NEXUS Transit Travel Time Calculator

Written by Paul Anderson (ande9474@umn.edu or paul.anderson@epfl.ch)
Documentation
May 11, 2012

Purpose

This program takes a database of transit schedules and returns a TAZ-to-TAZ travel time matrix. It was originally developed for use with the Metro Transit schedule format circa 2008 and as part of the Access to Destinations project for MnDOT. The latest version of this program (1.3) is compatible with the General Transit Feed Specification (GTFS) format and can be used with any GTFS schedule, as long as a few other inputs are provided.

Inputs

Several input files are needed. The assumed format is comma delimited with a one line header. File names can be set within the program.

stopline

[stop number, line number]
This is the driver for the program. One way to debug is to include a very small stopline file (~5 lines). For any schedule set, the user must ensure that all unique stop-line pairs are present in this file. (i.e. if stop 12345 is served by bus routes 2 and 3, the lines "12345,2" and "12345,3" should appear in this file)

stopconv

[stop number, TAZ, access/egress time (in minutes), block ID]
This file relates the stop locations to the nearest block centroid. The user would need to create this file to use this program with another city's data. Each block centroid is joined to the closest two stops (stops might be duplicated in this procedure, this is fine). Then, any remaining unjoined stops are joined to the closest block. This process guarantees that every block in the city will be considered, although its travel times may be too long to actually matter. Please note that the file provided in the example, stopconv_block2.csv, uses year 2000 TAZ and block definitions.

schedule

stop_times.txt (GTFS standard file)
[trip_id,arrival_time,departure_time,stop_id,stop_sequence,pickup_type,drop_off_type]
trips.txt (GTFS standard file)
[route_id,service_id,trip_id,trip_headsign,block_id,shape_id]
These files can be taken directly from a GTFS package. No additional processing is necessary.

transfer

[line1, stop1, line2, stop2]

This file contains all available transfers. In the Access to Destinations project, transfer files for the different scenarios were produced in ArcMap by setting a search tolerance of 400m (~1/4 mile).

tazfile
[TAZ1, TAZ2, walk time (in minutes)]
This file sets the base travel times. In the Access to Destinations project, this file contained interzonal walk times calculated using 1.2 times the centroid-to-centroid distance and a walking speed of 5 km/h. If the user only wants to see travel times on transit, this file could be filled in with a large number (such as 1440, the number of minutes in a day). This file is a travel time matrix and must include all possible TAZ to TAZ combinations. It also must be sorted, first by TAZ1, then by TAZ2.

Running the Program

First, place all input files in the same directory as Master1.java
Then, open Master1.java and change the file names to match your input files (search for "//SET FILE NAMES HERE"). At the bottom of this section are four additional variables. In every GTFS file, there are several different service IDs (weekday, weekend, special event, etc.). You probably only want one of these, so indicate which one in "serviceID". TAZnumber holds the number of TAZs. (For the Twin Cities in the year 2000 definition, there are 1201.) FirstScheduleTime and LastScheduleTime define the time period in seconds after midnight. For example, 21600 to 32400 represents 6-9 AM.
Now, check that the array sizes are large enough for your input files. (search for "//INCREASE AS NECESSARY"). The first number in the array definition must be larger than the number of lines in your input file.

Now, save Master1.java, open a command line window, navigate to your directory and type:

```
[your Java folder location]\Java\bin\javac Master1.java
[your Java folder location]\Java\bin\java Master1
```

The program will update you with each additional percentage point complete. It also tells you the elapsed time (in seconds) to complete the last percentage point, as well as the total run time. The results are saved to a file, results.csv, in the same directory as the program.

Example

This documentation was originally intended to be packaged with a complete example. This example will calculate travel times for the morning rush period (6-9am) using a GTFS schedule put out by Metro Transit and valid for late 2009/early 2010. The TAZ and block definitions are from year 2000.