Mobility data storage and analysis

Antònia Tugores, Pere Colet
Index

- Motivation
- Data storage
- Insert/Query performance
- Preliminary results and on going work
Motivation

Study urban mobility by using their tweets (specially the geolocated ones)

Compare with other on going studies
- Barcelona
- London
- Zurich

Not only tweets (in the future)
**JSON document**

JavaScript Object Notation
text-based open standard
designed for human-readable data interchange
language-independent
parsers available for many languages
alternative to XML
fields are in an arbitrary order

**GeoJSON**

open format for encoding collections of simple feature objects
points
line strings
polygons
and multi-part collections of points, lines and polygons
with their non-spatial attributes
using JavaScript Object Notation
Tweet

(+ RT info)

tweet content (140 characters)

id

geolocation (if enabled)

source

user information

creation time

place (if defined)

fields are in an arbitrary order

fields scheme can change
User Information

fields are in an arbitrary order

```
"user":
{
  "follow_request_sent": false,
  "profile_use_background_image": true,
  "contributors_enabled": false,
  "id": 260287521,
  "verified": false,
  "profile_image_url_https": "https://si0.twimg.com/profile_images/1533280952/tw_12403594_1315422640_normal.jpg",
  "profile_sidebar_fill_color": "252429",
  "profile_text_color": "609666",
  "followers_count": 128,
  "protected": false,
  "id_str": "260287521",
  "default_profile_image": false,
  "location": "L'Hospitalet de Llobregat",
  "utc_offset": 3600,
  "statuses_count": 1747,
  "description": "Necesito mettere algo dentro, un poco de café o algo... Y luego, de alguna manera, el mundo será un poco mejor. [Sam Yimes, Hombres de Armas, Terry Pratchett]",
  "friends_count": 570,
  "profile_link_color": "AA0000",
  "profile_image_url": "http://a0.twimg.com/profile_images/1533280952/tw_12403594_1315422640_normal.jpg",
  "notifications": null,
  "geo_enabled": true,
  "profile_background_color": "000000",
  "profile_background_image_url": "http://a0.twimg.com/profile_background_images/60083524/n9duka667b9ju3n9g4w4.jpeg",
  "screen_name": "jX09A",
  "lang": "ca",
  "following": false,
  "profile_background_tile": false,
  "favourites_count": 7,
  "name": "Jordi Cant\\u00f3\",
  "url": null,
  "created_at": "2011-03-03T15:45:46",
  "profile_background_image_url_https": "https://si0.twimg.com/profile_background_images/60083524/n9duka667b9ju3n9g4w4.jpeg",
  "time_zone": "Madrid",
  "profile_sidebar_border_color": "181A1E",
  "default_profile": false,
  "is_translator": false,
  "listed_count": 1
}
```
Data size estimation

2KB/tweet

Tweet: aprox. 20 key/value
User: aprox 30 key/value

- id
- content (140chars)
- geolocation
- creation time
- place
- user information
  - id
  - screen name
  - description
  - creation time
  - location
  - friends count
  - ....
  - ...

PyConES 2013
Data size estimation

Tweet: aprox. 20 key/value
User: aprox 30 key/value

id
ccontent (140chars)
geolocation
creation time
place
user information
id
screen name
description
creation time
location
friends count
...

2KB/tweet

× 15million tweets/day
30 GB/day

2500million tweets/year
6TB/year
big data is...
Big data is like teenage sex: everyone talks about it, nobody really knows how to do it, everyone thinks everyone else is doing it, so everyone claims they are doing it ...
Big data

Volume

Data Size

Data Complexity

Speed of Change

Velocity

Data Sources

Variety

PyConES 2013

http://ifisc.uib-csic.es
We need to...

Efficiently store data
Efficiently manage data (select subsets)
Efficiently analyse data
Requirements to store and manage the data

* Capability to store **billions** of documents

* Fast storage rate

* Scalability

* High availability

* High search performance

* Adaptative format (twitter can change the data format)
Databases

The Windows Club

http://blog.izenda.com/wp-content/uploads/2013/05/SQL-600x309.png
Databases

Relational (SQL) databases
- has a collection of tables of data items all of which is formally described and organized according to the relational model

Examples
- Oracle
- PostgreSQL
- MySQL

Non-relational databases
- A NoSQL database provides a mechanism for storage and retrieval of data that uses looser consistency models than traditional relational databases.

Examples
- MongoDB
- CouchDB
- HyperDex
Databases

Relational (SQL) databases

- Category
  - Name
  - Url

- User
  - Name
  - Email Address

- Article
  - Name
  - Slug
  - Publish date
  - Text

- Tag
  - Name
  - Url

- Comment
  - Comment
  - Date
  - Author
Non Relational databases
## DB comparison

<table>
<thead>
<tr>
<th></th>
<th>SQL</th>
<th>noSQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>High performance</td>
<td>Indexes</td>
<td>Indexes</td>
</tr>
<tr>
<td>High availability</td>
<td>Automatic master failover and recovery</td>
<td>Replica sets with automatic master failover and recovery</td>
</tr>
<tr>
<td>Easy scalability</td>
<td>Hardware: $$$$$$$ Soft: MySQL or $$$$$$  = $$$$$$$$$$</td>
<td>Hardware: Add ANY computer Soft: MongoDB = Automatic sharding</td>
</tr>
</tbody>
</table>
Empiric comparison: SQL vs noSQL
MySQL configuration

Physical computer:
16GB RAM
8 cores (2x Xeon L5520 @ 2.27GHz)
2TB, 7200 rpm

Indexing:
id
user.id
coordinates.coordinates
created_at
MongoDB minimal configuration

MongoDB “Data”
Holds the data
It can be
- A single server with the data
- A replica set with at least 3 servers with data replicated among them for security.
  One of the elements of the replica set acts as primary and the others are secondaries.
  Some of the secondary one can replicate the data with a delay time
  so that they can be used a back up.

Client / Application server
routes the reads and writes from applications to the data
MongoDB sharding

Several MongoDB shards (each formed by a replica set with at least three servers)
Data is distributed among the instances (sharding).

MongoDB configuration servers
distribute the data among the MongoDB instances (chunks)
hold the metadata of the cluster (relations between shard and chunks)

Client / Application server
routes the reads and writes from applications to the shards
MongoDB configuration

Virtual computers:
- 1-2GB RAM
- 1-2 cores

Physical computers:
- 16GB RAM
- 8 cores (2x Xeon L5520 @ 2.27GHz)
- 2TB, 7200 rpm

Indexing:
- id
- user.id
- coordinates.coordinates (2d)
- created_at

Config servers

1. mongod CF1
2. mongod CF2
3. mongod CF3

client

mongos CL

RS1
- mongod
- mongod
- mongod

RS2
- mongod
- mongod
- mongod

RS6
- mongod
- mongod
- mongod
Data collection

Twitter APIs.

- **StreamAPI**
  
  Client receives a small sample of all public statuses (aprox 1%)
  
  Only 12% of them are geolocalized
  
  and of those, only a small fraction are in the cities of interest.

- **REST API**

  Allows to get a users’ timeline.

  We use that for the most active users located in the cities of interest.

  After one year of data collection, in London there are 240K users and 6M tweets

We use **tweepy** to interface with the Twitter APIs

**simplejson** to convert from/to text/JSON

**pymongo** to insert the data to MongoDB database

**django-orm** to insert the data to MySQL database
Data collection

Stream

```python
from tweepy import Stream, OAuthHandler
from tweepy.streaming import StreamListener

class StdOutListener(StreamListener):
    def on_data(self, data):
        print(data)
        return True

auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(ACCESS_KEY, ACCESS_SECRET)
listen = StdOutListener()
stream = Stream(auth, listen, gzip=True)
stream.sample()
```
Data collection

Users timeline

continuously:
get user ids with tweets geolocated in the cities of interest
(sorted by number of geolocated tweets)
for the first 43200 uids (more active)
get timeline (from last stored tweet until now)

Users network

continuously:
get user ids with tweets geolocated in the cities of interest
(sorted by number of geolocated tweets)
for all of them
get current list of following and followers (uids)
Data collection

Write to DB (MySQL with Django-ORM)

```python
class Tweet(Model):
    twid = BigIntegerField(primary_key=True, db_index=True)
    place = ForeignKey(Place, null=True)
    text = CharField(max_length=2048, blank=True)
    retweet_count = IntegerField(null=True)
    parent_id = BigIntegerField(null=True)
    source = CharField(max_length=2048)
    coordinates = ForeignKey(BoundingBox, null=True)
    contributors = CharField(max_length=2048, null=True)
    retweeted = BooleanField()
    truncated = BooleanField()
    created_at = DateTimeField(null=True)
    user = ForeignKey(User)
    entities = ForeignKey(Entities, null=True)
    in_reply_to_status_id = BigIntegerField(null=True)
    in_reply_to_user_id = BigIntegerField(null=True)
    in_reply_to_screen_id = BigIntegerField(null=True)
    deleted = BooleanField()

    class Meta:
        app_label = 'twitter'
```

Idem for line in tweets_file:
```python
tweet = fillTweet(line)
tweet.save()
```
Data collection

Write to DB (MongoDB)

```python
mongoserver_uri = "mongodb://" + user + ":" + pwd + "@" +
    host + ":" + port + "/" + dbname
conection = MongoClient(host=mongoserver_uri)
db = conection[dbname]
collection = db[collname]

for line in tweetsfile:
    tweet = simplejson.loads(line.encode('utf8'))
    collection.insert(tweet)
```
Insertion performance
(time to insert 100K tweets)

**MySQL**

- Empty DB
- Django ORM (+ relations)

**MongoDB**

- 850 million tweets
- 3 replicaset
- JSON insertion + pymongo
  (200 tweets/second)
Query performance (MongoDB)

database with 1 thousand million tweets
geonear: get the closest documents to a given point with a maximum distance
tessellate the city maps with cells of approx 1km² (16MB size)
run a query per cell

Insertion speed when querying the primary node. (not production)

Queries timing histogram for Barcelona metropolitan area. Blue line shows the median and red line the 70th percentile.
MongoDB configuration improvements

2 mongos CL
Insertion in bunches
MongoDB issues

- Documents (JSON) and Javascript for queries
- Authorisation/authentication → 2.4
- Default write concern
- Synchronization between primary and secondaries
- Geoindexes: 2d and 2dsphere
- “Big” data backups are not easy → use delayed members in RS
Twitter (and other social network) data issues

- Bias: more tweets in cities
- Signal: people not represented
- Scale: sometimes, small data is better than big data
- Correlations between different subsets
- Big bad data (data quality analysis)
- ....
Preliminary results and on going work

2 billion tweets stored in the database.
250 million geolocated (aprox 13%)

5.9 millions located in London
1.5 millions located in Barcelona
0.2 millions located in Zurich

Users with at least one tweet in the city
260 thousand users in London
25 thousand users in Barcelona
3 thousand users in Zurich

Preliminary data shows a good agreement with population distribution and the transportation network of the cities

On going work: more detailed comparison with traffic data of the transportation networks with traditional datasets within the European project EUNOIA
#ViaCatalana

11/09/2013: demonstration in the center of Barcelona and a human chain crossing Catalonia from North to South along the Mediterranean coast.

http://ifisc.uib-csic.es/humanmobility

aprox 175000 tweets
4600 geolocated
The Hard Life of a NoSQL Coder

Part 1: The Outing

Thank you

Questions?