Router and Router Feedback How-To

This document provides basic information about using the TRANSIMS Router to generate travel plans for daily household trips. It also discusses how to review and resolve routing problems and implement a feedback process based on updated network travel times.

Revision History

10/25/2006	Created by AECOM Consult, Inc.
2/6/2007	Edited by AECOM Consult, Inc.
4/15/2010	Revised by RSG, Inc.

Table of Contents

1.0 Assumptions and Prerequisites

This document assumes you have installed TRANSIMS Version 4.0.3 on a Windows or Linux computer system and that you understand the basic procedures and terminology for executing TRANSIMS programs.

The TRANSIMS software and documentation can be downloaded from http://sourceforge.net/projects/transims/files/ → software. Basic TRANSIMS procedures and terminology are addressed in the Installation and Testing How-To available in the documentation set.

You should have a basic understanding of traditional transportation planning networks and path-building techniques, and you should have successfully run previous how-to exercises through the Trip Table Conversion How-To.

Text files are used to store the input and output information. You need to be able to review and edit these files using a standard text editor (e.g., vi, Pico, WordPad) or other software that can manipulate text files (e.g., Excel).

Because this document also describes how to generate ArcView shapefiles for displaying plan data in map format, familiarity with software that can read and display ArcView shapefiles is desirable, but not necessary.

1.1 Download the Case Study Files

Network information from Alexandria, Virginia, is used to demonstrate the procedures for using the Router, discuss outcomes, and describe concepts. The trip file included in the case study files was extracted from the traditional modeling process used by the Metropolitan Washington Council of Governments.

To download the Alexandria data to your computer or local area network, select http://sourceforge.net/projects/transims/files/ → test data → 4.0.3a Test Cases → Alexandria_4.0.3a.zip

You should create a directory with a name such as

c:\TRANSIMS\Alexandria	(Windows)
/home/TRANSIMS/Alexandria	(Linux)

and then unzip the downloaded file to that directory.

2.0 Introduction to TRANSIMS Routing

The Router builds travel paths, called plans in TRANSIMS, for the household activities included in an activity file or alternatively for the trips contained in a trip file. The activity file describing the activities and their locations can be generated using the **ActGen** program described in the Activity Generator How-To. In this case study we are going to route the individual trips generated in the Trip Table Conversion How-To. Given the origin activity location and the destination activity location, the trip start time, and the primary travel mode, the Router constructs a minimum impedance path between the origin and destination based on travel conditions at the specified time of day. The results are stored in the output plan file, which includes a separate set of records for each mode-specific leg of the trip.

For example, the plan for an automobile trip between home and work consists of three legs. The first leg represents the walk from the home activity location to the parking lot where the automobile is parked. The second leg represents the network links on the drive from the origin parking lot to the destination parking lot. The third leg represents the walk from the destination parking lot to the work activity location. The total impedance for all three legs is used to identify the optimal path for the trip.

The impedance for each link is determined by the user-defined combination of weighted walking time, waiting time, in-vehicle-travel time, transfer time, and cost. Different groups of travelers may have different weighting values. For example, cost may be less important to high-income travelers than to low-income travelers.

Link impedance is a function of the time of day the traveler enters the link. Link travel times are summarized for a user-specified time interval, usually 15 minutes. Vehicle use restrictions and turn prohibitions may also change by time of day. All of this can impact the path a given traveler will use.

The time dependent link travel times used for path building may be estimated in two different ways. The link delays generated by the Microsimulator provide the best estimate of travel times for path building. As an interim step or as a supplement to the Microsimulator, TRANSIMS also provides methods for estimating period travel times using traditional volume-to-capacity relationships via volume-delay functions such as the Bureau of Public Roads formula. The Router can update the travel times on each link at regular intervals during the path-building process. The PlanSum program can also use volume-to-capacity relationships to estimate the time-dependent travel times based on the travel plans for all travelers.

Additional time of day considerations affect transit path building. For each transit stop, schedules are provided for each run of each route that utilizes the stop. In other words, the Router knows that a bus on transit route 10 is scheduled to leave stop 100 at 8:32:19 in the morning. To determine the waiting time at each stop and the combination of routes that will minimize the total transit trip impedance, the Router considers the schedule information together with the time that the traveler starts a specific trip and the amount of time it will take the traveler to walk from the activity location to nearby bus stops. Transit paths can also be constrained by maximum walking distance, maximum waiting time, and maximum number of transfers. (See the Router documentation for more details - Router v4.0.pdf)

It is important to understand that TRANSIMS is based on the concept of iterative feedback. This means that the time-dependent link travel times produced by the Microsimulator or a traveler response procedure are used by the Router to repeatedly adjust travel plans so as to generate realistic estimates of traffic volume and transit ridership. A sophisticated model may involve 100 or more feedback loops. The newly generated travel plans are merged with the full regional plan set to create travel demand that is simulated by the Microsimulator**.** The results are then used as input for subsequent feedback decisions.

3.0 How to Route Trips on a TRANSIMS Network

This section describes how to use the Router to generate travel plans for the trips you generated following the steps outlined in the Trip Table Conversion How-To. Since the trip file contains automobile and transit trips, the resulting travel plans will include a variety of walk, drive, and transit legs.

3.1 Input Data Files

The Router needs the following information to generate multimodal paths:

- Highway network
- Transit network
- Travel demand by time of day
- Vehicle availability and location
- Time-dependent link travel times

The set of network files produced by the Highway Network Conversion How-To is sufficient to run the Router. At a minimum, the Router requires the node, link, lane connectivity, activity location, process link, and parking files. If transit routing is desired, the transit stop, transit route, and transit schedule files are also needed. A link delay file containing time-dependent link travel times is optional. If it is not provided, free-flow speeds will be used to initialize the travel times on each link. In subsequent iterations a link delay file representing the time-dependent link travel times from the initial routing would be used so that travel decisions are based on congested speeds.

The trips that are to be routed are defined in either a trip file or an activity file. A trip file defines the origin, destination, start time, and travel mode for each trip. In contrast, an activity file defines the start time, end time, and locations of the activities a traveler is engaged in over the course of the day. In this case, the Router prepares travel plans for movements from one activity location to the next. Both the trip file and the activity file also require a vehicle file, which identifies where each vehicle is parked at the beginning of the day.

3.2 Router Control File

In addition to specifying the input and output data files, the Router control file defines a variety of parameters that control the path-building process. A sample Router control file ("1.Alex.2005.Trip.Router.ctl") is provided in the control directory. The file is a text file that can be reviewed and edited using a standard text editor. The file records are listed below.

```
TITLE                              Route the Highway and Transit Trips for 1.Alex
DEFAULT_FILE_FORMAT                TAB_DELIMITED
PROJECT_DIRECTORY                   ../

NET_DIRECTORY                       ../network
NET_NODE_TABLE                     Node
NET_LINK_TABLE                     Link
NET_POCKET_LANE_TABLE              Pocket_Lane
NET_PARKING_TABLE                  Parking_2
NET_LANE_CONNECTIVITY_TABLE        Lane_Connectivity
NET_ACTIVITY_LOCATION_TABLE        Activity_Location_3
NET_PROCESS_LINK_TABLE             Process_Link_2
NET_TRANSIT_STOP_TABLE             Transit_Stop
NET_TRANSIT_ROUTE_TABLE            Transit_Route
NET_TRANSIT_SCHEDULE_TABLE         Transit_Schedule


HOUSEHOLD_LIST                      NULL
LINK_DELAY_FILE                    NULL

TRIP_FILE                          demand/Alex.2005.Trip.Trips
TIME_OF_DAY_FORMAT                 24_HOUR_CLOCK
VEHICLE_FILE                       demand/Alex.2005.Trip.Vehicles

NEW_PLAN_FILE                      demand/1.Alex.2005.Trip.TravelPlans
NEW_PROBLEM_FILE                   results/1.Alex.2005.Trip.Problems

NODE_LIST_PATHS                    YES
LIMIT_PARKING_ACCESS               YES
IGNORE_TIME_CONSTRAINTS            YES
WALK_PATH_DETAILS                  YES
WALK_SPEED                         1.0       //---- meters / second ----
WALK_TIME_VALUE                    20.0      //---- imped / second ----
VEHICLE_TIME_VALUE                 10.0      //---- imped / second ----
FIRST_WAIT_VALUE                   20.0      //---- imped / second ----
TRANSFER_WAIT_VALUE                20.0      //---- imped / second ----
DISTANCE_VALUE                     1.0       //---- imped / meter ----
COST_VALUE                         5.0       //---- imped / cent ----
TRANSFER_PENALTY                    1200     //---- impedance ----
MAX_WALK_DISTANCE                  2000      //---- meters ----
MIN_WAIT_TIME                      60        //---- seconds ----
LEFT_TURN_PENALTY                  300       //---- impedance ----
UTURN_PENALTY                      5000      //---- impedance ----
```

```
KISS_RIDE_STOP_TYPES              STOP, STATION, EXTERNAL
MAX_KISS_RIDE_DROPOFF_WALK        100          //---- meters ----
PARKING_HOURS_BY_PURPOSE          8.5, 2.5, 1.0, 1.0         //---- hours ----
```

This example uses a basic highway and transit network to construct travel plans for all trips included in the trip file. The result is the same as an all-or-nothing assignment using free-flow speeds. The path for each traveler is stored in the plan file, and any travelers that could not be routed will be listed in a problem file. In the troubleshooting section of this document the modeler can review cases that generate the most frequent routing error messages.

Each path will minimize the total impedance as defined by the input parameters. In our example, the time spent walking is assigned 20.0 impedance units per second, the waiting time at the first transit boarding is assigned 20.0 impedance units per second, and the waiting time at subsequent transit boarding locations is assigned 20.0 impedance units per second. Time spent in automobiles or transit vehicles is valued at 10.0 impedance units per second, and transit fares or tolls are valued at 5.0 impedance units per cent. A 1200 impedance unit penalty is also added for each transit transfer. In addition, each trip must include no more than 2000 meters of walking, no more than 60 minutes of waiting for any given bus, and no more than three transfers.

The Router emulates the different perceptions travelers have about the disutility of travel by introducing a stochastic element into the travel impedance. A non-zero value for the `PERCENT_RANDOM_IMPEDANCE` key causes the program to randomly adjust the link impedance each time it is considered by the path-building procedure. A value of 20 means that the link impedance perceived by the traveler can be as much as 20 percent less or 20 percent more than the unadjusted value.

3.3 Router Output Files

The Router program can be executed the batch file included in the batch directory:

TripModel.bat (Windows)

This batch will run the full model case. However, a new batch file that only runs through the first Router execution can be created by deleting the steps which follow this first Router call that runs the control file 1.Alex.2005.Trip.Router.ctl. The program will create the printout file "1.Alex.2005.Trip.Router.prn," as well as the plan and problem files. The routing process can take several minutes depending on the speed of your machine. For more information about running times refer to the Frequently Asked Questions section of this document.

The program reads the mode, start time, origin, destination, and traveler ID information for each trip contained in the trip file and writes a series of plan leg records in the plan file. A separate plan leg is generated for each change of mode included in the path. For example, the trip file includes the following trip record for household #53, person #1:

HHOLD	PERSON	TRIP	PURPOSE	MODE	VEHICLE	START	ORIGIN	ARRIVE	DESTINATION
52	1	1	1	2	53	6:36:18	4083	6:51:47	1859

The Router generates the following three plan leg records for this trip in the file 1.Alex.2005.Trip.TravelPlans:

```
5301 0 1 1
23778 4083 1 4083 2
30 23808 1 0 600
0 2
0

5301 0 1 2
23808 4083 2 1859 2
68 23876 1 0 1894
1 0
7
53 0
2158 2157 2565 1581 1572

5301 0 1 3
23876 1859 2 1859 1
30 23906 1 0 600
0 2
0
```

The first leg starts at 23778 seconds (6:36:18) at activity location 4083 and consists of walking for 30 seconds to parking lot 4083. The second leg consists of the drive from parking lot 4083 to parking lot 1859 starting at 23808 (6:36:48). The path travels through network nodes 2158, 2157, 2565, 1581, and 1572. The third leg defines the walk from parking lot 1859 to activity location 1859. The whole trip took 128 seconds (23906-23778), with 3094 impedance units (600 + 1894 + 600).

For more information about the data included in a plan file, refer to the Router documentation.

4.0 How to Create ArcView Shapefiles to View Travel Plans

Modelers often like to review the paths created by the Router to verify their validity or to check network travel times. The best way to accomplish this is to visualize selected travel plans on a network map. The ArcPlan utility enables you to convert the TRANSIMS plan file to an ArcView shapefile that can be displayed in ArcGIS or other mapping software. This section describes how to set up and run the ArcPlan program.

4.1 Input Data Files

The **ArcPlan** utility requires the network node, link, activity location, and parking files, in addition to the plan file. A network shape file can also be provided to properly draw the paths on curved links. If the plan set includes transit trips, then transit stop, transit route, and transit driver files are needed to visualize the transit vehicle path between the boarding and alighting stops.

4.2 Control File

A sample control file for the ArcPlan program is provided in the control directory. This file, “1.Alex.2005.Trip.ArcPlan.ctl,” is a text file that can be reviewed and edited using a standard text editor. The file records are listed below.

```
TITLE                               Display the 1.Alex Travel Plans
PROJECT_DIRECTORY                    ../
DEFAULT_FILE_FORMAT                 TAB_DELIMITED

NET_DIRECTORY                        ../network
NET_NODE_TABLE                      Node
NET_LINK_TABLE                      Link
NET_SHAPE_TABLE                     Shape
NET_PARKING_TABLE                   Parking_2
NET_ACTIVITY_LOCATION_TABLE         Activity_Location_3
NET_TRANSIT_STOP_TABLE              Transit_Stop
NET_TRANSIT_ROUTE_TABLE             Transit_Route
NET_TRANSIT_SCHEDULE_TABLE          Transit_Schedule
NET_TRANSIT_DRIVER_TABLE            Transit_Driver

PLAN_FILE                           demand/1.Alex.2005.Trip.TravelPlans
NODE_LIST_PATHS                     YES

SELECT_TIME_PERIODS                 7:00..9:00, 15:30..18:30
SELECT_RANDOM_PERCENTAGE            10        //---- percent ----
RANDOM_NUMBER_SEED                  11010

ARCVIEW_PLAN_FILE                    results/arcview/1.Alex.2005.Trip.TravelPlans.shp

TIME_OF_DAY_FORMAT                          24_HOUR_CLOCK
LINK_DIRECTION_OFFSET                       2.0        //---- meters ----
ACTIVITY_LOCATION_SIDE_OFFSET               15.0       //---- meters ----
PARKING_SIDE_OFFSET                         5.0        //---- meters ----
TRANSIT_STOP_SIDE_OFFSET                    8.0        //---- meters ----
TRANSIT_DIRECTION_OFFSET                    4.0        //---- meters ----

BANDWIDTH_SCALING_FACTOR                    30.0       //---- vehicles / meter ----
MAXIMUM_SHAPE_ANGLE                         45         //---- degrees ----
MINIMUM_SHAPE_LENGTH                        1          //---- meters ----


OUTPUT_COORDINATE_SYSTEM                    UTM, 18N, METERS
```

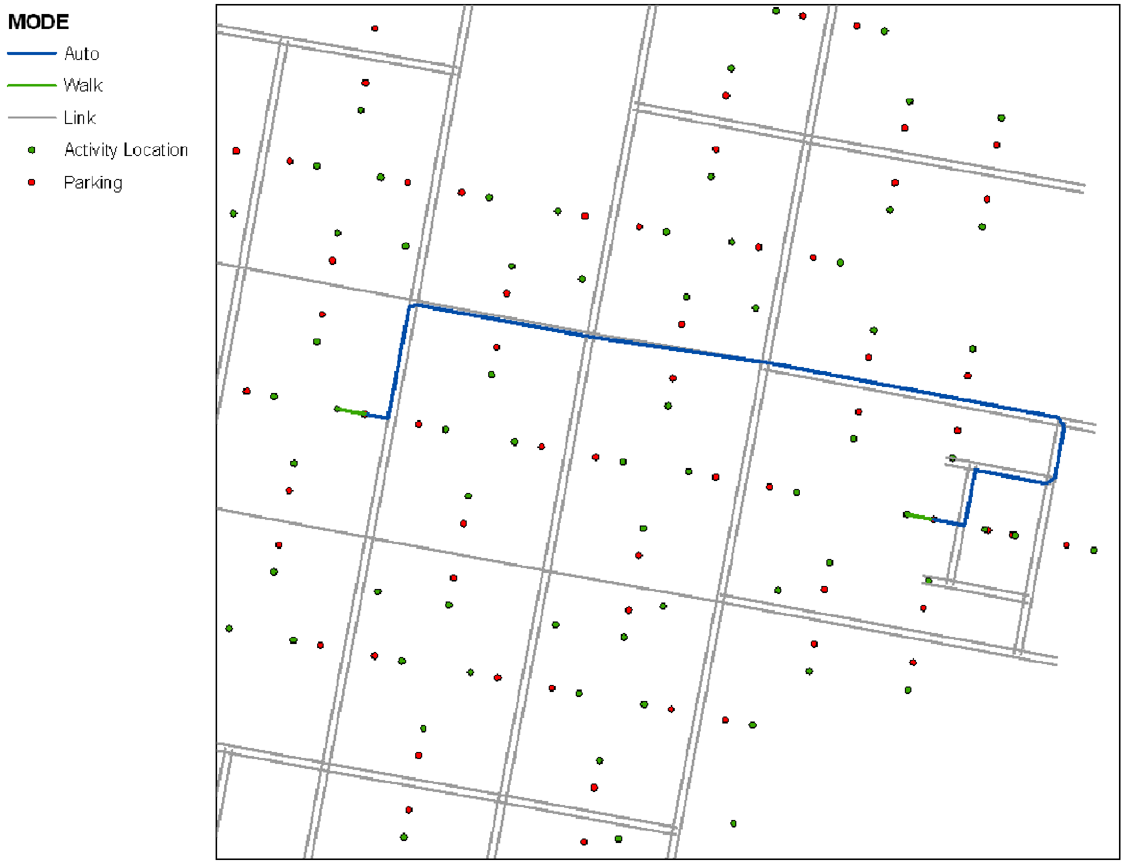
Shape records for the travel legs will be developed for travelers traveling on the network during the AM and PM peak periods (defined by the `SELECT_TIME_PERIODS` key). The offset parameters control how various components of the paths are drawn relative to the centerline of the roadway. In this case, parallel shapes for each direction of travel will be used to draw highway and transit vehicle paths. Movements to transit stops, parking lots, and activity locations are offset at different distances to help simplify the graphics. If the same offsets are used in the **ArcNet** program, the paths easily can be overlaid on the network features, allowing you to simultaneously review the network data and the path information.

4.3 Visualizing the Results

The **ArcPlan** program can be executed using the batch files included in the batch directory.

TripModel.bat (Windows)

This batch will run the full model case if unchanged. However, a new batch file that only runs through the first ArcPlan execution can be created by deleting the steps which follow this first **ArcPlan** call that runs the control file 1.Alex.2005.Trip.ArcPlan.ctl. The printout file "1.Alex.2005.Trip.ArcPlan.prn" will be created by the process, along with a new ArcView shapefile in the arcview subdirectory of the network directory. The "1.Alex.2005.Trip.TravelPlans.shp" and *shx files include one polyline record for each leg selected from the plan file. The *.dbf file contains information about the mode, start time, travel time, cost, and impedance for the trip leg. If a huge number of plans are included in the shapefile, it will be necessary to selectively query individual travelers to clearly visualize individual paths. To help distinguish walk legs from drive and transit legs, it is helpful to color code the legs by travel mode. The graphic below shows the path constructed for the plan included in Section 3.3.



5.0 How to Improve Results through Router Feedback

As described in Section 2, TRANSIMS uses an iterative feedback process to update link travel times and adjust travel plans. Traffic-related travel times are input to the Router through a link delay file (named Link_Delay in this example), which is one of the primary outputs of the Microsimulator. The link delay file includes the total volume and travel time on each link in 15-minute increments. Alternatively, the **PlanSum** program can approximate link delays using volume-to-capacity relationships in the form of volume-delay functions such as the Bureau of Public Roads formula. This exercise will demonstrate how to re-route a subset of trips using travel times generated by **PlanSum**. The Microsimulator How-To will demonstrate how to use the simulated travel times to re-route trips.

5.1 Link Travel Times by Time of Day

The Microsimulator simulates the travel plans generated by the Router and generates link travel times by time of day. These travel times are then used to re-route a subset of trips that are then re-simulated. After several iterations, the paths and travel times stabilize. If the initial set of travel plans is based on all-or-nothing paths using free-flow speeds, the number of iterations required can be significant. All-or-nothing paths are likely to generate more demand on critical links than the Microsimulator can handle. This results in cascading queues and gridlock that distort link travel times in large areas over many time periods.

It is therefore more efficient to refine the travel plans based on estimated travel time calculated by **PlanSum** before attempting the first simulation. This approach uses traditional volume-to-capacity relationships to approximate network congestion and adjust link demands. In many cases, these types of adjustments can result in a set of travel plans that the Microsimulator can handle in fewer iterations and with less processing time than methods that depend on purely the Microsimulator alone. Iterating the Router using PlanSum generated link delays in this fashion is intended to provide a warm-start for the microsimulation which will follow.

In these applications, the **PlanSum** program is used to estimate the demand for each link in 15-minute increments. This demand is divided by the 15-minute link capacity and read into a volume-delay equation that estimates the loaded travel time. The process is similar to a traditional capacity-restrained assignment. The results are stored in a link delay file that can be read by the Router to refine the travel plans.

A sample control file for the **PlanSum** program is provided in the control directory. This file, "1.Alex.2005.Trip.PlanSum.ctl," is a text file that can be reviewed and edited using a standard text editor. The file records are listed below.

```
TITLE                               Summarize Travel Plans
DEFAULT_FILE_FORMAT                 TAB_DELIMITED
PROJECT_DIRECTORY                   ../

NET_DIRECTORY                       ../network
NET_NODE_TABLE                      Node
NET_LINK_TABLE                      Link
NET_PARKING_TABLE                   Parking_2
NET_ACTIVITY_LOCATION_TABLE         Activity_Location_3
NET_LANE_CONNECTIVITY_TABLE         Lane_Connectivity
NET_TRANSIT_STOP_TABLE              Transit_Stop
NET_TRANSIT_ROUTE_TABLE             Transit_Route
NET_TRANSIT_SCHEDULE_TABLE          Transit_Schedule
NET_TRANSIT_DRIVER_TABLE            Transit_Driver

PLAN_FILE                           demand/1.Alex.2000.Act.TravelPlans
NODE_LIST_PATHS                     YES

SUMMARY_TIME_PERIODS                0:00..24:00
SUMMARY_TIME_INCREMENT              15       //---- minutes ----
NEW_LINK_DELAY_FILE                 results/1.Alex.2000.Act.Performance
NEW_LINK_DELAY_FORMAT               TAB_DELIMITED

EQUATION_PARAMETERS_1               BPR, 0.15, 4.0, 0.75
EQUATION_PARAMETERS_2               BPR, 0.10, 4.5, 0.75

NEW_RIDERSHIP_FILE                  results/1.Alex.2000.Act.Riders
NEW_RIDERSHIP_FORMAT                TAB_DELIMITED
```

NEW_LINK_VOLUME_FILE	results/1.Alex.2000.Act.Volumes
NEW_LINK_VOLUME_FORMAT	TAB_DELIMITED
NEW_ZONE_SKIM_FILE	results/1.Alex.2000.Act.Skims
NEW_ZONE_SKIM_FORMAT	TAB_DELIMITED
SKIM_TOTAL_TIME	YES
NEW_TRIP_TABLE_FILE	results/1.Alex.2000.Act.Trip_Tables
NEW_TRIP_TABLE_FORMAT	TAB_DELIMITED
ZONE_EQUIVALENCE_FILE	inputs/Zone_Equiv.txt
NEW_TRIP_TIME_FILE	results/1.Alex.2000.Act.Trip_Times
NEW_TRIP_TIME_FORMAT	TAB_DELIMITED
PLANSUM_REPORT_1	TRANSIT_RIDERSHIP_SUMMARY
PLANSUM_REPORT_2	TRAVEL_SUMMARY_REPORT
PLANSUM_REPORT_3	PRINT_ZONE_EQUIVALENCIES

This application reads the plan file and creates a link delay file with travel time estimates from two volume-delay equations defined by the `EQUATION_PARAMETERS_x` control keys as shown above. The index in the `EQUATION_PARAMETERS` control key defines the functional class of the roads for which the parameters are applied. The first volume-delay equation applies a Bureau of Public Roads (BPR) function to freeway and expressway links (functional classes 1 and 2), which results in the following equation:

$$time_t = time_0 \times \left(1 + 0.15 \times \left(volume_t / (capacity \times 0.75)\right)^{4.0}\right)$$

The second equation is applied to all arterial links (functional classes 3 through 10). (See the **PlanSum** quick reference documentation for additional functional forms and options.)

The **PlanSum** program can be executed by the batch file included in the batch directory:

TripModel.bat (Windows)

This batch will run the full model case if unchanged. However, a new batch file that only runs through the first PlanSum execution can be created by deleting the steps which follow this first PlanSum call that runs the control file 1.Alex.2005.Trip.PlanSum.ctl. The printout file "1.Alex.2005.Trip.PlanSum.prn" and a new link delay file will be created in the results directory. The link delay file will be read into the Router in Section 5.3.

5.2 Selecting Travelers to Re-route

One way to iterate the travel time estimates and the travel paths is to re-route all travelers based on updated travel speeds. This often results in oscillation effects, where the trips all take one route in one iteration and then all take another route in the next iteration. Since TRANSIMS simulates individual trips on a specific path, you cannot average the link volumes or simulate fractional trips, as can be done with traditional models. The desired

TRANSIMS approach is to re-route a subset (fraction) of the travelers and combine their travel plans with all the rest of the travel plans in a new simulation or travel time estimate.

A modeler might therefore want to re-route a subset of travelers, focusing as much attention as possible on congested locations and time periods. This will help to refine the network travel times and divert travelers away from congested locations using the fewest number of feedback iterations. The **PlanSelect** program selects households for re-routing using four basic criteria:

1) Volume-to-Capacity Ratio
2) Time of Day
3) Select Link or Node
4) Travel Time Difference

For each criterion, you define the random selection process, the selection criteria, and the maximum number of households to be selected. (Note: routing is done at the household level to properly account for shared travel arrangements and vehicle utilization and location by time of day.)

Volume-to-Capacity (v/c) Ratio: A household is eligible for re-routing if one or more of the household members travel through a link with a 15-minute volume-to-capacity ratio greater than the specified criteria (e.g., `SELECT_VC_RATIOS  1.2`).

Time-of-Day: A household is eligible for re-routing if one or more of the household members is actively traveling on the network during the specified time periods (e.g., `SELECT_TIME_PERIODS  7:00..9:00, 16:00..18:00`).

Select Link or Node: A household is eligible for re-routing if the path travels through one or more defined combinations of nodes. The path must pass through all of the nodes in the order they are defined. For example, the criteria "`SELECT_NODES_1 100, 300, 500`" will result in the selection of any traveler whose path includes nodes 100, 300, and 500, in that order. The path can include any number of other nodes between these values.

Travel Time Difference: The total travel time recorded in the plan file is compared with the cumulative travel time represented by the current link travel times to determine if a household is eligible for re-routing. In essence, the program "re-skims" the travel path for each traveler using the latest link travel times. If the absolute difference is greater than the user-specified criteria (e.g., `PERCENT_TIME_DIFFERENCE  10`), the household is eligible for re-routing.

A sample control file for the **PlanSelect** program is provided in the control directory. This file, "2.Alex.2005.Trip.PlanSelect.ctl," is a text file that can be reviewed and edited using a standard text editor. The file records are listed below.

```
TITLE                               Select Travelers to Re-Route in 2.Alex
DEFAULT_FILE_FORMAT                 TAB_DELIMITED
PROJECT_DIRECTORY                   ../

VEHICLE_FILE                        demand/Alex.2005.Trip.Vehicles

PLAN_FILE                           demand/1.Alex.2005.Trip.TravelPlans
LINK_DELAY_FILE                     results/1.Alex.2005.Trip.Performance

NEW_HOUSEHOLD_LIST                  demand/1.Alex.2005.Trip.HH_List

NET_DIRECTORY                       ../network
NET_NODE_TABLE                      Node
NET_LINK_TABLE                      Link
NET_PARKING_TABLE                   Parking_2
NET_LANE_CONNECTIVITY_TABLE         Lane_Connectivity

SELECT_TIME_RATIOS                  2.0     //---- travel time ratio ----

MINIMUM_TIME_DIFFERENCE             1       //---- minutes ----
MAXIMUM_TIME_DIFFERENCE             60      //---- minutes ----
SELECTION_PERCENTAGE                50      //---- percent ----
MAXIMUM_PERCENT_SELECTED            10      //---- percent ----
RANDOM_NUMBER_SEED                  255
```

This application selects travelers who travel through links with travel time ratios greater than 2.0. Up to 50 percent of the selected households, but no more than 10 percent of all households, will be randomly selected for re-routing. Their household IDs will be saved in the household list file.

The **PlanSelect** program can be executed using the batch file included in the batch directory:

TripModel.bat (Windows)

This batch will run the full model case if unchanged. However, a new batch file that only runs through the first **PlanSelect** execution can be created by deleting the steps which follow this first **PlanSelect** call that runs the control file 2.Alex.2005.Trip.PlanSelect.ctl. The printout file "2.Alex.2005.Trip.PlanSelect.prn" will be created, along with a new household list file, in the household directory. The household list file will be read into the Router in Section 5.3.

5.3 Re-routing Selected Travelers

In order to re-route selected travelers using updated link travel times, two additional keys are included in the Router control file. The `HOUSEHOLD_LIST` key is used to input the list of household IDs selected by the **PlanSelect** procedure (Section 5.2). The `LINK_DELAY_FILE` key is used to input the link volumes and travel times in 15-minute increments generated by the Microsimulator or the **PlanSum** process (Section 5.1). The output plan and problem file names are also modified to store the results in different files from the previous Router run iteration.

```
TITLE                              Route the Highway and Transit Trips for 2.Alex
DEFAULT_FILE_FORMAT                TAB_DELIMITED
PROJECT_DIRECTORY                  ../

NET_DIRECTORY                      ../network
NET_NODE_TABLE                     Node
NET_LINK_TABLE                     Link
NET_POCKET_LANE_TABLE              Pocket_Lane
NET_PARKING_TABLE                  Parking_2
NET_LANE_CONNECTIVITY_TABLE        Lane_Connectivity
NET_ACTIVITY_LOCATION_TABLE        Activity_Location_3
NET_PROCESS_LINK_TABLE             Process_Link_2
NET_TRANSIT_STOP_TABLE             Transit_Stop
NET_TRANSIT_ROUTE_TABLE            Transit_Route
NET_TRANSIT_SCHEDULE_TABLE         Transit_Schedule

HOUSEHOLD_LIST                     demand/1.Alex.2005.Trip.HH_List
LINK_DELAY_FILE                    results/1.Alex.2005.Trip.LinkDelay

TRIP_FILE                          demand/Alex.2005.Trip.Trips
TIME_OF_DAY_FORMAT                 24_HOUR_CLOCK
VEHICLE_FILE                       demand/Alex.2005.Trip.Vehicles

NEW_PLAN_FILE                      demand/2.Alex.2005.Trip.Plans
NEW_PROBLEM_FILE                   results/2.Alex.2005.Trip.Problems

NODE_LIST_PATHS                    YES
LIMIT_PARKING_ACCESS               YES
IGNORE_TIME_CONSTRAINTS            YES
WALK_PATH_DETAILS                  YES
WALK_SPEED                         1.0       //---- meters / second ----
WALK_TIME_VALUE                    20.0      //---- imped / second ----
VEHICLE_TIME_VALUE                 10.0      //---- imped / second ----
FIRST_WAIT_VALUE                   20.0      //---- imped / second ----
TRANSFER_WAIT_VALUE                20.0      //---- imped / second ----
DISTANCE_VALUE                     1.0       //---- imped / meter ----
COST_VALUE                         5.0       //---- imped / cent ----
TRANSFER_PENALTY                   1200      //---- impedance ----
MAX_WALK_DISTANCE                  2000      //---- meters ----
MIN_WAIT_TIME                      60        //---- seconds ----
LEFT_TURN_PENALTY                  300       //---- impedance ----
UTURN_PENALTY                      5000      //---- impedance ----
KISS_RIDE_STOP_TYPES               STOP, STATION, EXTERNAL
MAX_KISS_RIDE_DROPOFF_WALK         100       //---- meters ----
PARKING_HOURS_BY_PURPOSE           8.5, 2.5, 1.0, 1.0    //---- hours ----
```

The **Router** program is executed using the following batch files found in the batch directory:

TripModel.bat (Windows)

This batch will run the full model case if unchanged. However, a new batch file that only runs through the second **Router** execution can be created by deleting the steps which follow this second **Router** call that runs the control file 2.Alex.2005.Trip.Router.ctl. The printout file "2.Alex.2005.Trip.Router.prn" will be generated by the process, along with new plan and problem files. Note that the plan file includes travel legs only for the selected households; it will therefore need to be merged with the full regional plan file.

5.4 Merging Plan Files

The **PlanPrep** program merges the re-routed plans with the full plan set. The plans in the full plan set are replaced with the new plans for each traveler. Since plan files tend to be large, the merge is performed by reading both files sequentially and matching traveler IDs. This requires that both files be sorted by increasing traveler ID. Although the **PlanPrep** program can sort plan files, this will not be necessary as long as the input trip file is sorted by household and person. The output plan file contains the merged plans sorted by traveler ID.

A sample control file for the **PlanPrep** program is provided in the control directory. This file, "Alex.2005.Trip.PlanMerge.ctl," is a text file that can be reviewed and edited using a standard text editor. This file contains the following records:

```
TITLE                      Merge the Re-Routed Travel Plans for 2.Alex
DEFAULT_FILE_FORMAT        TAB_DELIMITED
PROJECT_DIRECTORY            ../

INPUT_PLAN_FILE            demand/2.Alex.2005.Trip.Plans
MERGE_PLAN_FILE            demand/1.Alex.2005.Trip.TravelPlans
OUTPUT_PLAN_FILE           demand/2.Alex.2005.Trip.TravelPlans
```

The **PlanPrep** program can be executed using a batch file included in the control directory:

TripModel.bat (Windows)

This batch will run the full model case if unchanged. However, a new batch file that only runs through the first **PlanPrep** execution can be created by deleting the steps which follow this first **PlanPrep** call that runs the control file 2.Alex.2005.Trip.PlanMerge.ctl. The printout file "2.Alex.2005.Trip.PlanMerge.prn" will be created by the process, along with the new plan file "2.Alex.2005.Trip.TravelPlans." This plan file can then be fed into the Microsimulator or processed by the **PlanSum** program to generate a number of summary reports or to create a new link delay file to feed updated speeds back to the **Router**. The **PlanSelect** program can be used to select additional travelers for re-routing. This iterative process should continue until the results stabilize.

5.5 Batch Feedback Loops

Ultimately, you will want to automate the feedback process to ensure that a consistent, repeatable modeling procedure is developed. To demonstrate this concept, a small batch file that executes 5 feedback iterations of the Router has been prepared. This process executes all of the steps outlined above and then continues selecting travelers for feedback using volume-to-capacity ratios and travel time difference selection techniques.

The batch processing script is included in the batch directory:

TripModel.bat (Windows)

This process will execute all of the programs in order and generate a number of printout and data files.

6.0 Troubleshooting

The **Router** problem file and the problem summary included at the end of the printout file provide important information for troubleshooting the routing process. If more than five percent of trips have problems, there are significant problems with the network or trip file that should be addressed.

Street directionality and lane connectivity are among the more likely sources of error. Remember, the connectivity between links at an intersection is defined in the lane connectivity file. As a result, two links with a common node are not connected unless there is a record in the lane connectivity file that connects the links.

The Router will probably not be able to construct travel paths for all travelers included in the trip file. A summary of problems will be listed at the end of the Router printout report. The listing will include the trip data and problem code for each trip that could not be routed. You should review the problems and problem types to determine if network coding or data processing errors are the reason a path could not be constructed. Once the network is corrected, the problem travelers can be re-routed and merged into the full plan set.

Some of the more frequently encountered messages in the problem file are listed here:

Path-Building Problem. The Router was not able to find a feasible path between the origin activity location and the destination activity location. This most likely reflects a network coding error that has to do with link directionality or lane connectivity. You should review the network in the vicinity of the origin and destination activities to make sure that these locations have the appropriate network connections.

Time Schedule Problem. The trip travel time exceeded its defined duration in the trip file plus any slack time defined in the Router control file. Such an error most likely reflects the disparity in the average travel speed used in the trip table conversion process and the simulated average travel speed. You can use the `IGNORE_TIME_CONSTRAINTS` key to ignore the time constraint and then re-run the Router to determine if network errors and not congestion are the reason for the time scheduling problem. If the time problem is the result of realistic network congestion, the messages will identify those travelers for whom the start and end times of their trips should be revised. This can be done using the **PlanTrips** program.

Walk Distance Problem. The path's cumulative walking distance exceeds the `MAX_WALK_DISTANCE` parameter. This is most likely to occur with transit trips, when the trip time is outside the service period of the transit routes required for the trip. For example, the trip may have started too late, causing the traveler to miss the last bus on a route between the origin and destination. The only solution is for the traveler to walk all the way from the origin to the destination. If the trip start time were moved a few minutes earlier, however, a more realistic travel plan could be constructed.

A complete list of the problem messages generated by the Router program is available in the Router documentation (see Router v4.0.pdf).

7.0 Frequently Asked Questions

How can the Router processing time be reduced?

> The Router run time is generally a linear function of the number of trips. It takes a Pentium IV machine with a 3.2GHz processor approximately 45 minutes to route one million trips. One way to reduce the total processing time is to run multiple Router applications on different CPUs, with separate household lists provided to each CPU to implement parallel processing. In a machine with multiple CPUs, each CPU can independently run an instance of the Router.
>
> Saving plan files as text will be necessary to help debug any errors and/or difficulties that are encountered during the model development process. However,you can eventually expedite the process by using binary files instead of ASCII files to store and retrieve the plans. Binary files occupy approximately half the storage space needed for ASCII files. This reduces reading and writing times by half.
>
> There are also a number of control keys that can be used to balance processing time against fully enumerated path building. For example, a `MAX_CIRCUITY_RATIO` of 2.0 has been found to reduce run times by 50 percent without biasing the results or producing too many problems. This key limits the links considered in the path building process to those whose distance to the origin and destination is less than twice the distance between the two activity locations.

Also, the MAX_WALKING_DISTANCE parameter has a significant impact on the time required to build transit paths.   Higher walk distances increase the number of transit stops and transfer options considered in the path building process.

How many iterations do we need to achieve convergence?

The number depends on the convergence criterion used and the method selected for achieving it. Studies in Portland, Oregon, and Washington, DC, have shown that after approximately 40 Router–Microsimulator feedback loops, the percentage of travelers who could improve their travel time by changing paths is less than 2 percent. Additional iterations do not significantly improve this result.

Can the Router build multimodal shortest paths?

The primary mode of each trip is specified in the input trip or activity file. The Router will build the path using the primary mode. Since walking is a component in all paths, the Router may find that walking from the origin to the destination is the minimum impedance path. In this case, the path will not include the primary mode.

The Router does, however, enable you to use control keys to change the primary trip mode. For example, you can redesignate auto trips as transit, park-and-ride, or bicycle trips. This is one way to generate mode-specific attributes for input to a mode choice model. See the Router documentation for implementation details.