simulation update of the vehicle movements involved in carrying out each traveler plan, the following steps are executed:

a) Perform lane changes for passing and lane following
b) Service Transit stop
c) Exit vehicles from parking places
d) Move vehicles in the same lane
e) Enter vehicles to parking places
f) Perform movements at Intersections: divided into
    i) Unsignalized Intersections
    ii) Signalized Intersections
g) Mark vehicles off-plan

Although the sequence for the time step update of vehicle movements is as stated above, the presentation of the material is given in a slightly different sequence in order to facilitate the understanding of the material by the reader.

6.8.3.1 Vehicle Movements In The Same Lane

Vehicles in the TRANSIMS network follow simple rules that govern their movements. These rules are intentionally kept simple to enhance the computation speed considering the millions of interactions taking place in the system. At the core, the rules for a vehicle movement in the same lane could be put simply as

- Acceleration whenever possible
- Deceleration only if necessary and sometimes for no reason.

The choice of a vehicle to accelerate or decelerate depends on the following:

a) The current vehicle speed and the minimum of the following three speeds which set the maximum velocity (Vmax) of the vehicle on a specific link:

i) The speed limit on the link

This is calculated by converting the link speed limit from m/s to cells per time-step and adding 0.5 cells and deceleration probability in cells to it and taking the lower integer bound of the number.

$$Speed\ Limit = Int\ [Speed\ Limit_{Network} / Cell\ Length + 0.5 + Deceleration\ Probability]$$

ii) The maximum vehicle attainable speed

This is calculated by converting the maximum attainable velocity of the vehicle obtained from the vehicle prototype file from m/s to cells per time-step and taking the upper bound integer ceiling of the number.

$$Max\ Vel_{\ (cells/time\text{-}step)} = ceil\ (Max\ Vel_{(m/s)} / Cell\ Length)$$

iii) The maximum global velocity,

This is a configuration key, and value currently being used is 5 cells per time-step.

b) The gap between the vehicle and the vehicle immediately ahead in the same lane.

Another factor that influences the movement of a vehicle is the deceleration probability (P_d), which can be thought of as the probability of a vehicle decelerating in the time step. All the vehicles in the TRANSIMS network are constrained by a maximum attainable speed that is specified ($V_{GlobalMax,}$ which is 5 cells/time step or about 80 mph).

Consider a vehicle traveling at a certain speed at a given time step. If the vehicle speed is greater than the gap ahead, the vehicle needs to reduce its speed to avoid a collision. The amount of deceleration depends on how large or small the current gap (Gc) is compared to the current speed. To model aggressive breaking an element of randomness in the form of P_d is used. If the probability of decelerating is greater than a certain threshold value (P_{noise}), the speed of the vehicle is further reduced beyond what is actually needed based on gap (G_c).

Considering the scenario where the gap ahead of the vehicle is larger than its current speed, then the vehicle can accelerate. The magnitude of acceleration is specified differently for each type of vehicle. For both autos and trucks, the maximum acceleration A_t, is specified in the vehicle prototype file. But for trucks and other vehicles, the current speed and the grade of the road play an important role in determining the magnitude of acceleration, the initial acceleration used when velocity is 1 cell/second or less will be the maximum acceleration as specified in the vehicle prototype file and if moving at a speed higher than 1 cell/time-step, the acceleration is calculated as assuming the constant power relationship, *"power= acceleration* x *speed = constant"*.

The acceleration expression for trucks at speeds more than 1 cell/second considering the grade is:

$$\textbf{Acceleration} = \textbf{Acceleration}_{(previous)}/\textbf{Speed}_{(current)} - 9.8 * \textbf{grade} / \textbf{CellLength}$$

If the speed is less than or equal to 1 cell/second, use the max acceleration in vehicle prototype,

$$\textbf{Acceleration} = [\textbf{Acceleration}_{prototype} - 9.8 * \textbf{grade}] / \textbf{CellLength}$$

All velocity and acceleration changes are integer values based on the number of cells/second or cells/second/second respectively. In those cases where the calculated A_t is fractional then it is randomly increased to an integer (60% of the time) or decreased to an integer (40% of the time). For example an acceleration of 1.6 cells/second/second is implemented as 2 cells/second/second (for 60% of the time) and 1 cell/second/second (for 40% of the time).

In the case that the vehicle is traveling at maximum allowable speed and has enough gap ahead to accelerate, the vehicle stays at maximum speed. As explained earlier, the probability for deceleration is randomly activated and the vehicle's speed may be reduced by 1 cell/second. A flowchart, illustrating the logic for vehicles movement in the same lane, is outlined in Figure-6.20.

To illustrate the above rules for general movement in a lane, the following pictorial examples on speeds are provided including their calculations.

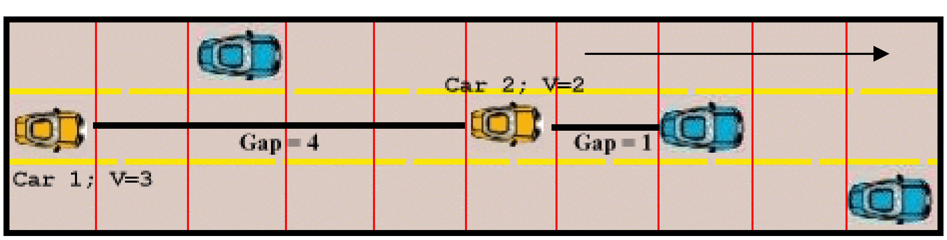


Figure-6.20: In-Lane Movement of Car 1 Based on Gaps at T=t

Consider the movement of car 1 shown in Figure-6.20. The gap ahead of it is 4 cells, and its current speed is 3 cells per time step or second. Since the gap ahead is more than the speed of the vehicle, acceleration is attempted. A random number is generated and for the sake of this example assume that this number is greater than the deceleration probability. Hence, the vehicle (car 1) will not maintain its speed. Now, the speed of car 1 in the next time step (t+1) would be equal to the gap ahead, which is 4 cells per time step. Car 1 would also move by an amount of the speed computed in the direction of motion i.e., by 4 cells as shown in Figure-6.21. This is true as long as the speed limit on the link allows a speed of 4 cells/second.

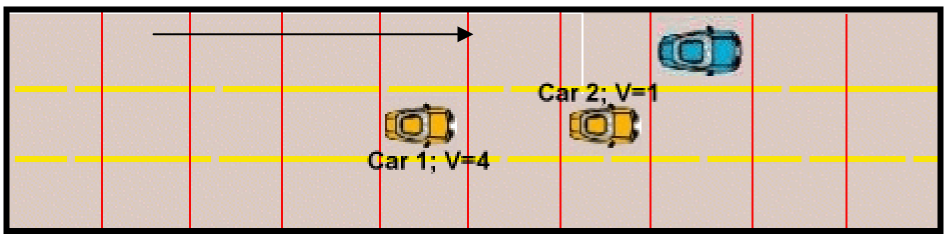
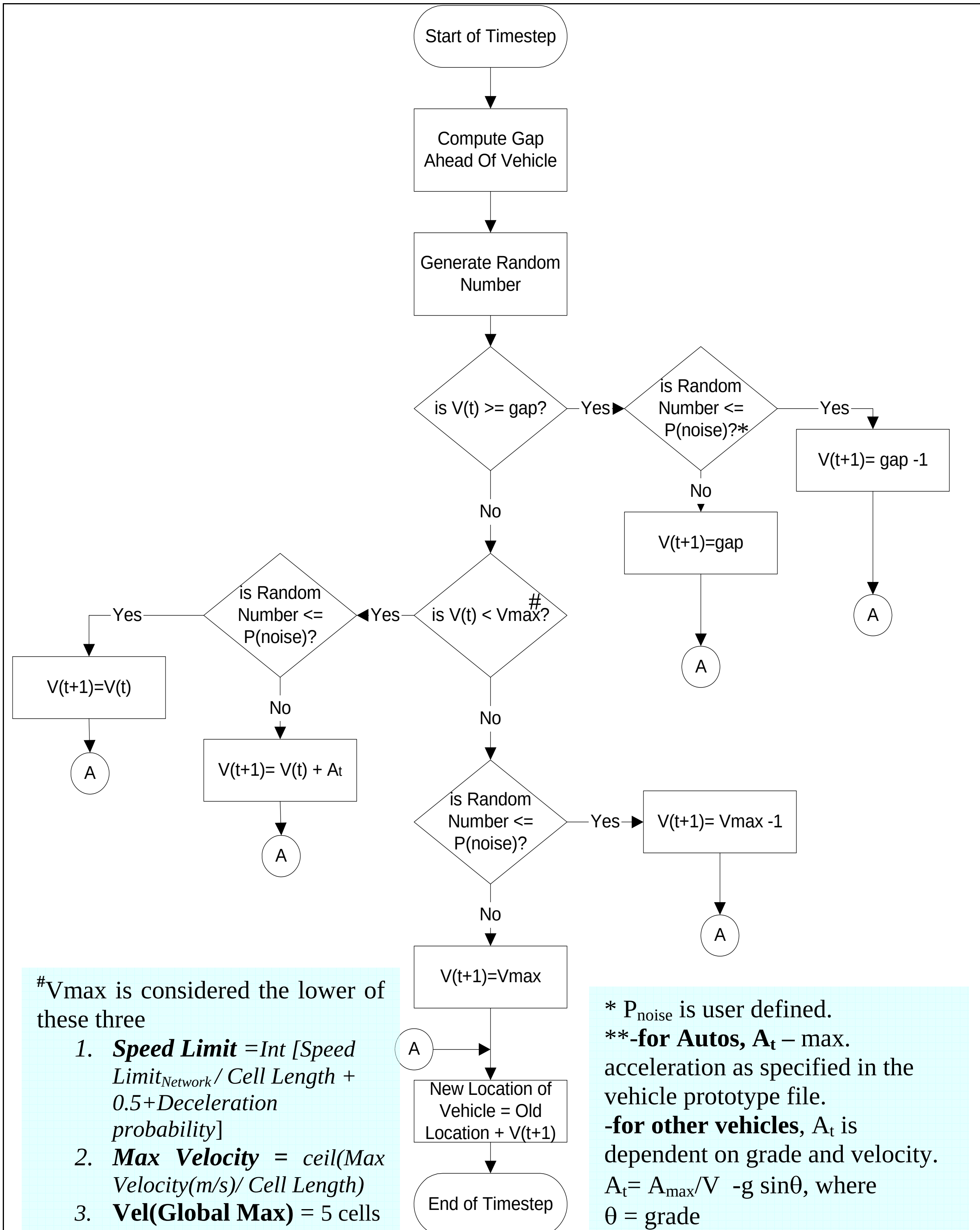


Figure-6.21: Position and Speed of Car 1 Based on Gaps At T=t+1

The analysis for in-lane movements of car 1 and car 2 is also presented in an algorithmic manner to provide a better understanding of the rules.

For Car 1: Current Velocity, V(t)=3 cells per time step.

Compute gap ahead Generate random number If V_t >= gap If random number <=P_{noise} Then V_{t+1} = gap –1 Else Vt_{+1} = gap Else If $V_t < V_{max}$ If random number <=P_{noise} Then $V_{t+1} = V_t$ Else if $V_{t+1} = V_t + A_t$ Else If (($Vt = V_{max}$) and (V_t < gap)) If random number <=P_{noise} Then $V_{t+1} = V_{max} - 1$ Else $V_{t+1} = V_{max}$	Gap ahead =4 cells Random number = 0.78; P_{noise} = 0.05 (assumed) V_t = 3 cells/time step and gap ahead = 4 cells Since (V_t < gap) AND ($V_t < V_{max}$) AND random number > P_{noise} $V_{t+1}=V_t+A_t$ (A_t is assumed to be 1 for this vehicle) V_{t+1} = 3 + 1 = 4 cells/time step Location at time t+1 = current cell + V_{t+1} = current cell + 4 cells



#Vmax is considered the lower of these three

1. ***Speed Limit*** =*Int [Speed Limit*$_{Network}$*/ Cell Length + 0.5+Deceleration probability*]
2. ***Max Velocity*** = *ceil(Max Velocity(m/s)/ Cell Length)*
3. **Vel(Global Max)** = 5 cells / time-step

* P_{noise} is user defined.
-for Autos, A_t** – max. acceleration as specified in the vehicle prototype file.
-for other vehicles, A_t is dependent on grade and velocity.
$A_t = A_{max}/V - g \sin\theta$, where
θ = grade

Figure-6.22: Flowchart for General Movement of Vehicles in the Same Lane

For Car 2: Current Velocity, V(t)=2 cells per time step

Compute gap
Generate random number
If V_t >= gap
  If random number <=P_{noise}
    Then V_{t+1} = gap –1
    Else V_{t+1} = gap
Else If V_t < V_{max}
  If random number <=P_{noise}
    Then V_{t+1} = V_t
    Else if V_{t+1} = V_t+ A_t
Else If ((Vt = V_{max}) and (V_t < gap))
  If random number <=P_{noise}
    Then V_{t+1} =V_{max} – 1
    Else V_{t+1} = V_{max}
	Gap =1 cell
	Random number = 0.57; P_{noise} = 0.05
	V_t = 2 cell/time step
	
	Since (V_t >= gap) and random number > P_{noise}
	
	V_{t+1} = gap = 1 cell/time step
	
	Location at time t+1 = current cell + V_{t+1}
	= current cell + 1 cell

6.8.3.2 Performing Lane Changes

The lane-changing maneuver of a vehicle in TRANSIMS Microsimulator occurs to pass a slower vehicle immediately ahead or to make turns at intersections following its current plan. The decisions for lane changes take place before the in-lane movement of the vehicles on the links occur. This ensures that the in-lane movement of the vehicles takes into account the effect of lane changes.

The Microsimulator treats lane changes into the left lane and into the right lane on alternating time steps. The left lane changes are made on even time steps while the right ones are made on odd time steps. Multilane roadways are processed from left to right during left lane changing and from right to left during right lane changing procedures. It should be noted that these lane change procedures are only explored if the cell on the adjacent lane in which the vehicle is trying to change into is vacant.

The above mentioned lane changing procedures are discussed in detail below under two separate categories. One category is for lane changing based purely on passing a slower vehicle, and the other one is based on making turns at intersections to follow plan.

6.8.3.2.1 Lane changes based on passing slower vehicles

The lane changes based on this criterion occur only if the speed of the vehicle under consideration is more than or equal to the gap ahead of it in the current lane (G_c). Another important consideration is the magnitude of the gaps in the adjacent lane into which the vehicle is attempting a lane change. The gap ahead of the vehicle in the new adjacent lane (G_f) should be larger than the one in the current lane (G_c). The vehicle, before making the necessary lane change, should also consider if the vehicle behind it in the new lane is sufficiently far away (G_b) to avoid any kind of collision.

The above ideas are captured in the TRANSIMS Microsimulator using three variables Weight 1, Weight 2 and Weight 3. The values of these weights are computed as shown in Table-6.27. For a vehicle to make a lane change the following three criteria should be satisfied: Weight 1 be greater than zero; Weight 1 be greater than Weight 2, and Weight 1 be greater than Weight 3.

Table-6.27: Computation of Weights for Lane Changes for Passing Slower Vehicles

Parameter	Description	Equation
Weight 1	An integer value based on the gap in the current vehicle, the potential speed of the vehicle in this time step, and the gap forward in the new lane	Weight1=1 if ($V+1>G_c$) AND ($G_f > G_c$)
Weight 2	An integer value based on the gap forward in the new lane and the speed of the vehicle	Weight2= $V-G_f$
Weight 3	An integer value based on the gap backward and the maximum speed of a vehicle in the simulation	Weight3= $V_{Max} - G_b$

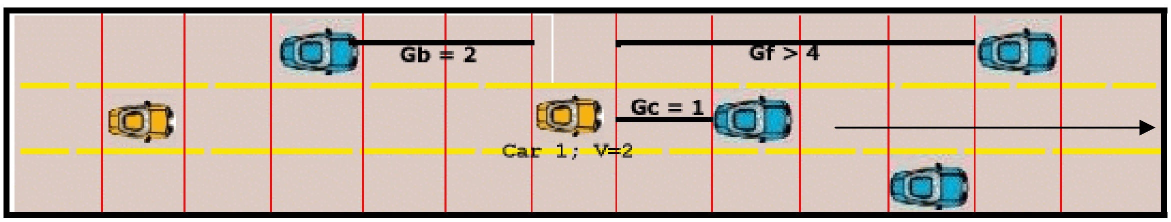


Figure-6.23: Left Lane Change Considerations for Car 1 at T=t

An example of the lane changing procedures based on passing slower vehicles is shown in Figure-6.23. The example shows car 1 moving at a speed of 2 cells per time step. Consider a time step when left lane changes are done first. As shown in Figure-6.23, the gap ahead in the current lane or G_c is 1 cell, the gap forward or G_f in the adjacent lane into which the vehicle is considering lane change is 4 cells. Using this information Weight 1 is calculated to be 1. Weight 2 is computed as ($V-G_f$) = 2-4 = -2. The gap backward or G_b in the new lane is 2 cells. Weight 3 is computed as 5-2 ($V_{Max} – G_b$) = 3. Considering the three Weights, a check is made to see if car 1 will make the left lane change.

Condition 1: Weight 1 > 0 (TRUE)
Condition 2: Weight 1 > Weight 2 (TRUE)
Condition 3: Weight 1 > Weight 3 (FALSE)

Since all the three conditions are not satisfied, car 1 cannot make a lane change into the left lane. The same analysis is presented next for car 1 attempting to make a lane change into the right lane in the next time step, as shown in Figure-6.24. These calculations are done in an algorithmic manner and provided in a tabular form below.

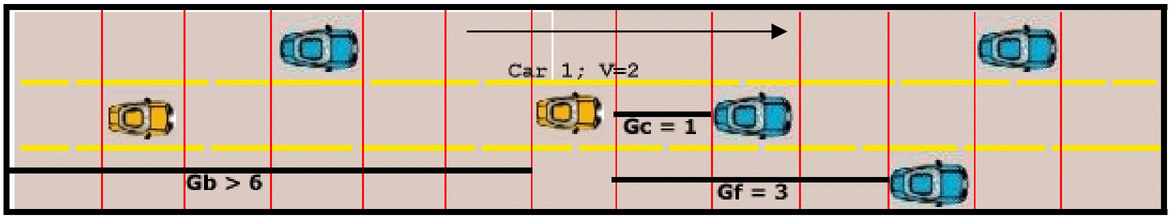


Figure-6.24: Right Lane Change for Car1

The results show that Car 1 is allowed to make the right lane change.

Lane change to get into the left lane for Car 1

If neighboring position in adjacent lane is empty	Neighboring position in adjacent lane empty
Calculate gap in current lane G_c Calculate gap forward in new lane G_f Calculate gap backward in new lane G_b	G_c = 1 cell G_f = 4 cells G_b = 2 cells
Using G_c, G_f, G_b calculate Weight1= $(V+1>G_c)$ AND $(G_f > G_c)$ Weight2= V-G_f Weight3= VMax - G_b	Weight1 = 1 = ((2+1 > 1) AND (4>1)) Weight2 = -2 = (2-4) Weight3 = 3 = (5-2)
If weight 1 > 0 And weight1> weight2 and Weight1> weight3 And lane change probability is affirmative And lane change not merge, turn or next link Move vehicle to new lane	Weight1 =1 (TRUE) Weight1 > Weight 2 (TRUE) Weight 1 > Weight 3 (FALSE) Since the three conditions are not satisfied, lane change into the left lane is not allowed.

Lane change to get into the right lane for Car 1

If neighboring position in adjacent lane is empty	Neighboring position in adjacent lane empty
Calculate gap in current lane G_c Calculate gap forward in new lane G_f Calculate gap backward in new lane G_b	G_c = 1 cell G_f = 3 cells G_b = 7 cells (not shown clearly in figure)
Using G_c, G_f, G_b calculate Weight1= $(V+1>G_c)$ AND $(G_f > G_c)$ Weight2= V-G_f Weight3= VMax - G_b	Weight1 = 1 = ((2+1 > 1) AND (3>1)) Weight2 = -1 = (2-3) Weight3 = -2 = (5-7)
If weight 1 > 0 And weight1> weight2 and Weight1> weight3	Weight1 =1 (TRUE) Weight1 > Weight 2 (TRUE) Weight 1 > Weight 3 (TRUE)
And lane change probability is affirmative And lane change is not a merge, turn or next link	Lane change probability affirmative and Lane is not a merge or turn or next link one.
Move vehicle to new lane	Hence lane change into the right lane is allowed based on passing a slower vehicle.

6.8.3.2.2 Performing Lane changes based on plan following

As a vehicle enters a link, acceptable lanes for transition to the next link in its plan are determined. From this, a particular lane is chosen to be the preferred destination lane. The preferred destination lane is generally the current lane if allowed onto the next link. In the event that the current lane is not acceptable, a preferred destination lane is chosen at random from the allowable set of lanes.

Lane changes based on plan following are triggered only when the vehicle is within a set distance from the intersection. This distance is specified by D_{pf}, the point on the link where a vehicle starts to consider lane changes to follow its plan. It can be easily understood that the urgency for a lane change to get into the desired lane based on plan following increases with the vehicle getting closer and closer to the intersection. It can also be understood that this urgency also increases with the number of lanes between the current lane and the preferred lane. Microsimulator uses these two factors in modeling a parameter (Weight 4) which represents the bias to make a lane change based on plan following. The value of Weight 4 is dependent on various conditions. Weight 4 is allocated a value based on which of the following conditions is true:

1. If the vehicle is currently in a merge lane,

Weight4 = Global_Max_Speed

2. If the vehicle is within the plan following cells distance and is not in its correct lane,

$$Weight4 = V_{\max} - \frac{(V_{\max} - 1) * D_i}{n * D_{pf}}$$ Where,

V_{max} is the max speed attainable by vehicle
D_i is the distance of the vehicle from the intersection
D_{pf} is the set distance from intersection where a vehicle starts to consider lane changes to follow its plan (specified in configuration file).
N is the number of lanes changes necessary to get into the preferred lane.

It can be seen from the above equation that, as D_i goes from $n*D_{pf}$ to 0, the values of Weight4 go from 1 to V_{max} indicating that it should always be a positive value. Weight4 is initially set to 0. As discussed earlier, left and right lane changes occur on alternating time steps even for lane changes based on plan following.

3. If the vehicle is within plan following distance and is in the correct lane and the adjacent lanes are not appropriate to allow vehicle turning at its intersection,

Weight4 = -1 i.e. the vehicle is restricted from making any lane changes

4. If the vehicle is in correct lane for plan following cells and next lane is acceptable, vehicle is still allowed to do lane changes based on gaps.

Weight4=0

The overall decision to change lane considers both plan following and gaps. The parameters are adjusted to reflect these conditions.

Weight1 = Weight1 (based on Gaps) + Weight4.

However if weight 4 is set to –1, it will prevent any passing lane changes based on gaps

The overall conditions for lane change remain the same as those based on passing slower vehicles i.e., Weight 1 >0; Weight 1> Weight 2 and Weight 1 > Weight 3.

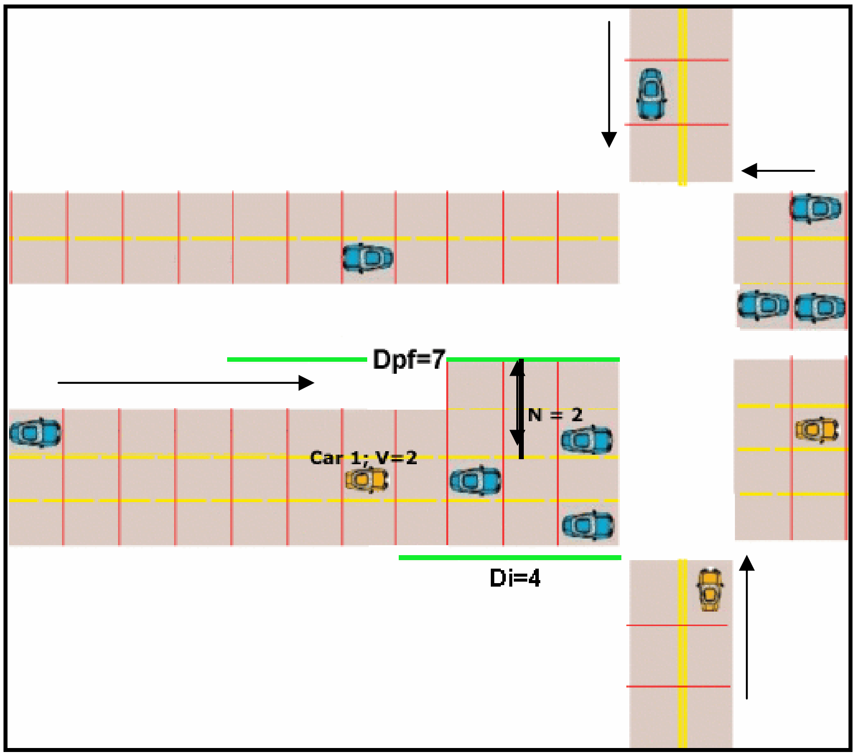


Figure-6.25: Example for Lane Change Based on Plan Following

An illustration of a lane change based on plan following is shown in Figure-6.25 above. In this example, the analysis for lane change is presented for Car 1, which is moving with a velocity of 2 cells per timestep. Let us assume at this point that this vehicle needs to make a left turn at the intersection, and hence needs to get into the left pocket lane. To get into the left pocket lane, two left lane changes need to occur i.e., n=2. It can also be clearly seen that car 1 is 4 cells away from the intersection or D_i = 4. For this particular example let us also consider that the lane change for plan following is considered when a vehicle is within 7 cells from the intersection i.e., D_{pf} = 7 cells. Taking into consideration all these factors, Weight 4 is calculated using the equation defined earlier.

$$Weight4 = V_{\max} - \frac{(V_{\max} - 1) * D_i}{n * D_{pf}}$$

= 5 – (5-1)*4/(2*7) =5- 1.14 = 3.87

Using Weight 4, Weight 1 is calculated as Weight1 = Weight 1 + Weight 4; The analysis then continues exactly as that for lane change based on gaps. For Example;

Weight 1 = 1+3.87 = 4.87
Weight 2 = 2 –3 = -1
Weight 3 = 5 –5 = 0

Since Weight 1>0, and Weight 1> Weight 2, and Weight 1 > Weight 3, then the lane change into the adjacent lane is approved. It is important to note that a car making 2 lane changes to reach its desired plan following lane needs to again calculate Weight 4 and the other three weights at timestep t+2, after the in-lane movement in the adjacent lane is carried in this timestep. For this example, the car needs to execute a turn pocket lane change at time t+2.

6.8.3.2.3 Merge Lanes

Merging is handled by using the same lane-change logic as described above. Vehicles in the merge lanes are forced to make lane changes in the direction of the merge. In an event where a lane has a merge pocket and a turn pocket further down towards the intersection, vehicles are prohibited from entering the turn pocket lane until past the end point of the merge pocket.

6.8.3.2.4 Turn Pocket Lanes

Vehicles attempting a lane change to enter a turn pocket lane from an adjacent lane are subject to speed restrictions that prevent movement of the vehicles past the start of the turn pocket.

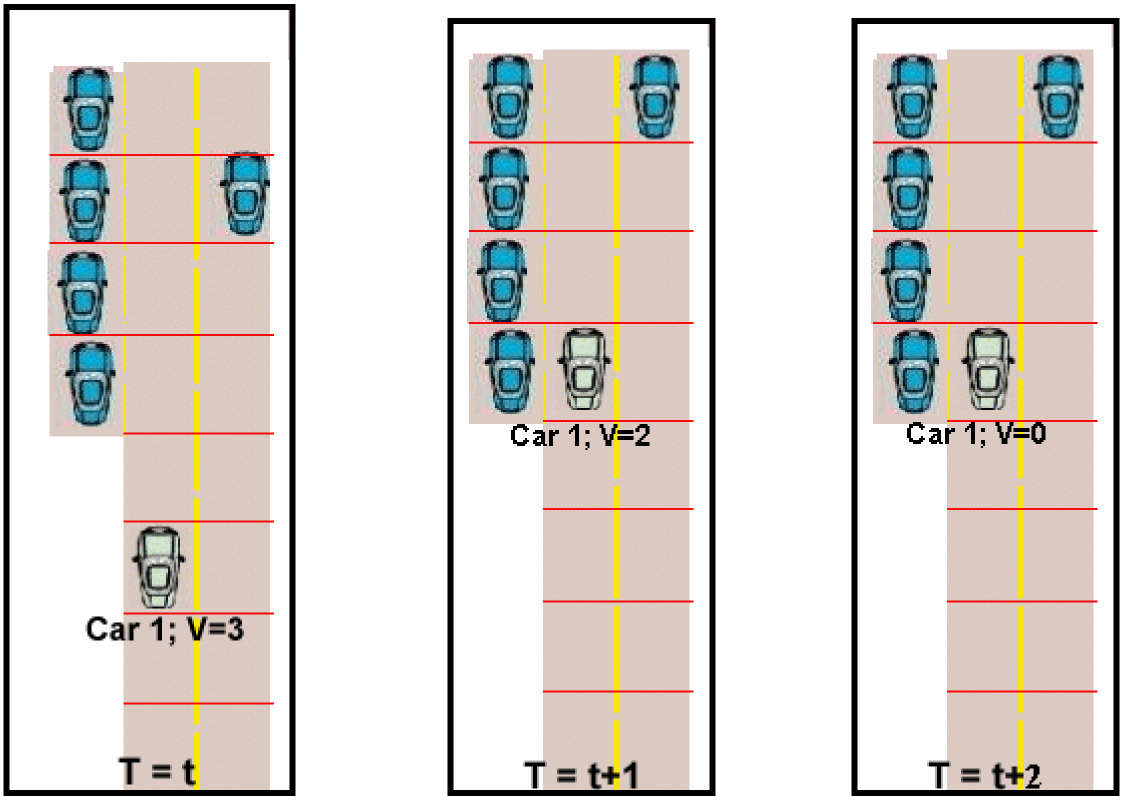


Figure-6.26: Queue Formation at a Turn Pocket

This restriction may cause vehicles to queue on the adjacent lane until a lane change is feasible into the turn pocket lane.

A typical case of queue formation on the lane adjacent to the turn pocket is shown in Figure-6.26. The first of the three figures shows the state of the simulation at time step t when Car 1 is traveling with a speed of 3cells/timestep. Since the vehicle will pass the start point of the turn pocket if it continues with the same speed, the vehicle decelerates and in the next time step (t+1) reaches the start of the turn pocket. The speed now is 2 cells/time step. In the next time step (t+2), since car 1 is already at the start of the turn pocket and the turn pocket is full, car 1 does not move any further and its speed drops to 0 cells/time step. Its speed will be constrained to zero until a lane change into the turn pocket is possible.

6.8.3.2.5 Look Ahead Across Links

Some vehicles may be unable to make the required lane changes into acceptable approach lanes on short multilane links with multiple lane connectivity at the intersections. Thus, looking ahead across links increases the time that a vehicle has to make a plan following lane change.

The acceptable approach lanes are determined based on a plan look ahead distance. The distance is used to determine how many links in the plan will be considered when determining the approach lanes on the current link. A distance of 262.5 meters (35 grid

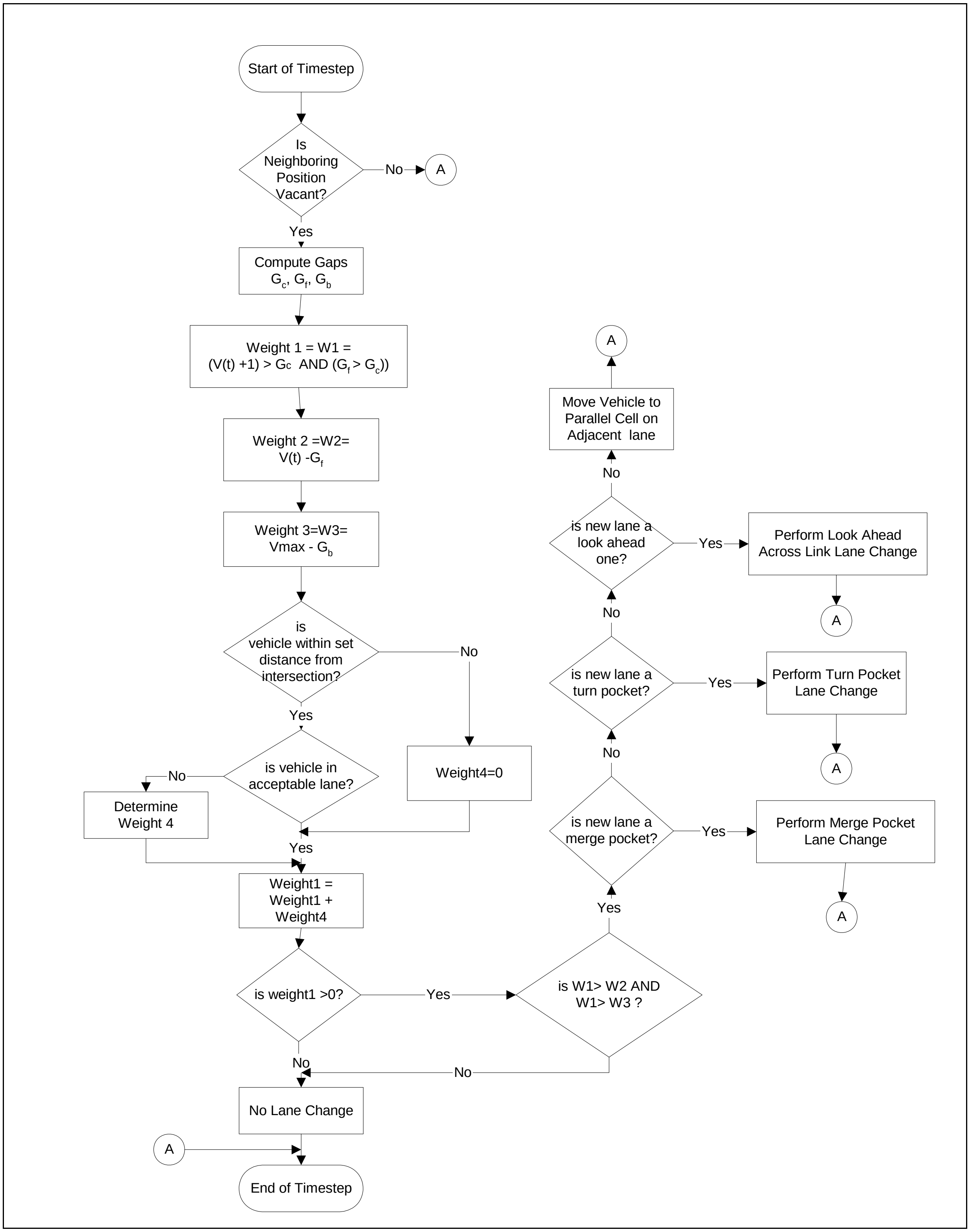


Figure-6.27: A Flowchart Representing the Lane Change Procedures

cells) is the default value. A value of 0.0 implies approach lanes are being determined by considering the next link only. A flowchart depicting the logic for all cases of lane changing in shown in Figure-6.27.

6.8.3.3 Servicing Transit Stops

Mass transit vehicles have a priority in lane changes in order to keep to their schedule and prevent them from going off-plan. Transit vehicles are allowed to enter transit stops during the lane changes. A transit vehicle with a specified route enters a transit stop if it is not full and there is a queue of people waiting to be served for that route ID at that stop. This also takes place, if any passenger wishes to get off at the stop and if the driver plans include a scheduled departure time for this stop. Entry and exit of passengers takes place simultaneously. The configuration keys set the mean rate at which travelers enter and exit.

Travelers leaving the transit vehicle who have completed a leg of their plan are placed into the Arrived Travelers list to trigger the Read plans process in order to find the next leg of their plans. If a traveler’s plan calls for him or her to take the route that this vehicle is servicing, and the number of passengers already aboard does not exceed the capacity for this type of vehicle specified by the vehicle prototype file, he or she will enter the vehicle. In that case, travelers are popped off the Arrived Travelers queue until either the maximum number of travelers who can board in a single time step is reached, or only those travelers whose next departure times are later than the current simulation time are left in the queue.

At the transit stop the vehicle is either left occupying the grid cells or is taken off the grid, depending on the type of transit stop. If the transit vehicle stays on the grid then it will attempt to get into the rightmost lane and its speed is constrained to zero. If all the passengers exiting at this stop have been taken care of and either the bus is full or no more passengers are waiting to board, the vehicle is placed back on the grid, and its speed constraint is removed. A flowchart depicting the logic of servicing a transit stop is shown in Figure-6.28.

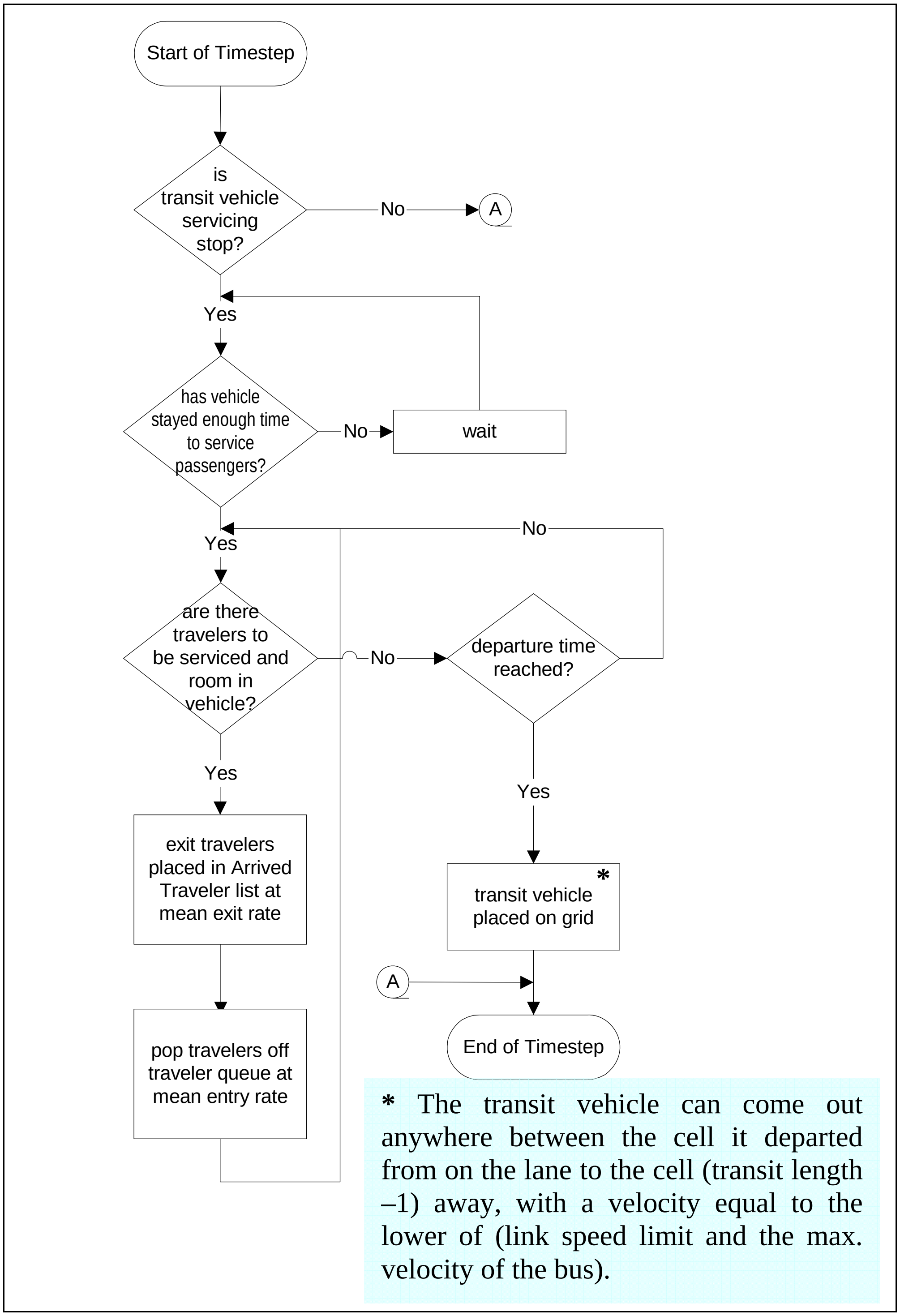


Figure-6.28: Flowchart Representing Servicing a Transit Stop

6.8.3.4 Vehicle Movements at Intersections

TRANSIMS Microsimulator uses a separate set of rules to model the vehicles leaving a link and passing through an intersection. Every vehicle arriving at an intersection, determines a destination lane on the next link onto which it will travel considering the lane use and HOV restrictions. By default the current lane is chosen if allowed on the next link, otherwise a lane is picked randomly from the set of allowable lanes.

Vehicles at signalized intersections have different behaviors from those at unsignalized intersections, though they share some common features. This section presents the way the Microsimulator handles both these cases. But first, the six conditions to be satisfied for a vehicle to enter an intersection are provided below.

Condition One: The vehicle has to be the first one on the link going toward the intersection. This is necessary since the Microsimulator allows only one vehicle per lane to enter an intersection in a single time step.

Condition Two: The speed of the vehicle trying to enter the intersection must be greater than or equal to the number of empty cells between it and the end of the link.

Condition Three: The vehicle entering the intersection should satisfy the conditions of the traffic control at the intersection. The state of the Traffic Control indicates if a vehicle must consider oncoming traffic gaps. This would ensure that a vehicle facing a red does not enter the intersection. For unsignalized intersections the traffic control would specify a condition of yield or stop. For signalized intersections it would represent a protected, permitted, or a wait traffic control state. This would specify if the oncoming traffic has to be considered for an entry into the intersection. The traffic controls and conditions at an intersection are shown in Table-6.28.

Table-6.28: Traffic Control States and Corresponding Actions

Traffic Control State	**Action**	**Conditions**
S* - Protected	Proceed	None
S – Wait	Stop	None
S - Permitted	Evaluate	G_i on IL (Interfering Lanes)
S - Caution	Proceed	None
U** -None	Proceed	None
U – Stop	Wait Evaluate	Stopped < 1 Timestep G_i on IL, Stopped ≥ 1 Timestep
U – Yield	Evaluate	G_i on IL

* S = Signalized intersection
** U = Unsignalized intersection

Condition Four: If the traffic control specifies that the oncoming traffic has to be considered then this condition checks to ensure that there is an acceptable gap between the turning vehicle and oncoming traffic. This is generally important for unsignalized intersections and permitted movements in signalized intersections.

Condition Five: This condition is only checked for vehicles trying to enter a signalized intersection. The intersection buffer for the signalized intersection is checked to verify if it can accommodate the entering vehicle. This is explained more in detail later in the section. Practically, this would represent a check to see if there is any space in the intersection to accommodate another vehicle.

Condition Six: A last check before a vehicle enters the intersection is to check if the destination cell on the destination link into which the vehicle will move is unoccupied. In the case that the destination cell is unoccupied and all other conditions are satisfied, the vehicle leaves the intersection buffer and follows the procedures pertaining to the destination link.

The remaining section looks at the procedures followed for signalized and unsignalized intersections separately.

6.8.3.4.1 Unsignalized Intersections

A major difference in the way in which Signalized and Unsignalized intersections are handled is the way vehicles can enter and exit an unsignalized intersection in a single time step. An unsignalized intersection with stop/yield type *traffic controls* requires every entering vehicle to consider oncoming traffic before it can move onto the next link. The TC (Traffic Control) state may require that the distance between the intersection and the on-coming traffic (interfering lane gap) meet certain criteria before the vehicle can enter the intersection. The turning vehicle uses the gap between the oncoming vehicles and the intersection to determine whether the intersection can be entered. If the gap is acceptable, the vehicle traverses the intersection and arrives on the destination link during a single update step in the Microsimulator.

The interfering lane gap (G_i) is the distance between the oncoming vehicle and the intersection, as shown in Figure-6.29. The oncoming vehicle must be on a link connected to the intersection, which limits the look-back distance for oncoming traffic to the length of a single link. The speed of the oncoming vehicle (V_{OV}) and the Gap Velocity Factor (GVF) are used to calculate the Desired Gap. The Gap Velocity Factor, which is the perceived safe distance which an oncoming vehicle is away from the intersection relative to its speed is specified by the configuration key, and has a default value of 3.

$$\text{Desired Gap } (G_d) = V_{OV} * \text{GVF}$$

On links where the desired gap is greater than the number of cells on the link, the number of cells on the link is used as the desired gap. If

$G_i \geq G_d$, then Interfering Gap is Acceptable
$G_i < G_d$, then Interfering Gap is Not Acceptable

It is important to note that for an oncoming vehicle with a speed of 0, G_d will also be 0, which allows movement through intersections in congested conditions where both G_d and G_i = 0.

For the example shown in Figure-6.29, Car 1 is not allowed to enter the intersection because its interfering gap (G_i) for the oncoming car 4 on link 4 is 4, which is less than the required G_d of 6 obtained by multiplying the car 4 velocity of 2 by the Gap Velocity Factor of 3.0.

If the interfering gap is not acceptable and the vehicle is at a stop or yield sign and the interfering lane is also controlled by a stop/yield sign, there will be a deadlock resolution in which the vehicle will proceed with probability determined by the value of the configuration key CA_IGNORE_GAP_PROBABILITY.

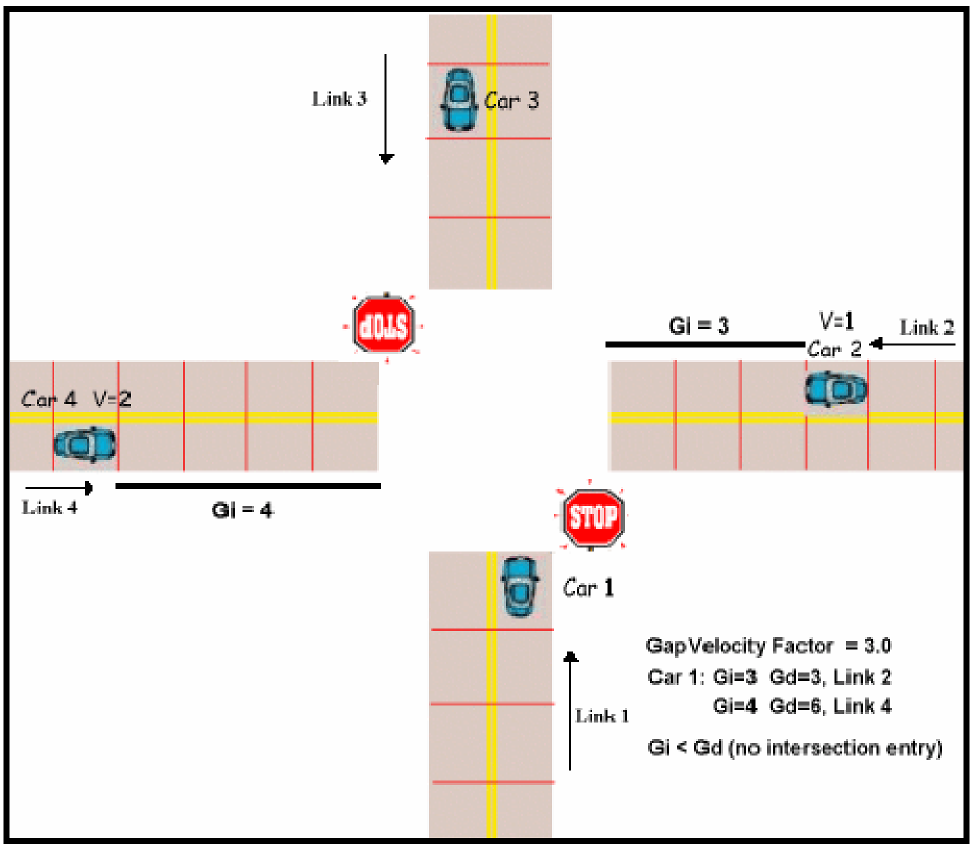


Figure-6.29: Vehicle Movement at an Unsignalized Intersection

The primary destination cell on the destination link to which the vehicle will exit from the intersection is determined by considering the current speed of the vehicle. In case the primary destination cell is occupied, a cell closer to the intersection is tried. Should it be occupied, a cell a bit closer to the intersection is tried until an unoccupied one between the intersection and the primary destination cell is found. Upon finding a vacant cell, a marker is placed on the cell to reserve the position and an internal state variable is set to indicate that the vehicle can proceed. This variable is further used during the movement procedure to determine whether to remove a vehicle from a link or decrease its speed. Vehicles traversing unsignalized intersections are placed on their destination link during the clean-up procedure at the end of a time step.

6.8.3.4.2 Signalized Intersections

At signalized intersections the vehicles do not traverse the intersection in the current Microsimulation time step. Instead they are placed in the *internal queued buffers* maintained by the signalized intersection. Each intersection has one queued buffer for each incoming lane. If the conditions of the signalized traffic control have been satisfied, a vehicle must check whether the appropriate buffer has space to receive the vehicle, which is one of the conditions for entry into the intersection.

The time that the vehicle spends in the queued buffer models the time necessary to traverse the intersection. Also the state of the TC at the intersection is an important factor in the decision on whether the vehicle can enter the intersection. The TC must indicate a permitted, protected, or caution movement for the current lane in order for the vehicle to enter the intersection. Vehicles with permitted, but not protected traffic control at a signalized intersection consider the oncoming traffic before entering the intersection just as in an unsignalized intersection.

After the specified time period has expired in the intersection buffer, the vehicle exits from the buffer to the first cell on the destination link if the cell is vacant. If not, the vehicle waits in the intersection buffer until the cell becomes vacant. Also these buffers have a fixed size so if the buffer is full, the vehicle cannot enter the intersection and must wait on the link. Figure-6.30 shows the logical flowchart for vehicle movements at an intersection.

6.8.3.5 Mark Vehicles Off-Plan

Any vehicle becomes off-plan if it is not in the acceptable approach lane while entering an intersection and thus cannot follow its assigned plan. Also, vehicles attempting to enter an intersection are marked off-plan if they have not moved for the duration of time specified by the configuration key CA_MAX_WAITING_SECONDS.

TRANSIMS Microsimulator deals with the off-plan vehicles by keeping the vehicle in simulation until a time again specified by the CA_OFF_PLAN_EXIT_TIME configuration key is reached, after which the vehicle exits at the nearest parking location. In order to keep the off-plan vehicle in simulation, a new destination link is selected from the links that are connected to the vehicles current lane. This method of choosing random links occurs until the exit time for the vehicle is reached.

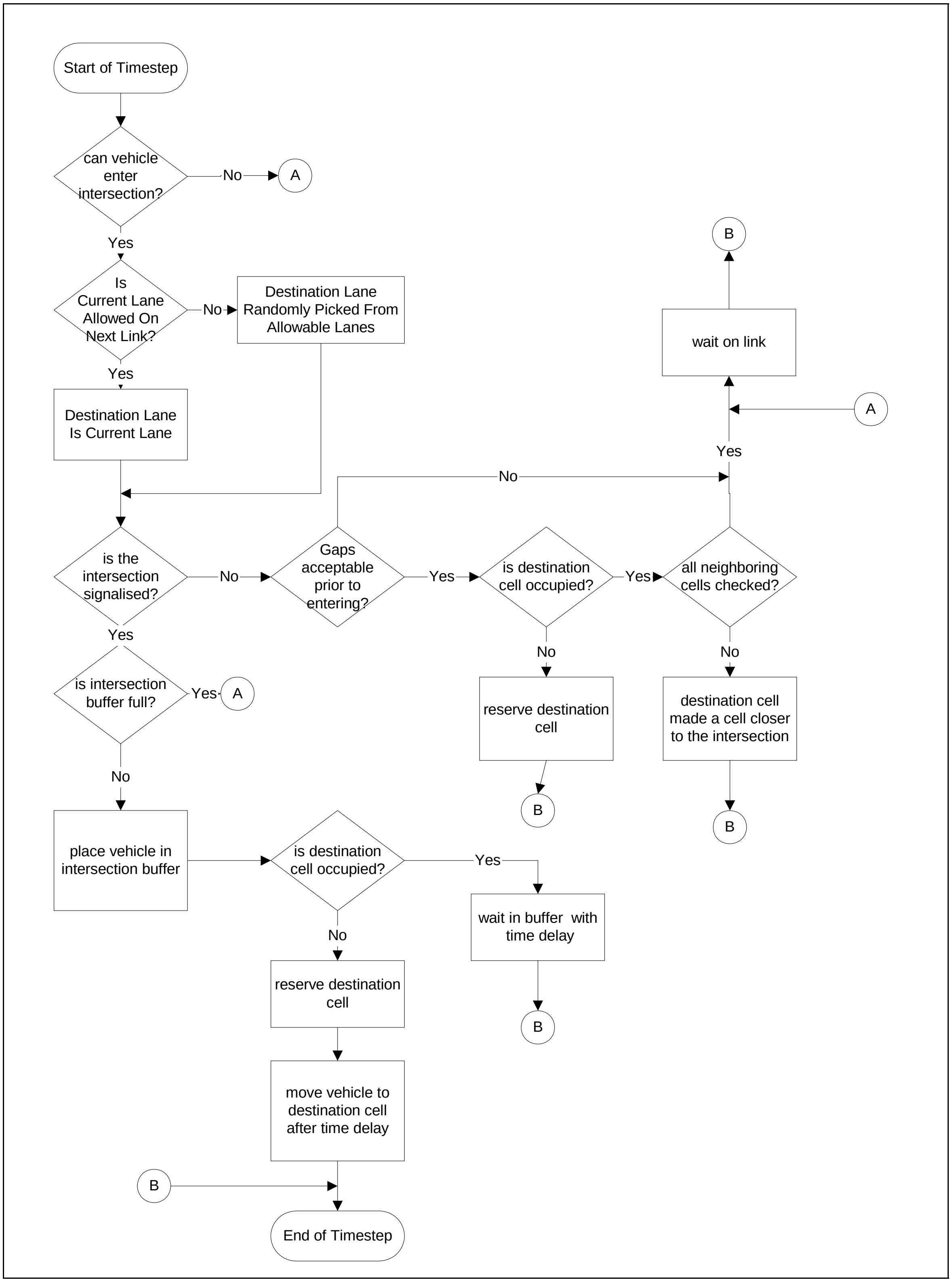


Figure-6.30: Flowchart for Movement of Vehicles at Intersections

6.8.3.6 Exit Travelers and Vehicles from a Parking Place

A parking place accessory has a list of IDs for the vehicles that are present (either because they begin the simulation there or they have arrived there during the course of the simulation). It also has a queue of travelers and their associated plans. This procedure handles each traveler in the traveler queue whose departure time has arrived.

If the traveler is waiting for a vehicle, he or she cannot leave unless the assigned vehicle with appropriate ID is present. If the vehicle is not there, the traveler's departure time is incremented and he or she is placed back in the queue with a delay time of one second. A vehicle whose ID is on the list will have been present in the simulation only if it has arrived here from somewhere else. Otherwise, a new vehicle with this ID must be created using the type implied by the traveler's plan.

The traveler is added to the vehicle as a driver or passenger, depending on the traveler's plan. If the driver has not yet been added to the vehicle, the vehicle passenger is not assigned until the vehicle driver is assigned. So the vehicle passenger is not popped off the queue and continues waiting. Otherwise, the driver checks to see how many passengers are anticipated. (This information is contained in the driver's plan, along with the IDs of the expected passengers). If any passenger is missing, the driver is placed back in the queue so that the vehicle will try to leave again on the next time step. If the driver and all passengers are present, the vehicle attempts to find a place on the grid in the local CPU in any lane, traveling at a speed Vmax governed by the aforementioned three speed limits.

The appropriate grid for the planned direction of travel is determined, and the grid is searched upstream for a distance of V_{Max} cells. If a vehicle is found to occupy that cell in a lane, that lane and its adjacent lanes right and left are eliminated from consideration. All lanes are searched and if a lane is available, the vehicle is placed on the lane at the cell corresponding to the Vmax cell in the direction of travel from the parking place. If there is no room on the grid at Vmax cell, the nearby cells to the parking place are searched to determine if one of them is available according to previous rules. If no place is available the driver is returned to the traveler queue.

6.8.3.7 Enter Vehicles into Parking Places

Vehicles are removed from the roadway at destination parking places by checking all of the cells in all lanes downstream from a parking place for a distance of $V_{GlobalMax}$ cells. If a vehicle is found on the last step of the current leg of its plan and with this parking place as its destination, the vehicle is removed from the roadway. Its ID is placed onto the list of vehicles present at that parking place.

6.8.4 Preparing For a Timestep

Preparing for a time step by the Microsimulator essentially involves updating the signals and preparing the nodes. The timing tables containing the information of every signal are used to update the signals at each time step. The signalized traffic controls are initialized at the beginning of the simulation to the first interval of the signal cycles first phase when the offset is 0. When the offset is not zero, the signal is initialized to the phase and interval that corresponds to the simulation time 0 in the offset cycle.

At every time step each intersection is checked to see if there are any vehicles ready to be ejected from the intersection. Vehicles are ejected from the intersections after they have stayed in the intersection buffer for more than the intersection specified residence time. These vehicles exit from the queued buffers onto the first cell in the destination lane of the destination link. These vehicles reserve their destination cell before the vehicles on links calculate movement, thus giving the vehicles exiting from intersection buffers precedence over vehicles on links. A temporary vehicle marker is placed on the next grid for each vehicle that will leave the intersection in this time step.

6.8.5 Cleaning Up After a Timestep

This procedure essentially deals with the migration of travelers and vehicles from one CPU to another and the clean up of the nodes and edges. As the simulation progresses, a situation may arise when a traveler or a vehicle needs to transfer from a region being handled by one CPU to another one being handled by a different CPU. These migrating vehicles or travelers need to be handed over to another CPU. Encoding the vehicles and travelers into a message and passing them over to the other CPU takes care of the migrating travelers and vehicles and is carried out on a link-by-link basis for all the shared links as discussed below.

6.8.5.1 Migrate Vehicles

Any vehicle that has passed from a region of a link controlled by a CPU into a region controlled by its neighbor must be encoded in a message and sent to that neighbor. The Migrate Vehicles process is done on a link-by-link basis.

6.8.5.2 Migrate Travelers

Some travelers not in vehicles may have been placed in the Migrating Travelers list during the time step. The Migrate Travelers procedure encodes those travelers into messages and passes them onto the desired CPUs, thus clearing out the list as it goes.

6.8.5.3 Clean up Nodes

The Clean up Nodes procedure causes each intersection to eject the first vehicle in each of its buffers into previously reserved locations on the destination link. Vehicles are transferred from the buffers to their reserved destination cells during the cleanup phase, which takes place after movement changes for all the vehicles are executed. Vehicle speed does not change during intersection entry/exit at a signalized intersection. Vehicles are placed in the first cell on the destination link with the same velocity that they entered the intersection buffer.

6.8.5.4 Clean up Edges

The Clean up Edges procedure clears all temporary vehicle markers from the grids on the links. In addition, if the cleanup action state variable for a vehicle is `eject`, it places the vehicle in the intersection buffer if it is buffered; otherwise, it places it directly onto the next edge. If the cleanup action is `migrate`, it deletes the vehicle (which has already been sent to its destination CPU in the migration step).

6.8.6 Supporting Parallel Computation

The Microsimulator runs on multiple CPUs, where available. This section discusses the way the Microsimulator deals with issues of parallel computation and use of multiple CPUs. It also highlights how a transportation network is partitioned to enable parallel computation, the use of multiple CPUs to maximize computational speed, and how the information flows occur across the different CPUs.

6.8.6.1 Transportation Network Partition

To support parallel computation, the transportation network is partitioned among the available CPUs, with each CPU receiving a set of nodes and links. The proper partitioning of the transportation network means the distribution of a part of the transportation network to each CPU in such a way that there is no burdening of one CPU, but uniform distribution of the load with regards to the processor capabilities, as discussed earlier in the key concepts section.

TRANSIMS Microsimulator uses the Orthogonal Bisection (OB) Algorithm or the METIS graph-partitioning library (a public domain package) whichever is available to partition the network among the CPUs. The algorithm that is used for the partitioning of the network is determined at the run time by a combination of the configuration keys (PAR_PARTITION_FILE, PAR_USE_METIS_PARTITION, PAR_USE_OB_PARTITON).

The partitioning process is carried out by an algorithm that assigns cost functions to links and nodes in the transportation network. The cost functions can be based on the number of cells on the links attached to the node. They could also be based on information collected in prior partitioning calculations written into a file. The Microsimulator provides a feature of capturing information about the amount of CPU time devoted to processing each link and node, as the simulation progresses. The configuration keys that control these operations are PAR_RTM_INPUT_FILE and PAR_RTM_PENALTY_FACTOR. The information stored could be retrieved for later use in a subsequent simulation run if so desired, particularly if neither METIS nor OB partitioning algorithm have been requested. This is controlled by configuration keys PAR_SAVE_PARTITION, PAR_PARTITION_FILE.

6.8.6.2 Distributed Links and Information Flow

6.8.6.2.1 Distributed links

Links that are shared between two CPUs are referred to as distributed links. These distributed links are split in the middle and shared between CPUs, each of which is responsible for its half of the link. The Microsimulator also assigns each distributed link an equal number of grid cells to a CPU, to ensure that links with odd cells are divided consistently.

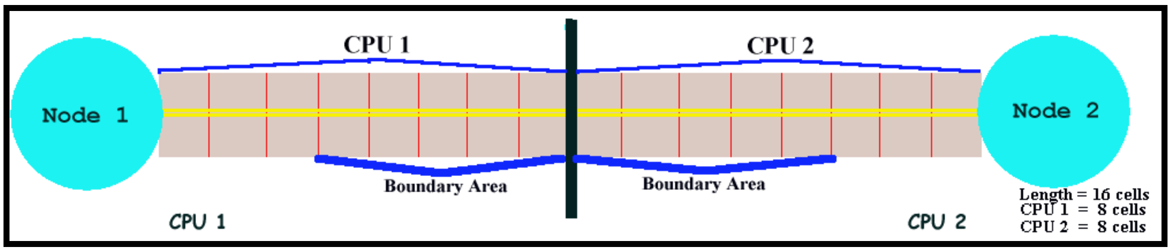


Figure-6.31: A Distributed Split Link

The region close to the middle where the distributed links are split is referred to as the boundary area as shown in Figure-6.31. The boundary area is equal to the value of $V_{GlobalMax}$ cells (5 cells) on each side of the split. Every distributed split link should be longer than a certain number of cells to be split (specified by PAR_MIN_CELLS_TO_SPLIT). The boundary width defines the maximum distance (forward or backward) that can be used for gap calculations.

6.8.6.2.2 Boundary Information Flow

Vehicles are transferred between CPUs as they traverse the split link. Each split link introduces a message-passing delay during the update sequence so that messages are passed among the vehicles crossing these split links. The messages that are exchanged between CPUs can be identified as one of two types Vehicle Migration Messages and Boundary Exchange Messages, as shown in Figure-6.32.

<u>Vehicle Migration Messages</u>

The vehicle migration messages occur for all vehicles that have completed the traversal of the active cells of a CPU. These messages carry information about the vehicle, its occupants, and their plans and are sent to the CPU that handles the part of the transportation network the vehicle is traveling into. After the vehicle moves from one part of the network to the other and the Vehicle Migration message is sent, the vehicle is removed from the originating CPU.

Upon the receipt of the message, the other CPU creates a vehicle with occupants and their corresponding travel plans using the information contained in the message. After which it places them in an appropriate position on its half of the distributed link.

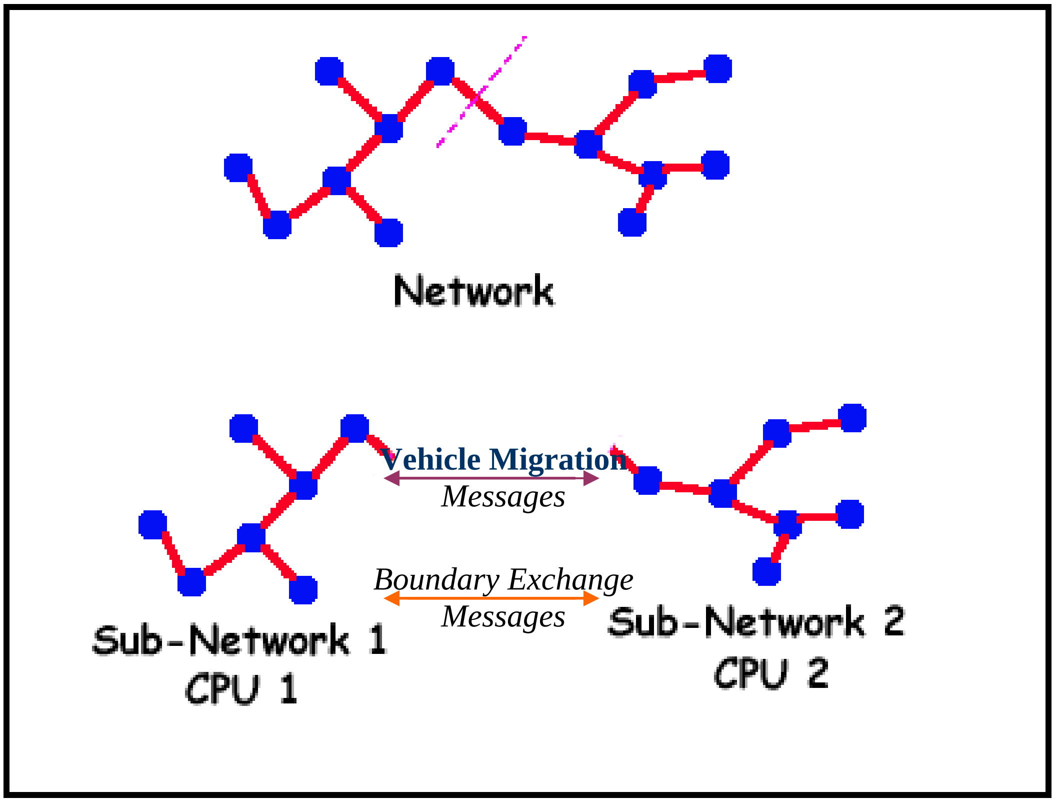


Figure-6.32: A Pictorial Representation of Boundary Information Exchange

<u>Boundary Exchange Messages</u>

Exchange of boundary information between CPUs referred to as Boundary Exchange Messages, are necessary to correctly calculate position changes (movement and lane changes) for vehicles in a CPU's boundary area. Information about vehicles in the next $V_{GlobalMax}$ cells (or preceding $V_{GlobalMax}$ cells, depending on the direction of traffic flow) is necessary to execute the appropriate gap calculations for lane changes and movement.

Each CPU maintains a list of its distributed links and of the CPU owners of the other half of the links. Boundary exchanges must be conducted before lane changes and again before vehicle movement. Each CPU initiates the exchanges at the appropriate time. Each CPU waits until it receives all of the boundary exchange messages from neighboring CPUs.

6.8.6.3 Parallel Computation Sequence and Synchronization Points

The TRANSIMS Microsimulator, which is a distributed object simulation, uses the master/slave(s) paradigm. One CPU acts as a master when there are multiple CPUs being used. The master CPU starts the slave processes, handles the initialization sequence, and also serves as a synchronization point for the slave processes.

The slave processes start to work after the initialization and do all the work in the simulation. After initialization, each slave process completes successive update cycles until the end of the specified simulation run. The slave processes synchronize with the master process at the beginning of each time step or at the beginning of a sequence of time steps, depending on the value of the configuration key CA_SEQUENCE_LENGTH. The master/slave process is depicted pictorially in Figure-6.33.

6.8.6.3.1 Initialization Sequence

The master process starts by reading the network information from the database, constructs a copy of the transportation network, and also constructs or reads a partition. Then, the master creates and initializes the following five-step slave process:

Step 1: Initiates/Starts Slave Processes.

Step 2: Sends each slave ID lists of its local nodes and links and lists of those connected to it by distributed links.

Step 3: Sends each slave a mapping from node IDs to CPU IDs, and optionally (depending on the setting of configuration key CA_BROADCAST_ACC_CPN_MAP) a mapping from accessory IDs to CPU IDs.

Step 4: Tells each slave to construct its transportation sub-network from database information.

Step 5: Tells slaves to read in the initial plans, queue initial vehicles on parking places, and initially place vehicles on the links at the given simulation start time.

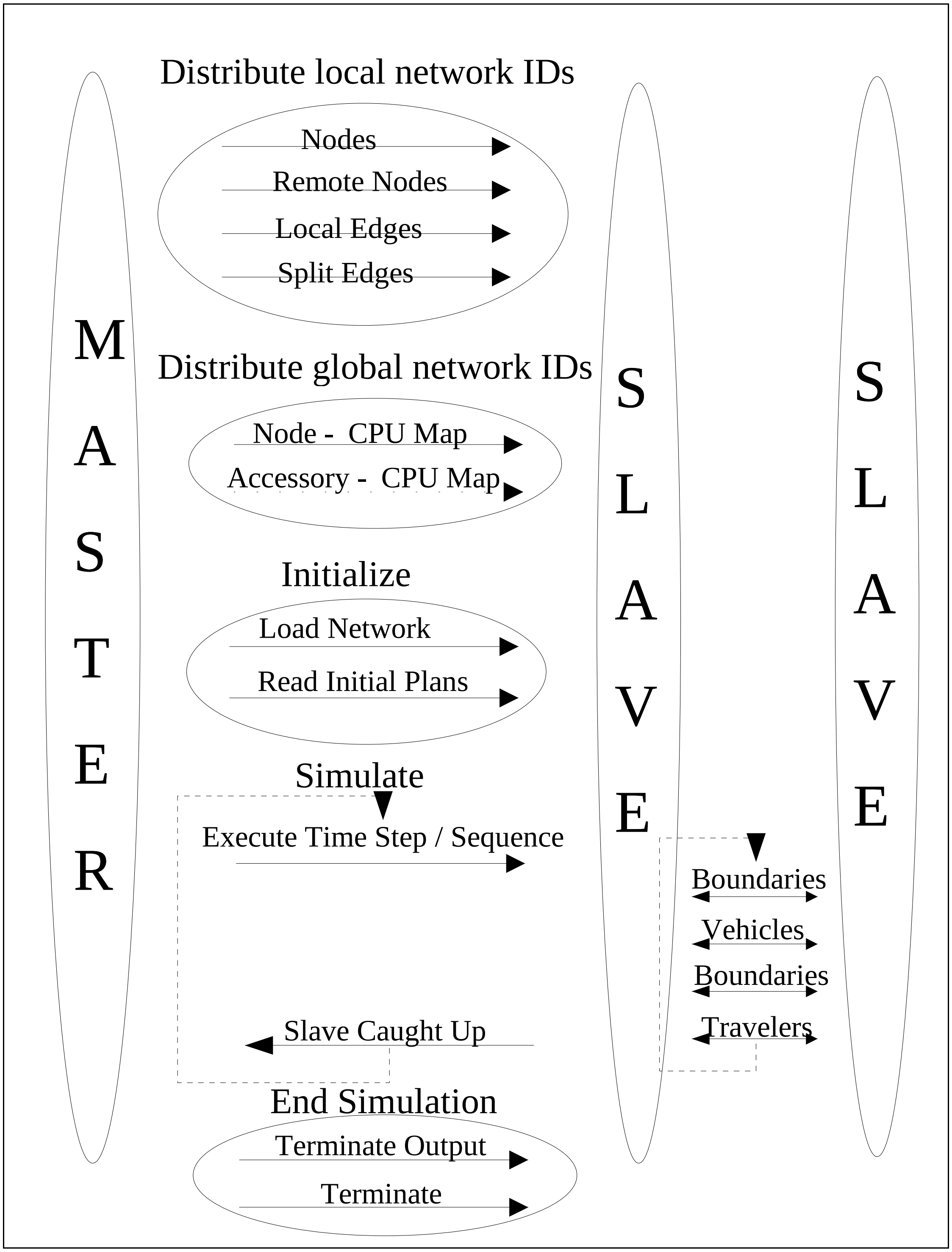


Figure-6.33: Communication and Procedure for Parallel Computation

6.8.6.3.2 Simulation

Once the initialization sequence is complete, the master starts the simulation by instructing the slaves to execute the first time step. The master process then waits until all the slaves complete execution of a fixed number of procedures within a time step. After this occurs it sends a message to the slaves to execute the next time step sequence.

6.8.6.3.3 Termination

Termination occurs by the master sending messages to the slaves to shut down the parallel I/O system and to exit when the requested number of time steps has been executed as specified by the user.

6.8.6.3.4 Overlapping Computation and Communication

For efficiency, The TRANSIMS Microsimulator uses a parallel code that overlaps communication whenever possible, enabling the CPUs to continue execution of useful work while still waiting for the responses from other CPUs.

The Microsimulator notes the links under a single CPU's control and those shared. After sending boundary information, each CPU can update its entire non-shared links before it must make use of the boundary information received from other CPUs. The configuration key CA_LATE_BOUNDARY_RECEPTION, if set, will allow the simulation to make computations that way.

6.8.6.3.5 Output Collection

Slaves generate in parallel all output information from the Traffic Microsimulator. Each slave sends a message to the master indicating what sort of information it would like to write and how many bytes the information will require on disk. The master collates the requests from all the slaves and responds to each, indicating an offset into a file for writing the information. Each slave then writes its information to disk at the indicated location. The message traffic generated by the output system is shown in Figure-6.12 in the Output Visualizer section.

APPENDIX-A

Configuration Parameters

The configuration parameters control how the drivers and vehicles behave in the traffic microsimulation. The variations in behavior among drivers are accomplished by allowing certain behaviors to vary randomly within specified limits. These configuration parameters are listed below in Table-6.29. Other configuration keys that control the installation and the setup files of the Microsimulator are not listed here.

Table-6.29: Configuration Parameters

Configuration Parameter	Description
CA_DECELERATION_PROBABILITY	Variation in the traffic is enhanced by having each driver randomly decide whether to decelerate for no apparent reason at each time step. The probability of decelerating is a value in the range 0.0 to 1.0 [default = 0.2].
CA_LANE_CHANGE_PROBABILITY	Variation in the traffic is reduced by not allowing every driver who would change lanes based on vehicle speed and gaps in the traffic to do so at each time step. This is done to prevent *lane hopping*. The probability that a driver will change lanes when speed and gaps permit is a value in the range of 0.0 to 1.0 [default = 0.99].
CA_PLAN_FOLLOWING_CELLS	Plan Following Cells specifies a count of the number of cells preceding the intersection within which a vehicle will make lane changes to get in an appropriate lane to transition to the next link in its plan. Beyond this distance, lane-changing decisions are based only on vehicle speed and gaps in the traffic. Within this distance, the lane required by the vehicle's plan is also taken into account. As the vehicle nears the intersection, the bias to be in the lane required to stay on plan is increased. Valid values are positive or zero [default = 70 cells].
CA_LOOK_AHEAD_CELLS	The preferred lane for a vehicle to be in as it approaches an intersection depends on the connectivity from the current link to the next link in the plan. In some situations, it is advantageous for the driver to look beyond the next link to subsequent links in the plan when deciding the preferred lane. Look Ahead Cells controls how far ahead the driver will look. A value of 0 indicates that the driver will not look beyond the next link. A positive value indicates that the driver will look at least one additional step beyond the next step in the plan. The number of additional links that will be considered is determined by the lengths of the subsequent links, with link lengths being summed until the accumulated distance is greater than or equal to Look Ahead Cells. Valid values are positive or zero [default = 35 cells].
CA_INTERSECTION_WAIT_TIME	Intersection Wait Time specifies the number of seconds that a vehicle requires to pass through a signalized intersection. A vehicle resides in an intersection queued buffer for this amount of time and is then placed on the next link if the first cell on that link is unoccupied. It will remain in the intersection for a longer time if entry to the next link is blocked by another vehicle. Valid values are positive [default = 1 second].

Configuration Parameter	Description
CA_OFF_PLAN_EXIT_TIME	Off Plan Exit Time specifies the number of seconds a vehicle is allowed to deviate from its plan before being removed from the simulation. This prevents off-plan vehicles from wandering on the transportation network. Valid values are positive [default = 1 second].
CA_IGNORE_GAP_PROBABILITY	Drivers at unsignalized intersections wait for a suitable gap in cross traffic before proceeding through the intersection. The deadlock that would occur when vehicles are waiting for each other at multi-way stop/yield signs is prevented by allowing each driver to ignore the gap constraint with some probability. The probability that the drivers at multi-way stop/yield signs will ignore the constraint is a value in the range of 0.0 to 1.0 [default = 0.66].
CA_GAP_VELOCITY_FACTOR	At unsignalized intersections and during protected movements at signalized intersections, drivers wait for a suitable gap in cross traffic before proceeding through the intersection. The number of empty cells in a suitable gap is based on the speed of the cross traffic and the gap velocity factor. The suitable gap is calculated for each lane of the cross traffic. Gap = Speed of Oncoming Vehicle * Gap Velocity Factor The gap velocity factor must be greater than 0.0. The default value is 3.0. Note that vehicles with a speed of 0 result in a suitable gap size of 0, which improves traffic flow in congested conditions.
CA_MAX_WAITING_SECONDS	Max Waiting Seconds determines the number of seconds that a vehicle will try to enter an intersection. If the vehicle has not moved from the link into or through the intersection in Max Waiting Seconds, the vehicle will abandon its plan and try an alternative movement through the intersection, if one exists. Max Waiting Seconds must be greater than 0 and should be greater than the longest red phase of the traffic controls in the simulation. The default value is 600 seconds.
CA_INTERSECTION_CAPACITY	Intersection Capacity determines the number of vehicles that can be held by each intersection's buffers. This is interpreted as the effective number of cells available in the buffer; that is, a vehicle that is 2 cells long will not fit in an intersection if its capacity is 1.
CA_MAXIMUM_SPEED CA_MAXIMUM_ACCELERATION	Maximum speed (in cells / time step) and acceleration (in cells/timestep/timestep) are applied to vehicles of type auto in the movement phase of traffic dynamics.
CA_BUS_LENGTH CA_BUS_CAPACITY CA_BUS_MAXIMUM_SPEED CA_BUS_MAXIMUM_ACCELERATION	Bus Length specifies the length (in cells) of every transit-type vehicle. Bus Capacity specifies the maximum number of occupants (including the driver) allowed in a transit-type vehicle. Bus Maximum Speed and Acceleration are the same as Maximum Speed and Acceleration, except these are applied to transit-type vehicles.
CA_TRANSIT_INITIAL_WAIT	Transit Initial Wait specifies the number of time steps a transit vehicle must be present at a transit stop before any passengers get on or off.
CA_ENTER_TRANSIT_DELAY CA_EXIT_TRANSIT_DELAY	These specify the mean number of time steps it takes for a single traveler to enter or exit a transit vehicle.
PLAN_FILE	The plan file specifies the name of the file where plans reside or a string to which *.tim.idx* and *.trv.idx* can be appended to find the time-sorted and traveler-id-sorted indexes into a plan file(s). The plans should include all travelers—for example, plans created by the Route Planner, transit driver plans, freight plans, etc. The name should be given as an absolute path name, since the slave executables are not always run from the current working directory.
VEHICLE_FILE	The vehicle file specifies the name where vehicles reside, or a string to which *.veh.idx* can be appended to find the vehicle-id-sorted index into a vehicle file(s). The vehicle file must include all vehicles to be used in the simulation.

Configuration Parameter	Description
CA_USE_PARTITIONED_ROUTE_FILES	It is more efficient for slaves to read only those plans that start in the part of the network for which they are responsible. If the partitioning to be used by the simulation is available (from a prior run of the simulation, for example), the *DistributePlans* utility will create a separate pair of indexes for each slave into one common plan file. If Use Partitioned Route Files is set, the slaves will look for these slave-specific indexes. If they do not exist, the simulation will fall back to using a single global pair of indexes.
CA_NO_TRANSIT	If this flag is set, travelers whose plans originate or end at a transit stop are removed from the simulation. None of their remaining legs are used. (The transit drivers' plans do not fall into this category, thus transit vehicles can still be present in the simulation, but no passengers will use them.)
CA_RANDOM_SEED1 CA_RANDOM_SEED2 CA_RANDOM_SEED3	These three values are combined to initialize the random number generator. Note that the actual sequence of random numbers generated on a slave also depends on the number of slaves and the partitioning in general.
CA_SIM_START_HOUR CA_SIM_START_MINUTE CA_SIM_START_SECOND	These values are combined to give the time of day at which the simulation starts. Plans whose estimated arrival time is before the start time will not be executed.
CA_SIM_STEPS	The simulation executes Sim Steps time steps before exiting.
CA_SEQUENCE_LENGTH	The slaves are implicitly synchronized among themselves by the actions of passing boundaries and migrating vehicles. They are also explicitly synchronized by the master every Sequence Length time steps. It may be more efficient to allow the implicit synchronization to control the simulation.
CA_SLAVE_PRINT_MASK CA_MASTER_PRINT_MASK	These variables control which logging messages to ignore. They are set within the code based on the values of the LOG_ configuration keys and should not be set directly.
CA_SLAVE_MESSAGE_LEVEL CA_MASTER_MESSAGE_LEVEL	Only warning messages whose severity is at least as high as Message Level will be written to the master or slave's log file.
CA_USE_NETWORK_CACHE	If set, use a cached binary representation of the network. This representation would have been created by a prior run of the simulation.
CA_OUTPUT_BUFFER_SIZE CA_OUTPUT_BUFFER_COUNT CA_OUTPUT_WRITE_INTERVAL	The parallel I/O system buffers output from each slave until the master provides a file offset to which it can be written and Output Write Interval time steps have occurred. The Output Buffer Size determines the size of these buffers in bytes. The Output Buffer Count determines the maximum number of buffers that can be opened for each output stream. If the total amount of data buffered exceeds the product of the size and count, the simulation will exit.
CA_USE_ROMIO_FOR_OUTPUT	If Use Romio For Output is set, and the executable was compiled with the USE_ROMIO and USE_MPI flags defined, the parallel output system will use ROMIO files instead of UNIX files
CA_PRESORT_EVENT_FILES	Event output is not necessarily produced in globally time-ordered fashion by the different slaves. If this flag is set, the event file output should be correctly sorted by simulation time.
CA_LATE_BOUNDARY_RECEPTION	If Late Boundary Reception is set, the simulation will try to overlap computation and communication.
CA_RTM_SAMPLING_INTERVAL CA_RTM_OUTPUT_FILE	The partitioning algorithms try to find the partition that spreads the computation associated with nodes and links evenly while simultaneously trying to minimize the communication costs associated with split links. The costs for each node and link can be estimated using run time costs from prior runs. These costs are sampled at the interval defined by RTM Sampling Interval and written out to the file named by RTM Output File. They are currently read in from a file whose name is hard-coded to be RunTimeMeasurements, which is expected to be found in the directory named by OUTPUT_DIRECTORY.

Configuration Parameter	Description
CA_BROADCAST_ACC_CPN_MAP CA_BROADCAST_TRAVELERS	If Broadcast Travelers is set, migrating travelers are broadcast to every CPU. Since only one CPU will eventually make use of the traveler, this is inefficient. If Broadcast Acc CPN Map is set, each CPU knows which CPU is associated with every accessory, so traveler migration messages can be targeted to only the single CPU that needs them. If the CPN Map is not broadcast, travelers must be broadcast.
PAR_SLAVES	This key sets the number of slave processes to spawn. It must be smaller than the number of host CPUs available (to allow one process for the master).
PAR_HOST_COUNT	The number of distinct machines that make up the parallel machine environment.
PAR_HOST_I PAR_HOST_CPUS_I PAR_HOST_SPEED_i	These variables describe the parallel machine environment to the simulation. There should be one set of these three variables, with *i* replaced by an integer from 0 to the value of PAR_HOST_COUNT - 1, for each host. Host should be a string containing the name of the machine. Host CPUs should give the number of CPUs available for use on the machine. Host Speed should give the relative speeds of the different machines in arbitrary units. The sum of all the values of Host CPUs must be at least one larger than the number of slaves requested.
PAR_BROADCAST_CONFIG	Broadcast parallel configuration information to all hosts. This must be set to use MPI.
PAR_MPI_SPIN	Activates spinning while waiting for messages.
PAR_COMMUNICATION	One of “MPI” or “PVM.” This is evaluated only by the script Msim.pl, which can be used to run the simulation.

REFERENCES

1. Los Alamos National Laboratory. *TRANSPORTATION ANALYSIS SIMULATION SYSTEM (TRANSIMS)* (1998).

2. Los Alamos National Laboratory. *TRANSPORTATION ANALYSIS SIMULATION SYSTEM (TRANSIMS) Version: TRANSIMS-LANL-1.0*. NM (1999).

3. Los Alamos National Laboratory. *TRANSPORTATION ANALYSIS SIMULATION SYSTEM (TRANSIMS) Version: TRANSIMS-LANL-1.1*. NM (2000).

4. Simulation Modeling & Analysis second edition, Averill M. Law W. David Kelton, McGraw-Hill, Inc., 1991

5. Intersection Controller Training Course, NHI Course No. 13336, National Highway Institute, U.S. Department of Transportation, Federal Highway Administration, Pub. No. FHWA-HI-88-013, June 1988.

6. Kai Nagel. Fast Low Fidelity Microsimulation of Vehicle Traffic on Supercomputers. *TRANSIMS Report Series*. Los Alamos National Laboratory, TMIP, DOT, LA-UR-94-008, September 1, 1993.