# OpenStreetMap 数据分析

## 1. 地图范围

芝加哥，伊利诺伊，美国

- http://metro.teczno.com/#chicago (http://metro.teczno.com/#chicago)

我在2014年10月去过芝加哥，并且本课也以芝加哥作为案例，所以我想继续深入分析芝加哥数据。

## 2. 地图的问题

由于芝加哥地图文件比较大（1.9G），我用“项目详情”中提供的代码提取了样本数据(这部分数据有**190M**，即原始数据的**1/10**)，用data.ipynb文件（该文件大部分内容由“案例研究”的最后一个练习提供）输出了5个csv文件，从这些CSV文件中我发现了如下问题：

- 街道名过于简化，如Street写成St，Avenue写成Ave；
- 州名Illinois，简写为IL；
- 名称重复，如："Cook,Illinois,Ill.,IL,USA"显然应该是"Cook,Illinois"
- 有些key值有大写字母，如：“gnis：County”，(拆分后County会变成key值，gnis变成type值)，因为大部分county都为小写字母，格式不统一；
- 大量的value值含有'_'，如bus_station

注：在对“样本数据”清洗成功后，再对原始文件（1.9G）进行清洗，以下所有计算的数据都来自于原始文件（1.9G）。

## 3. 对数据进行清洗

### 街道名的清洗

街道名的清洗借鉴了“案例研究”中"练习"的思路，首先定义一个mapping的字典，然后编写一个替换函数update_name。

In [7]:

```
mapping = { "St": "Street",
            "St.": "Street",
            "Rd.":"Road",
            "Ave":"Avenue",
            "Rd":"Road",
            "IL":"Illinois"}
street_type_re = re.compile(r'\b\S+\.?$', re.IGNORECASE)
def update_name(name):
    m = street_type_re.search(name).group()
    if m in list(mapping):
        name = name.replace(m,mapping[m])
    return name
```

在应用中，要判断是否type == 'addr' and key == 'street'，如果为真，就把value值带入上述update_name函数中进行分析，返回更正后的value值。

## 州名的替换

州名的替换仍然用上述方法，所不同的是要先判断key值为['is_in','county','state']中的一个。

In [ ]:

```
if key in ['is_in','county','state']:
    ele.attrib['v'] = update_name(ele.attrib['v'])
```

## 名称重复

目前只找到："Cook,Illinois,Ill.,IL,USA"这个值，直接替换即可。

In [ ]:

```
if ele.attrib['v'] == 'Cook,Illinois,Ill.,IL,USA':
    ele.attrib['v'] = 'Cook,Illinois'
```

## **key**值大写字母

用lower()函数

## 含有**"_"value**值

'_'主要集中在key值为'highway'，'railway'，'amenity'对应的value值中，用replace()函数直接替换

In [ ]:

```
if key in ['highway','railway','amenity','service','leisure','grass']:
    ele.attrib['v'] = ele.attrib['v'].replace('_',' ')
```

In [ ]:

```
```

### 对数据清洗的益处和一些预期的问题

显然，我们做以上的清洗工作就是为了减少数据错误，将数据导入数据库后，查询更方便，不出现漏查的情况。

- 对街名简化的纠正的好处是，比如我想查某条大街房屋的数量，比如"Michigan Avenue",原始数据可能有的写成了"Michigan Ave",如果不清洗数据，这时候我用"Michigan Avenue"就查不到"Michigan Ave"相关的信息；预期的问题：仍然有一些简写没有改正，这样的数据会让非英语国家的人难以理解。
- 对于州名清洗和上述原理一样，可能在本项目并不明显，因为都是伊利诺伊州的数据，但是如果下载更大的地图，比如要查伊利诺伊的图书馆数量，限制条件是写“IL”还是“Illinois”呢，显然还是统一格式比较好；预期的问题：有时候针对特定范围的问题，有些清洗没有意义。在本项目，对**“IL”**都改成**‘Illinois’**可能就没有必要，因为所有的数据都在伊利诺伊，查询的时候就没有**"Illinois"**的限制条件。
- 名称重复“'Cook,Illinois,Ill.,IL,USA'”是错误，改正过来比较好；预期的问题：这种重复的错误肯定在数据集的其他位置中还存在。
- 对于key值大小写问题，还是应统一写成小写，因为Sqlite对大小写敏感，查询的时候不容易漏项；预期的问题：无。我只是对**key**值大小写问题进行清洗；（如果对**value**值大小写更改，就会改变英语的语法习惯）。
- 带有"_"的值是否该替换成空格？这仍是个格式统一的问题，根据语言习惯，需要进行纠正。预期的问题：下横杠的清洗可能会清洗一掉一些特定名称的下横杠。

# 4. 对数据库进行分析

### 文件大小

In [ ]:

```
chicago.osm ..............1,965,808 KB
chicago.db................1,449,367 KB
nodes.csv.................723,680 KB
node_tags.csv.............8,496 KB
ways.csv..................75,576 KB
ways_nodes.csv............232,644 KB
ways_tags.csv.............177,897 KB
```

### 节点数量

In [ ]:

```
sqlite> SELECT COUNT(*) FROM nodes;
```

8410774

## 途径数量

In [ ]:

```
sqlite> SELECT COUNT(*) FROM ways;
```

1189174

## 多少个唯一用户

In [ ]:

```
sqlite> SELECT COUNT(DISTINCT(e.uid))
FROM (SELECT uid FROM nodes UNION ALL SELECT uid FROM ways) e;
```

2327

## 排名前**10**的贡献用户

In [ ]:

```
sqlite> SELECT e.user, COUNT(*) as num
FROM (SELECT user FROM nodes UNION ALL SELECT user FROM ways) e
GROUP BY e.user
ORDER BY num DESC
LIMIT 10;
```

In [ ]:

```
chicago-buildings|5626906
Umbugbene|1100615
alexrudd (NHD)|232625
woodpeck_fixbot|225389
patester24|109155
TIGERcnl|105497
mpinnau|104667
asdf1234|101209
g246020|99595
Sundance|84049
```

通过上面的数据，我们发现排名前10的用户贡献了81.1%的数据，特别是排名第一的用户贡献了58.6%的数据，有563万条数据，这么庞大的数据显然是机器人程序输入的。我认为用户为Openstreetmap提供数据的主要动机是获得成就感。显然Openstreetmap为了鼓励这种行为，应该在网页的显著位置提供

贡献者的“天梯排名”，这样可以激发越来越多的用户参与到上传地图信息的活动中来。另外Opensteetmap应该对排名靠前的用户提供奖励，毕竟前几名提供了大部分的数据。可以按月统计贡献量，颁发证书或者送一些小礼品。

## 排名前十的便利设施的数量

In [ ]:

```
sqlite> SELECT value, COUNT(*) as num
FROM nodes_tags
WHERE key='amenity'
GROUP BY value
ORDER BY num DESC
LIMIT 10;
```

In [ ]:

```
place of worship|3057
school|1950
restaurant|1424
fast food|840
parking|593
cafe|399
fuel|348
bench|331
bicycle rental|327
bank|307
```

## 图书馆的数量和分布

我去过“芝加哥公共图书馆”，它是芝加哥的标志性建筑，拥有“世界上最大的公共图书馆”的美誉，所以我想对芝加哥的图书馆进行统计。

In [ ]:

```
sqlite> SELECT count(distinct(id)) from nodes_tags
        WHERE value = 'library'
        AND key = 'amenity';
```

图书馆的数量为130个。另外我想统计一下芝加哥下属各个县的图书馆数量。

In [ ]:

```
sqlite> SELECT nodes_tags.value, COUNT(*) as num
        FROM nodes_tags
        JOIN (select distinct(id) from nodes_tags where value = 'library'and key = 'am
enity') i

        ON nodes_tags.id = i.id
        WHERE nodes_tags.key = 'county_name'
        GROUP BY nodes_tags.value
        ORDER BY num DESC;
```

In [ ]:

```
Cook|44
Lake|21
DuPage|11
Will|6
Kane|3
Kendall|2
McHenry|1
```

发现库克县（Cook）有44个图书馆，占了整个大芝加哥地区图书馆数量的1/3。在网上查资料得知，库克县是伊利诺伊州人口最多的县，也是全美人口第二多的县，仅次于洛杉矶县。所以图书馆多也不足为奇了。另外在查询的时候，发现很多的图书馆的信息不完整，有的给出了位置信息如city：Evanston， city：Chicago，county_name:Cook等等，有的什么也没有,这说明地图信息还是有改进的空间。

我建议，可以通过节点的经纬度进行定位从而获得缺失的位置信息，可程序化实现。 说明：我的以上的查询是在**node_tage**表下查询的，本身就是**node_tage**表就和**node**表进行关联，而**node**表有经纬度的信息，这样就知道这个特定便利设施的经纬度了；我们还需要一个数据集（外部的数据集），就是给出经纬度就可以知道在哪个城市或者哪个县，这样就可以补充便利施舍的位置信息了。

这么做的益处：在经纬度的检查时候，可能上述**130**个图书馆有重复的（即坐标重合的），那么找出重复数据并剔除，最终就可以精确给出芝加哥“市内”有多少个图书馆，周边各个县有多少个图书馆。

预期的问题：需要“外部数据集”的数据收集，可能要涉及各个城市和县边界的处理，而且有些建筑本身就建在两个城市的边界处，可能还得需要人工判断。

# 结论

本作业，首先清洗了数据，对过于简化的拼写和多余的'_'进行了修正，另外还清理了key值大写问题，并且对清洗的益处和预期的问题进行了讨论。 在数据分析过程中，分析了文件文件的大小、唯一用户数量、节点和途径的数量、常用便利设施的数量，另外对图书馆的总数量和芝加哥周边各县的图书馆数量进行了统计。在分析过程中发现，地图文件并不完整，有些设施缺少位置信息，并对该问题给出了建议。由于排名前十的用户提供了80%以上的信息，我建议Openstreetmap应该鼓励这些用户，比如在网页的显著位置展示天梯图，提供一些奖励等，激发一些潜在用户上传地图信息。