Measuring Urban Mobility and Accessibility Using OpenTripPlanner Analyst: Methodology and Data

Matt Conway

April 22, 2012

This paper is the companion to the poster "Measuring Urban Mobility and Accessibility Using OpenTripPlanner Analyst;" it describes the methodology used to produce the maps on the poster and describes the data used for them.

Contents

All maps use OpenStreetMap data as a basemap. These maps should be taken as proofs of concept only; there are several things that could introduce sufficient error to make them useless for planning. First of all, they are all at specific times rather than being across a range of times; no one will leave to catch a bus or train planning on waiting 30 minutes for their first transit vehicle. Flexible times can improve travel times [1, 295-6].

Also, the graph used has not been extensively tested; there are likely lurking data issues. This is one advantage of OpenTripPlanner: if it is deployed as an agency's customer-facing trip planner, data issues will be found by customers and other departments far more effectively than they could be found by a planning department alone. For instance, when I initially built this graph, the Sacramento Valley Station light rail stop was connected to a path that went nowhere, so the planner could not access the station (that particular issue was fixed for this project).

1 Figure 1

This is a basic isochrone map; it was generated directly from a GeoTIFF out of OpenTripPlanner Analyst. First, OpenTripPlanner Analyst was configured using the graph configuration found in B.5. The origin parameter was set to be the center of the UC Davis Quad[1] at 8am on February 29, 2012. Then, the GeoTIFF was fetched using URL 1 in 6 and styled in Quantum GIS to produce this map.

2 Figure 2

This map shows the difference in travel times between weekdays and weekends, for trips starting at 8am at the UC Davis Quad. It would be far more meaningful if average travel times were used instead of travel times that specifically start at 8am; interactions between the two schedules[2] would not be prominently featured if that were the case. Flexible trip times can reduce total transit time [1, 295-6]. The raster used in Figure 1 (URL 1) was subtracted from a similar query for Saturday, February 25, 2012 (URL 3) using the QGIS raster calculator and the code in B.1.

3 Figure 3

This is a Hägerstrand's prism visualization, showing areas where remaining time is greater than 20 minutes for a trip on February 29, 2012 departing the UC Davis Quad (as above) at 8am and arriving at the California State Capitol[3] at 10:30am. This was issued against the same OpenTripPlanner instance, with URL 2 and styled in QGIS to show areas with more than 20 minutes of remaining time.

4 Figure 4

This map shows the calculation of accessibility via the isochrone method using an Analyst raster. In this case, a raster for 5pm on a weeknight was used (URL 4), since Analyst cannot yet produce

[1] 38.54165° N, 121.74948° W

[2] For example, one might just catch a bus at 8am on Saturday, but just miss it on a weekday.

[3] 38.57671° N, 121.49424° W

a raster for arrive-by and it didn't make sense to calculate how many people could potentially leave the campus at 8am and be home in 90 minutes. The raster was combined with data from the 2010 US Census. First, the 2010 TIGER/Line shapefile for all blocks in California was downloaded and processed using ogr2ogr B.2. Then, the Census 2010 Summary File 1 was downloaded and population data extracted from the ZIP file using the Unix tools unzip, cut and awk B.4. The data were then joined in an SQLite database and ultimately loaded into PostGIS.

The raster was converted to a mask for all areas accessible in under 90 minutes, then polygonized using GDAL. This polygon (which is shown on the final map) was then loaded into PostGIS, and a spatial SQL query B.3 was used to find all of the blocks that touched the 90 minute area.[4] The final step was to sum all of the selected blocks.

5 Figure 5

This is simply a demo of the OpenTripPlanner customer-facing trip planning software, using the same graph as was used for analysis.

6 Figure 6

This is a demo of the OpenTripPlanner Analyst webapp, previewing a Hägerstrand's prism visualization.

[4]The touch operation was used because there are many small holes in the 90 minute polygon, in areas where there are no roads. A contains operation left out many blocks that should clearly have been included.

A URLs to Produce GeoTIFFs

These URLs were used to produce GeoTIFFs for various figures. Note that, although parameters for point B are often included in travel time requests, they are not used (this is a side effect of the way the web application works).

1	Travel time from the UC Davis Quad (Fig. 1)	http://localhost:8080/opentripplanner-analyst-core/wms?layers=traveltime&format=image/geotiff&srs=EPSG:26942&resolution=15&bbox=2004103.9238304757,576304.6520648422,2063760.7649179231,621263.7160801375&DIM_ORIGINLAT=38.54165382313658&DIM_ORIGINLON=-121.7494797706604&time=2012-02-29T16:00:00Z&DIM_ORIGINLATB=38.57670545237843&DIM_ORIGINLONB=-121.49424076080324&DIM_TIMEB=2012-02-29T18:30:00Z
2	Time remaining from the UC Davis Quad to the California State Capitol (Fig. 2)	http://localhost:8080/opentripplanner-analyst-core/wms?layers=hagerstrand&format=image/geotiff&srs=EPSG:26942&resolution=15&bbox=2004103.9238304757,576304.6520648422,2063760.7649179231,621263.7160801375&DIM_ORIGINLAT=38.54165382313658&DIM_ORIGINLON=-121.7494797706604&time=2012-02-29T16:00:00Z&DIM_ORIGINLATB=38.57670545237843&DIM_ORIGINLONB=-121.49424076080324&DIM_TIMEB=2012-02-29T18:30:00Z
3	Travel time from UC Davis Quad, Saturday 8am	http://localhost:8080/opentripplanner-analyst-core/wms?layers=traveltime&format=image/geotiff&srs=EPSG:26942&resolution=15&bbox=2004103.9238304757,576304.6520648422,2063760.7649179231,621263.7160801375&DIM_ORIGINLAT=38.54165382313658&DIM_ORIGINLON=-121.7494797706604&time=2012-02-25T16:00:00Z&DIM_ORIGINLATB=38.57670545237843&DIM_ORIGINLONB=-121.49424076080324&DIM_TIMEB=2012-02-29T18:30:00Z
4	Travel time from UC Davis Quad, Evening	http://localhost:8080/opentripplanner-analyst-core/wms?layers=traveltime&format=image/geotiff&srs=EPSG:26942&resolution=15&bbox=1984955,576304.6520648422,2077494,621263.7160801375&DIM_ORIGINLAT=38.54165382313658&DIM_ORIGINLON=-121.7494797706604&time=2012-03-01T01:00:00Z&DIM_ORIGINLATB=38.57670545237843&DIM_ORIGINLONB=-121.49424076080324&DIM_TIMEB=2012-02-29T18:30:00Z

B Listings

B.1 Raster Calculator Expression for Subtracting Rasters

This code was used to subtract the weekday raster from the Saturday one, setting NoData areas (areas with a value of 255 in the original rasters) to -32768. Comments cannot actually be included in the raster calculator, but are included here for clarity.

```
# Subtract them
((DavisQuad2012-02-25T16_00_00Z@1  -  DavisQuad2012-02-29T16_00_00Z@1)*
# Multiply by 1 if neither is 255 (NoData), 0 otherwise
(DavisQuad2012-02-25T16_00_00Z@1 != 255 AND DavisQuad2012-02-29T16_00_00Z@1 != 255))
# Subtract 32768 if either one was NoData, giving us -32768 for NoData.
-
(32768*(DavisQuad2012-02-25T16_00_00Z@1 = 255 OR DavisQuad2012-02-29T16_00_00Z@1 =
    255))
```

B.2 Using ogr2ogr to Extract Sacramento-area Census Blocks

This command was used to extract relevant Census blocks from the California TIGER/Line 2010 Shapefile.

```
ogr2ogr -f sqlite -spat -121.966 38.344 -120.99 38.774 accessibilitydata.sqlite
    tl_2010_06_tabblock10.shp
```

B.3 Spatial SQL to Find Blocks that Touch the Polygonized Isochrone

This SQL creates a new table with all the blocks that touch the isochrone. It takes many minutes to calculate, so using a new table is wise. Once it is done, it is trivial to sum all of the population values in the resulting table.

```
CREATE TABLE analystposter.blockswithin90 AS
SELECT * FROM analystposter.blockpop
 WHERE ST_Intersects(transform(the_geom, 26942), (SELECT ST_Union(the_geom) FROM
     analystposter.within90));
```

B.4 Extracting Data from Census Summary File 1

This parses the Summary File 1 ZIP directly, joins the Geo header to the population (table P1) and makes a table with TIGER GeoIDs that can be imported to an SQL database for joining to TIGER.

```
# Parse from the geo table: summary level, logrec, FIPS for state, county, tract,
    block
# TIGER/Line GeoIDs are concatenations of FIPS codes for state, county, tract and
    block.
# then print out a space-delimited mapping of all logrecs to geoids. Finally, filter
     to just our counties.
unzip -p ca2010.sf1.zip cageo2010.sf1 | cut --output-delimiter=',' -b
    9-11,19-25,28-29,30-32,55-60,62-65 | awk -F , '$1 == 101 && ($4 == 005 || $4 ==
    017 || $4 == 061 || $4 == 067 || $4 == 095 || $4 == 101 || $4 == 113) {print $2"
    "$3$4$5$6}' > logrec-geoid10.ssv
```

```
# Now parse P1 from the second file and join to the GeoIDs from the
first file
join -j1 <(unzip -p ca2010.sf1.zip ca000012010.sf1 | awk -F , '{print $5" "$6}') <(
    cat logrec-geoid10.ssv) > joined.ssv

# End result: lines like "logrec p1 geoid10", which can be imported to an SQL
    database for joining with TIGER/Line
```

B.5 Graph Builder Configuration for OpenTripPlanner Analyst

This is the graph builder configuration used to build the Sacramento graph for OpenTripPlanner Analyst. The graph was built using OpenStreetMap and transit agency data from March 16, 2012.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.
        springframework.org/schema/beans/spring-beans-2.5.xsd
    http://www.springframework.org/schema/context http://www.springframework.org/
        schema/context/spring-context-2.5.xsd">

    <bean id="graphBundle" class="org.opentripplanner.model.GraphBundle">
        <property name="path" value="/var/otp/graphs/davis/" />
    </bean>


    <bean id="gtfsBuilder" class="org.opentripplanner.graph_builder.impl.
        GtfsGraphBuilderImpl">
      <property name="gtfsBundles">
        <bean id="gtfsBundles" class="org.opentripplanner.graph_builder.model.
            GtfsBundles">
          <property name="bundles">
            <list>
              <!-- Unitrans -->
              <bean class="org.opentripplanner.graph_builder.model.GtfsBundle">
                <property name="url" value="http://www.gtfs-data-exchange.com/agency
                    /unitrans-davis/latest.zip" />
                <property name="defaultAgencyId" value="Unitrans" />
                <property name="defaultBikesAllowed" value="true" />
              </bean>

              <!-- Sac RT -->
              <bean class="org.opentripplanner.graph_builder.model.GtfsBundle">
                <property name="url" value="http://www.gtfs-data-exchange.com/agency
                    /sacramento-regional-transit/latest.zip"/>
                <property name="defaultAgencyId" value="SacRT"/>
                <property name="defaultBikesAllowed" value="true"/>
              </bean>

              <!-- Yolobus -->
              <bean class="org.opentripplanner.graph_builder.model.GtfsBundle">
                <property name="url" value="http://iportal.sacrt.com/gtfs/Yolobus/
                    google_transit.zip"/>
                <property name="defaultAgencyId" value="Yolo"/>
```

```
            <property name="defaultBikesAllowed" value="true"/>
          </bean>

          <!-- Capitol Corridor -->
          <bean class="org.opentripplanner.graph_builder.model.GtfsBundle">
            <property name="url" value="http://www.gtfs-data-exchange.com/agency
                /capitol-corridor-joint-powers-authority/latest.zip"/>
            <property name="defaultAgencyId" value="CC"/>
            <property name="defaultBikesAllowed" value="true"/>
          </bean>

        </list>
      </property>
    </bean>
  </property>
</bean>

<bean id="osmBuilder" class="org.opentripplanner.graph_builder.impl.osm.
    OpenStreetMapGraphBuilderImpl">

    <!-- Use an OSM provider that reads street data from an .osm file -->
    <property name="provider">
        <bean class="org.opentripplanner.openstreetmap.impl.
            AnyFileBasedOpenStreetMapProviderImpl">
            <property name="path" value="/otp/osm/sac2012-03-17T04:00:00Z.osm"
                />
        </bean>
    </property>

    <!-- Street Traversal Permisssion and Biking weighting is configurable, but
        we'll use the defaults for OSM...

         Street Traversal Permisssion: what modes are allowed on certain road/
            surface types (e.g., no cars on bike lanes, etc...)

         Bike weighting: the planner is using a 'shortest path' algorithm to
            choose one route from another.  To prefer certain paths
                        for biking (e.g., bike lanes), we can make the
                            theoretical lenght of a street type shorter or
                            longer, thus
                        more or less attractive to the routing engine.  Weights
                             of > 1.0 make a road type shorter (more attactive)

         @see http://wiki.openstreetmap.org/wiki/OSM_tags_for_routing/Access-
            Restrictions
         @see https://github.com/openplans/OpenTripPlanner/blob/master/
            opentripplanner-graph-builder/src/main/java/org/opentripplanner/
            graph_builder/impl/osm/DefaultWayPropertySetSource.java
    -->
    <property name="defaultWayPropertySetSource">
        <bean class="org.opentripplanner.graph_builder.impl.osm.
            DefaultWayPropertySetSource" />
    </property>
</bean>
```

```
    <bean id="transitStreetLink" class="org.opentripplanner.graph_builder.impl.
        TransitToStreetNetworkGraphBuilderImpl" />
    <bean id="optimizeTransit" class="org.opentripplanner.graph_builder.impl.
        OptimizeTransitGraphBuilderImpl" />


    <bean id="graphBuilderTask" class="org.opentripplanner.graph_builder.
        GraphBuilderTask">
        <!-- typical config -->
        <property name="graphBundle" ref="graphBundle" />
        <property name="graphBuilders">
            <list>
                <ref bean="gtfsBuilder" />
                <ref bean="osmBuilder" />
                <ref bean="transitStreetLink" />
                <ref bean="optimizeTransit" />
            </list>
        </property>
    </bean>
</beans>
```

References

[1] Lei, T. L. and R. L. Church. "Mapping Transit-Based Access: Integrating GIS, Routes and Schedules". *International Journal of Geographical Information Science* 24. 2 (2010): 283–304. 12 Apr. 2012.