Router *(version 4.0)*

Revision History

1/8/2010	Edited by AECOM Consult, Inc.
4/20/2010	Edited by RSG, Inc.

The **Router** program is used to:

1. Generate travel plans for household activities that are connected by walk, drive, transit, park-&-ride, kiss-&-ride, bicycle, and magic move modes.
2. Generate travel plans for household and itinerant trips by walk, drive, transit, park-&-ride, kiss-&-ride, bicycle, and magic move modes.
3. Build travel plans from specified origins to specified destinations at specified times of day using a specified travel mode.
4. Selectively route activities or trips from specified origins, to specified destinations, at specified times of day, and\or by specified modes.
5. Generate problem files for those activities or trips that could not be routed for specific reasons.
6. Implement an incremental capacity restrained assignment algorithm.
7. Re-route selective trips with a household person's tour.
8. Update the plans in an input plan file.
9. Dump out an ArcView shapefile showing the links reached by a path building task that experienced problems of a selected type.
10. Routing by selected trip purposes.
11. A vehicle file is optional for plans that are not simulated.

Router is a console-based program that runs in a command window on either Windows or Linux. The command syntax is:

Router [-flag] [control_file] [partition]

The control_file is the file name of an ASCII file that contains the control strings expected by the program. The control_file is optional. If a file name is not provided, the program will prompt the user to enter a file name. The flag parameters are also optional. Any combination of the following flag parameters can be included on the command line:

-Q[uiet]	= execute without screen messages
-H[elp]	= show program syntax and control keys
-K[eyCheck]	= list unrecognized control file keys
-P[ause]	= pause before exiting
-N[oPause]	= never pause before exiting
-B[atch]	= execute in batch processing mode

The partition parameter is optional. It is used to specify the partition number for a particular execution. Partitions are used in the TRANSIMS process for parallel execution on multiple CPUs. If a partition number is provided, the program will build plans for the households included in the corresponding **Router** household list file and store the results in the corresponding plan and problem files. A partition number of "0" corresponds to partition files ".tAA", "1" equals ".tAB", etc.

The program automatically creates a printout file based on the control_file name. If the file name includes an extension, the extension is removed and ".prn" is added. The printout file will be created in the current working directory and will overwrite an existing file with the same name. If a partition parameter is provided, the file extension is replaced by "_#.prn" where "#" is the partition parameter.

Control File Parameters

Control parameters are defined using a control key followed by a string or number. The control parameters can be specified in any order. If a given key is defined more than once, the last instance of the key is used. The default value for each key is 0 or "Null". Null parameters do not need to be included in the file. Note that comment lines or extraneous keys can be included in the file. They will be ignored by the program.

A typical **Router** control file is shown below:

```
TITLE                             Mode Router Test

#---- Network Files -----#
NET_DIRECTORY                     \u1\projects\mpo5\network
NET_SIZE_TABLE                    Size
NET_NODE_TABLE                    Node
NET_LINK_TABLE                    Link
NET_LANE_CONNECTIVITY_TABLE       Lane_Connectivity
NET_PARKING_TABLE                 Parking
NET_ACTIVITY_LOCATION_TABLE       Activity_Location
NET_PROCESS_LINK_TABLE            Process_Link
NET_TRANSIT_STOP_TABLE            Transit_Stop
NET_TRANSIT_FARE_TABLE            Transit_Fare
NET_TRANSIT_ROUTE_TABLE           Transit_Route
NET_TRANSIT_SCHEDULE_TABLE        Transit_Schedule

#---- Input Files ----#
LINK_DELAY_FILE                   \u1\projects\mpo5\msim\LinkDelay.90.tim
HOUSEHOLD_LIST                    \u1\projects\mpo5\population\hhold_list.100
VEHICLE_FILE                      \u1\projects\mpo5\vehicle\Total_Vehicle
ACTIVITY_FILE                     \u1\projects\mpo5\activity\Total_Activity

#---- Output Files ----#
NEW_PLAN_FILE                     \u1\projects\mpo5\plans\Total_Plans.100
NEW_PROBLEM_FILE                  \u1\projects\mpo5\output\Problems.100

#---- Options -----#
NODE_LIST_PATHS                   YES
ROUTE_SELECTED_MODES              1,2,3,4,5,6,7,8,9
ROUTE_WITH_SPECIFIED_MODE         3
LIMIT_PARKING_ACCESS              YES
```

```
IGNORE_TIME_CONSTRAINTS           NO
PERCENT_RANDOM_IMPEDANCE          20
RANDOM_NUMBER_SEED                0

#---- Parameters -----#
WALK_SPEED                        1.0
BICYCLE_SPEED                     4.0
WALK_TIME_VALUE                   20.0
BICYCLE_TIME_VALUE                15.0
FIRST_WAIT_VALUE                  20.0
TRANSFER_WAIT_VALUE               20.0
VEHICLE_TIME_VALUE                10.0
DISTANCE_VALUE                    0.0
COST_VALUE                        144.0        #---- $5\hour of walk time ----
TRANSFER_PENALTY                  0.0
RAIL_BIAS_FACTOR                  1.0

#---- Constraints -----#
MAX_WALK_DISTANCE                 2000
MAX_BICYCLE_DISTANCE              10000
MAX_WAIT_TIME                     60
MAX_NUMBER_OF_TRANSFERS           5
MAX_PARK_RIDE_PERCENTAGE          50
MAX_CIRCUITY_RATIO                2.0
MIN_CIRCUITY_DISTANCE             2000
MAX_CIRCUITY_DISTANCE             20000
MAX_ROUTING_ERRORS                100000
MAX_LINK_DELAY_ERRORS             100000

#----- V\C Controls ----#
LINK_DELAY_UPDATE_RATE            0
EQUATION_PARAMETERS_1             BPR, 0.15, 4.0, 0.75
```

This example creates plans for the households in the "hhold_list.100.t*" file. The program initializes the 15 minute link volumes and travel times from information provided in the link delay file "LinkDelay.90.tim". All plans will be routed as Transit with a minimum impedance path based on 10 times the in-vehicle travel time, 20 times the walking time, 20 times the wait time, and 144 times the fare. This value is randomly adjusted by as much as plus or minus 10 percent for each traveler. The cumulative walk distance must be less than 2000 meters, the wait time must be less than 60 minutes, and the number of transfers must be less than 5.

The keys recognized by the **Router** program are listed below. These keys can be defined in a variety of different ways to perform different tasks.

TITLE

Any text string can be used on this line. This text is printed on the top of each output page.

REPORT_FILE

The report file name is optional. If a file name is not provided, the program automatically creates a report file name based on the input control file name plus the partition number. The report file will overwrite an existing file with the same name if the Report Flag key is False or not specified.

REPORT_FLAG

The report flag key is optional. If it is specified as Yes or True, the report file or default printout file will be opened in "Append" mode rather than "Create" mode. This permits the user to consolidate the output of several programs into a single report file.

MAX_WARNING_MESSAGES

When the program generates a warning message, a counter is incremented and the total number of warning messages is reported and a warning return coded (2) is set at the end of the execution. By default the program prints up to 100,000 warning messages to the print-out file. If more than 100,000 warning messages are sent, the program stops printing additional messages to the file or terminates the program with an error message based on the MAX_WARNING_EXIT_FLAG. This parameter enables the user to modify the default warning limit.

MAX_WARNING_EXIT_FLAG

If the maximum number of warning messages is exceeded, this flag directs the program in what to do. If the flag is TRUE (the default), the program is terminated with an error message about the warning messages. If the flag is FALSE, the program continues execution, but no additional warning messages are sent to the screen or written to the printout file. The warning message counter continues to count the messages and reports the total at the end of the execution.

PROJECT_DIRECTORY

The project directory key is not required. If it is specified, it is added to all non-network file names required by the program. If it is not specified, all non-network file names should fully specify the file path.

NET_DIRECTORY

The network directory key is not required. If it is specified, it is added to all network table names. If it is not specified, the network table names should fully specify the file path.

NET_NODE_TABLE

The node table key is required. It specifies the name of the TRANSIMS node file within the network directory. The full path and file name for the node table is constructed by appending the value of this key to the value of the NET_DIRECTORY key.

NET_LINK_TABLE

The link table key is required. It specifies the name of the TRANSIMS link file within the network directory. The full path and file name for the link table is constructed by appending the value of this key to the value of the NET_DIRECTORY key.

NET_LANE_CONNECTIVITY_TABLE

The network lane connectivity table key is required. It specifies the name of the TRANSIMS lane connectivity file within the network directory. The full path and file name for the lane connectivity table is constructed by appending the value of this key to the value of the NET_DIRECTORY key.

NET_PARKING_TABLE

The network parking table key is required. It specifies the name of the TRANSIMS parking table file within the network directory. The full path and file name for the parking table is constructed by appending the value of this key to the value of the NET_DIRECTORY key.

NET_ACTIVITY_LOCATION_TABLE

The network activity location table key is required. It specifies the name of the TRANSIMS activity location file within the network directory. The full path and file name for the activity location table is constructed by appending the value of this key to the value of the NET_DIRECTORY key.

NET_PROCESS_LINK_TABLE

The network process link table key is required. It specifies the name of the TRANSIMS process link file within the network directory. The full path and file name for the process link table is constructed by appending the value of this key to the value of the NET_DIRECTORY key.

NET_LANE_USE_TABLE

The network lane use table key is optional. It specifies the name of the TRANSIMS lane-use file within the network directory. The full path and file name for the lane-use table is constructed by appending the value of this key to the value of the NET_DIRECTORY key.

NET_TOLL_TABLE

The network toll table key is optional. It specifies the name of the TRANSIMS toll file within the network directory. The full path and file name for the toll table is constructed by appending the value of this key to the value of the NET_DIRECTORY key. Records in the toll file will override the optional toll field in the link file. The toll file permits differential tolls by vehicle type and time of day.

NET_TURN_PROHIBITION_TABLE

The network turn prohibition table key is optional. It specifies the name of the TRANSIMS turn prohibition file within the network directory. The full path and file name for the turn prohibition table is constructed by appending the value of this key to the value of the NET_DIRECTORY key. Optional fields may be included in the turn file to restrict turning movements by vehicle type and include turn penalties in addition to turn prohibitions.

NET_TRANSIT_STOP_TABLE

The transit stop table is optional. If the stop table is not provided, transit paths cannot be built. This key specifies the name of the TRANSIMS transit stop file within the network directory. The full path and file name for the transit stop table is constructed by appending the value of this key to the value of the NET_DIRECTORY key.

NET_TRANSIT_FARE_TABLE

The transit fare table is optional. It specifies the name of the transit fare file within the network directory. The full path and file name for the transit fare table is constructed by appending the value of this key to the value of the NET_DIRECTORY key. The fare file is a comma, space, or tab delimited ASCII file with the following column headers:

FROM_ZONE	Boarding Fare Zone Range
TO_ZONE	Alighting Fare Zone Range
FROM_MODE	TRANSIMS Mode Labels for the Previous Transit Mode
TO_MODE	TRANSIMS Mode Labels for the Current Transit Mode
PERIOD	Time Period Range
CLASS	Fare Classification Range
COST or FARE	Fare in Cents

The fare zones are defined on the Transit Route file. The Mode options include NONE, BUS, EXPRESS, TROLLEY, STREETCAR, LIGHTRAIL, RAPIDRAIL, REGIONRAIL, and ANY. A sample transit fare table is shown below:

FROM_ZONE	TO_ZONE	FROM_MODE	TO_MODE	PERIOD	CLASS	FARE	NOTES
0..2	0..2	NONE	ANY	0:00..24:00	CASH	100	Base Fare
0..2	0..2	ANY	ANY	0:00..24:00	CASH	25	Transfer Fare
0..2	0..2	RAPIDRAIL	RAPIDRAIL	0:00..24:00	CASH	0	Free transfer on rail
0..2	0..2	ANY	BUS	7:00..9:30	CASH	35	Bus Transfer Fare
1	2	BUS	BUS	0:00..24:00	SPECIAL	25	Transfer Fare
2	0..2	NONE	ANY	15:00..18:00	CARD	85	Card Discount

NET_TRANSIT_ROUTE_TABLE

The transit route table is required if the transit stop file is provided. This key specifies the name of the TRANSIMS transit route file within the network directory. The full path and file name for the transit route table is constructed by appending the value of this key to the value of the NET_DIRECTORY key.

NET_TRANSIT_SCHEDULE_TABLE

The transit schedule table is required if the transit stop file is provided. This key specifies the name of the TRANSIMS transit schedule file within the network directory. The full path and file name for the transit schedule table is constructed by appending the value of this key to the value of the NET_DIRECTORY key.

LINK_DELAY_FILE

The link delay file key is optional. If the key is provided, the program uses the information in the link delay file to initialize the link volumes and travel times for each time period. The header record in the link delay file is used to determine the size of each time period. The time periods are typically 15 minutes long. If a link delay file is not provided (or the key is "NULL"), free flow speeds are used for all times of day. Free flow speeds are also used for all links and time periods not included in the link delay file.

The link delay file can be created by the PlanSum or Microsimulator programs. The following Microsimulator configuration keys are used to create the file:

```
OUTPUT_SUMMARY_FILE_1           LinkDelay
OUTPUT_SUMMARY_FILE_1           TAB_DELIMITED
OUTPUT_SUMMARY_INCREMENT_1      900
OUTPUT_SUMMARY_START_TIME_1     0
OUTPUT_SUMMARY_END_TIME_1       86400
OUTPUT_SUMMARY_TURN_FLAG_1      YES
```

These commands will create 15-minute (i.e., 900 second) volume and travel time summaries for 24 hours (i.e., 0-86400 seconds) for each link in the network. If the file format is not VERSION3, the volumes and travel times for each turning movement are generated. The turning penalties will be used by the **Router** during the highway path building process. The full path and file name is constructed by appending the value of this key to the value of the PROJECT_DIRECTORY key.

LINK_DELAY_START_TIME

The link delay start time is optional and defaults to zero (midnight). The value can be coded as seconds, hours, or clock time. This key can be used in conjunction with the LINK_DELAY_END_TIME key to control the way time periods in the input link delay file are used for travel time calculations. By default, volume and travel time after midnight are added to the corresponding early morning time periods and the trips on the network after midnight use these early morning travel times. The new keys can be used to extend the time period processing beyond 24 hours. The maximum allowed start time is 5 days.

LINK_DELAY_END_TIME

The link delay start time is optional and defaults to 24 hours or midnight. The value can be coded as seconds, hours, or clock time. This key can be used in conjunction with the LINK_DELAY_START_TIME key to control the way time periods in the input link delay file are used for travel time calculations. By default, volume and travel time after midnight are added to the corresponding early morning time periods and the trips on the network after midnight use these early morning travel times. The new keys can be used to extend the time period processing beyond 24 hours. The maximum allowed end time is 5 days.

HOUSEHOLD_LIST

The household list file is optional and may not be coded if the HOUSEHOLD_RECORD_FILE is provided. If neither key is provided, plans will be generated for all households in the trip and\or activity files. If it is provided, the key is appended to the value of the PROJECT_DIRECTORY key to identify the full path to one or more household list files. A household list file is a simple list of the household ID numbers that will be processed by a particular **Router** application. A sample household list is shown below.

```
3
20
32
49
100
120
```

The household list key can be the path to a specific file or the root path to a group of partitioned files. If the command line includes a partition parameter, the program will add ".t*" to the household list key. If the partition number is "0", the household list will include the "tAA" extension. If the partition number is "1", the "tAB" extension is used….

HOUSEHOLD_RECORD_FILE

The household record file is optional and may not be coded if a HOUSEHOLD_LIST is provided. If neither key is provided, plans will be generated for all households in the trip and\or activity files. If it is provided, the key is appended to the value of the PROJECT_DIRECTORY key to identify the full path to one or more household record files. A household record file is a standard TRANSIMS data file with tab-delimited field headers (or a *.def file). Three fields are recognized for selecting households, persons, and trips for routing. A sample household record file is shown below.

```
HHOLD  PERSON TRIP
3      1      0
20     3      5
32     0      0
49     2      0
100    0      0
120    4      3
```

Note that zero is used to specify that all trips or persons defined for the household within the trip or activity file will be processed. The first record, therefore, instructs the **Router** to build paths for all trips made by person #1 within household #3. Conversely, the second record indicates that only trip #5 for person #3 within household #20 will be re-Routed. The third record has exactly the same affect as the household list file. In other words, all persons and trips within household #32 will be routed.

The household record file key can be the path to a specific file or the root path to a group of partitioned files. If the command line includes a partition parameter, the program will add ".t*" to the household list key. If the partition number is "0", the household list will include the "tAA" extension. If the partition number is "1", the "tAB" extension is used….

HOUSEHOLD_FILE

The household file is optional. If the file is provided, it can be used to modify the path selection criteria based on the household type of the traveler. The HOUSEHOLD_TYPE_SCRIPT is used to define the household type assigned to a given traveler. The type value is used as an index into the impedance value keys. The full path and file name is constructed by appending the value of this key to the value of the PROJECT_DIRECTORY key.

HOUSEHOLD_TYPE_SCRIPT

The household type script is required when a HOUSEHOLD_FILE is provided. If needed, this key appended to the value of the PROJECT_DIRECTORY key to define the full path to an ASCII file containing a User Program script. The script can reference any field in the Household File using the syntax "Household.Field". The script should return the index into the impedance value keys to be used for the given household.

PARKING_PENALTY_FILE

The parking penalty file is optional. It is a simple delimited text file with a field for the parking lot ID and a impedance penalty field. The parking penalties are applied to all drive or park-&-ride trips that park their vehicle at the parking lot. They do not affect kiss-&-ride trips. The penalty is typically used to adjust the demand for a parking lot to the parking lot capacity. It is often used to adjust the station choice component of rail transit routes.

TRANSIT_PENALTY_FILE

The transit penalty file is optional. It is a simple delimited text file with fields for the transit stop, transit route and\or transit run and an impedance penalty field. This file can be used to input impedance penalties for combinations of stop, route, and\or run to adjust the likelihood that a stop will be included in a path based on the capacity constraints of the transit service. The program also allows the user to drop the stop number to apply the specified penalty to all stops on a route and\or run. It allows for both, a high-level and a fine-tuned transit ridership calibration.

VEHICLE_FILE

The vehicle file is required if a trip or activity file are processed and the IGNORE_VEHICLE_ID key is false. The vehicle file contains the parking location of each vehicle used for drive activities. The full path and file name is constructed by appending the value of this key to the value of the PROJECT_DIRECTORY key.

SORT_VEHICLES

The sort vehicles key is optional. If provided, it tells the program if the vehicle file needs to be sorted. The default value is TRUE. This implies that the vehicle file needs to be sorted by vehicle ID. If the key is FALSE and the vehicle file is not sorted, the program is likely to take considerably long to read the vehicle file. Setting this key to TRUE can significantly reduce the time required to read the vehicle file.

IGNORE_VEHICLE_ID

The vehicle file processing is made optional based on this key value. If TRUE, the vehicle file is not processed and the location of the vehicle is assumed connected to the parking lot attached to the origin and destination activity locations. The default value is FALSE. This implies that the vehicle file is read and the location of the vehicles is check and repositioned based on the path building result. Setting this key to TRUE can save processing time, but it is primarily used to build drive plans for transit trips as input to a PlanSum process to generate travel time skims for a mode choice model.

VEHICLE_TYPE_FILE

The vehicle type file is optional and is used to map vehicle types and sub-types in the vehicle file to specific lane or link use restrictions. The full path and file name is constructed by appending the value of this key to the value of the PROJECT_DIRECTORY key.

TRIP_FILE

A trip or activity file is required if origin-destination routing is not specified. The program can process both a trip and activity file in the same application. If both files are provided, the trip file will be processed first. The trip file contains the trips and travel modes for each traveler. The full path and file name is constructed by appending the value of this key to the value of the PROJECT_DIRECTORY key. If the file name includes ".t*" or ".*", the file is processed as a partitioned file with an extension based on the partition parameter on the command line.

ACTIVITY_FILE

The activity or trip file is required if origin-destination routing is not specified. The program can process both a trip and activity file in the same application. If both files are provided, the trip file will be processed first. The activity file contains the activity schedule for each traveler and the travel modes used to travel from one activity location to the next. The full path and file name is constructed by appending the value of this key to the value of the PROJECT_DIRECTORY key. If the file name includes ".t*" or ".*", the file is processed as a partitioned file with an extension based on the partition parameter on the command line.

PLAN_FILE

An input plan file is optional. When appended to the PROJECT_DIRECTORY key, the key specifies the file name for the input plan file that is copied to the output plan file as specified trips are re-routed. If the key is not provided, the new plans can be merged with the original plan file using the PlanPrep program. If the command line includes a partition parameter, the program will add ".t*" to this key. If the partition number is "0", the "tAA" extension is added. If the partition number is "1", the "tAB" extension is added…. If the plan files have a companion *.def file, the PLAN_FORMAT and NODE_LIST_PATHS keys are not required.

PLAN_FORMAT

The plan format key is optional. If provided, it defines the file format for the input plan file. The default value is VERSION3 (unformatted text) format. This parameter enables the user to input plans in BINARY format.

NEW_PLAN_FILE

The new plan file key is optional. When appended to the PROJECT_DIRECTORY key, it specifies the file name for the output plan file created by the program. If the key is not provided, plans will be built but not saved. If the command line includes a partition parameter, the program will add ".t*" to this key. If the partition number is "0", the "tAA" extension is added. If the partition number is "1", the "tAB" extension is added….

NEW_PLAN_FORMAT

The new plan format key is optional. If provided, it defines the file format for the output plan file. The default value is VERSION3 (unformatted text) format. This parameter enables the user to output the plan file in BINARY format.

NODE_LIST_PATHS

The node list paths key is optional and when provided specifies the way the path is identified in the output plan file. The key is "true" by default. This means that the output plans will include a list of the node ID numbers along the travel path. If the key is "false", the program saves the path as a list of link ID numbers. If the first character of the key is "0", "N", "n", "F", or "f", the key is interpreted as "false". If input plans are provided, these plans must have same node list format.

NEW_PROBLEM_FILE

The new problem file key is optional. When appended to the PROJECT_DIRECTORY key, it specifies the file name for the output problem file created by the program. If the key is not provided, the path building problems will be summarized but not saved. If the command line includes a partition parameter, the program will add ".t*" to this key. If the partition number is "0", the "tAA" extension is added. If the partition number is "1", the "tAB" extension is added….

The problem file is a tab-delimited ASCII file that lists the households and persons for whom complete plans could not be generated. A sample problem file generated by the **Router** is shown below.

```
HHOLD   PERSON TRIP   MODE   PROBLEM       START  ORIGIN ARRIVE DESTINATION
178     1      1      2      3      3272   5      3485   7
343     1      1      2      3      1089   24     1302   22
356     1      1      2      3      3178   11     3391   10
359     1      1      2      3      3264   11     3477   10
503     1      1      2      3      997    10     1210   11
```

The PROBLEM field identifies the problem type for a given record. The message corresponding to each problem code is listed below. An interpretation of the messages generated by this program is provided in the algorithm section of this document.

1. Path Building
2. Time Schedule
3. Zero Node
4. Vehicle Type
5. Path Circuity
6. Travel Mode
7. Vehicle Access
8. Walk Distance
9. Wait Time
10. Walk Access
11. Path Size
12. Park-&-Ride Lot
13. Bike Distance
14. Departure Time
15. Arrival Time
16. Link Access
17. Lane Connectivity
18. Parking Access
19. Lane Merging
20. Lane Changing
21. Turning Speed
22. Pocket Merge
23. Vehicle Spacing
24. Traffic Control
25. Access Restriction
26. Transit Stop
27. Activity Location
28. Vehicle Passenger
29. Activity Duration
30. Kiss-&-Ride Lot
31. Vehicle ID
32. Data Sort
33. Walk Location
34. Bike Location
35. Transit Location

TIME_OF_DAY_FORMAT

The time of day format defines how the start and end times are written to the output files. The default format will display values in hours. The format options include HOURS, SECONDS, 24_HOUR_CLOCK, and 12_HOUR_CLOCK.

ROUTE_SELECTED_MODES

By default, the **Router** builds paths for all households in the household list file. This parameter permits the user to select the modes on the activity file to be routed. The key is a comma separated list of the mode codes used in the activity file (1..14). All modes will be routed if the key is "1,2,3,4,5,6,7,8,9,10,11,12,13,14". Only transit trips are routed if the key is "3".

ROUTE_WITH_SPECIFIED_MODE

Normally a trip is routed using the mode code on the activity file. If this option is used, the mode code is replaced by the specified value. This enables the user to quickly route a selected set of households (or selected modes if the ROUTE_SELECTED_MODES options is used) by a different mode. A value of "3" means that all households will be routed as transit. The mode codes are:

1	Walk
2	Drive
3	Transit
4	Transit with Rail Bias
5	Park-&-Ride Outbound
6	Park-&-Ride Inbound
7	Bicycle
8	Magic Move
9	School Bus
10	Carpool 2
11	Carpool 3
12	Carpool 4+
13	Kiss-&-Ride Outbound
14	Kiss-&-Ride Inbound

ROUTE_FROM_SPECIFIED_LOCATIONS

This key is used to select origin activity locations from a trip or activity file for routing or to specify a list of origins to build plans from if a trip or activity file is not provided. The default value is "ALL". The activity locations are interpreted as a comma-delimited list of numbers or number ranges. A sequential range of activity locations are specified by providing the first ID in the range and the last ID in the range separated by two periods (e.g., 1000..2000).

ROUTE_TO_SPECIFIED_LOCATIONS

This key is used to select destination activity locations from a trip or activity file for routing or to specify a list of destinations to build plans to if a trip or activity file is not provided. The default value is "ALL". The activity locations are interpreted as a comma-delimited list of numbers or number ranges. A sequential range of activity locations are specified by providing the first ID in the range and the last ID in the range separated by two periods (e.g., 1000..2000).

ROUTE_AT_SPECIFIED_TIMES

This key is used to select start times from a trip or activity file for routing or to specify time periods for path building when a trip or activity file is not provided. The default value is “ALL”. The time periods are interpreted as a comma-delimited list of time ranges. Time ranges can be specified in seconds of the day or 24 hour clock time (e.g., 7:30..9:30).

ROUTE_BY_TIME_INCREMENT

This key is only used when trip or activity files are not provided. It is used to specify the time increment at which paths will be built over the specified time range. The default value is zero which means that only one path will be built for each time period range. The key is interpreted as minutes and can range from zero to 240 minutes (4 hours). For example, if the specified time period is 7:30..9:30 and the time increment is 30 minutes, paths between each origin-destination combination will be built at 7:30, 8:00, 8:30, and 9:00.

ROUTE_SELECTED_PURPOSES

This key is used to select trip purposes from a trip or activity file for routing. The default value is “ALL”. The key value is interpreted as a comma-delimited list of trip purpose ranges (integers). Trip purposes are defined by the user and have not fixed meaning within TRANSIMS (except that trip purpose zero is HOME).

LIMIT_PARKING_ACCESS

The limit parking access key is optional and when provided controls the way vehicles are associated with activity locations. The key is “true” by default. This means that the vehicle must be parked at a parking lot directly connected to the activity location through a process link. If the key is “false”, the program will build a walk path between the activity location and the vehicle. If the first character of the key is “0”, “N”, “n”, “F”, or “f”, the key is interpreted as “false”.

IGNORE_TIME_CONSTRAINTS

The ignore time constraints key is optional and when provided controls how the activity start time impacts path building. The key is “false” by default. This means that the trip must be completed before the upper bound of the activity start time. If the trip takes too long, a time schedule error is registered in the problem file. If the key is “true”, the program will continue building the path without consideration of the activity schedule. The start time of the next trip will be the arrival time of the previous trip plus the duration of the activity. If the first character of the key is “0”, “N”, “n”, “F”, or “f”, the key is interpreted as “false”. Anything else is interpreted as “true”.

END_TIME_CONSTRAINT

The end time constraint is optional and only applied if the IGNORE_TIME_CONSTRAINTS key is “false”. This parameter enables the user to add a time buffer to the end time of the trip to limit the time constraint errors to those instances where the travel exceeds the end time plus the end time constraint. The parameter is defined in minutes. The default is zero.

PERCENT_RANDOM_IMPEDANCE

The percent random impedance key is optional and specifies the amount of random impedance effects. The key can range from zero to 100 percent. The default value is zero. Zero implies that all travelers perceive the impedance on a given link in exactly the same way. Non-zero parameters cause the program to randomly adjust the link impedance each time it is considered by the path-building algorithm. A value of 20 means that the impedance perceived by the traveler may be as much as 10 percent less or 10 percent more than the "actual" impedance.

RANDOM_NUMBER_SEED

The random number seed key is optional. This key specifies the random number seed to be used for the random impedance calculations. Any positive integer can be specified. If the value is zero or if no key is provided, the program uses the system clock to set the random number seed.

WALK_SPEED

The walk speed is optional and when provided specifies the walking speed in meters per second. The value can range from 0.5 to 10.0 meters per second. The default value is 1.0. Link lengths are divided by this value to convert distance into walk time.

BICYCLE_SPEED

The bicycle speed is optional and when provided specifies the bicycling speed in meters per second. The value can range from 1.0 to 20.0 meters per second. The default value is 4.0. Link lengths are divided by this value to convert distance into bicycle time.

WALK_TIME_VALUE

The walk time value key is optional and when provided specifies the impedance values for time the traveler spends walking. The values can range from zero to 1000.0. The default value is 20.0. This value is multiplied by the time spent walking on network and process links. If household types are defined, this key can include a list of values corresponding to each household type. For example, 20, 25, 30 can be specified to define the walk time value for household types 1, 2 and 3+, respectively.

BICYCLE_TIME_VALUE

The bicycle time value key is optional and when provided specifies the impedance values for time the traveler spends bicycling. The values can range from zero to 1000.0. The default value is 15.0. This value is multiplied by the time spent bicycling on network links. If household types are defined, this key can include a list of values corresponding to each household type. For example, 15, 20, 25 can be specified to define the bicycle time value for household types 1, 2 and 3+, respectively.

FIRST_WAIT_VALUE

The first wait value key is optional and when provided specifies the impedance values for time the traveler spends waiting for the first transit vehicle. The values can range from zero to 1000.0.

The default value is 20.0. This value is multiplied by the difference between the time of day when the traveler arrives at a transit stop and the time when the next transit vehicle is scheduled to leave that stop. If household types are defined, this key can include a list of values corresponding to each household type. For example, 20, 25, 30 can be specified to define the first wait time value for household types 1, 2 and 3+, respectively.

TRANSFER_WAIT_VALUE

The transfer wait value key is optional and when provided specifies the impedance values for time the traveler spends waiting to transfer to another transit vehicle. The values can range from zero to 1000.0. The default value is 20.0. This value is multiplied by the difference between the time of day when the traveler arrives at a transit stop and the time when the next transit vehicle is scheduled to leave that stop. If household types are defined, this key can include a list of values corresponding to each household type. For example, 20, 25, 30 can be specified to define the transfer wait time value for household types 1, 2 and 3+, respectively.

VEHICLE_TIME_VALUE

The vehicle time value key is optional and when provided specifies the impedance values for time the traveler spends in a vehicle. The values can range from zero to 1000.0. The default value is 10.0. This value is multiplied by the travel time on each link at the time of day when the traveler's path enters the link or the difference in the schedule time between the boarding and alighting transit stops. If household types are defined, this key can include a list of values corresponding to each household type. For example, 10, 15, 20 can be specified to define the in-vehicle time value for household types 1, 2 and 3+, respectively.

DISTANCE_VALUE

The distance value key is optional and when provided specifies the impedance values for the distance traveled. The values can range from zero to 1000.0. The default value is 0.0. This value is multiplied by the length of the each link. It is only used for driving trips. If household types are defined, this key can include a list of values corresponding to each household type. For example, 0, 5, 10 can be specified to define the distance value for household types 1, 2 and 3+, respectively.

COST_VALUE

The cost value key is optional and when provided specifies the impedance values for travel cost. The values can range from zero to 1000.0. The default value is 0.0. This value is multiplied by the cost value on Process Links, parking lots, and the transit fare. The program also looks for a "COST" field on the Link file. If household types are defined, this key can include a list of values corresponding to each household type. For example, 0, 5, 10 can be specified to define the cost value for household types 1, 2 and 3+, respectively.

LEFT_TURN_PENALTY

The left turn penalty key is optional and when provided specifies an additional impedance value for lane connections identified as left turns. The values can range from zero to 10000.0. The default value is 0.0. This value is added to the impedance of the departure link of a drive path. If household types are defined, this key can include a list of values corresponding to each household type. For example, 0, 100, 200 can be specified to define the left turn penalty for household types 1, 2 and 3+, respectively.

RIGHT_TURN_PENALTY

The right turn penalty key is optional and when provided specifies an additional impedance value for lane connections identified as right turns. The values can range from zero to 10000.0. The default value is 0.0. This value is added to the impedance of the departure link of a drive path. If household types are defined, this key can include a list of values corresponding to each household type. For example, 0, 100, 200 can be specified to define the right turn penalty for household types 1, 2 and 3+, respectively.

UTURN_PENALTY

The U-turn penalty key is optional and when provided specifies an additional impedance value for lane connections identified as U-turns. The values can range from zero to 10000.0. The default value is 0.0. This value is added to the impedance of the departure link of a drive path. If household types are defined, this key can include a list of values corresponding to each household type. For example, 0, 100, 200 can be specified to define the U-turn penalty for household types 1, 2 and 3+, respectively.

TRANSFER_PENALTY

The transfer penalty key is optional and when provided specifies an additional impedance value for transferring from one transit route to another. The values can range from zero to 10000.0. The default value is 0.0. This value is added to the impedance of the access link to the second, third, etc, transit boarding stop. If household types are defined, this key can include a list of values corresponding to each household type. For example, 0, 100, 200 can be specified to define the transfer penalty for household types 1, 2 and 3+, respectively.

STOP_WAITING_PENALTY

The stop waiting penalty is similar to a transfer penalty, but applies to all boardings at a transit stop coded with the “STOP” type. This is typically used to distinguish boardings at bus stops from boardings at rail stations. The default value is 0.0. The values can range from zero to 100,000 impedance units. If household types are defined, this key can include a list of values corresponding to each household type.

STATION_WAITING_PENALTY

The station waiting penalty is similar to a transfer penalty, but applies to all boardings at a transit stop coded with the “STATION” type. This is typically used to distinguish boardings at a rail station from boardings at a bus stop. The default value is 0.0. The values can range from zero to 100,000 impedance units. If household types are defined, the key can include a list of values corresponding to each household type.

BUS_BIAS_FACTOR

The bus bias factor is optional and when provided factors up the total impedance value for each segment of a transit trip that uses a local or express bus. The value can range from 1.0 to 3.0.

The default value is 1.0. If household types are defined, this key can include a list of factors corresponding to each household type.

BUS_BIAS_CONSTANT

The bus bias constant is optional. When provided, the total impedance value for each local or express bus segment of a transit trip is adjusted by this value. The value must be greater than zero and is applied after the bus bias factor is applied. If household types are defined, this key can include a list of impedance values corresponding to each household type.

RAIL_BIAS_FACTOR

The rail bias factor is optional and when provided factors down the total impedance value for each segment of a transit trip that uses rail. The value can range from 0.1 to 1.0. The default value is 1.0. This value factors the impedance of rail legs. Rail is defined as any transit mode other than bus (i.e., TROLLEY, STREETCAR, LIGHTRAIL, RAPIDRAIL, REGIONRAIL). If household types are defined, this key can include a list of factors corresponding to each household type. For example, 1.0, 0.9, 0.8 can be specified to define the rail bias factor for household types 1, 2 and 3+, respectively.

RAIL_BIAS_CONSTANT

The rail bias constant is optional. When provided, the total impedance value for each rail segment of a transit trip is adjusted by this value. The value should be negative impedance units. It is applied after the rail bias factor is applied. The resulting impedance will not be less than zero. Rail is defined as any transit mode other than bus (i.e., TROLLEY, STREETCAR, LIGHTRAIL, RAPIDRAIL, REGIONRAIL). If household types are defined, this key can include a list of factors corresponding to each household type. For example, -300, -400, -500 can be specified to define the rail bias constant for household types 1, 2 and 3+, respectively.

MAX_WALK_DISTANCE

The maximum walk distance key is optional. It defines the maximum cumulative walk distance for a given trip. This includes walks to and from the vehicle and any walks required by transfers. If the value is zero, no walk limitations are imposed. Otherwise the value can range from 100 to 10,000 meters. The default value is 2,000 meters. If household types are defined, this key can include a list of values corresponding to each household type. For example, 4000, 3000, 2000 can be specified to define the maximum walk distance for household types 1, 2 and 3+, respectively.

MAX_BICYCLE_DISTANCE

The maximum bicycle distance key is optional. It defines the maximum cumulative bicycle distance for a given trip. If the value is zero, no distance limitations are imposed. Otherwise the value can range from 1000 to 20,000 meters. The default value is 10,000 meters. If household types are defined, this key can include a list of values corresponding to each household type. For example, 10000, 8000, 6000 can be specified to define the maximum bicycle distance for household types 1, 2 and 3+, respectively.

MAX_WAIT_TIME

The maximum wait time key is optional. It defines the maximum time a person will consider waiting at each transit stop to board a vehicle. If the value is zero, no wait time limitations are imposed. Otherwise the value can range from 1 to 180 minutes. The default value is 60 minutes. If household types are defined, this key can include a list of values corresponding to each household type. For example, 60, 50, 40 can be specified to define the maximum wait time for household types 1, 2 and 3+, respectively.

MIN_WAIT_TIME

The minimum wait time key is optional. It defines the minimum time in seconds a person must wait at each transit stop to board a vehicle. The key defaults to zero, which means that the traveler to board the next transit route immediately after alighting from the previous route. The key is intended to discourage transfers between routes with very tight schedule coordination requirements. Since transit running times are affected by vehicle interactions in the Microsimulator, it is likely that a tight transfer will fail and cause the traveler to wait for the next vehicle on the route. This parameter builds some slack time into the path so the traveler is more likely to complete the trip successfully.

MAX_NUMBER_OF_TRANSFERS

The maximum number of transfers key is optional. It defines the maximum number of times the traveler can transfer between transit routes during a given trip. The value can range from zero to 10 transfers. The default value is 5 transfers. If household types are defined, this key can include a list of values corresponding to each household type. For example, 5, 4, 3 can be specified to define the maximum number of transfers for household types 1, 2 and 3+, respectively.

MAX_PARK_RIDE_PERCENTAGE

The maximum park-&-ride percentage key is optional. It defines the maximum percentage of the total trip length that can be used to access a park-&-ride lot. The length is calculated as the straight-line distance between the trip origin and the park-&-ride lot and the park-&-ride lot and the trip destination. All lots that satisfy the search area are selected for consideration by a park-&-ride trip. The value can range from 1 to 100 percent. The default value is 50 percent. If household types are defined, this key can include a list of values corresponding to each household type. For example, 50, 60, 70 can be specified to define the maximum park-&-ride percentage for household types 1, 2 and 3+, respectively. The algorithm section of this document describes park-&-ride paths in more detail.

MAX_KISS_RIDE_PERCENTAGE

The maximum kiss-&-ride percentage key is optional. It defines the maximum percentage of the total trip length that can be used to access a kiss-&-ride lot. The length is calculated as the straight-line distance between the trip origin and the kiss-&-ride lot and the kiss-&-ride lot and the trip destination. All lots that satisfy the search area are selected for consideration by the kiss-&-ride trip. The value can range from 1 to 100 percent. The default value is 35 percent. If household types are defined, this key can include a list of values corresponding to each household type.

KISS_RIDE_TIME_FACTOR

Since the drive leg of a kiss-&-ride trip requires a driver of the vehicle, the impedance for the time spent driving to the kiss-&-ride lot is factored to apply some value to the driver's time. The kiss-&-ride time factor accounts for the driver's time in the selection of the kiss-&-ride lot. The default value is 2.5 which implies a value of 1.0 for the kiss-&-ride traveler and a value of 1.5 for the driver. This assumes the driver will return home after dropping off the kiss-&-ride traveler, but the value of the driver's time is less than the kiss-&-ride traveler. The value can range from 1.0 to 4.0. If household types are defined, this key can include a list of values corresponding to each household type.

KISS_RIDE_STOP_TYPES

Kiss-&-ride stop types and dropoff walk distance are used to identify kiss-&-ride lots. Kiss-&-ride lots are standard parking lots within a specified walk distance of a transit stop of the specified stop types. The kiss-&-ride stop types key lists the transit stop types where kiss-&-ride activities are permitted. By default all transit stop types are eligible for kiss-&-ride. The key can include any combination of STOP, STATION, and EXTERNAL stop types.

MAX_KISS_RIDE_DROPOFF_WALK

The maximum kiss-&-ride drop-off walk key defines the maximum walk distance between a standard parking lot and a transit stop that will be considered as a potential kiss-&-ride drop-off location. This key is combined with the kiss-&-ride stop types key to identify kiss-&-ride lots. The default value is 100 meters. The accepted range is 10 to 500 meters.

MAX_LEGS_PER_PATH

This parameter is optional. It defaults to 1000. It is used to protect against infinite loops that can occasionally occur building constrained transit paths. The parameter represents the maximum number of legs (drive, walk, and transit segments) that are permitted on a transit path. If the path exceeds this value, a Path Problem message is generated.

ADD_WAIT_TO_TRANSIT_LEG

This parameter is optional. It is "false" by default. The default behavior is to add the waiting time at a transit stop to the walk access leg entering the stop. This helps to separate out-of-vehicle time from in-vehicle time when summarizing the components of a transit trip for mode choice analysis. If this key is "true", the waiting time is included in the transit leg. This has the effect of setting the start time for the transit leg based on the time the traveler starts waiting as opposed to the time when the transit vehicle is scheduled to depart. This provides additional flexibility in the Microsimulator to control transit routes based on link travel times, traffic control delays, and passenger service times rather than a fixed schedule at each stop.

FARE_CLASS_DISTRIBUTION

The transit fare table has the option of specifying a fare for three classes of travelers (CASH, CARD, and SPECIAL). This key enables the user to randomly distribute transit travelers to one of these three classes. The key accepts up to three percent values representing the probability that a traveler will be assigned to cash, card, or special fare classes. The values are normalized to sum to 100 percent. The default value is 100 percent cash fares.

PARKING_HOURS_BY_PURPOSE

The Version 4 parking table includes two fields for parking cost (HOURLY and DAILY). This key enables the modeler to specify the average parking duration in hours by trip purpose. Trip purpose codes are included in the trip and activity files and can be configured by the modeler is many different ways. For example, the modeler may assign purpose 1 to work trips, purpose 2 to shopping trips, and purpose 3 to other trips. This key can be used in this case to estimate the parking duration for each trip purpose as "9.0, 1.5, 2.5". This means that work trips will park their car for 9 hours, shopping trips for 1.5 hours and other trips for 2.5 hours. When each trip attempts to park at a given parking lot, the number of hours is multiplied by the hourly parking cost and compared to the daily parking cost. The parking cost assigned to the trip will be based on the minimum of the two numbers. The default parking duration for each trip purpose is zero. This will result is zero cost for parking.

LOCAL_ACCESS_DISTANCE

This parameter is optional. The default value is 2000 meters. If the LOCAL_FACILITY_TYPE key is not EXTERNAL, drive paths are restricted to higher facility types when more than the value of this parameter away from their trip origin or destination.

LOCAL_FACILITY_TYPE

This parameter is optional. The default value is EXTERNAL which disables the local facility restrictions for drive paths. Other values include MAJOR, MINOR, COLLECTOR, and LOCAL. When one of these values is provided, drive paths will be restricted to higher facility types when more than the LOCAL_ACCESS_DISTANCE away from their trip origin or destination. This parameter defines the upper end of the facility type restriction. The lower end is always defined as LOCAL. In other words, if the parameter is MINOR, no link with facility types MINOR, COLLECTOR, or LOCAL will be available for the line-haul portion of the trip.

MAX_CIRCUITY_RATIO

The maximum circuity ratio key is optional. The value can be zero or between 1.0 and 10.0. The default value is 2.0. This key defines the maximum permissible ratio between the sum of the distance a path node is from the trip origin and destination and the straight-line distance between the trip origin and destination. If the value is zero, no circuity checks are made. A value of 2.0, implies that the length of the travel path is limited to approximately twice the straight-line distance between the origin and the destination.

This parameter is designed to reduce the processing time of the **Router**. By focusing the path-building algorithm on links that are generally between the origin and the destination, the program can avoid wasting computational time considering paths in the wrong direction. Experience has shown that the total processing time can be reduced by approximately 50 percent when a circuity ratio of 2.0 is used. This may result in as many as one percent of the trips failing due to path circuity errors. The user can re-run these households through the **Router** using a circuity ratio of zero to determine if logical paths can be found.

MIN_CIRCUITY_DISTANCE

The minimum circuity distance key is optional. The value can range from zero to 10,000 meters. The default value is 2,000 meters. This value is used in conjunction with the maximum circuity ratio to focus the path-building algorithm on links that are generally located between the trip origin and destination. If the origin and destination locations are relatively close from a straight-line distance perspective, but not as close from a network perspective, the ratio algorithm can limit the path building in illogical ways. This parameter is designed to help avoid this problem by permitting the algorithm to consider all nodes that are within a minimum distance of the origin and destination. For example, this parameter can allow the algorithm to consider nodes that are up to 2,000 meters away when the trip origin and destination are close to each other but on different streets.

MAX_CIRCUITY_DISTANCE

The maximum circuity distance key is optional. The value can range from zero to 100,000 meters. The default value is 20,000 meters. This value is used in conjunction with the maximum circuity ratio to focus the path-building algorithm on links that are generally located between the trip origin and destination. If the origin and destination locations are far apart, the ratio calculation has little impact on the number of nodes that are considered by the path-building algorithm. This parameter is designed to help narrow the focus of long distance trips to more direct paths. The algorithm uses the minimum of the straight-line distance between the origin and destination multiplied by the circuity ratio and the maximum circuity distance to determine if a node is out of range. For example, this parameter will limit the trip distance to 20,000 meters longer than the straight-line distance between the origin and destination.

MAX_ROUTING_PROBLEMS

The maximum number of routing problems key is optional. It defaults to 100,000 problems. This parameter defines the number of path building problems that are permitted before the program terminates execution. A value of zero will disable this feature.

MAX_LINK_DELAY_ERRORS

The maximum number of link delay errors key is optional. It defaults to 100,000 errors. This parameter defines the number of link delay processing errors that are permitted before the program terminates execution. A processing error can occur if a Link-Node combination found in the link delay file is not included in the input Link file. This typically means that the input network is not fully compatible with the network used to create the link delay file. In most cases, a small number of differences are acceptable, but significant differences will invalidate the path building process. Considering that the link delay file typically includes records for each lane in each direction for each 15 minute time period, changing a single link in the network can easily result in 200 or more data processing errors.

LINK_DELAY_UPDATE_RATE

The link delay update rate key is optional. The value can range from zero to 1,000,000 trips. The default value of zero disables link delay updates. If the key is one or greater, the program will use the BPR volume-delay function to calculate the travel times on each link based on the current volumes in each time period. Time periods are defined by the input link delay file or default to 15 minutes if a link delay file is not provided. This parameter defines the number of trips that are loaded between travel time updates. The update process is described in more detail in the algorithm section of this document.

Note that this parameter should only be used for initial incremental loading. Link delay updates are appropriate when the input link delay file is generated by the PlanSum program. This parameter should be zero if the input link delay file was generated by the TRANSIMS Microsimulator. Microsimulator delays can be significantly different from volume to capacity based delays. If link delay updates are enabled, the delays from the Microsimulator are destroyed by the first update cycle.

LINK_DELAY_VOL_FACTOR

The link delay volume factor can be used in partitioned applications to provide the **Router** with a better approximation of the volume-to-capacity ratio during travel time updates. Since each **Router** in a partitioned application assigns a percentage of the total trips to the network, the volume on assigned to each link does not reflect the total traffic from all trips. This key permits the modeler to multiple the volume loaded by each partition by a factor that reflects the number of other **Router** applications that are loaded traffic to the network at the same time. The factor is only applied when the volume-delay functions update the travel times. The factored volume is not saved to the network or the plan file.

The default volume factor is 1.0. The program accepts any value greater than 1.0. Normally the factor would be set equal to the number of partitions.

UPDATE_PLAN_RECORDS

The update plan records key is optional. The default value is “false”. If true, the program re-skims the travel time and impedance components of existing plan records while building paths for other records. In other words, the generalized cost is updated for all the output plan records based on the current specifications. The plan update can be executed without a household list or an input trip and\or activity file. The print file includes a message about the number of updates that were made.

PRINT_UPDATE_WARNINGS

The print update warnings key is optional. The default value is “false” and is used in along with the UPDATE_PLAN_RECORDS key. If true, warning messages during plan update process are sent to the print file.

EQUATION_PARAMETERS_#

The equation parameters key is optional. A volume-delay equation is used by the link delay updates. The “#” at the end of the key refers to the facility type. For example, EQUATION_PARAMETERS_1 specifies the volume-delay equation used for Freeways. If an equation is not provided for a given facility type, the program uses the equation from a lower facility type code. Each key requires four or five values. The first is the functional type code. Four options are currently available: BPR, BPR_PLUS (or BPR+), EXPONENTIAL (or EXP), and CONICAL (or CON). These are followed by the up to four parameters as floating point numbers separated by a comma. The BPR function default values are 0.15, 4.0, and 0.75. The BPR equation for computing the link travel time is:

$$t = t_0 * (1 + (alpha * (Volume / Capacity)^{Beta}))$$

Where

$$t = LoadedTravelTime(seconds)$$

$$t_0 = BaseTravelTime(seconds)$$

alpha = BPR "A" parameter = 0.15

Beta = BPR "B" parameter = 4.00

Volume = Volume on the link in a given time period

Capacity = Adjusted Capacity of a link in a given time period

Capacity of the link for a given time period is estimated as follows:

$$Capacity = BPRFactor * HourlyCapacity * (TimeIncrement / 3600)$$

Where

BPRFactor = BPR "C" parameter = 0.75
TimeIncrement = TimePeriod (in seconds)

If the BPR Plus option is selected, a fourth parameter can be provided. This parameter represents the minimum travel speed that can result from the calculation. If the fourth parameter is not provided, it will default to 1.0 meters per second. The value must be greater than 0.0 and less then or equal to 7.0 meters per second.

The Exponential function is of the form:

$$delay = \min(\alpha * e^{\beta * v / c}, \chi)$$

where *delay* is represented as minutes per kilometer. This value would therefore be multiplied by the link length and added to the free flow travel time.

The Conical function is of the form:

$$t_v = t_0\left(2 - \beta - \alpha(1 - v/c) + \sqrt{\alpha^2(1 - v/c)^2 + \beta^2}\right)$$

Where

$$\beta = \frac{2\alpha - 1}{2\alpha - 2}$$

In other words, the function takes only one parameter α. Values between 4 and 10 are typically used for different facility types.

ARCVIEW_PROBLEM_DUMP

The problem dump key is optional. When appended to the PROJECT_DIRECTORY key, it specifies the file name of an ArcView shape file that is created for dumping link status information for problem paths. Since the output generated by this key can be huge, it is important to limit the type and number of problem plans that are output. The PROBLEM_DUMP_TYPE and PERCENT_PROBLEMS_DUMPED keys help to limit the number of plans that are written to the shapefile.

PROBLEM_DUMP_TYPE

The problem dump type key is optional. It is used to control the types of problems that are included in the ArcView problem dump file. One of the problem code numbers listed about under the new problem file key is entered using this key to focus the problem dump on a specific problem type. If undefined, households with any type of problem will be written to the output shapefile.

PERCENT_PROBLEMS_DUMPED

The percent problems dumped key is optional. It is used to control the number of problem trips that are written to the ArcView problem dump file. The key value is a floating point percentage of the selected problem types. If undefined, all of the households with problems of the specified problem type will be written to the output shapefile.

ROUTER_REPORT_#

Reports are optional. The "#" at the end of the report keyword represents the report number (e.g., ROUTER_REPORT_1). The key can be provided with additional numbers to specify additional reports. The reports are generated in numerical order (i.e., 1, 2, 3…).

The string parameter associated with a report keyword is limited to the following options:

HOUSEHOLD_TYPE_SCRIPT
HOUSEHOLD_TYPE_STACK
FARE_DATA_REPORT

The above reports are printed in the "*.prn" file that is generated in the same directory as the control file used to run the **Router** program. Each of above reports is described below:

Household Type Script

A Household Type Script is read when a Household File is provided. This report lists the commands found in the script file. An example is shown below.

```
Household Type Script

      IF (Household.Persons > 2) THEN
           IF (Household.Persons > 3) THEN
                RETURN (13)
           ELSE
                IF (Household.AgeLT5 == 1) THEN
                        RETURN (12)
                ELSE
                      IF (household.Age5to17 == 1) THEN
                           RETURN (11)
                      ELSE
                           RETURN (10)
                      ENDIF
                ENDIF
           ENDIF
      ELSE
           IF (Household.Persons == 2) THEN
                IF (Household.AgeLT5 == 1) THEN
                     RETURN (6)
                ELSE
                     RETURN (5)
                ENDIF
           ELSE
                RETURN (3)
           ENDIF
      ENDIF
```

Household Type Stack

The program compiles the household type script into a command processing stack for execution. This report shows how the script was converted to processing commands. The stack commands corresponding to the script listed above are shown below.

```
Household Type Stack

  1) Integer      Household.PERSONS
  2) Integer      2
  3) Relation     GT
  4) Logical      If False, Jump to 29
  5) Integer      Household.PERSONS
  6) Integer      3
  7) Relation     GT
  8) Logical      If False, Jump to 12
  9) Integer      13
 10) Return       Integer
 11) Logical      Jump to 28
```

```
12) Integer      Household.AGELT5
13) Integer      1
14) Relation     EQ
15) Logical      If False, Jump to 19
16) Integer      12
17) Return       Integer
18) Logical      Jump to 28
19) Integer      Household.AGE5TO17
20) Integer      1
21) Relation     EQ
22) Logical      If False, Jump to 26
23) Integer      11
24) Return       Integer
25) Logical      Jump to 28
26) Integer      10
27) Return       Integer
28) Logical      Jump to 45
29) Integer      Household.PERSONS
30) Integer      2
31) Relation     EQ
32) Logical      If False, Jump to 43
33) Integer      Household.AGELT5
34) Integer      1
35) Relation     EQ
36) Logical      If False, Jump to 40
37) Integer      6
38) Return       Integer
39) Logical      Jump to 42
40) Integer      5
41) Return       Integer
42) Logical      Jump to 45
43) Integer      3
44) Return       Integer
45) End
```

Fare Data Report

The transit fare table is often defined using ranges of fare zones, transit modes, and time periods. The **Router** expands these values into unique combinations of boarding fare zone, alighting fare zone, previous transit mode, current transit mode, time period, and fare class. This report prints the fare for all of these unique combinations. If the fare policy is complex, the size of this report can be huge. A few sample lines generated from the transit fare example included above are shown below.

```
Transit Fare Data Report

 From   To          From          To                          Fare   Fare
 Zone  Zone         Mode         Mode        Time Period      Class  Cents

    1     2         NONE    RAPIDRAIL         0:00..7:00       CASH    100
    1     2         NONE    RAPIDRAIL         7:00..9:30       CASH    100
    1     2         NONE    RAPIDRAIL        9:30..15:00       CASH    100
    1     2         NONE    RAPIDRAIL       15:00..18:00       CASH    100
    1     2         NONE    RAPIDRAIL       18:00..24:00       CASH    100
    1     2         NONE   REGIONRAIL         0:00..7:00       CASH    100
```

```
1      2          NONE   REGIONRAIL        7:00..9:30       CASH    100
1      2          NONE   REGIONRAIL       9:30..15:00       CASH    100
1      2          NONE   REGIONRAIL      15:00..18:00       CASH    100
1      2          NONE   REGIONRAIL      18:00..24:00       CASH    100
1      2           BUS          BUS        0:00..7:00       CASH     25
1      2           BUS          BUS        0:00..7:00    SPECIAL     25
1      2           BUS          BUS        7:00..9:30       CASH     35
1      2           BUS          BUS        7:00..9:30    SPECIAL     25
1      2           BUS          BUS       9:30..15:00       CASH     25
1      2           BUS          BUS       9:30..15:00    SPECIAL     25
1      2           BUS          BUS      15:00..18:00       CASH     25
1      2           BUS          BUS      15:00..18:00    SPECIAL     25
1      2           BUS          BUS      18:00..24:00       CASH     25
1      2           BUS          BUS      18:00..24:00    SPECIAL     25
1      2           BUS      EXPRESS        0:00..7:00       CASH     25
1      2           BUS      EXPRESS        7:00..9:30       CASH     25
1      2           BUS      EXPRESS       9:30..15:00       CASH     25
```

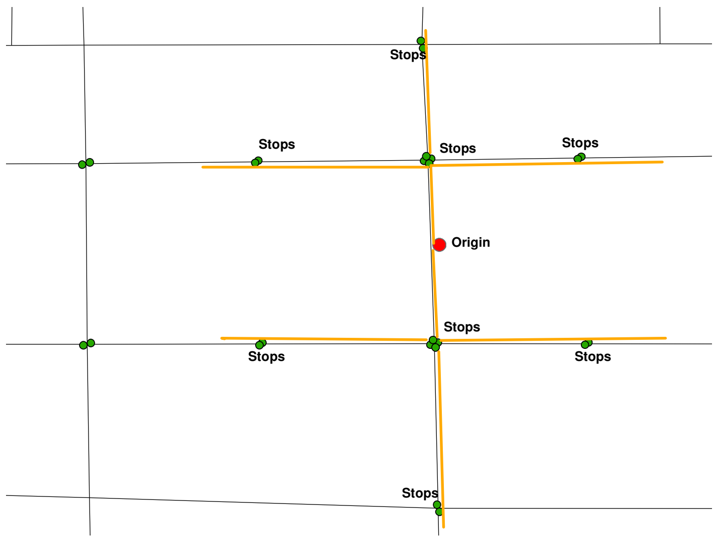
Algorithm Notes

This program uses a vine-based path building procedure for drive trips and a node-based path building procedures for walk, bike, and transit trips. The vehicle type restrictions defined for each link are used to limit the links considered by auto, truck, walk, and bicycle trips. Walk and bicycle trips are allowed to travel the wrong direction on a one-way street. Transit trips are limited by the transit stops, routes and schedules. Process links are used to connect parking lots and transit stops to activity locations.

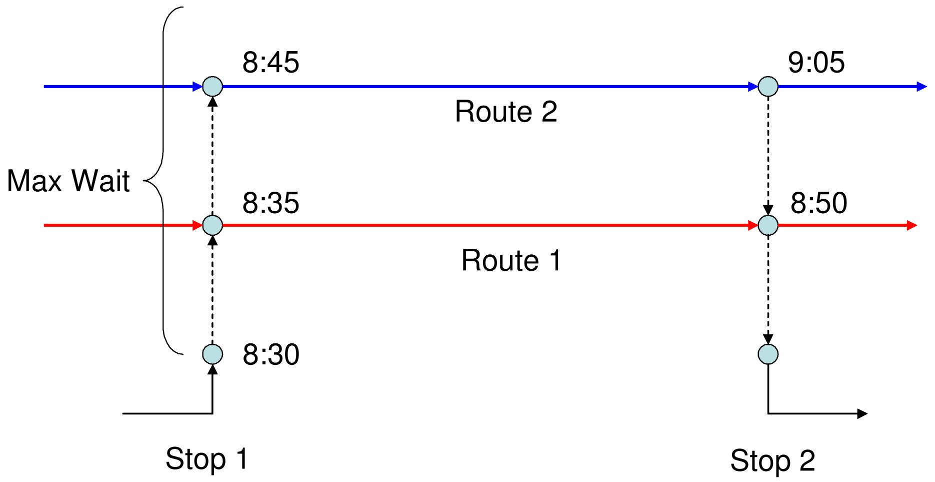
In networks that include turn restrictions, a vine-based algorithm is able to build paths that pass through the same intersection more than once. In the TRANSIMS context, turn restrictions are represented as link combinations where no lane connectivity is defined. Turn restrictions do not apply to the walk or bicycle networks. Path building on these networks use a node-based algorithm for computational efficiency.

Transit Path Building

Transit paths are built using boarding and alighting legs, schedule departure times, and transfer stages. The process begins by walking as far as possible within the time and distance constraints imposed by the user. As transit stops are found on these paths, they are added to a stop queue. If the traveler is able to walk all of the way to the destination, the destination cumulative impedance is set and considered by all subsequent travel options.



When the walk options are exhausted, the program considers all of the route legs leaving all of the stops in the stop queue. The first scheduled departure of each route leaving each stop is considered as long as the first departure is less than the maximum wait time from the stop arrival time. The program then considers each alighting option given the boarding vehicle. The in-vehicle travel time is calculated as the difference in the departure time of this vehicle from the boarding and alighting stops. The transit fare is determined from the boarding fare zone, the alighting fare zone, and the transit mode.



The leg impedance is added to the cumulative impedance at the boarding stop to determine the impedance at the alighting stop. If this impedance is less than the best destination impedance and less than any previous path to the alighting activity location, the alighting activity location is added to the walk queue for consideration during the next stage of the trip.

If transfers are permitted, the program will repeat the process of building walk paths away from each alighting activity location and add new stops to the boarding queue. After the last transfer the program considers walk options to reach the final destination.

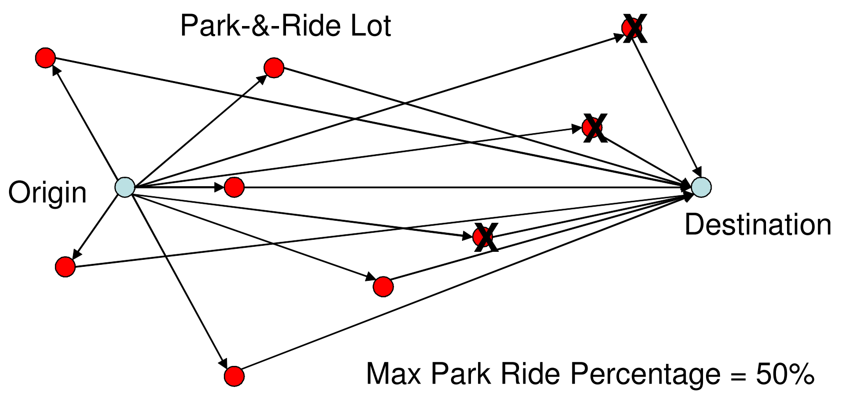
Park-&-Ride and Kiss-&-Ride Paths

Two types of park-&-ride and kiss-&-ride trips are built by the **Router**. Mode 5 is a park-&-ride trip that starts by driving to a park-&-ride lot to take transit to the final destination. Mode 6 is the return trip that takes transit to the park-&-ride lot where the vehicle is parked and then drives to the final destination. Whereas mode 6 trips must know where the vehicle is parked and can travel directly to the parking lot, mode 5 trips must select the best park-&-ride lot for the given trip.

Kiss-&-ride trips work the same way. Mode 13 is a kiss-&-ride trip that starts with the person as a passenger in a vehicle located at the trip origin and drives to a kiss-&-ride lot to take transit to the final destination. Mode 14 is the return trip that takes transit to the same kiss-&-ride lot to be driven to the final destination. Whereas mode 14 trips must know where to meet the vehicle, mode 13 trips must select the best kiss-&-ride lot for the given trip.

The park-&-ride and kiss-&-ride lot selection process is based on the relative location of each lot to the origin and destination of the trip. The straight-line distance from the origin to the lot is compared with the straight-line distance from the lot to the destination to determine the likelihood that the traveler would consider a particular park-&-ride or kiss-&-ride lot for a given trip. The MAX_PARK_RIDE_PERCENTAGE and MAX_KISS_RIDE_PERCENTAGE parameters establish the upper limit of the drive portion of the trip to the total trip length (as measured by the straight-line segments). The program considers all park-&-ride or kiss-&-ride lots that meet this criterion. The lot that provides the minimum total impedance for the drive and transit legs of the trip will be selected.

The program builds the minimum impedance drive path to each of these park-&-ride or kiss-&-ride lots. The impedance and travel time to each lot is considered by the transit path building algorithm that builds paths from each lot to the final destination. The lot with the minimum total impedance is selected. The vehicle is driven to the park-&-ride lot and must therefore be retrieved from this lot before this traveler can make any subsequent drive trips. This is typically done using a mode 6 trip to retrieve the vehicle.



For kiss-&-ride trips, a vehicle is driven to the kiss-&-ride lot and waits for the return trip, but is not officially "parked" at the parking lot. This trip is not assigned parking cost or parking penalties. In activity-based applications, the driver will typically be another household member who will use the vehicle for other activities.

Parking Lot Access Options

For a drive trip the traveler must move from the activity location to the parking lot where the vehicle is located. Normally, this is a parking lot that is connected to the activity location by a process link. The traveler will then park the vehicle at a parking lot attached to the destination activity location. If the LIMIT_PARKING_ACCESS key is "false", the program builds a walk path between the activity location and the parking lot. At the destination end of a drive trip, the program selects up to ten near-by parking lots to consider. The parking lot with the best total impedance to the destination is selected. This permits the program to consider lots that may have lower parking costs. The trade-off between parking cost and walk time is defined by the relative impedance values. Note that the default value of cost is zero, which means cost is not considered by the path building process.

Impedance Calculations

The path with the minimum cumulative impedance between the origin and destination activity locations whose total travel time fits within the specified activity schedule is saved to the output plan file. Link impedance is defined as the weighted combination of travel time, distance, and cost. Travel time is broken down in to walk time, wait time, and in-vehicle time. Separate weights are provided for each component of impedance.

The program also permits the user to include random impedance variance. The random variance is calculated each time the path building process considers a link. The percent random impedance parameter defines the minimum and maximum variance. The equation is:

$$\text{Impedance} = \text{impedance} * (1 + \text{percent} * (\text{probability} - 0.5) \backslash 100)$$

Impedance may also vary by time-of-day. If a link delay file is provided or link delay updates are enabled, the program keeps track of travel times on each link by time period. In most cases, 15 minute time periods are used. The travel time used for a particular link is based on the time-of-day that the path enters the link.

The activity file defines lower and upper bounds for the start and end of each activity. This program assumes the trip starts at the mid-point of the lower and upper bounds of the previous activity's end time. The trip must be completed before the upper bound of the start time of the current activity. If the trip cannot be completed in time, a time schedule error is recorded and no travel plan is generated. The program does not check if the trip arrives at the destination before the lower bound of the start time.

Output Plan Records

When a plan is generated, it typically includes at least five components. The first component is the initial at-home activity. At the end of this activity a trip is generally needed to travel to the next activity. A vehicle trip is stored with three legs. The first leg is the walk trip from the origin activity location to the parking lot where the vehicle is located. The second leg is the in-vehicle component from the origin parking lot to a parking lot attached to the destination activity. The program can consider up to ten parking lots attached to the destination activity. The lot that results in the minimum total impedance to the destination activity location is selected. The last leg is the walk from the parking lot to the destination activity location.

Transit plans are considerably more complex. They generally begin by walking to a transit stop. This is following by a transit boarding leg that includes the process link delay and cost values plus the waiting time for the transit vehicle. The next leg is the transit in-vehicle leg between the boarding and alighting stops of a particular route. This leg will contain the in-vehicle time and the transit fare. The alighting leg only includes the process link delay and cost values from the alighting stop to the activity location. Then the trip may walk to the final destination, walk to another transit stop, or board another route at the current stop.

Park-&-Ride plans will include all of the components of a drive trip and a transit trip. All walk, bicycle, and drive access legs include a full enumeration of all of the links or nodes used by the path.

The program permits the user to store the path between the origin and destination in one of two ways. At present, the TRANSIMS Microsimulator can only work with node list plans. A node list plan records the network nodes between the origin and destination parking lots or activity locations. If the parking lots are located on the same link, no nodes are included in the path. Since the TRANSIMS Microsimulator cannot process a drive plan with no nodes, this program records a zero-node error message.

The link list plan avoids this problem by listing the link numbers between the origin and destination parking lots or activity locations. This means that at least one link number is included in the plan even if the parking lots are on the same link. Positive and negative link numbers are used to indicate the direction of travel on the link. A positive link number represents travel in the A→B direction. A negative link number represents travel in the B→A direction.

V/C Delay Method

As mentioned earlier, time-of-day impedances can be based on link delay updates. A volume to capacity ratio relationship is used to adjust the travel time in each period based on the number of travelers that utilize the link. If a link delay file is provided, the volumes and travel times are initialized with information found in the file. If a file is not provided, or the file does not include data for all links and time periods, the travel time for each time period is based on the free flow speed.

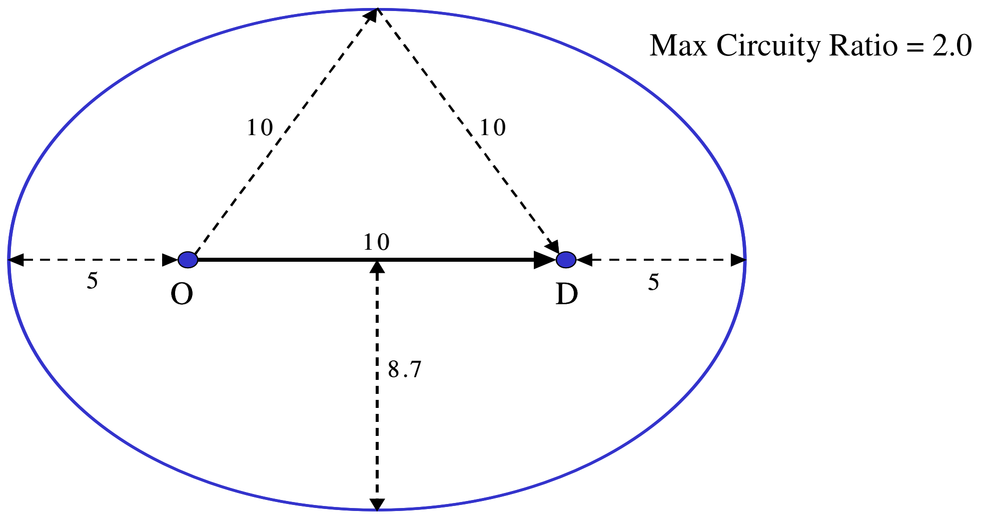
As the path for each trip is written to the plan file, the program increments the link volume for the appropriate time period. When the number of trips reaches the link delay update rate value, the program updates the travel times on each link and time period based on the total volume and the volume-delay equation. The equation assigned to the facility type is used to calculate the new travel times. These times are used to construct the next set of paths. If this program is being executed with partitioned data, it is important to note that the volume-delay calculation is based on the sum of the input volumes and the trips included in this particular partition. It does not include the trips loaded to the link by other partitions.

The **Router** builds a new path for each traveler based on the latest travel times. Only travelers from households in the household list are processed. The travelers are processed in the order they are found in the activity file. The order of the households in the household list does not impact the order of processing. The **Router** also does not read the vehicle or activity index files.

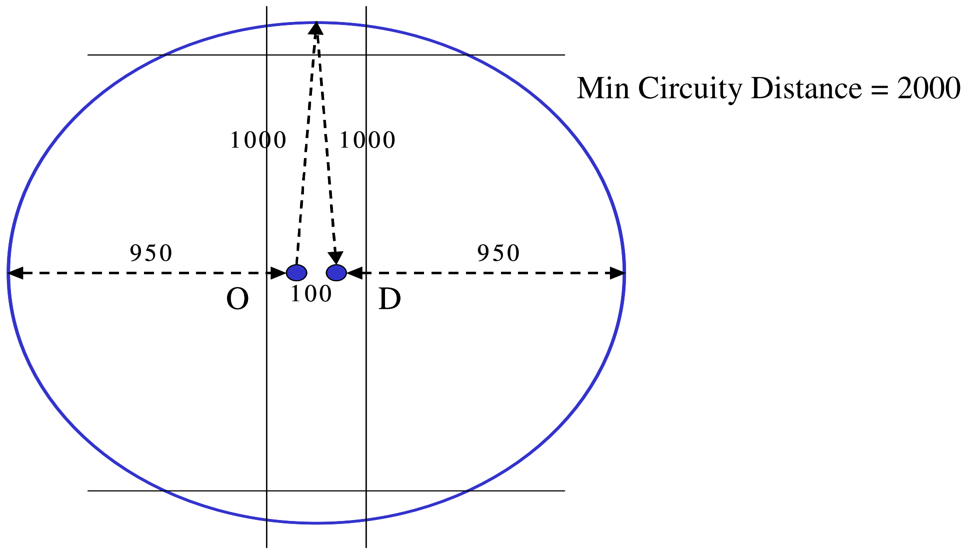
Path Circuity Restrictions

Since a new path is constructed for each traveler, minor efficiencies in the path-building algorithm can have significant impacts on the overall processing time. The circuity parameters are one way the user can focus the algorithm on reasonable options and thereby reduce the processing time. These parameters improve performance by limiting the number of links the program considers to those links that are generally between the origin and destination activity locations.

For most trips, the maximum circuity ratio controls the area the program uses to construct the minimum impedance path. The default value of 2.0 generates the search radius shown in the following graphic. If the straight-line distance between the origin and destination is 10 kilometers, the algorithm will consider links that are as much as 5 kilometers in the wrong direction or 5 kilometers beyond the destination. At the mid-point between the origin and destination, the link can be as much as 8.7 kilometers away from the direct path between the origin and destination. This is a generous search radius for most networks.

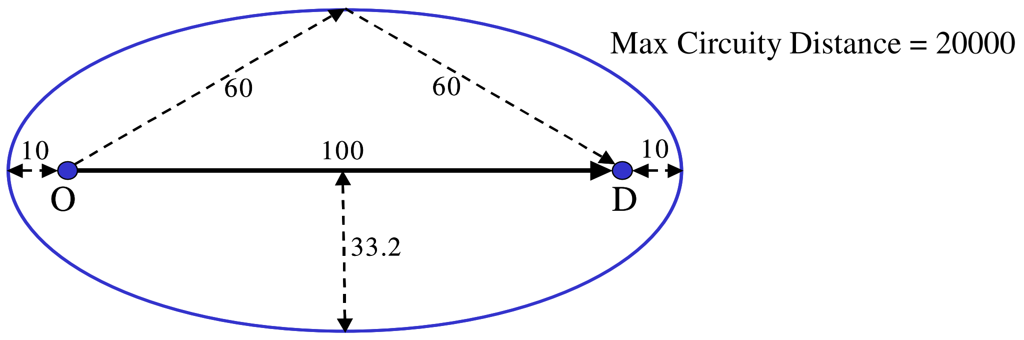


There are, however, situations where the ratio-based search area is too restrictive or too generous. In these situations, the minimum circuity distance or the maximum circuity distance can help minimize the impacts of the ratio method on the path building process. The minimum



circuity distance increases the search area for short trips and the maximum circuity distance reduces the search area for long trips. The following example shows how the default minimum circuity distance of 2000 meters helps to avoid path-building problems between activity locations that are physically close, but not as close given the roadway network.

If the trip is long, the ratio method tends to generate a search area that is inefficiently large. The maximum circuity distance parameter helps to focus the search area more closely. The default value is 20,000 meters or 20 kilometers. The example below shows the impact of this parameter on an origin and destination that are 100 kilometers apart. The algorithm restricts the search area to 10 kilometers in the wrong direction and 10 kilometers beyond the destination. At the mid-point between the origin and destination the path can be as much as 33.2 kilometer away from the direct path between the origin and destination.



Path Building Problem Messages

If a problem is encountered during the path building process, a message is written to the Problem file and added to the summary statistics. The most likely explanation for a given problem message is listed below:

1. *Path Building Problem* – This message typically means that there is no feasible path between the origin and destination. It could be caused by lane connectivity or one-way street conditions or by a network coding error.
2. *Time Schedule Problem* – This message indicates that the trip travel time exceeded the upper bound of the activity start time. It could be caused by excessive congestion or no path options.
3. *Zero Node Path Problem* – The zero-node error occurs when the origin and the destination activity locations lie on the same link and node list plans are requested.
4. *Vehicle Type Problem* – The origin parking lot is located on a link that does not permit the corresponding vehicle type. This most often occurs when autos are loaded to transit only links or trucks to auto only links.
5. *Path Circuity Problem* – A circuity error indicates that the path building process was limited by one or more of the circuity parameters. It either means that a path does not exist or the path is highly circuitous. The user can set the maximum circuity ratio parameter to zero to eliminate these messages. If a path does not exist, a path building or time schedule message will be generated.
6. *Travel Mode Problem* – The **Router** records a travel mode error when the mode on the activity file cannot be built. This generally means that the transit, walk, or bike networks have not been enabled.
7. *Vehicle Access Problem* – An access error is generated when the vehicle listed in the activity file is not found in the vehicle file or when the vehicle is located at a parking lot that is not attached to the activity location with a process link.
8. *Walk Distance Problem* – This message is generated when the cumulative walk distance required by the path exceeds the MAX_WALK_DISTANCE parameter.
9. *Wait Time Problem* – This message indicates that potential transit routes exist, but the wait time required to board the routes exceed the MAX_WAIT_TIME parameter.
10. *Walk Access Problem* – This message is generated when the link associated with the origin or destination activity location does permit travel by the chosen mode. It most often indicated a walk or bike access restriction at one of the trip ends.
11. *Path Size Problem* – This message is generated when the number of links or nodes in the path exceeds the maximum path size. The maximum path size parameter is defined in the Network Size file and defaults to the number of network nodes. Assuming the maximum path size is a relatively large number, this error typically indicates a problem in the path-building algorithm that resulted in an infinitely looping path. This can only occur if the percent random impedance is high. Re-running the **Router** will most likely eliminate the problem.

12. *Park & Ride Lot Problem* – In order to building a park-&-ride trip (mode 5), the must be a parking lot designated with the PARKRIDE style in the general proximity of the trip origin. The MAX_PARK_RIDE_PERCENTAGE parameter determines how far away from the origin the software can search for possible park-&-ride lots. If no lots are found within the search area, the park-&-ride lot error message is recorded.
13. *Bicycle Distance Error* – This message is generated when the bicycling distance exceeds the MAX_BICYCLE_DISTANCE parameter.

Sample Printout

The printout file generated by the **Router** will look something like the example below. It is an ASCII text file with a maximum of 95 characters per line and 65 lines per page. The file can be viewed or printed using a variety of text editors. For best results in a word processor, use a 10-point Courier font and 0.5 inch margins on all sides.

```
********************************************
|                                          |
|          Router - Version 4.0.63         |
|   Copyright (c) 2009 by AECOM Consult    |
|          Mon Apr 19 13:11:47 2010        |
|                                          |
********************************************

Control File = ..\control\1.Alex.2000.Act.Router.ctl
Report_File  = ..\control\1.Alex.2000.Act.Router.prn (Create)

Route the Highway and Transit Trips for 1.Alex

Project Directory = ..\

Default File Format = TAB_DELIMITED

Network Directory = ..\network
Node File = ..\network\Node
Link File = ..\network\Link
Lane Connectivity File = ..\network\Lane_Connectivity
Parking File = ..\network\Parking_2
Activity Location File = ..\network\Activity_Location_3
Process Link File = ..\network\Process_Link_2
Transit Stop File = ..\network\Transit_Stop
Transit Route File = ..\network\Transit_Route
Transit Schedule File = ..\network\Transit_Schedule

Vehicle File = ..\demand\Alex.2000.Act.Vehicles
Vehicle File will be Sorted by Vehicle ID

Activity File = ..\demand\Alex.2000.Act.Activities

Time of Day Format = 24_HOUR_CLOCK

New Plan File = ..\demand\1.Alex.2000.Act.TravelPlans
Plan File will contain Node List Paths

Access to Parking Lots will be Limited to Process Links
Walk Path Details will be written to the Plan File
Activity Schedule Constraints will be Ignored
Routing Errors will Terminate Plan Construction

Random Impedance Adjustments are Disabled
```

```
Walk Speed = 1.00 meters per second
Bicycle Speed = 4.00 meters per second

Walk Time Value = 20.00 impedance units per second
Bicycle Time Value = 15.00 impedance units per second
First Wait Time Value = 20.00 impedance units per second
Transfer Wait Time Value = 20.00 impedance units per second
Vehicle Time Value = 10.00 impedance units per second
Distance Value = 1.00 impedance units per meter
Cost Value = 5.00 impedance units per cent

Left Turn Penalty = 300.00 impedance units per left turn

Transfer Penalty = 1200.00 impedance units per transfer
Stop Waiting Penalty = 0.00 impedance units per boarding
Station Waiting Penalty = 0.00 impedance units per boarding
Bus Bias Factor = N\A impedance factor
Bus Bias Constant = N\A impedance units
Rail Bias Factor = N\A impedance factor
Rail Bias Constant = N\A impedance units

Route the Highway and Transit Trips for 1.Alex
Wed Jan 20 13:35:18 2010  Router  page 2


Maximum Walk Distance = 2000 meters
Maximum Bicycle Distance = 10000 meters
Maximum Waiting Time = 60 minutes
Minimum Waiting Time = 60 seconds
Maximum Number of Transfers = 3
Maximum Number of Paths = 4
Maximum Park-&-Ride Percentage = 50
Maximum Kiss-&-Ride Percentage = 35
Kiss-&-Ride Time Factor = 2.50
Kiss-&-Ride Stop Types = STOP  STATION  EXTERNAL
Maximum Kiss-&-Ride Dropoff Walk Distance = 100 meters
Waiting Time will be Added to the Access Leg
Parking Hours by Trip Purpose = 8.50 2.50 1.00 1.00 hours

Link Delay Updates are Disabled

New Problem File = ..\results\1.Alex.2000.Act.Problems
New Problem File Format = VERSION3
Maximum Number of Routing Problems = 100000


Number of Node File Records = 2573

Number of Link File Records = 3607
Number of Directional Links = 6774

Number of Lane Connectivity File Records = 16141
Number of Lane Connectivity Data Records = 14113

Number of Parking File Records = 7736

Number of Activity Location File Records = 8656

Number of Transit Stop File Records = 953

Number of Process Link File Records = 17378

Number of Transit Route File Records = 1377
Number of Transit Route Data Records = 20

Number of Transit Schedule File Records = 60441

Number of Vehicle File Records = 174389
Number of Links in the Walk Network = 3417
Number of Links in the Bicycle Network = 3401
Number of Directional Links in the Highway Network = 6742

Number of Highway Link Connections = 14054

Number of Park-&-Ride Lots = 2

Number of Kiss-&-Ride Lots = 2374

Number of Activity Records = 1133718
Number of Activities Processed = 1075924

Number of Households Processed = 78296
Number of Vehicle Trips Saved = 590986

Route the Highway and Transit Trips for 1.Alex
Wed Jan 20 13:38:59 2010  Router  page 3


Number of Output Plans = 3215887
Number of Output Records = 21305580
Number of Output Travelers = 188623
Number of Output Trips = 599748

Percent of Total Trips with Problems = 1.7%

Total Number of Problems = 14499
Number of Path Building (#1) Problems = 244 (1.7%)
Number of Zero Node (#3) Problems = 2087 (14.4%)
Number of Vehicle Access (#7) Problems = 1813 (12.5%)
Number of Walk Distance (#8) Problems = 10196 (70.3%)
Number of Bike Distance (#13) Problems = 16 (0.1%)
Number of Access Restriction (#25) Problems = 30 (0.2%)
Number of Vehicle ID (#31) Problems = 113 (0.8%)

Mon Apr 19 13:11:47 2010 -- Process Complete (0:03:41)
```