Advanced Geospatial Data Analytics:

MapReduce and Geospatial data handling

Junjun Yin

CyberGIS Center for Advanced Digital and Spatial Studies
Department of Geography and Geographic Information Science
National Center for Supercomputing Applications (NCSA)
University of Illinois at Urbana-Champaign, IL, 61801
jyn@illinois.edu

March 8th, 2016

Outline

- Python programing basics
- Hadoop Streaming API with Python
  - Take advantage of the mapper for parallel processing
- Geospatial processing in Hadoop
  - Case study
  - Existing tools and frameworks
  - Write your own geospatial processing pipeline with Python
  - Leveraging the existing geospatial libraries as packages in Python

Python Programing Basics

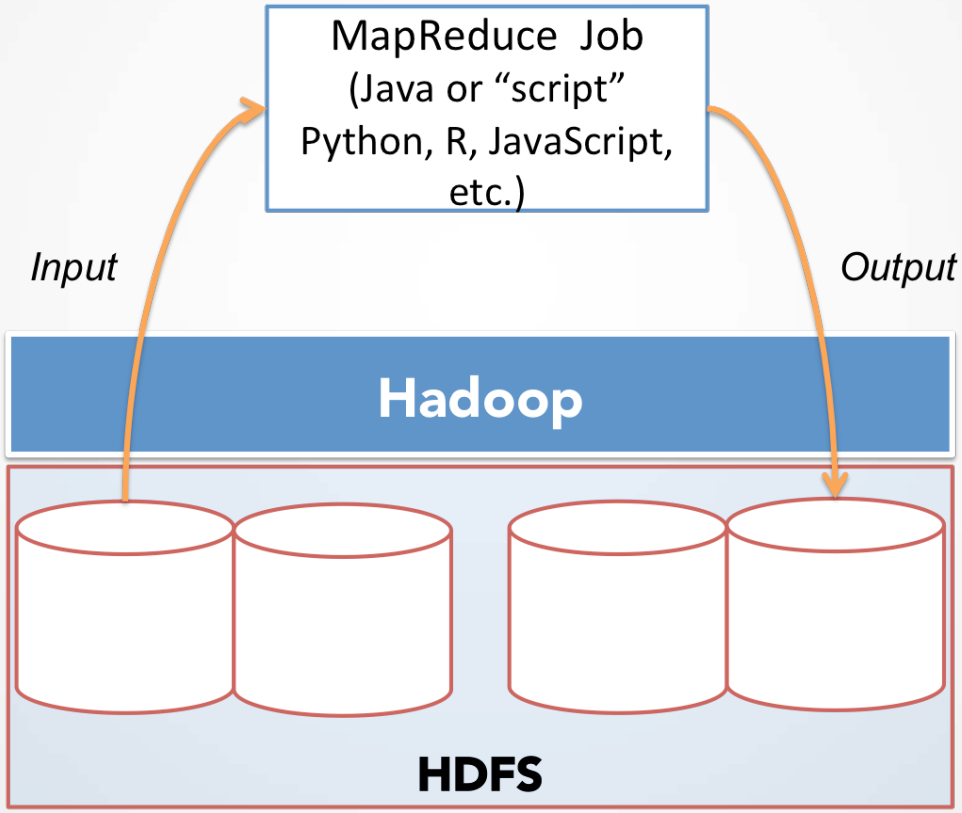
- For this unit, let's follow the attached document and review some basics in Python programing
- Different coding environments
  - Choose your favorite code editor

Hadoop Streaming API with Python

MapReduce

- Let's recap on what is the MapReduce paradigm
- Can you separate each step as individual process
  - If yes, how?
- Can you come up with some scenarios?

Delegation of Map and Reduce tasks



Example

```
import time
import datetime

mFile = open("ny_taxi_2013.csv","rb")
mOutput = open("ny_taxi_2013_unix.csv","wb")

for line in mFile:
    lineArray = line.split(',')
    pickup = lineArray[5]
    dropoff = lineArray[6]
    pickup_date = pickup.split(" ")[0]
    dropoff_date = dropoff.split(" ")[0]
    pickup_unix = datetime.datetime.strptime(pickup,"%Y-%m-%d %H:%M:%S").strftime("%s")
    if pickup_date > '2013-03-09' and pickup_date < '2013-11-03':
        pickup_final = int(pickup_unix) - 5 * 3600
    else:
        pickup_final = int(pickup_unix) - 6 * 3600
    ...
    newLine = line.replace("\n",",") + str(pickup_final) + ',' + str(dropoff_final) + '\n‘
    mOutput.write(newLine)
```

Hadoop streaming API with Python

```
#!/usr/bin/env python
import sys
import os
import time
import datetime

for line in sys.stdin:
    lineArray = line.strip().split(',')
    pickup = lineArray[5]
    dropoff = lineArray[6]
    ...
    newLine = line.replace("\n",",") + str(pickup_final) + ',' + str(dropoff_final) + '\n‘

    print newLine
```

Run Hadoop Streaming Job

- **hadoop jar** /usr/hdp/2.3.2.0-2602/hadoop-mapreduce/hadoop-streaming-2.7.1.2.3.2.0-2602.jar **-file** mapper.py **-mapper** mapper.py **-input** ny_taxi_2013.csv **-output** ny_taxi_2013_unix.csv

- **yarn jar** /usr/hdp/2.3.2.0-2602/hadoop-mapreduce/hadoop-streaming-2.7.1.2.3.2.0-2602.jar **-file** mapper.py **-mapper** mapper.py **-input** ny_taxi_2013.csv **-output** ny_taxi_2013_unix.csv

Using Python libraries for data analytics

- One step further
  - Numpy, Scipy, Pandas, scikit-learn
- Problems
  - Module load does not work
  - Each node in Hadoop cluster should have the python framework enabled
- Solution
  - Virutalenv (python virtual environment)

Using Python libraries for data analytics

- >>> virtualenv-2.7 demoenv
- New python executable in demoenv/bin/python2.7 ... done.
- >>> virtualenv-2.7 --relocatable demoenv
- >>> source demoenv/bin/activate
- >>> pip install “some module”
- >>> zip –r ../demoenv.zip *
- >>> hdfs dfs –copyFromLocal demoenv.zip

Using Python libraries for data analytics

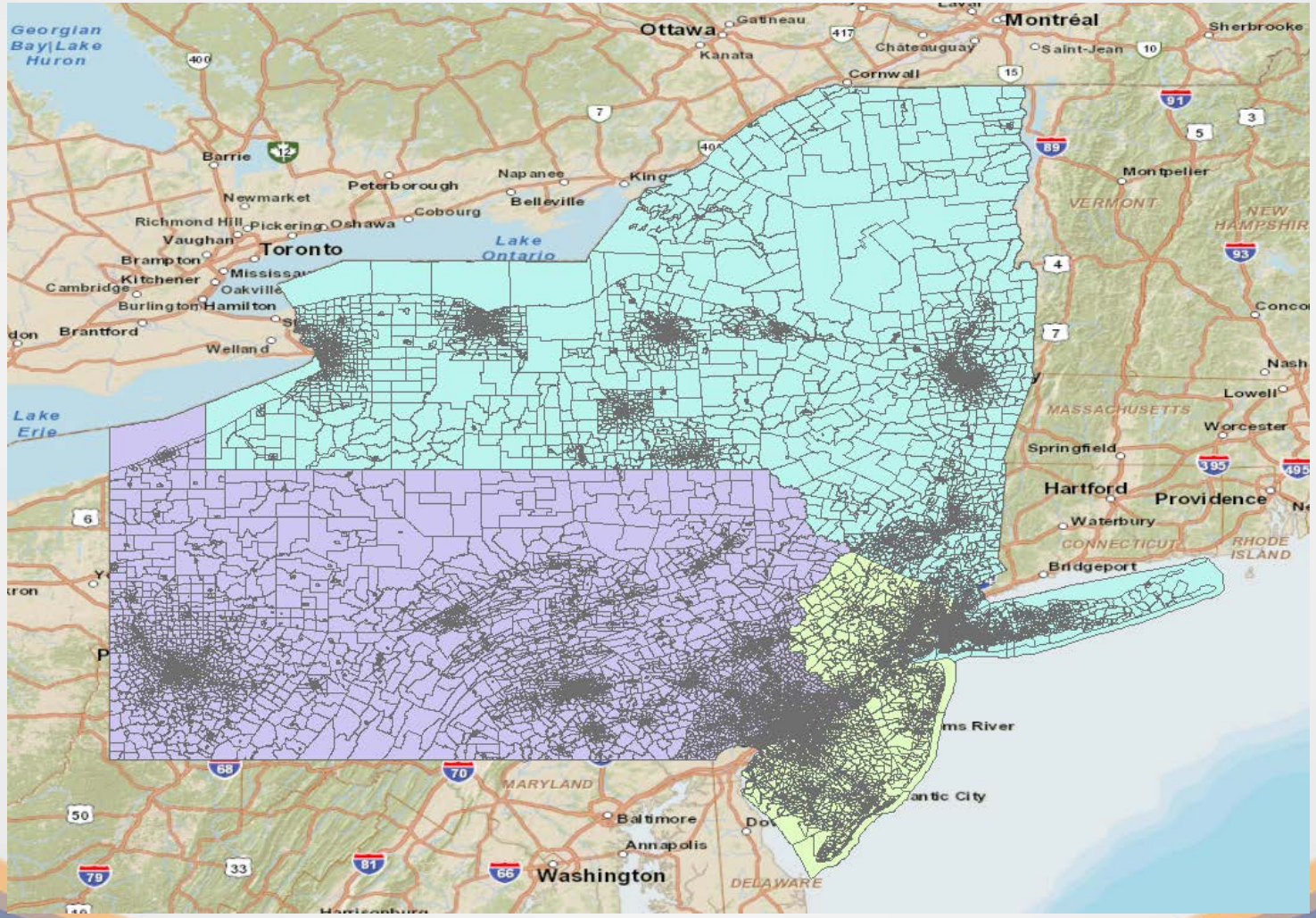
```
hadoop jar /usr/hdp/2.3.2.0-2602/hadoop-mapredu
    ce/hadoop-streaming-2.7.1.2.3.2.0-2602.jar
....files
-archives hdfs://cg-hm11.ncsa.illinois.edu/user/jyn/
    demoenv.zip#demoenv
```

In your python script

```
#!./demoenv/bin/python
import your_module
```

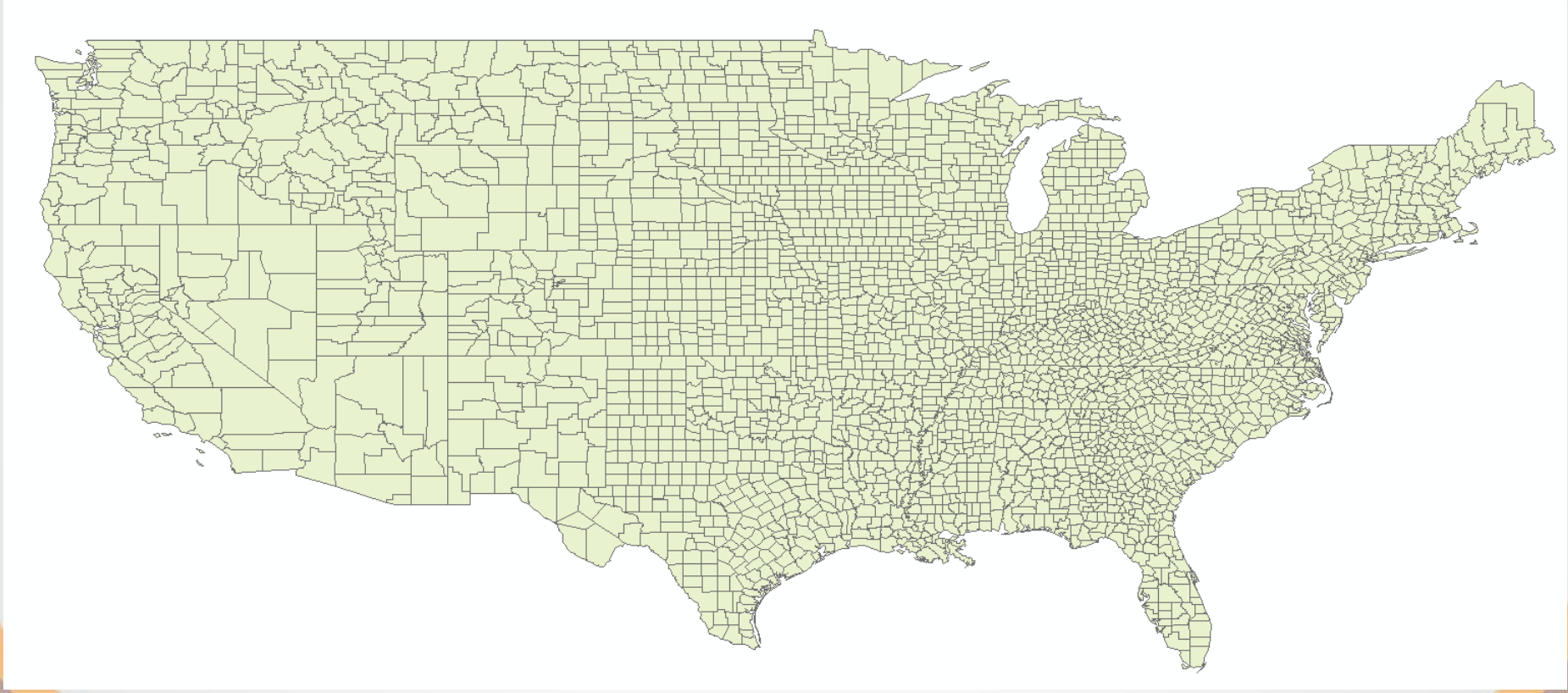
Geospatial processing in Hadoop

Case study



Aggregating New York taxi records to traffic analysis zones

Case study



Aggregating Twitter data to the corresponding county

Ideas?

Geospatial Operations

- Topology operations
  - Validating geometry, geometry relationships
  - Intersection
  - Intersect
  - Overlapping
  - Touch (sharing a boundary)
  - etc.
- Spatial Query
  - Geo-within (Range query)
  - K-NN (nearest neighbors)
  - etc.

Open Source Geospatial Libraries

- SDBMS (spatial database management system)
- QGIS
- ArcGIS
- GRASS GIS

- Open source libraries: OSGeo
  - GDAL
  - Fiona
  - Shapely
  - Shapefile.py

Existing distributed computing resources

- SpatialHadoop
  - http://spatialhadoop.cs.umn.edu/
- SpatialSpark
  - https://github.com/syoummer/SpatialSpark

Shapefile.py

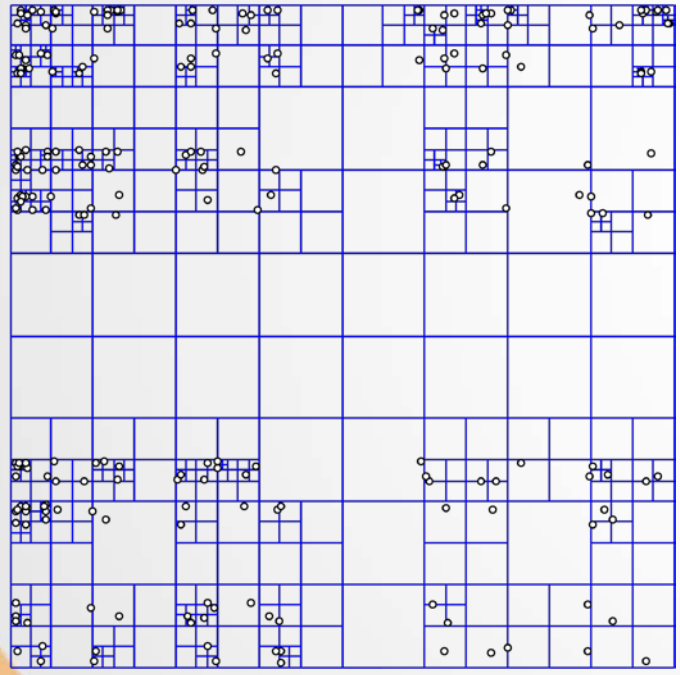
- Basically, instead of
- https://pypi.python.org/pypi/pyshp

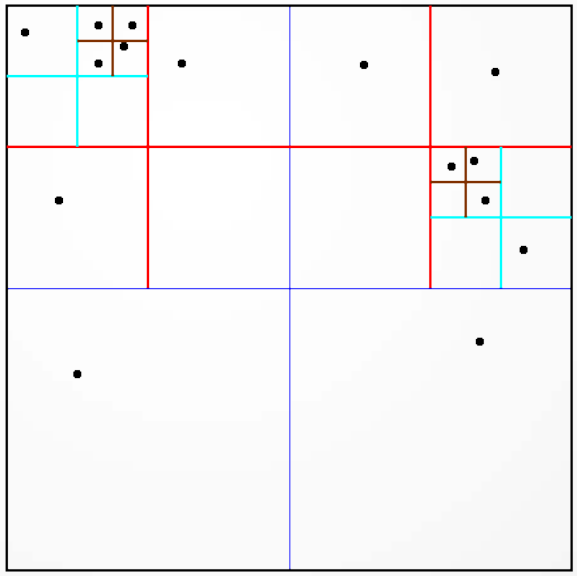
- Example usage
- >>> myshp = open("shapefiles/blockgroups.shp", "rb")
- >>> mydbf = open("shapefiles/blockgroups.dbf", "rb")
- >>> r = shapefile.Reader(shp=myshp, dbf=mydbf)

Improve efficiency with spatial indexing

- The purposes of spatial index
  - Improve search efficiency
  - Deal with multi-dimensional dataset
- Different types of spatial index
  - Quad-Tree
  - R-Tree
  - octree
  - Kd-tree
  - ....
- For this course, we will demonstrate and use Quad-Tree

Illustration of Quad-Tree for indexing points





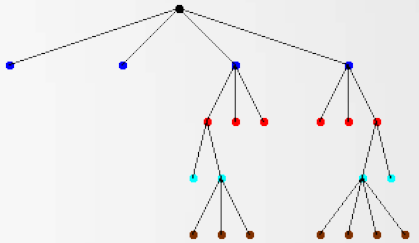
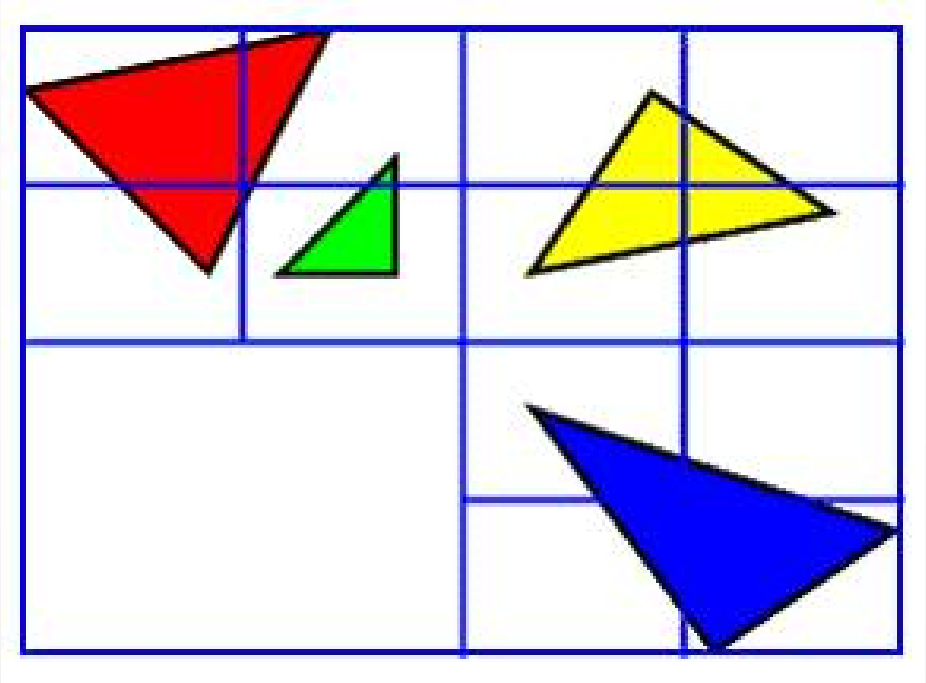
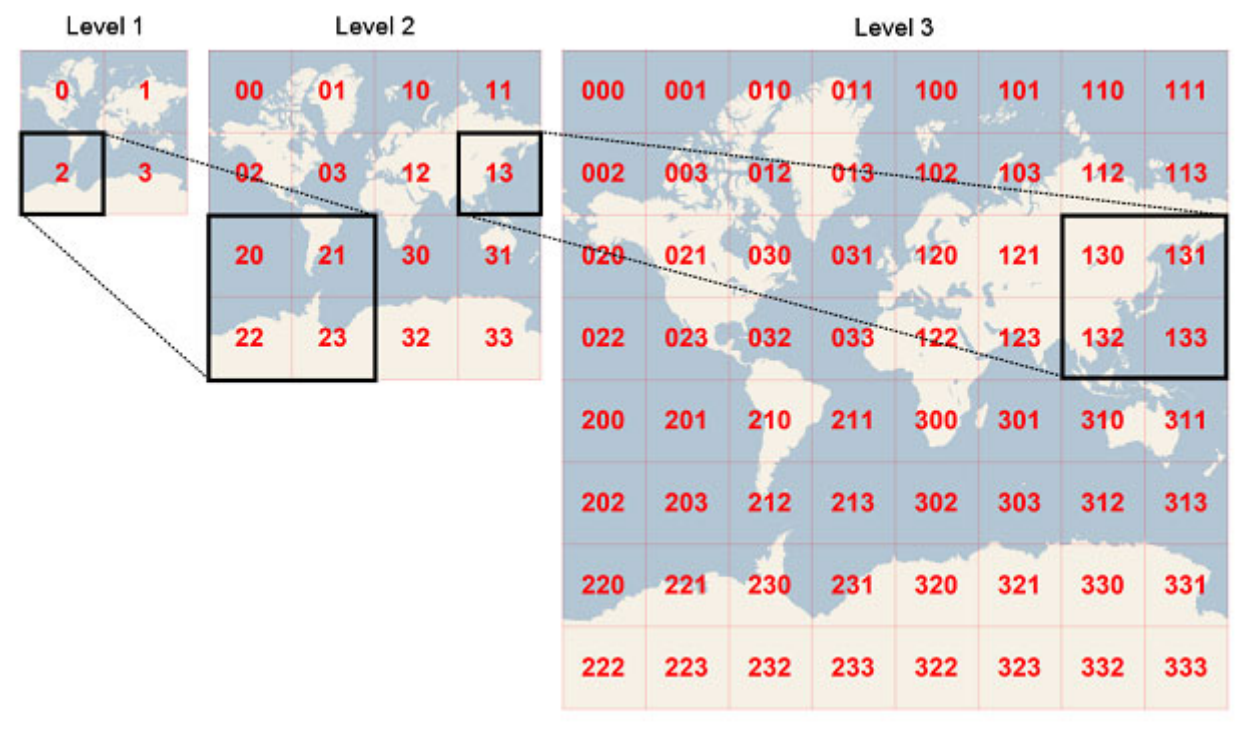


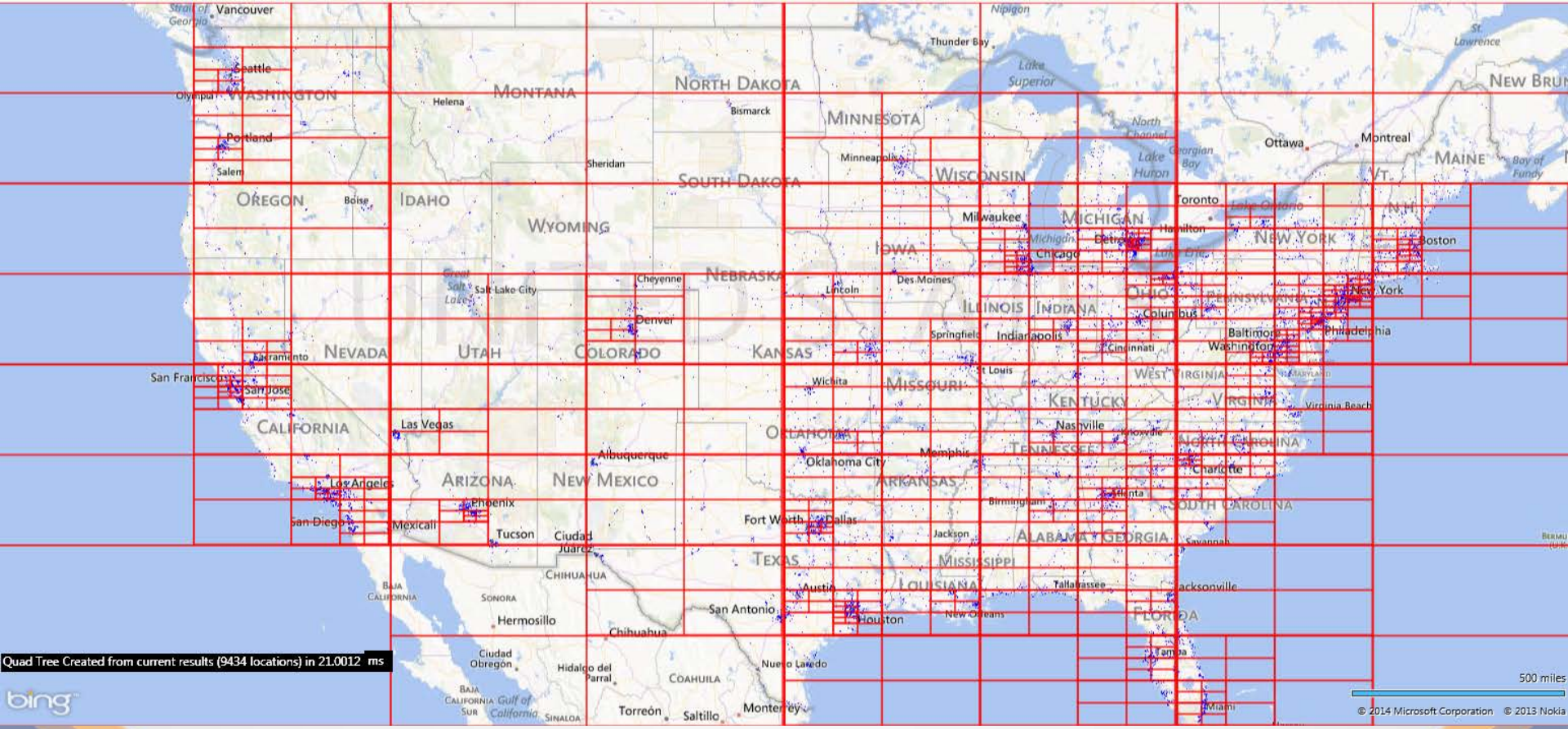
Illustration of Quad-Tree for polygons



Applications of Quad-Tree



Applications of Quad-Tree



Source: *http://softwareunwound.com/2014/03/18/adventures-in-mapping-big-data-sets-in-real-time-map-controls-part-2-enter-the-quadtree/*

Put things together

- Step1
  - We need to create spatial index (quad-tree) for the data we are about to process
  - Which data then? The point data or the polygon layer data?
- Step2
  - What we should do to determine a point belongs to a specific polygon?
  - Which geospatial operation should you pick?
- Step3
  - How can we take advantage of Hadoop Streaming API with Python
- Step4
  - How to run our program?

Notes

The detailed procedures will be practiced in the our lab this Thursday.