Chapter 7

Feedback Processes

CONTENTS

Feedback Processes

Chapter Outline

TRANSIMS consists of a series of modules that produce synthetic households, activities for individuals within these households, the choice of routes for movements among these activities, and the microsimulation of these movements to create traffic dynamics on the network. The TRANSIMS framework allows each module to be executed in any desired order by a set of scripts specified by the user in the Feedback Controller.

TRANSIMS is first and foremost a model framework. The information generated by the Population Synthesizer, Activity Generator, Route Planner, and Traffic Microsimulator can be passed between these modules and user-developed modules to model travel demand behaviors and transport system responses. Feedback enables the overall computational system to reflect "learned" behavior within the simulated population represented. Information about each individual's experiences may be used to select those with sub optimal choices and feed them back to another module to estimate the costs of other choices

Feedback may also be used during model development and calibration. Network coding errors, survey record errors, poorly selected activity sets and various travel choice problems can be identified and even corrected using iterative feedback techniques.

Figure-7.1 shows main functions of the Feedback Module within the TRANSIMS framework and its relations to other modules.

This chapter begins with an overview of the Feedback Module, then discusses the terminology involved in the Feedback Module and provides the key concepts used in the process. It also highlights the major inputs and outputs to the module, including the various module interfaces, the values for relevant configuration files and the detailed discussion of the model tools used in this module.

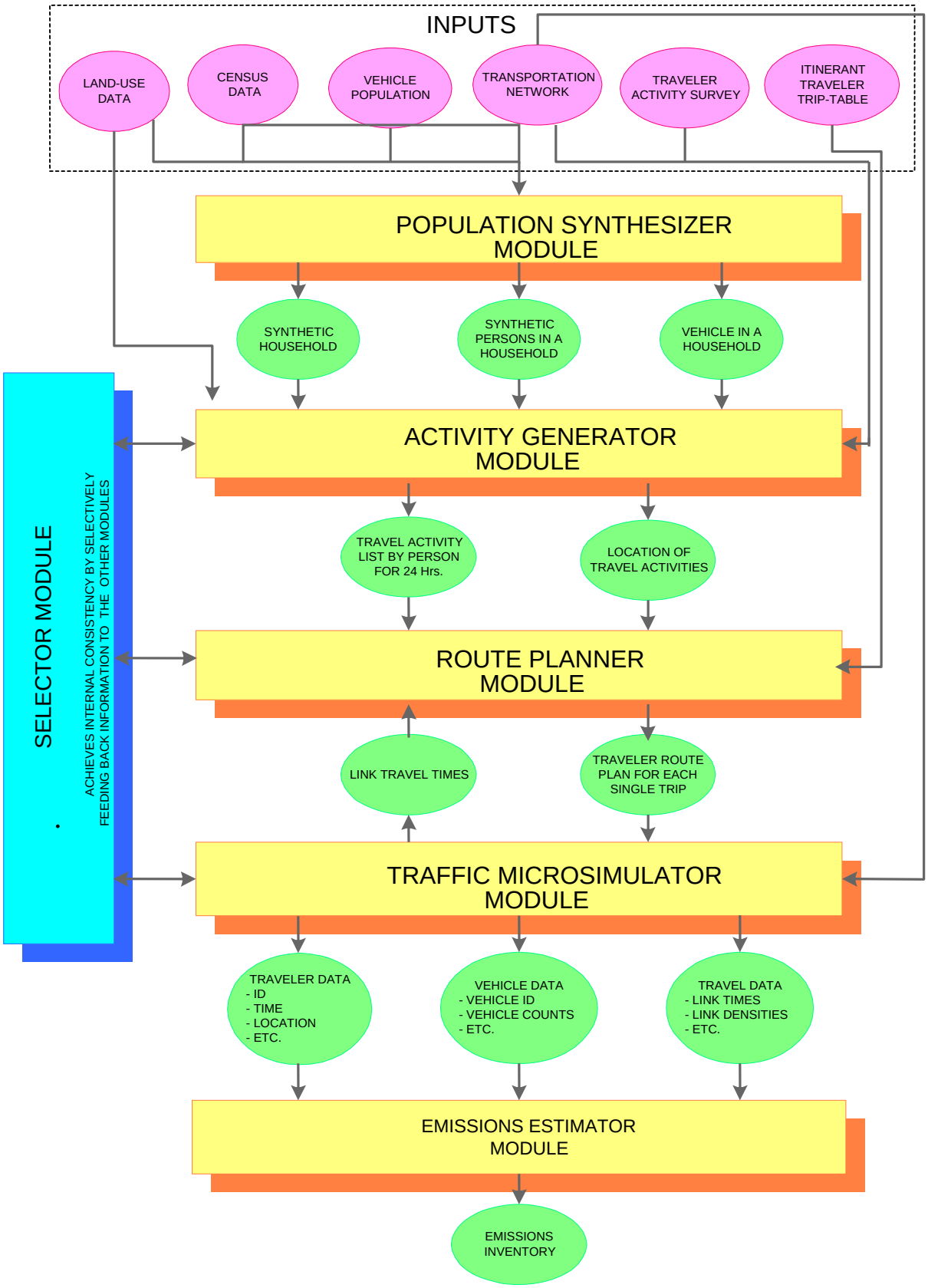

Figure-7.1: TRANSIMS framework

7.1 Overview

A key distinguishing feature of TRANSIMS is the process known as iterative feedback. Feedback provides a natural way to tailor models to specific, possibly overlapping, subpopulations. The Feedback module uses an iteration script provided by the user to control the overall TRANSIMS Framework. A typical TRANSIMS study involves repeated iteration between modules. There is no single, “standard” iteration script because different study designs involve different iteration schemes.

One important example of feedback is in solving the traffic assignment problem. The simplest version of this uses a loop between the Route Planner and Traffic Microsimulator modules to find out the actual travel times from the Microsimulator and consequently feed them back to the Route Planner to find the shortest routes for the travelers in the network. On the first iteration of the Route Planner, routes are chosen under the hypothesis that travel times are represented by the free speeds on the network. Of course, there are many more information flows in TRANSIMS than just the travel time table. Every TRANSIMS module can be used to update a selected subpopulation using information provided by the feedback process. In effect, this is like providing a separate model for every conceivable subdivision of the population without the need for fitting each model separately. For example, work location is chosen using a single simple model for the entire population. If people who commute by bus across a river are assigned work locations poorly, selecting that subpopulation and running the work location assignment model with slightly different input information can change the poorly selected locations for that subpopulation with no change in the model itself.

The structure of the Feedback Module is shown in Figure-7.2. It has three modeling tools, Collator, Stratifier and Selector.

The Collator gathers data from some input files of TRANSIMS, i.e., Network file and Transit Route file. It also extracts data from the output files of the Population Synthesizer, Activity Generator, Route Planner, and Traffic Microsimulator. All data collected are then put into the Collator Iteration Database.

The Stratifier divides trips into Binnings and Stratifications based on user-defined criteria. The numeric identifier of each Binning and Stratification is added to the iteration database and the database is developed into the Stratifier Iteration Database.

The Selector model’s responsibility is to pick up a subset of travelers from the Stratifier Iteration Database. This is done in two steps. In step 1, the targeted cells are picked up from the stratification, where a subset of travelers or trips is selected within the cell.

In step 2, the output of the Selector module on each selected subset will consist of one Activity Feedback file and one Route Planner Feedback file. The Activity Feedback file is then sent to the Activity Regenerator, and the Route Planner feedback file is sent back to the Route Planner. The user is free to use either or both of them according to his/her intention.

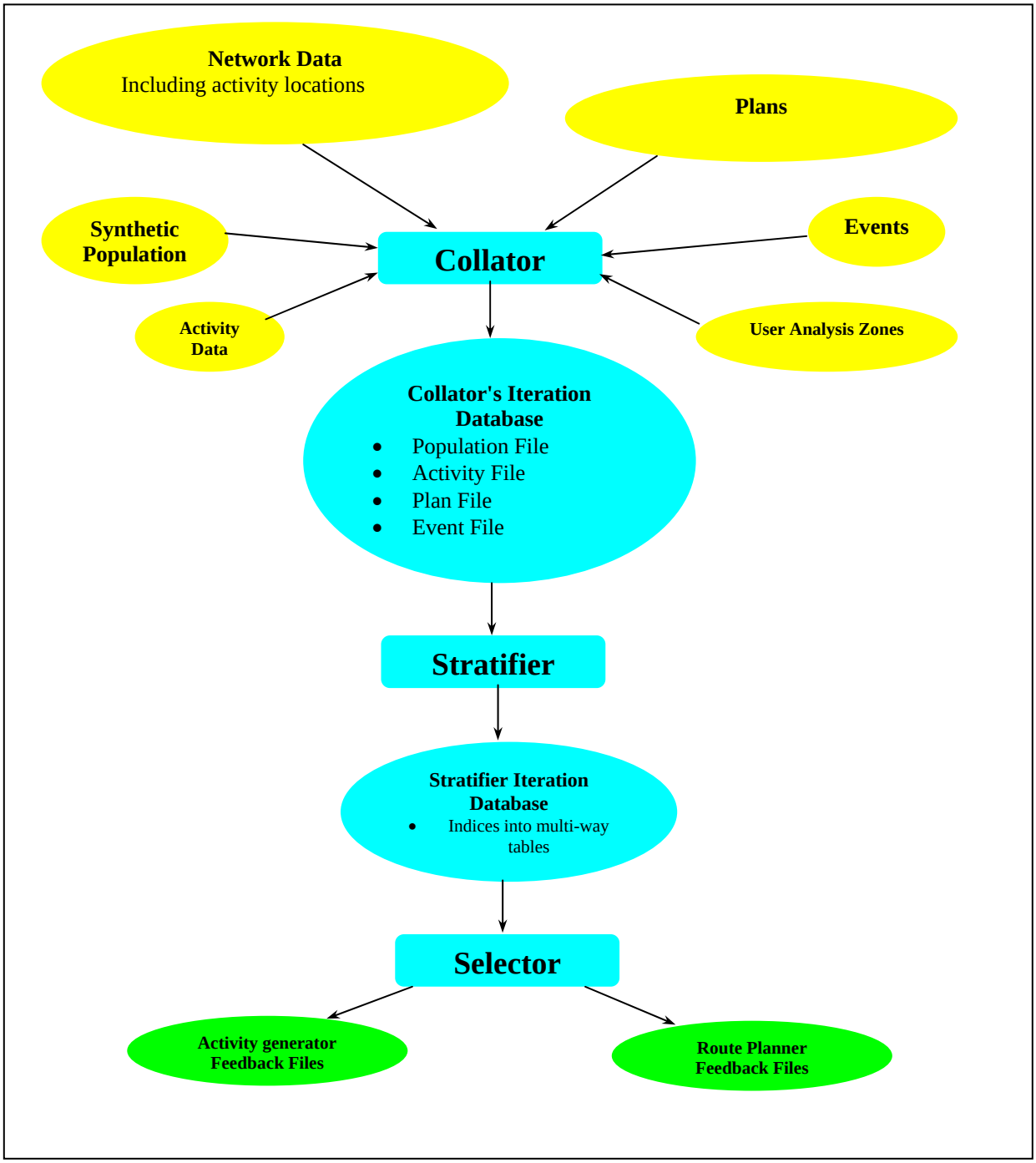

Figure-7.2: Data flow diagram of the TRANSIMS Feedback Module

7.2 Terminology

- **Iteration:** Execution of one TRANSIMS program (e.g., Activity Generator, Route Planner, Traffic Microsimulator).

- **Iteration Database:** The archive of information about travelers across iterations. It is used by the Selector to make its selection decisions. It enables user to filter results and run repeated iterations.

- **Iteration Script**: A programmatic approach to implementing a model design. The iteration script controls process and information flows within a TRANSIMS model.

- **Collator:** One of the tools providing Selector/Iteration Database functionality. Its main function is to gather data from disparate sources (e.g., activity files, plan files, event files) into a single table keyed by traveler ID and trip number. It accumulates data over an entire trip and provides some commonly used processing algorithms. It can be run after each module and will fill in all of the fields in Iteration Database that depend on that module with the most recent data available.

- **Stratifier:** One of the tools providing Selector/Iteration Database functionality. It uses a combination of built-in algorithms on the information contained in ITDB to stratify or classify trips.

- **Selector:** One of the tools providing Selector/Iteration Database functionality. Its goal is to either reroute the traveler or to make use of one of the feedback pathways defined by the Activity Generator. It uses the Iteration Database to select a set of travelers. It has a set of algorithms and each algorithm may require its own set of parameters and is associated with a name, goal and a cost function.

- **Tour:** This concept is only used in the Selector Module for the user's convenience. Each trip starting from the home location is defined as the start of a new tour.

- **Subtour:** This concept is only used in the Selector Module for the user's convenience. Each trip from an anchor activity that returns to the same anchor location before returning to the home location is defined as the start of a new subtour.

- **Binning:** Each discretized variable is known as a "binning". Discretization is implemented in the Stratifier and binnings are added to the Iteration Database. Each binning is given a numeric identifier.

- **Bin:** During discretization, each item in a binning is called a “bin”. Each bin is given a numeric identifier. Discretization is implemented by the Stratifier.

- **Stratification:** A k-way table created by the Stratifier from the binnings. Each stratification is given a numeric identifier. Each cell in the k-way table is given an index and represents a group of travelers with similar preferences, demographics and/or experiences.

- **Selector Choices:** Files that list the travelers who will be reassigned activities, re-planned, re-simulated, etc.

- **Selector Statistics:** Statistics that provide a basic summary of the choices the Selector makes.

- **Feedback Controller:** This controls the *Collator*, *Stratifier* and *Selector* tools. It reroutes selected travelers, updates travel times, regenerates activities and microsimulates all travelers.

7.3 Key Concepts

The Iterative Process

The most important function of the Feedback Module is the iterative process. During each iteration, the user invokes the Collator, Stratifier, and Selector to do the following:

- Read information about the travelers from the Iteration Database.
- Examine each traveler and decide whether to:
  - Re-generate his or her activities using the Activity Generator,
  - select a new route between his or her existing activities using the Route Planner, or
  - retain his or her existing activities and the planned route between them.
- Write the selections made for each traveler into data files that when executed can be read by the Activity Generator and the Route Planner.

Figure-7.3 illustrates a step in the iterative process conducted by the Selector module.

After the Selector completes the selection process for all travelers, the Activity Generator, Route Planner, or Traffic Microsimulator runs to calculate the updated activity set, plan set, or microsimulation output files, respectively (according to the decisions made by the Feedback Controller).

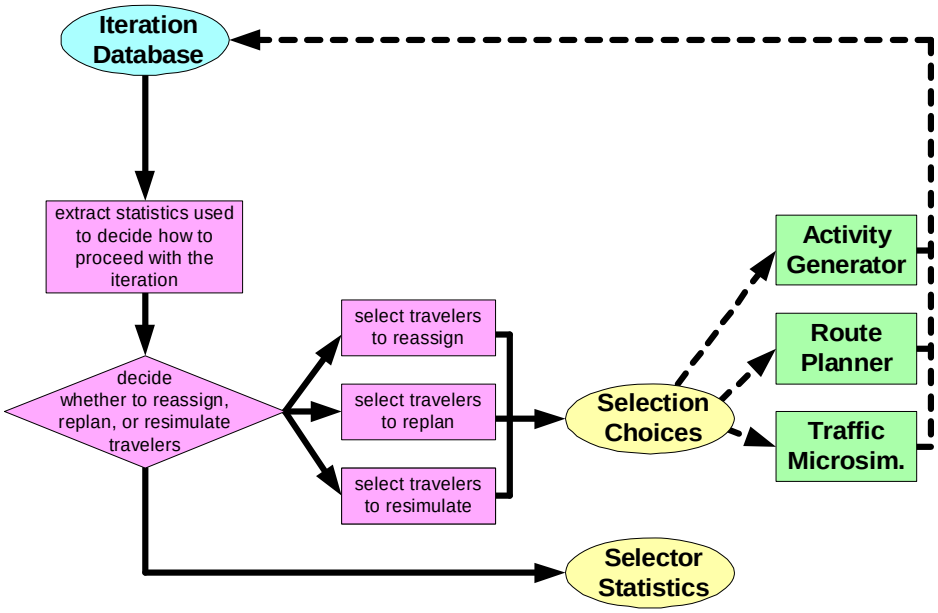

Figure-7.3: Typical Selector/Iteration Database logic

The iteration script will re-invoke a Selector at the start of the next iteration in the study. Figure-7.4 shows an example of one possible progression, as determined by the Selector.

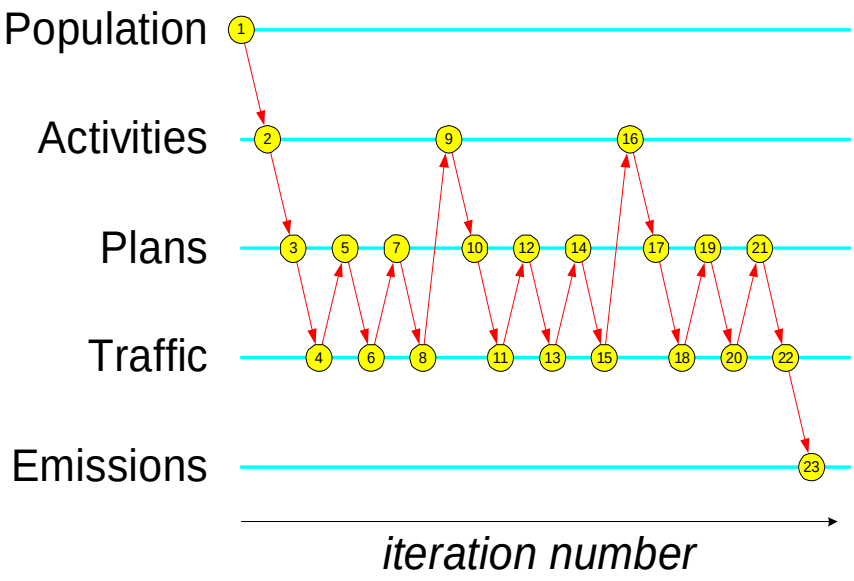

Figure-7.4: An example of iteration progressions

The Iteration Database is the archive of information about travelers across iterations. The Selector uses this information to make selection decisions and they are chosen by the user from:

- The population, activity, and plan files fields—for example, income, mode preference, or the expected duration of a trip.
- Information extracted from detailed Traffic Microsimulator event output—for example, the actual duration of a trip.
- Information deduced from combinations of the previous two—for example the duration of a trip relative to its expected duration.

The left side of Figure-7.5 shows this data flow into the Selector/Iteration Database.

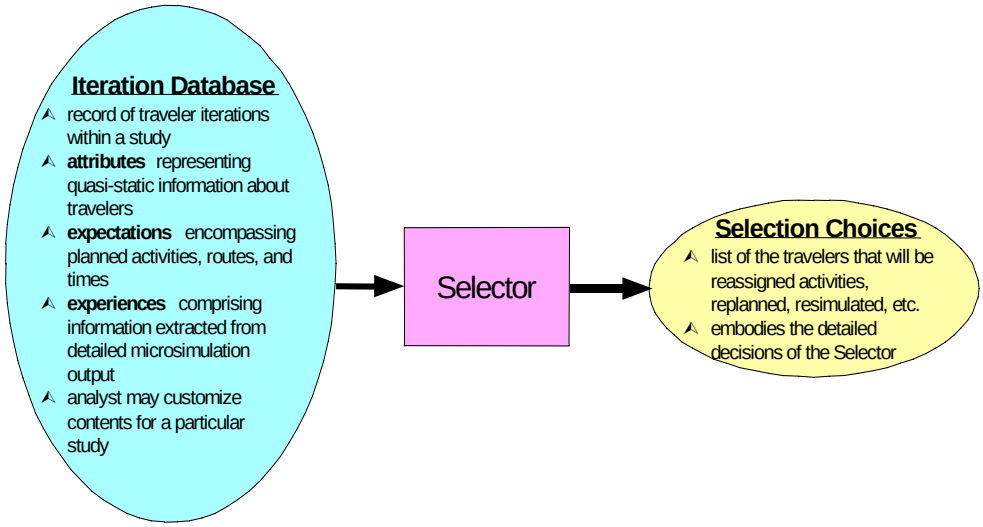

Figure-7.5: Typical Selector/Iteration Database data flow

TRANSIMS Framework Flexibility

The Feedback Module is very flexible and user-controlled. It provides information pathways and tools for manipulating information. Users have to design the order in which the modules are called and how the feedback process is executed.

The Framework's flexibility allows for countless variations in the iteration process. For example, in some studies, the Selector may run after the Activity Generator or Route Planner completes its execution. Thus, the Selector can decide which of the generated activities or plans will be accepted for travelers. Those not accepted are discarded and new activities or plans are produced.

The iteration script has the potential to make additional choices, such as the following:

- which version of the Activity Generator, Route Planner, or Traffic Microsimulator will run during the present iteration;
- if transit schedules will be adjusted or vehicles added or removed from the transit fleet;
- if network characteristics (such as traffic signal timing, congestion pricing, or roadway information signs) will be altered;
- which travelers receive data from traffic information systems; or
- whether to complete the study (i.e., end the iteration) because the iterations have converged sufficiently (or diverged).

Several implementations have been written that were used in the demonstration project for a typical transportation planning study. For example, Figure-7.3 shows a typical iteration scheme that is set up by the Selector/Iteration Database script. In this scheme, activities, plans, and microsimulations are iterated until traffic behavior on the network stabilizes. It is not difficult for analysts to write additional iteration scripts for their own specialized studies

7.4 Major Data Inputs

Feedback Module can extract data from any input files according to the user's intention. The Network, Transit Route and Transit Schedule files are typically used in the feedback processes.

7.5 Major Data Outputs

There are no direct outputs from the Feedback Module to users. All outputs from the Feedback Module need to be sent back to other Modules for feedback purposes.

7.6 Module Interfaces

The information available to TRANSIMS about travelers consists of traveler-specific data contained in the population, activity, plan, vehicle, and simulation output files. For feedback purposes, Feedback Module can extract data from the output files of all other TRANSIMS modules, except the Emissions Estimator. After some kinds of analyses, selected data will be sent back to the Activity Generator and Route Planner modules.

An outline for the module interfaces involving the Feedback Module is displayed in Figure-7.6.

7.6.1 Inputs received from the Population Synthesizer Module

All output data of the Population Synthesizer Module can be used as inputs to the Selector Module.

The major output of the Population Synthesizer module is a synthetic population of households containing a set of information associated with each household and each person within the household. The Population Synthesizer also generates and assigns private vehicles to households.

Household demographic data include tract ID, block group ID, household ID, number of persons living in the household, number of autos in the household, the home location ID, the PUMS household ID, the presence of persons under the age of 18 in the household, number of workers in the family and the household income.

Person demographic information includes household ID, person ID, age of the person, and relationship of that person to the householder, sex of the person and whether that person worked in the year 1989.

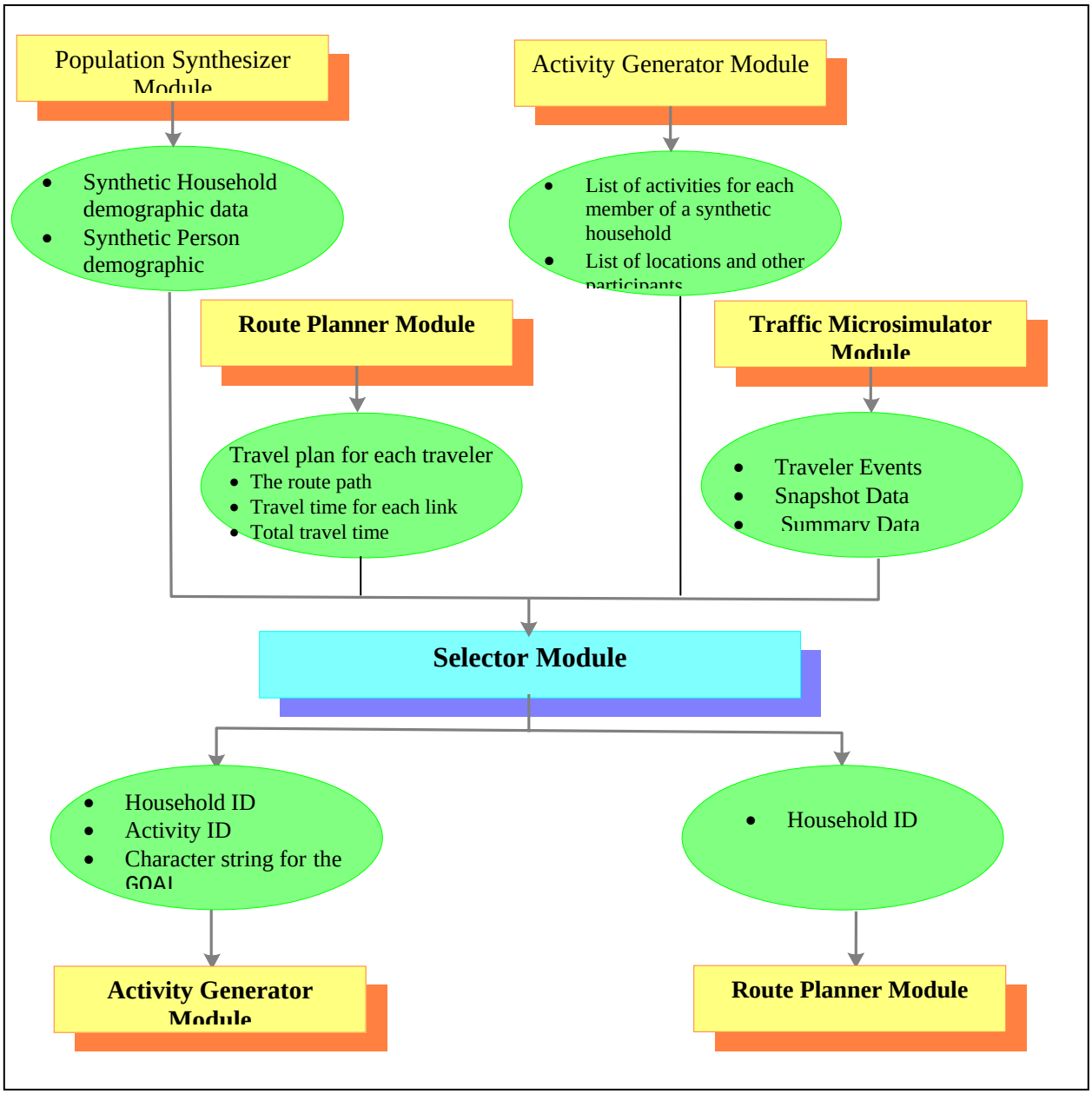

Figure-7.6: Flowchart of the module interfaces for the Feedback Module

7.6.2 Inputs received from the Activity Generator Module

All output data of the Activity Generator Module can be used as inputs to the Selector Module.

The primary output of the Activity Generator Module is the list of activities for each member of a synthetic household. Each activity list is comprised of the following:

- An activity type (e.g., home or school, etc.) and its priority,
- Starting, ending and duration time preferences,
- A preferred travel mode,
- A vehicle preference (if appropriate),
- A list of possible locations for an activity, and
- A list of other participating household members for shared activities.

7.6.3 Inputs received from the Route Planner Module

All output data of the Route Planner Module can be used as inputs to the Selector Module.

The Route Planner Module develops the route plans based on the demand represented in the Activities data file. The major outputs of the Route Planner for each traveler are information about the travel plans, which comprise the route path (nodes, links, and travel modes), travel time for each link, and total travel time. In addition, TRANSIMS displays the non-transportation activities in the output as well, in order to maintain a record for every activity for each member over the 24-hour horizon.

Generally, the outputs are formatted to show the relevant information for **each leg** on a trip. A leg must start and end at an activity location, a parking location, or a transit stop (notice that there is a special leg for the non-transportation activity that starts and ends at the same location).

7.6.4 Inputs received from the Traffic Microsimulator Module

All output data of the Traffic Microsimulator Module can be also used as inputs to the Selector Module.

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

7.6.5 Outputs sent to the Route Planner Module

The households identified to be re-routed need to be sent to the Route Planner Module through Route Planner feedback files. In addition, it is likely that each household for which an activity has been changed should be re-planned for a new route. There is only household ID recorded in the Route Planner feedback files. The Route Planner feedback files should be concatenated, sorted, and duplicate lines removed so that each household appears only once.

7.6.6 Outputs sent to the Activity Generator Module

The households selected that their activities need to be re-generated need to be sent to the Activity Generator Module through Activity Generator feedback files. Household ID, activity ID, and a character string are recorded in Activity Generator feedback files.

7.7 Configuration File

In this section all files used to run the Route Planner along with their descriptions, including the default values, are listed.

7.7.1 Iteration Database General Configuration File Keys

Configuration File Key	Description
ROUTER_IGNORABLE_PRIORITIES	See the Route Planner documentation.
ACT_HOME_ACTIVITY_TYPE	The number of the home activity type (non-negative integer). This key should be specified if an activity file is specified.
ACT_SCHOOL_ACTIVITY_TYPE	The number of the school activity type (non-negative integer). This key should be specified if an activity file is specified.
ACT_WORK_ACTIVITY_TYPE	The number of the work activity (non-negative integer). This key should be specified if an activity file is specified.
NET_ACTIVITY_LOCATION_TABLE	The activity location table name. This key is required.
NET_DIRECTORY	The directory where the network files reside. This key is required.
NET_LINK_TABLE	The link table name. This key is required.
NET_NODE_TABLE	The node table name. This key is required.
NET_PARKING_TABLE	The parking table name. This key is required.
NET_PROCESS_LINK_TABLE	The process link table name. This key is required.
NET_TRANSIT_STOP_TABLE	The transit stop table name. This key is required.
SEL_ACTIVITY_FILE	The activity file for use by the Collator. If not present, the ACTIVITY_FILE configuration file key is used. Either SEL_ACTIVITY_FILE or ACTIVITY_FILE must be specified.
SEL_EVENT_FILE	The event file for use by the Collator.
SEL_ITDB_FILE	The full pathname of the output Iteration Database file generated by the Selector Collator. An iteration number extension is automatically added to the end of this name.
SEL_MESSAGE_LEVEL	Sets the message level for the Collator, Stratifier, and Selector modules. Message levels range from –1 to 4 with the higher numbers reporting more.

Configuration File Key	Description
SEL_PLAN_FILE	The plan file for use by the Collator. If not present, the PLAN_FILE configuration file key is used.
SEL_POPULATION_FILE	The population file for use by the Collator. If not present, the ACT_POPULATION_FILE configuration file key is used. Either SEL_POPULATION_FILE or ACT_POPULATION_FILE must be specified.
SEL_STRAT_OUT_FILE	The full pathname for the output Iteration Database created by the Stratifier. There may be several of these for each Collator run. Default = strat
SEL_UAZ_FILE_n	The full pathname of the polygon file specifying User Analysis Zone(s) n, where n is an integer starting at 1.

7.7.2 Iteration Database Activity Configuration File Keys

Configuration File Key	Description
SEL_USE_ACT_HH_ID	If set, directs the Collator to add the household ID from the activity file for the ending activity for the trip to the output Iteration Database
SEL_USE_ACT_LAST_OK_ITER	If set, directs the Collator to add a field containing the number of the last iteration on which no problems were reported for this household in the Activity Generator or Regenerator's Problem File. A value of –1 indicates there has been a problem reported on every iteration.
SEL_USE_ACT_PERSON_ID	If set, directs the Collator to add the person ID from the activity file for the ending activity for the trip to the output Iteration Database
SEL_USE_END_ACT_GROUP_NUM	If set, directs the Collator to add the activity group number from the activity file for the ending activity for the trip to the output Iteration Database
SEL_USE_END_ACT_ID	If set, directs the Collator to add the activity ID from the activity file for the starting activity for the trip to the output Iteration Database
SEL_USE_END_ACT_LOCATION	If set, directs the Collator to add the first of the possible locations from the activity file for the ending activity for the trip to the output Iteration Database
SEL_USE_END_ACT_TYPE	If set, directs the Collator to add the activity type from the activity file for the ending activity for the trip to the output Iteration Database
SEL_USE_END_DUR_LB	If set, directs the Collator to add the duration lower bound from the activity file for the ending activity for the trip to the output Iteration Database. The value is converted from fractional hours to seconds.
SEL_USE_END_DUR_UB	If set, directs the Collator to add the duration upper bound from the activity file for the ending activity for the trip to the output Iteration Database. The value is converted from fractional hours to seconds.
SEL_USE_END_TIME_LB	If set, directs the Collator to add the end time lower bound from the activity file for the ending activity for the trip to the output Iteration Database.

Configuration File Key	Description
SEL_USE_END_TIME_UB	If set, directs the Collator to add the end time upper bound from the activity file for the ending activity for the trip to the output Iteration Database.
SEL_USE_END_MODE_PREF	If set, directs the Collator to add the mode preference from the activity file for the ending activity for the trip to the output Iteration Database
SEL_USE_END_OTHER_PARTICIPANTS	If set, directs the Collator to add the number of other participants from the activity file for the ending activity for the trip to the output Iteration Database
SEL_USE_END_POSS_LOC	If set, directs the Collator to add the number of possible locations from the activity file for the ending activity for the trip to the output Iteration Database
SEL_USE_END_PRIORITY	If set, directs the Collator to add the activity priority from the activity file for the ending activity for the trip to the output Iteration Database
SEL_USE_END_START_TIME_LB	If set, directs the Collator to add the start time lower bound from the activity file for the ending activity for the trip to the output Iteration Database
SEL_USE_END_START_TIME_UB	If set, directs the Collator to add the start time upper bound from the activity file for the ending activity for the trip to the output Iteration Database
SEL_USE_END_VEHICLE_ID	If set, directs the Collator to add the vehicle ID from the activity file for the ending activity for the trip to the output Iteration Database
SEL_USE_START_ACT_GROUP_NUM	If set, directs the Collator to add the activity group number from the activity file for the starting activity for the trip to the output Iteration Database.
SEL_USE_START_ACT_ID	If set, directs the Collator to add the activity ID from the activity file for the starting activity for the trip to the output Iteration Database.
SEL_USE_START_ACT_LOCATION	If set, directs the Collator to add the first of the possible locations from the activity file for the starting activity for the trip to the output Iteration Database.
SEL_USE_START_ACT_TYPE	If set, directs the Collator to add the activity type from the activity file for the starting activity for the trip to the output Iteration Database.
SEL_USE_START_DUR_LB	If set, directs the Collator to add the duration lower bound from the activity file for the starting activity for the trip to the output Iteration Database. The value is converted from fractional hours to seconds.
SEL_USE_START_DUR_UB	If set, directs the Collator to add the duration upper bound from the activity file for the starting activity for the trip to the output Iteration Database. The value is converted from fractional hours to seconds.
SEL_USE_START_END_TIME_LB	If set, directs the Collator to add the end time lower bound from the activity file for the starting activity for the trip to the output Iteration Database. The value is converted from fractional hours to seconds.
SEL_USE_START_END_TIME_UB	If set, directs the Collator to add the end time upper bound from the activity file for the starting activity for the trip to the output Iteration Database. The value is converted from fractional hours to seconds.

Configuration File Key	Description
SEL_USE_START_MODE_PREF	If set, directs the Collator to add the mode preference from the activity file for the starting activity for the trip to the output Iteration Database.
SEL_USE_START_OTHER_PARTICIPANTS	If set, directs the Collator to add the number of other participants from the activity file for the starting activity for the trip to the output Iteration Database.
SEL_USE_START_POSS_LOC	If set, directs the Collator to add the number of possible locations from the activity file for the starting activity for the trip to the output Iteration Database.
SEL_USE_START_PRIORITY	If set, directs the Collator to add the activity priority from the activity file for the ending activity for the trip to the output Iteration Database.
SEL_USE_START_START_TIME_LB	If set, directs the Collator to add the start time lower bound from the activity file for the starting activity for the trip to the output Iteration Database.
SEL_USE_START_START_TIME_UB	If set, directs the Collator to add the start time upper bound from the activity file for the starting activity for the trip to the output Iteration Database.
SEL_USE_START_VEHICLE_ID	If set, directs the Collator to add the vehicle ID from the activity file for the starting activity for the trip to the output Iteration Database.

7.7.3 Iteration Database Microsimulation Event Configuration File Keys

Configuration File Key	Description
SEL_USE_ACCELS	If set, directs the Collator to add the ACCELS field from the Traffic Microsimulator event output data to the output Iteration Database. This value is summed across events from every leg of the trip.
SEL_USE_ANOMALY	If set, directs the Collator to add the ANOMALY field from the Traffic Microsimulator event output data to the output Iteration Database. Note that this gives only the value for the last event of the trip.
SEL_USE_DISTANCE_SUM	If set, directs the Collator to add the DISTANCE_SUM field from the Traffic Microsimulator event output data to the output Iteration Database. This value is summed across events from every leg of the trip.
SEL_USE_EVENT_LEG_ID	If set, directs the Collator to add the LEG field from the Traffic Microsimulator event output data to the output Iteration Database. Only the value for the first event of each trip is reported.
SEL_USE_EVENT_PERSON_ID	If set, directs the Collator to add the TRAVELER field from the Traffic Microsimulator event output data to the output Iteration Database.
SEL_USE_EVENT_TRIP_ID	If set, directs the Collator to add the TRIP field from the Traffic Microsimulator event output data to the output Iteration Database.
SEL_USE_EVENT_USER	If set, directs the Collator to add the USER field from the Traffic Microsimulator event output data to the output Iteration Database.

Configuration File Key	Description
SEL_USE_EVENT_VEHICLE_ID	If set, directs the Collator to add the VEHICLE field from the Traffic Microsimulator event output data to the output Iteration Database. Only the value for the first event of each trip is reported.
SEL_USE_LINK	If set, directs the Collator to add the LINK field from the Traffic Microsimulator event output data to the output Iteration Database. Note that this gives only the value for the last event of the trip.
SEL_USE_LOCATION	If set, directs the Collator to add the LOCATION field from the Traffic Microsimulator event output data to the output Iteration Database. Note that this gives only the value for the last event of the trip.
SEL_USE_NODE	If set, directs the Collator to add the NODE field from the Traffic Microsimulator event output data to the output Iteration Database. Note that this gives only the value for the last event of the trip.
SEL_USE_ROUTE	If set, directs the Collator to add the ROUTE field from the Traffic Microsimulator event output data to the output Iteration Database. Only the value for the first event of each trip is reported.
SEL_USE_SIGNALS	If set, directs the Collator to add the SIGNALS field from the Traffic Microsimulator event output data to the output Iteration Database. This value is summed across events from every leg of the trip.
SEL_USE_STATUS	If set, directs the Collator to add the STATUS field from the Traffic Microsimulator event output data to the output Iteration Database. Note that this gives only the value for the last event of the trip.
SEL_USE_STOPPED	If set, directs the Collator to add the STOPPED field from the Traffic Microsimulator event output data to the output Iteration Database. This value is summed across events from every leg of the trip.
SEL_USE_STOPS	If set, directs the Collator to add the STOPS field from the Traffic Microsimulator event output data to the output Iteration Database. This value is summed across events from every leg of the trip.
SEL_USE_TIME	If set, directs the Collator to add the TIME field from the Traffic Microsimulator event output data to the output Iteration Database. Only the value for the last event of each trip is reported.
SEL_USE_TIME_SUM	If set, directs the Collator to add the TIME_SUM field from the Traffic Microsimulator event output data to the output Iteration Database. This value is summed across events from every leg of the trip.
SEL_USE_TURN	If set, directs the Collator to add the TURN field from the Traffic Microsimulator event output data to the output Iteration Database. Note that this gives only the value for the last event of the trip.
SEL_USE_VEH_SUBTYPE	If set, directs the Collator to add the VSUBTYPE field from the Traffic Microsimulator event output data to the output Iteration Database. Only the value for the first event of each trip is reported.
SEL_USE_VEH_TYPE	If set, directs the Collator to add the VEHTYPE field from the Traffic Microsimulator event output data to the output Iteration Database.

Configuration File Key	Description
SEL_USE_YIELDS	If set, directs the Collator to add the YIELDS field from the Traffic Microsimulator event output data to the output Iteration Database. This value is summed across events from every leg of the trip.

7.7.4 Iteration Database Router/Plan Configuration File Keys

Configuration File Key	Description
SEL_USE_COST	If set, directs the Collator to add the cost from the Plan file for the trip to the output Iteration Database. Note that there are multiple legs in a trip, each of which contains a value for this field. The Collator will use the sum over all legs.
SEL_USE_DEP_TIME	If set, directs the Collator to add the departure time from the Plan file for the trip to the output Iteration Database. Note that there are multiple legs in a trip, each of which contains a value for this field. The Collator will use the first leg.
SEL_USE_DRIVER	If set, directs the Collator to add the driver flag from the Plan file for the trip to the output Iteration Database. The Collator will fill this field with NA, since there are multiple possible values per trip.
SEL_USE_DURATION	If set, directs the Collator to add the (expected) duration from the Plan file for the trip to the output Iteration Database. Note that there are multiple legs in a trip, each of which contains a value for this field. The Collator will use the sum over all legs.
SEL_USE_END_ACC	If set, directs the Collator to add the ending accessory ID from the Plan file for the trip to the output Iteration Database. Note that there are multiple legs in a trip, each of which contains a value for this field. The Collator will use the last leg.
SEL_USE_END_ACC_TYPE	If set, directs the Collator to add the ending accessory type from the Plan file for the trip to the output Iteration Database. Note that there are multiple legs in a trip, each of which contains a value for this field. The Collator will use the last leg.
SEL_USE_GCF	If set, directs the Collator to add the generalized cost function from the Plan file for the trip to the output Iteration Database. Note that there are multiple legs in a trip, each of which contains a value for this field. The Collator will use the sum over all legs.
SEL_USE_LEG_ID	If set, directs the Collator to add the leg ID from the Plan file for the trip to the output Iteration Database. The Collator will fill this field with "NA", since there are multiple legs for each trip.

Configuration File Key	Description
SEL_USE_MAX_TIME	If set, directs the Collator to add the max time flag from the Plan file for the trip to the output Iteration Database. Note that there are multiple legs in a trip, each of which contains a value for this field. The Collator will use the last leg.
SEL_USE_MODE	If set, directs the Collator to add the mode from the Plan file for the trip to the output Iteration Database. The Collator will fill this field with NA, since there are multiple possible values per trip.
SEL_USE_PLAN_PERSON_ID	If set, directs the Collator to add the person ID from the Plan file for the trip to the output Iteration Database.
SEL_USE_PLAN_USER	If set, directs the Collator to add the person ID from the Plan file for the trip to the output Iteration Database. Note that there are multiple legs in a trip, each of which contains a value for this field. The Collator will use the first leg.
SEL_USE_ROUTER_LAST_OK_ITER	If set, directs the Collator to add a field containing the number of the last iteration on which no problems were reported for this traveler in the Router's Problem File. A value of –1 indicates there has been a problem reported on every iteration.
SEL_USE_START_ACC	If set, directs the Collator to add the starting accessory from the Plan file for the trip to the output Iteration Database. Note that there are multiple legs in a trip, each of which contains a value for this field. The Collator will use the first leg.
SEL_USE_START_ACC_TYPE	If set, directs the Collator to add the starting accessory type from the Plan file for the trip to the output Iteration Database. Note that there are multiple legs in a trip, each of which contains a value for this field. The Collator will use the first leg.
SEL_USE_STOP_TIME	If set, directs the Collator to add the (expected) arrival time from the Plan file for the trip to the output Iteration Database. Note that there are multiple legs in a trip, each of which contains a value for this field. The Collator will use the last leg.
SEL_USE_TRIP_ID	If set, directs the Collator to add the value in the user field from the Plan file for the trip to the output Iteration Database.

7.7.5 Iteration Database Population Configuration File Keys

Configuration File Key	Description
SEL_USE_<pop_file header field>	If set, directs the Collator to add the corresponding demographic variable from the population file to the output Iteration Database.
SEL_USE_BLOCK_GROUP	If set, directs the Collator to add the BLOCK_GROUP field from the population file to the output Iteration Database.
SEL_USE_HH_ID	If set, directs the Collator to add the HH_ID field from the population file to the output Iteration Database.

Configuration File Key	Description
SEL_USE_HOME_LOCATION	If set, directs the Collator to add the HOME_LOCATION field from the population file to the output Iteration Database.
SEL_USE_NUMBER_PERSONS	If set, directs the Collator to add the NUMBER_PERSONS field from the population file to the output Iteration Database.
SEL_USE_NUMBER_VEHICLES	If set, directs the Collator to add the NUMBER_VEHICLES field from the population file to the output Iteration Database.
SEL_USE_PERSON_ID	If set, directs the Collator to add the PERSON_ID field from the population file to the output Iteration Database.
SEL_USE_TRACT	If set, directs the Collator to add the TRACT field from the population file to the output Iteration Database.

7.7.6 Iteration Database Stratifier Configuration File Keys

Configuration File Key	Description
SEL_ALGORITHM	A comma- or semicolon-separated list of names of algorithms used to select travelers. Currently, the allowed values are: HI_VAR, LO_VAR, HI_RANGE, LO_RANGE, HI_MEAN, LO_MEAN, HI_SIGMA, and LO_SIGMA. These values instruct the Selector to pick the cell with the highest or lowest variance, range, mean, or ratio of mean to standard deviation, respectively. In addition, it is possible to pick a cell by index using the argument PICK_CELL <n>, where <n> is an integer. If the k binnings that make up a cell have n0, n1, …, nk bins each, a set of indexes into each binning of the form (i0, i1, …, ik) is equivalent to the cell index ik + n(k-1) * (i(k-1) + … + * (i2 + n1 * (i0))…). One algorithm must be supplied for each name in the SEL_COST argument.
SEL_BIN_BOUNDS_n	If specified, the bin boundaries used for binning n. The argument is a comma- or semicolon-separated list of values. Bins will be created extending from the smallest value to - infinity and from the largest value to + infinity. Over-ridden if SEL_BINS_NUMBINS_n is present.
SEL_BIN_FIELD_n	The column name of the input Iteration Database used for creating binning n.
SEL_BIN_NAME_n	The column name for binning n in the output Iteration Database. n must start at 0.
SEL_BIN_NUMBINS_n	If specified, the number of bins to create for binning n. Bin boundaries will be chosen to create bins with equal numbers of elements. The number of bins may be adjusted within the code if the distribution of elements is concentrated on a few values.

Configuration File Key	Description
SEL_BIN_SEL_ALGO	A semicolon-separated list of names of algorithms to use in selecting trips from within the chosen cell of the stratification. Possible values are: RANDOM, TAIL, or ALL. There must be one algorithm supplied for each name in the SEL_COST argument. The RANDOM value takes two optional parameters: the first is the fraction of elements to select; the second is the absolute number of elements to select, which overrides the first if it is smaller. For example: RANDOM, 0.2, 100; TAIL takes one required and three optional arguments. The first is 0 if the lowest cost tail is to be selected and non-zero otherwise; the second is the fraction of trips to select; the third is an absolute threshold to apply; and the last is an absolute number of elements to select.
SEL_COST	A comma- or semicolon-separated list of names of columns in the input Iteration Database to associate with stratifications for use by the Selector in selecting trips. One selected set will be created for each name.
SEL_GOAL	The Activity Regenerator command to be associated with the selected set of trips. One goal must be supplied for each name in the SEL_COST argument. The entire goal string is written to the Activity Generator feedback file after each selected traveler ID.
SEL_STRAT_BINS	A semicolon-separated list of comma-separated strings specifying the names of binnings in the Iteration Database to be used in stratifying the data.
SEL_USE_STRATIFICATION	Each semi-colon separated list in the value of SEL_STRAT_BINS creates one stratification, indexed beginning with 0. One stratification must be supplied for each name in the SEL_COST argument.

7.7.7 Iteration Database Algorithm Configuration File Keys

Configuration File Key	Description
SEL_USE_AND	Directs the Collator to include a field (for each pair) which is the logical AND between the values of the two fields. The argument is a semicolon-separated list of comma-separated Iteration Database field names.
SEL_USE_CROSS_BOUND	Adds a field to the output Iteration Database which is true if the starting and ending activity locations for the trip are in different polygons. The argument is a comma-separated list of UAZ ID, as above, but without the polygon identifier. The value of this field is true if the starting and ending activity locations for the trip are in different polygons.
SEL_USE_DIFF	Directs the Collator to include a field (for each pair) which is the difference between the values of the two fields. The argument is a semicolon-separated list of comma-separated Iteration Database field names.

Configuration File Key	Description
SEL_USE_DRIVES_PASSENGER	Directs the Collator to include a field which is true if, on any leg of the trip, the Plan file specifies that the traveler drives a vehicle with passengers. Does not apply to transit vehicle drivers.
SEL_USE_EFFSPEED	Directs the Collator to include a field giving the ratio of the Euclidean distance between start and end activities to the total time (as calculated for SEL_USE_T_TOTAL).
SEL_USE_END_ACT_USER_DATA	Directs the Collator to include a field giving the value of any user-specified field in the Activity Location network table for the activity location at the end of the trip. The argument is a semicolon-separated list of field names.
SEL_USE_END_IN_REGION	See SEL_USE_START_IN_REGION. The value of this field will be true if the ending activity location is inside the polygon.
SEL_USE_END_REGION	The number (n) of the User Analysis Zone specified by the configuration file key SEL_UAZ_FILE_n. The value of the field in the iteration database is the number of the polygon in the UAZ that contains the ending activity location or –1 if not in any defined polygon.
SEL_USE_EUCLID	Directs the Collator to include a field giving the Euclidean distance between the starting and ending activity locations, in the same units the network tables use.
SEL_USE_FINISH_TRIP	Directs the Collator to include a field that is true if an "end trip" event is found for this trip in the event output file.
SEL_USE_MODE_LEG_COUNT	Directs the Collator to include a field giving the number of legs on the trip using the specified mode. The argument is a comma-separated list of modes. Currently, only the following modes are distinguished: w - walk i - bicycle t, l, or b - transit c - non-transit vehicle (as driver or passenger) a - activity
SEL_USE_MODE_STRING	Directs the Collator to include a field reflecting the modes used on every leg of this trip. The value is a string with one letter for each leg, starting from the left. Information comes from the Plan file, which does not know about as many modes as the activity file. Currently the letters used and their meanings are: c - driving a vehicle p - passenger in a non-transit vehicle t - transit w - walk i - bicycle a - activity (no transportation)
SEL_USE_NUMLEGS	Directs the Collator to include a field giving the number of legs in this trip, as found in the Plan file.
SEL_USE_OR	Directs the Collator to include a field (for each pair) that is the logical OR between the values of the two fields. The argument is a semicolon-separated list of comma-separated Iteration Database field names.

Configuration File Key	Description
SEL_USE_PRODUCT	Directs the Collator to include a field (for each pair) that is the product of the values of the two fields. The argument is a semicolon-separated list of comma-separated Iteration Database field names.
SEL_USE_RATIO	Directs the Collator to include a field (for each pair) that is the ratio between the values of the two fields. The argument is a semicolon-separated list of comma separated Iteration Database field names.
SEL_USE_RELDIFF	Directs the Collator to include a field (for each pair) that is the relative difference between the values of the two fields. The argument is a semicolon-separated list of comma-separated Iteration Database field names. SEL_USE_RELDIFF A, B creates a field with values (A - B) / B.
SEL_USE_START_ACT_USER_DATA	Directs the Collator to include a field giving the value of any user-specified field in the Activity Location network table for the activity location at the beginning of the trip. The argument is a semicolon-separated list of field names.
SEL_USE_START_IN_REGION	Adds a field to the Iteration Database that has the value true if the starting activity location is in the specified polygon. The argument is a semicolon-separated list of UAZ region identifiers. Each region identifier is of the form <UAZ_ID>, <polygon id>; where <UAZ_ID> is an integer referring to a User Analysis Zone file specified by the UAZ_FILE_NAME configuration file key and <polygon id> is the ID of a polygon contained within that file. The Iteration Database column will be named START_IN_REGION_<UAZ_ID>_<polygon ID>. Its value will be a Boolean, which is true if the starting activity location for the trip is inside the polygon.
SEL_USE_START_REGION	The number (n) of the User Analysis Zone specified by the configuration file key SEL_UAZ_FILE_n. The value of the field in the iteration database is the number of the polygon in the UAZ that contains the starting activity location or –1 if not in any defined polygon.
SEL_USE_SUM	Directs the Collator to include a field (for each pair) which is the sum of the values of the two fields. The argument is a semicolon-separated list of comma-separated Iteration Database field names.
SEL_USE_T_MODE	Directs the Collator to include a field giving the total time (in seconds) spent in a particular mode on this trip, as found from the expected duration of those legs in the Plan file. See SEL_USE_MODE_LEG_COUNT for a list of the modes that can be distinguished. The argument should be a semicolon-separated list of mode characters.
SEL_USE_T_TOTAL	Directs the Collator to include a field giving the total time spent on this trip as found from the difference in the TIME field between the last and first events for this trip. The event output file should capture events with "change on trip" status.

Configuration File Key	Description
SEL_USE_T_WAIT	Directs the Collator to include a field giving the total time spent waiting as found by summing over the times between "begin waiting" and "end waiting" events in the event output file. The event output file must capture these events and must not filter out the TIME field.
SEL_USE_T_WALK	Directs the Collator to include a field giving the total time (in seconds) spent walking on this trip, as found from the expected duration of walk legs in the Plan file.
SEL_USE_TRAVERSE_REGION	The number (n) of the User Analysis Zone specified by the configuration file key SEL_UAZ_FILE_n. The value of the field in the iteration database is true if one of the start/end activity locations is in the specified UAZ and the other is outside the UAZ.
SEL_USE_XOR	Directs the Collator to include a field (for each pair) which is the logical exclusive OR between the values of the two fields. The argument is a semicolon-separated list of comma-separated Iteration Database Field names.

7.8 Algorithm

Feedback processes are important in the framework of TRANSIMS. Models are developed from a series of feedback loops between TRANSIMS modules that changes the behavior of selected individuals of the synthetic population. Also, feedback is necessary to stabilize the traffic.
Feedback is used to solve problems with current travel plans, to consider new travel options, and to stabilize traffic. To fulfill the above objectives, the Feedback Module adjusts travel activities by changing:

- Travel path,
- Mode of travel,
- Activity location,
- Activity schedule, and
- Number of activities.

The user determines who to change, what to change, and how to implement the change using the scripts. Feedback rules and decision weights may change based on the type of study. As stated earlier, Feedback is a very flexible and user-directed module. The User should define the convergence criteria and decides when the Feedback module should stop iterating. Feedback is typically stopped when no significant changes are shown in key system statistics after several iterations.

TRANSIMS provides three tools for choosing the sets to be acted on:

- Collator
- Stratifier
- Selector

Basically, the Collator collects input and output data the user is interested in for each trip and creates a Collator Iteration Database. In the process, all data are translated from diverse TRANSIMS-formats into a simple, but possibly large, ASCII, comma-separated format. The Stratifier allows the user to

group trips in the Collator Iteration Database based on criteria the user sets, called Discretization. The criteria are added to the database and thus a Stratifier Iteration Database is generated. The Selector chooses all or part of trips of some groups for feedback from the Stratifier Iteration Database, according to the users' intentions. Rudimentary database queries and a reporting system are implemented in Stratifier and Selector. Figure-7.7 shows the relationship between the three model tools.

7.8.1 Collator

The Collator gathers data from some input files of TRANSIMS, i.e., Network file and Transit Route file. It can also extract data from output files of the Population Synthesizer, Activity Generation, Route Planner, and Traffic Microsimulator. The Collator calculates values based on data from several different sources: for example, it uses the activity location table together with an activity file to calculate Euclidean distance between activities.

The Collator creates a database with one record for each trip made by each traveler from every household specified in its household file, or in the population file, if no household file is specified.

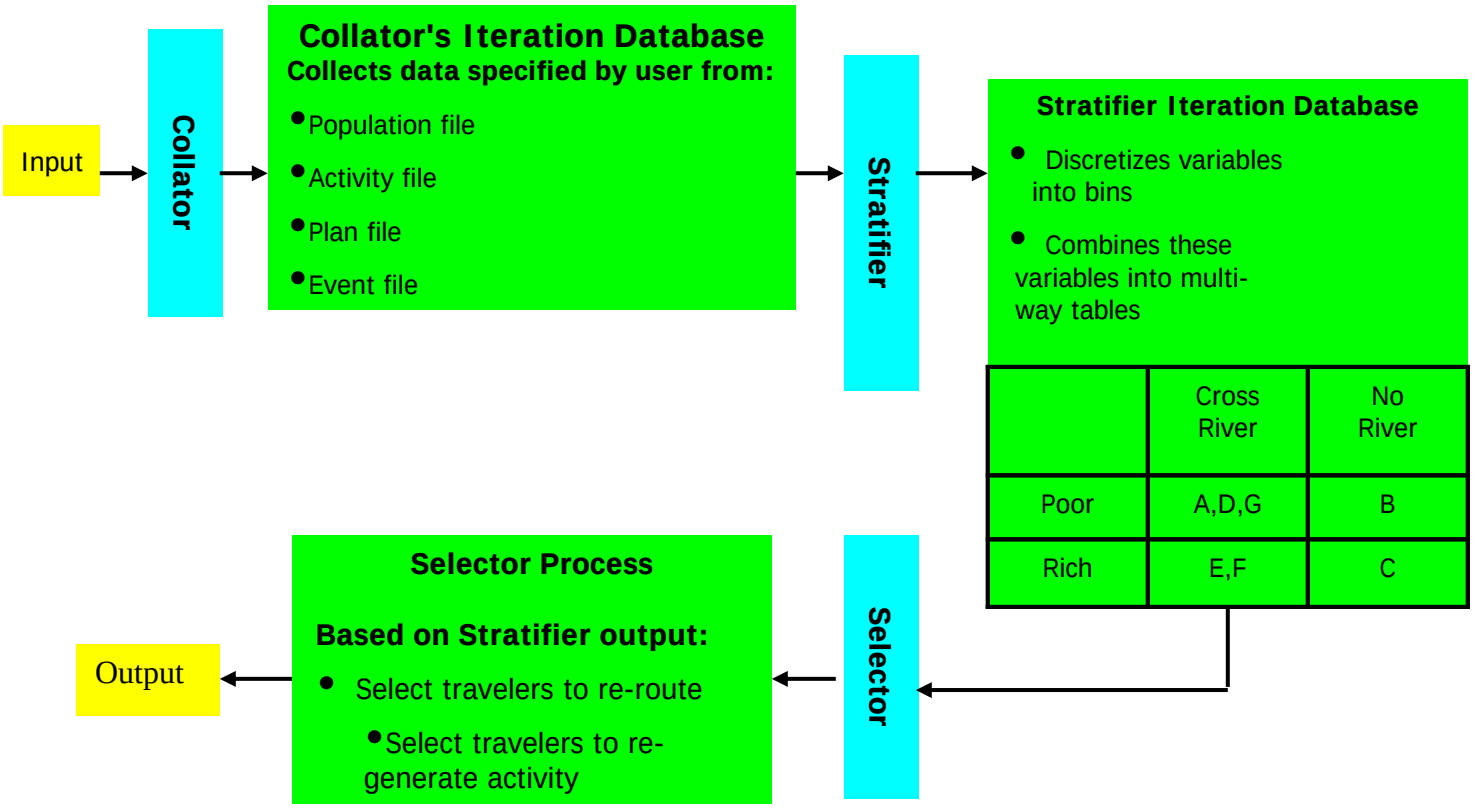

Figure-7.7: Relationship between the three model tools

Each record contains a few pre-determined fields and as many other fields as the user has requested. Fields that are present in every record are: `household id`, `traveler id`, `trip id` (as it would appear in the traveler's plan), `starting` and `ending activity ids` (as they appear in the activity file), and `tour` and `subtour id`.

The user requests fields by turning on any of the configuration file keys shown in the Configuration Files part of this chapter.

If the data for a given field is not available—for example, if there is no plan corresponding to a particular trip—that column will be blank in the database. If the data cannot be calculated—for example, if a denominator is zero—an "NA" will appear in that column. Records with blank or "NA" values are not used in Stratifier or Selector calculations that rely on that field.

After the run of Collator, if the user wishes to analyze the data by tour rather than by trip, he/she could use text-processing tools to aggregate the data and prepare a new file for later use by the Stratifier and Selector. The only requirements imposed by the software are:

- Two lines of header information, the second of which lists all the field names
- Fields are comma-separated ASCII text;
- The fields `HH`, `TRAV`, `TRIP`, `START_ACT_ID`, and `END_ACT_ID` must be used.

The Collator output can be indexed if desired. If the user requests an index, Collator Iteration Database from previous runs will be kept for the current invocation. If the underlying data files have not changed since the most recent Collator run or if data is missing from the data files, the most recent value from a previous database will be used. If an index is present, certain fields can be added to the database, such as a number indicating the last iteration on which a variable changed. Indexing can be time-consuming, and is not often necessary.

Collator Example

In this example, we have two tasks. One is to find all trips that start in region 1 and end in region 2 or 3, with duration of 20 minutes or less. The other is to find all the trips made by people older than 18 years old.

To implement these two tasks, we need to collect the information about whether the trip crosses a river, the duration of the trip, the age of the traveler and the ratio of expected travel times in modes ‘w' and 'c' with Collator.

First, we create a set of polygons that distinguish the two sides of a river. These are stored in the file `$TRANSIMS_ROOT/network/rivers.polygons`, called a "User Analysis Zone" or UAZ file by the Collator. In our case, there are two rivers that merge and thus three polygons as shown if Figure-7.8.

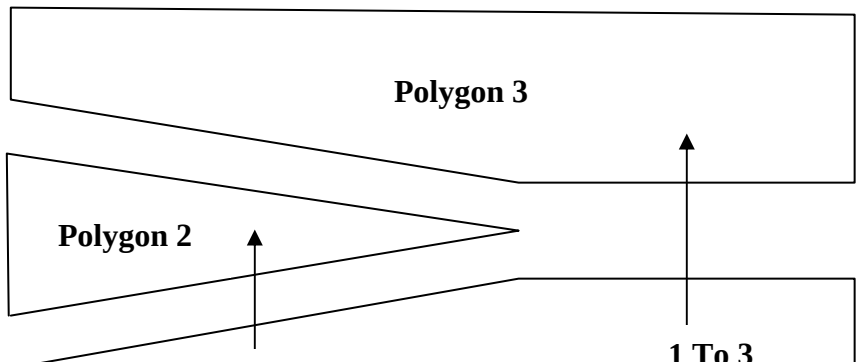

Figure-7.8: Three Polygons and Two kinds of trips mentioned in Collator example

To find out whether a trip's origin and destination are in different polygons, we add the following keys to a configuration file:

```
SEL_UAZ_FILE_1           $TRANSIMS_ROOT/network/rivers.polygons
SEL_USE_CROSS_BOUNDARY   1
```

If we wanted to be more specific about which polygonal region the trip started or ended in, we could also add:

```
SEL_USE_START_REGION     1
SEL_USE_END_REGION       1
```

Finally, if we are only interested in trips that start and end in specific polygons, say starting in polygon 1 of UAZ file 1 and ending in polygon 2 or 3 of UAZ file 1, we could use the following configuration file keys:

```
SEL_USE_START_IN_REGION    1, 1
SEL_USE_END_IN_REGION      1, 2; 1, 3
SEL_USE_AND                START_IN_REGION_1_1, END_IN_REGION_1_2;
START_IN_REGION_1_1, END_IN_REGION_1_3
```

The `SEL_USE_AND` configuration file key creates two fields, each of which is the logical `AND` of its arguments. In this case, the first field is true if, and only if, the trip starts in polygon 1 of UAZ file 1 and ends in polygon 2 of UAZ file 1.

In this example, we have included all of these different `REGION` and `BOUNDARY` configuration file keys.

Only the `START_IN_REGION`, `END_IN_REGION`, and `AND` configuration file keys are required for what follows.

To find the expected travel time (from the Plan file), we add the following configuration file key:

```
SEL_USE_DURATION   1.
```

For the age of the traveler and the ratio of expected travel times in modes ‘w' and 'c', we use the following configuration file keys:

```
SEL_USE_AGE      1
```

```
SEL_USE_T_MODE    w; c
SEL_USE_RATIO     T_MODE_c, T_MODE_w
```

Figure-7.9 shows a sample of the Collator output database for these configuration file keys.

							1	2	3	4
HH	TRAV	TOUR	SUBTOUR	TRIP	START_ACT_ID	END_ACT_ID	AGE	START_IN_REGION_1_1	END_IN_REGION_1_2	END_IN_REGION_1_3
2	4	0	0	1	1	1	59	TRUE	FALSE	FALSE
2	4	1	0	2	1	2	59	TRUE	TRUE	FALSE
2	4	1	0	3	2	2	59	FALSE	TRUE	FALSE
2	4	1	0	4	2	3	59	FALSE	FALSE	FALSE
2	4	1	0	5	3	3	59	TRUE	FALSE	FALSE
2	4	2	0	6	3	4	59	TRUE	FALSE	FALSE
2	4	2	0	7	4	4	59	TRUE	FALSE	FALSE
2	4	2	0	8	4	5	59	TRUE	FALSE	FALSE
2	4	2	0	9	5	5	59	TRUE	FALSE	FALSE
2	5	0	0	1	6	6	56	TRUE	FALSE	FALSE
64	198	0	0	1	1	1	63	TRUE	FALSE	FALSE
64	198	1	0	2	1	2	63	TRUE	FALSE	FALSE
64	198	1	0	3	2	2	63	TRUE	FALSE	FALSE
64	198	1	0	4	2	3	63	TRUE	FALSE	FALSE

5	6	7	8	9	10	11	12	13
START_REGION_1	END_REGION_1	CROSS_BOUND_1	DURATION	T_MODE_w	T_MODE_c	*RATIO_T	AND_START_IN*(1)	AND_START_IN*(2)
1	1	FALSE	28688	0	0	NA	FALSE	FALSE
1	2	TRUE	1012	46	986	21	TRUE	FALSE
2	2	FALSE	31385	0	0	NA	FALSE	FALSE
2	1	TRUE	1018	46	972	21.13043	FALSE	FALSE
1	1	FALSE	4672	0	0	NA	FALSE	FALSE
1	1	FALSE	581	46	535	11.63044	FALSE	FALSE
1	1	FALSE	2479	0	0	NA	FALSE	FALSE
1	1	FALSE	583	46	537	11.67391	FALSE	FALSE
1	1	FALSE	57900	0	0	NA	FALSE	FALSE
1	1	FALSE	99242	0	0	NA	FALSE	FALSE
1	1	FALSE	27197	0	0	NA	FALSE	FALSE
1	1	FALSE	410	46	364	7.913044	FALSE	FALSE
1	1	FALSE	36222	0	0	NA	FALSE	FALSE
1	1	FALSE	408	46	362	7.869565	FALSE	FALSE

*RATIO_T_MODE_c_T_MODE_w
*(1) AND_START_IN_REGION_1_1_END_IN_REGION_1_2
*(2) AND_START_IN_REGION_1_1_END_IN_REGION_1_3

Figure-7.9: Example Iteration Database

Notice that "NA" appears for the value of a field when it requires division by 0. In general, inappropriate fields are either left blank or given an "NA".

7.8.2 Stratifier

The Stratifier groups trips in two steps:

1. Discretized variables, i.e., group trips according to a specific criterion. For example, one criterion could be age of 18. Therefore, the trips made by people older than 18 years old fall into one group, whereas trips made be people younger or equal to 18 years old fall into another group. Another criterion could be to group trips based on whether trip duration is shorter than 1200 sec or not. Each discretized variable is known as a "binning", and each binning is given a numeric identifier. 'People older than 18' and 'trip duration shorter than 1200' are two binnings.

2. Combine binnings into multi-way tables, i.e., group trips based on the combination of multiple criteria. In the previous example, we will group trips according to both binnings and get a 2-way table. Each binning can be used in any number of different tables. Each table is called a "stratification," and each stratification is given a numeric identifier. Furthermore, each cell in the k-way table is given an index.

Figure-7.10 shows an example of the two steps employed by the Stratifier.

…	TRIP	…	age bin (0/1)	duration bin (0/1)
	1		1	1
	2		0	1
	3		1	0

⇩

STRATIFICATION

	Age	
	0	1
Duration 0	0	1
1	2	3

			0	1	
…	TRIP	…	age bin	duration bin	STRAT-0
	1		1	1	3
	2		0	1	2
	3		1	0	1

Figure-7.10: An example showing the two steps in STRATIFICATION

Discretization is accomplished in any of the following three ways;

- Automatically which is the best for fields that already contain discrete data with only a few different values. Each different value is assigned a numeric bin id.
- Manually, specifying the number of bins. The data in a field is sorted and placed into the user-specified number of quantiles. If the data is discrete and heavily concentrated on a few values, the algorithm may reduce the number of quantiles.
- Manually, specifying the bin boundaries.

The user chooses one of these methods for each binning using configuration file keys. The boundaries for each binning are listed in the header of the Stratifier output database.

The Stratifier output database contains one record for each record in the Collator's database. Each record contains the pre-determined fields `household id`, `traveler id`, `trip`, and starting and ending activity for the trip. It also contains one field for each binning specified and one field for each stratification. The stratification field's value is the index of the cell into which this record falls.

Stratifier Example (cont'd on the Collator Example)

In the first step, we want to group trips based on whether they start in region 1 and end in region 2 or 3 or not, and whether the trips have a time duration of 20 minutes or less or not, and whether these trips are made by people older than 18 years old or not respectively. We will add binnings that address each of these categories to the database.

1. Specify a binning for the `DURATION` field into two bins with a boundary of 20 minutes (= 1200 seconds):

```
SEL_BIN_NAME_0      duration_bin
SEL_BIN_FIELD_0     DURATION
SEL_BIN_BOUNDS_0    1200
```

2. Create separate binnings for trips that start in region 1 and end in region 2 or 3. Since these are Boolean fields, we need not specify bin boundaries.

```
SEL_BIN_NAME_1      1_to_2
SEL_BIN_FIELD_1     AND_START_IN_REGION_1_1_END_IN_REGION_1_2

SEL_BIN_NAME_2      1_to_3
SEL_BIN_FIELD_2     AND_START_IN_REGION_1_1_END_IN_REGION_1_3
```

3. Bin trips by travelers' ages:

```
SEL_BIN_NAME_3      age_bin
SEL_BIN_FIELD_3     AGE
SEL_BIN_BOUNDS_3    18
```

In the second step, we create three k-way tables, or stratifications, which we will use in the selection process. The first two stratifications are about whether a trip starts in region 1 and ends in region 2 or 3, with duration of 20 minutes or less. The third one is about whether the traveler is older than 18 years old.

The first two stratifications are a pair of two-way tables: one from binnings 0 and 1; and one from binnings 0 and 2.

We will use the age binning (number 3) by itself to generate the third stratification. The configuration file key `SEL_STRAT_BINS` specifies this arrangement:

```
SEL_STRAT_BINS     0, 1; 0, 2; 3
```

The Stratifier takes less than two minutes to run on the 258,000 record database created by the Collator above. Figure-7.11 shows a sample of the Stratifier output database for the same trips shown in the Collator database sample above.

					1	2	3	4	5	6	7
HOUSE	TRAVELER	TRIP	STARTACT	ENDACT	duration bin	1 to 2	1 to 3	age bin	STRAT 0	STRAT 1	STRAT 2
2	4	1	1	1	1	0	0	1	1	1	1
2	4	2	1	2	0	1	0	1	2	0	1
2	4	3	2	2	1	0	0	1	1	1	1
2	4	4	2	3	0	0	0	1	0	0	1
2	4	5	3	3	1	0	0	1	1	1	1
2	4	6	3	4	0	0	0	1	0	0	1
2	4	7	4	4	1	0	0	1	1	1	1
2	4	8	4	5	0	0	0	1	0	0	1
2	4	9	5	5	1	0	0	1	1	1	1
2	5	1	6	6	1	0	0	1	1	1	1
64	198	1	1	1	1	0	0	1	1	1	1
64	198	2	1	2	0	0	0	1	0	0	1
64	198	3	2	2	1	0	0	1	1	1	1

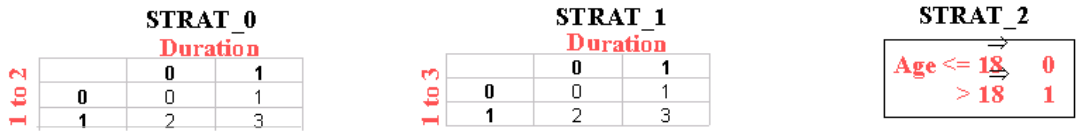

Figure-7.11: Example Stratifier Database

The first line of the file (not shown here) describes each of the binnings. There are three different types: rational, categorical, and ordinal. In a "rational" binning the elements are assumed to be floating-point numbers, and bin bounds are as indicated; in a "categorical" binning, the elements take on one of the few indicated discrete values, each of which is one bin; in an "ordinal" binning, they take on all integer values between the indicated upper and lower bounds.

The Stratifier adds the same `household`, `traveler`, `trip`, and `start and end activity` fields as the Collator. It does not add `tour` or `subtour` information.

7.8.3 Selector

There are two steps for the Selector to pick up a subset of travelers in the Stratifier Iteration Database: choosing from stratifications and selecting travelers

First, choose cells from stratifications. The cells can be picked by index (using the PICK_CELL selection algorithm). The index of a desired cell can be determined as follows:

$$\sum_{j=1}^{k}\{i(j)\prod_{m=0}^{j-1}n(m)\}$$

where:

k = the total number of binnings, i.e., the number of dimensions of stratification.
j = the index of the binning
$i(j)$ = the index of the bin for the desired cell in binning j
$n(m)$ = total number of bins in binning m, n(0) = 1.

Figure-7.13 shows an example of choosing a cell.

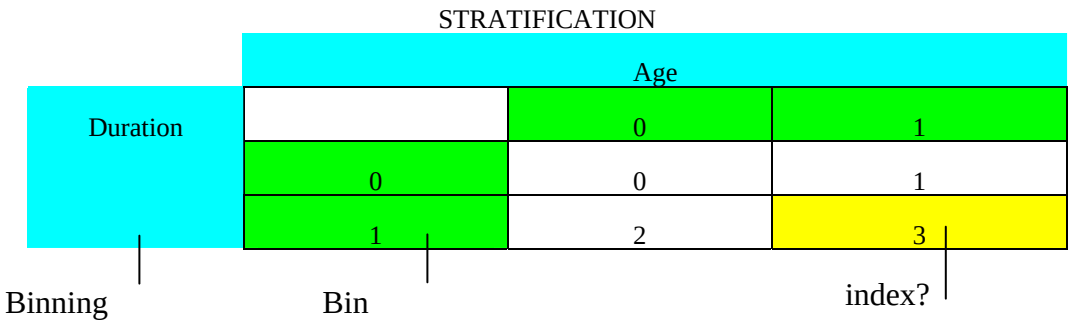

Figure-7.13: Pick up a cell by index from a Stratification

Since there are two binnings, Age and Duration, k is 2. And because there are two bins in each binning, n(1) = 2, n(2) = 2. n(0) always equals to 1. As the numerical identifier for the cell in Age binning is 1, i(1) = 1. i(2) = 1, based on similar reason. Therefore, we can get the index of the cell according to the formula, which is coincident with what we got before.

$$\text{Index} = i(1) + i(2) * (n(0) * n(1) = 1 + 1 * 2 = 3$$

It is probably worthwhile for the user to verify that the cell index used corresponds to the desired one by looking at the values of variables of interest for a traveler assigned to that cell.

After choosing a stratification, the next step is selecting a subset of travelers or trips within the cell. There are three ways to do that:

- The user can specify that all elements of the cell be chosen.
- Elements can be chosen uniformly at random.

- Elements can be chosen based on the cost function. Usually, each entry in a stratification can be associated with a cost function, such as travel time. So the user can pick the cell that has the highest or lowest mean travel time; or the cell that has the largest or smallest standard deviation or range in travel times.

A single run of the Selector can be used to select many different subsets of trips or travelers—each stratification in the Stratifier output database can be associated with a cost function one or more times.

There are two possible goals for selection. One is to identify activities to be changed using the Activity Regenerator. The other is to identify households to be re-routed.

The output of Selection on each subset will consist of one Activity Feedback file and one Route Planner Feedback file. The Activity Feedback file is then sent to the Activity Regenerator, and the Route Planner feedback file is sent back to the Route Planner. The user is free to use either or both of them according to his/her intention.

In the Activity Feedback file, a traveler id and an activity id are recorded. For each selection, the user specifies a "goal". The corresponding character string from the `GOAL` configuration file key is also included in the Activity Feedback file. In the Route Planner Feedback file there is only a household id reported, as we always need to re-plan the whole household.

Each selection creates a different pair of Activity Regenerator and Route Planner feedback files. If we make several selections at the same time, we will get multiple Activity Regenerator feedback files and Route Planner feedback files. The Activity Regenerator files should be concatenated so that all activities for a given household can be updated simultaneously. The ordering of commands for the Activity Regenerator is significant. The user should be careful to concatenate Activity Regenerator files in the order s/he wishes the Regenerator commands to be applied. The Route Planner feedback files should also be concatenated, sorted, and duplicate lines removed so that each household appears only once.

Selector Example (cont'd on Stratifier Example)

The following example illustrates one way of performing the desired selections. It is not unique.

As we mentioned before, we have two tasks. One is to find all trips that start in region 1 and end in region 2 or 3, with duration of 20 minutes or less. The other is to find all the trips made by people older than 18 years old. By now, we have already divided the trips according to these criteria. Next, we will pick up trips from the Stratifier Iteration Database based on these criteria.

We can find the cell corresponding to bin 1 of the first binning (which has a total of 2 bins) and bin 1 of the second binning satisfying our criteria of the first task. This cell, according to the formula above, gives a cell index of:

$$1 + 1 * 2 = 3$$

We would like to pick up all the elements of the cell. Even though no cost function is required for the selection, the current implementation of the Selector requires one to be provided. It is best if the cost function not have any "NA" values in it, because that will unnecessarily remove some records from consideration. Hence we choose `HH`, which is guaranteed to be defined for every record.

In practice, we would create a similar selection from the second stratification. We don't use this selection in the following example because it does not illustrate anything different from the first selection.

For the second task, we will again pick a specific cell, but not all of the elements of that cell are needed. In this case, we are using bin 1 of the only binning, so the cell index is also 1. We use the ratio of travel times in walk and car mode as the cost function:

```
SEL_USE_STRATIFICATION   0; 2
SEL_ALGORITHM            PICK_CELL 3; PICK_CELL 1
SEL_COST                 HH; RATIO_T_MODE_c_T_MODE_w
SEL_BIN_SEL_ALGO         ALL; TAIL, 1, , 1.5
```

For 1st selection *For 2nd selection*

The parameters to the "TAIL" within-cell selection algorithm specify that the elements with costs above 1.5 are to be chosen.

Finally, we want to assign a goal for each selection. The goal of the first task is to relocate the ending activity for each selected trip. The goal of the second task is to reroute.

```
SEL_GOAL       LS 0.01; REROUTE
```

The output of a Selector run with this configuration file will consist of four files. There will be one Activity Feedback file and one Router Feedback file for each of the two selections. As we won't be generating new activities for the selection set 2 the Activity Feedback file for selection 2 won't be used because we intend only to re-route these households.

If several different Activity Regenerator commands were generated, the user should concatenate the files in the order he/she wishes the commands to be applied. For example, it makes little sense to change an activity location after a household's activities have been regenerated from the survey, so files containing a "regenerate activities from the survey" command should be appended after all other files. Similarly, the Router Feedback household files should all be concatenated and sorted and duplicate lines should be removed.

The base filenames for Selector output are specified by two configuration file keys:

```
ACT_FEEDBACK_FILE         /home/eubank/test/feedback.act
ROUTER_HOUSEHOLD_FILE     /home/eubank/test/feedback.router
```

The actual filenames will have "_<n>" appended to them, where "<n>" is an integer indicating which selection they represent.

The Selector takes less than a minute to run on the 258,000 record database created by the Stratifier above. Here is the first line of the first Activity Regenerator Feedback file:

```
2 2 LS 0.01
```

And here is the first line of the second Router Feedback file:

```
2
```

References

1. Beilner, H. and Jacobs, F. (1972). Probabilistic aspects of traffic assignment, Proc. 5th Intern.

2. Hensher, D., Stopher, P. (1977). “Behavioral Travel Modelling”, Croom Helm London

3. Nagel, K., Beckman, R.J., and Barrett C.L., (1999), “ TRANSIMS for transportation planning”, Los Alamos National Laboratory

4. Nagel, K., Rickert, M., Simon, P.M., and Pieck, M., (2000), “ The dynamics of iterated transportation simulation”, Los Alamos National Laboratory, LA-UR 98-2168

5. Nagel, K., Barrett C.L. (1997), “ Using Microsimulation Feedback for trip adaptation for realistic traffic in Dallas”, Los Alamos National Laboratory, LA-UR 97-1334

6. Jacob, R., Marathe, M.V., and Nagel, K., (1998), “ A computational study of routing algorithms for realistic transportation networks”, Los Alamos National Laboratory, LA-UR 98-2249

7. Watson, P. L. (1973). Predictions of intercity modal choice from disaggregate, behavioural stochastic models. Highway Research Record, No. 446, pp. 28-35.

8. Watson, P.L. (1974). Comparison of the model structure and predictive power of aggregate and disaggregate models of intercity mode choice. Transportation Research Record, No. 528, Transportation Research Board, Washington, DC, pp. 59-65.

9. TRANSIMS_LANL website, http://transims.tsasa.lanl.gov/

10. TRANSIMS_DOT website, http://www.transims.net/