P3: Wrangle OpenStreetMap Data

Edmund Wong

Map Area: Syracuse, NY, United States

1. Problems encountered in map

I surveyed the data using Python's ElementTree module. After parsing the entire Syracuse OSM file, I prepared the list of node tags, way tags, and relation tags from the XML data.

```
import xml.etree.ElementTree as ET
fname = 'syracuse_new-york.osm'
tree = ET.parse(fname)

nodetags = tree.findall('node/tag')
waytags = tree.findall('way/tag')
relationtags = tree.findall('relation/tag')
```

To audit this data, I quickly inspected the data to see what kind of common problems there are in fields I suspected to likely have problems. Most phone numbers are in the "XXX XXX XXXX" format. However, I found some phone numbers that were inconsistent to that format. Some of those problematic phone numbers contained dashes and some began with "+1" dialing code. Here is a query I used to detect and clean/fix problematic phone numbers that do not agree with this format. It requires the python phonenumbers module.

```
for idx, tag in enumerate(nodetags):
    if tag.attrib['k'] == 'phone':
        #Get value of phone number
        pn = tag.attrib['v']
        #Check whether phone number matches XXX XXX XXXX format
        if not pn[0:3].isdigit() or pn[3] != ' ' or not pn[4:7].isdigit() or pn[
7] != ' ' or not pn[8:12].isdigit() or len(pn) > 12:
            try:
                phone_obj=phonenumbers.parse(pn, 'US')
                #replace phone number with correct format
                nodetags[idx].attrib['v'] = phonenumbers.format_number(phone_obj
, phonenumbers.PhoneNumberFormat.mro).replace('-',' ')
                print "Fixed:", pn, nodetags[idx].attrib['v']
            except NumberParseException:
                print pn, "Not valid phone number"
```

Another problem is that the website URLs of organizations and businesses in Syracuse are not standardized. Some top-level-domain URLs have missing trail slashes. Here is an example where I checked and repaired for missing trailing slashes in the website URLs of. Website data exists in nodes, ways, and relations (the example below is waytag specific).

```
for idx, tag in enumerate(waytags):
    #only top-level domains
    if tag.attrib['v'].endswith(('.com', '.org', '.edu', '.gov', '.mil', '.net', '.int')):
        if tag.attrib['k'] == 'website' and not tag.attrib['v'].endswith('/'):
            waytags[idx].attrib['v'] = tag.attrib['v'] + '/'
```

2. Overview of the Data

Before getting into the overview of the data, the XML data was converted to JSON. Additionally, the data was restructured in a fashion where fields that should belong together are grouped into the same object. For example, here I grouped the "timestamp", "version", "uid", and "user" fields under "created". It also makes sense that I grouped address fields together. This format was based on James Kao. (https://jameskao.me)

An original OSM Node in XML:

```
<node id="1694014652" lat="43.0525611" lon="-76.1546779" version="6" timestamp="2015-01-12T02:07:11Z" changeset="28077497" uid="599436" user="zeromap">
    <tag k="name" v="Dinosaur BBQ"/>
    <tag k="phone" v="315 476 4937"/>
    <tag k="amenity" v="restaurant"/>
    <tag k="cuisine" v="american"/>
    <tag k="website" v="http://www.dinosaurbarbque.com/bbq-syracuse"/>
    <tag k="addr:city" v="Syracuse"/>
    <tag k="addr:state" v="NY"/>
    <tag k="addr:street" v="West Willow Street"/>
    <tag k="addr:country" v="US"/>
    <tag k="addr:postcode" v="13202"/>
    <tag k="opening_hours" v="Mo-We 11:00-23:00; Th 11:00-24:00; Fr-Sa 11:00-1:00; Su 12:00-22:00"/>
    <tag k="addr:housenumber" v="246"/>
</node>
```

The same node formatted and restructured in JSON:

```
{
    "website": "http://www.dinosaurbarbque.com/bbq-syracuse",
    "cuisine": "american",
    "amenity": "restaurant",
    "name": "Dinosaur BBQ",
    "created": {
        "changeset": "28077497",
        "user": "zeromap",
        "version": "6",
        "uid": "599436",
        "timestamp": "2015-01-12T02:07:11Z"
    },
    "opening_hours": "Mo-We 11:00-23:00; Th 11:00-24:00; Fr-Sa 11:00-1:00; Su 12:00-22:00",
    "pos": [
```

```
        43.0525611,
        -76.1546779
    ],
    "phone": "315 476 4937",
    "address": {
        "city": "Syracuse",
        "country": "US",
        "state": "NY",
        "street": "West Willow Street",
        "postcode": "13202",
        "housenumber": "246"
    },
    "type": "node",
    "id": "1694014652"
}
```


Once the data was imported into mongodb using mongoimport, I performed a couple of queries that will provide a statistical overview of the Syracuse dataset.

Size of XML: 64.2 MB
Size of JSON (after restructuring): 166.2 MB

Number of nodes:

```
> db.syracuse.find({"type":"node"}).count()
279521
```

Number of ways:

```
> db.syracuse.find({"type":"way"}).count()
35510
```

Number of unique users:

```
> db.syracuse.distinct("created.user").length
234
```

Number of restaurants:

```
> db.syracuse.find({"type":"node","amenity":"restaurant"}).count()
77
```

Number of unique contributing users:

```
>db.syracuse.distinct("created.user").length
234
```

Kinds of amenities:

```
>db.syracuse.distinct("amenity")
[ “parking_entrance”, “school”, place_of_worship”, … , “driving_range”]
```

Number of restaurants that are classified by a cuisine type:

```
> db.syracuse.find({"amenity":"restaurant", "cuisine":{$exists:1}}).count()
89
```

3. Other ideas about the datasets

Inclusion of elevation data:
From glancing at the fields of a node, there is missing elevation information that can potentially extend OpenStreetMap to allow bikers and walkers to figure out the difficulty of the commute. Elevation data in node points can also create surface data on the map to visualize hills and other topographical details. In MongoDB, the benefit of its schemaless nature allows us insert and update new kinds of data (such as elevation) into the existing collection easily. Also elevation data can be extended to Ways (Ways that dont share a start and end Node such as highways, railways, and roads) in such a way where overall elevation gain or loss can be calculated.

Potentials problems are that elevation data can be problematic to standardize, different units of measurement. It is popularily measured in both feet and meters, so some standard must first be established before incorporating elevation data into OSM. Also, accurate elevation data is not as easily acquirable as the other existing fields and may require expertise of those in the geographical sciences to validate the accuracy of such data.

Additional exploratory analyses:

In addition, I have performed other additional exploratory analyses to figure out the count of how often elements have been updated.

```
> db.syracuse.aggregate([{"$match":{"created.version":{"$exists":1}}},
{"$group":{"_id":"$created.version", "count":{"$sum":1}}}, {"$sort":{"count":-
1}}])
```

{ "_id" : "1", "count" : 202433 }
{ "_id" : "2", "count" : 93249 }
{ "_id" : "3", "count" : 12521 }
{ "_id" : "4", "count" : 3495 }
{ "_id" : "5", "count" : 1484 }
{ "_id" : "6", "count" : 703 }
{ "_id" : "7", "count" : 409 }
{ "_id" : "8", "count" : 278 }
{ "_id" : "9", "count" : 186 }
{ "_id" : "10", "count" : 91 }
{ "_id" : "11", "count" : 47 }

{ "_id" : "12", "count" : 33 }
{ "_id" : "13", "count" : 26 }
{ "_id" : "14", "count" : 25 }
{ "_id" : "16", "count" : 13 }
{ "_id" : "15", "count" : 9 }
{ "_id" : "17", "count" : 8 }
{ "_id" : "19", "count" : 8 }
{ "_id" : "18", "count" : 7 }

This output shows the most of these elements (nodes, ways) have been updated only once, and almost less frequently for each additional version change.