Towards Parallel Detection of Moving Flock Patterns in Large Spatiotemporal Datasets

Andres Calderon

December 1, 2016
Trajectory Datasets

- Sensors, sensors everywhere...
  - Smart phones, GPS, RFID, WiFi, Bluetooth, IoT, Remote sensing...

Hurricane Trajectories, 1851-2012, with 10 Simulated Ones Atop

http://tinyurl.com/h4mbvxxz
Applications

http://tinyurl.com/jfm8qfu
GeoLife: Building Social Networks Using Human Location History

Established: February 6, 2009

GeoLife is a location-based social-networking service, which enables users to share life experiences and build connections among each other using human location history. Dr. Yu Zheng started this project in 2007 with his team.

Application Scenarios

- GeoLife enables user to share travel experience using GPS trajectories.
- By mining multiple users’ location histories, GeoLife can discover the top most interesting locations, classical travel sequences and travel experts in a given geospatial region, hence enable a generic travel recommendation.
- By understanding individual location history, GeoLife can measure the similarity between users and perform personalized friend & location recommendation.

http://tinyurl.com/hpd4nxl
Applications

**eBird animated migration map**

An animated map of the Western Hemisphere shows the paths of more than 100 bird populations as they migrate throughout the year.

The map was created by researchers at the Cornell Lab of Ornithology, who plotted the routes of these groups to understand their paths across land and the open ocean.

As revealed in the moving map, the team found wide similarities in the migration routes of different groups of species.

[Map of migration routes](http://tinyurl.com/hkc6ahl)

Color-coded dots show the trajectories of these birds as they head southward in the fall. Dark blue dots show the birds during January, with light green representing June-July, and red showing December.
Outline

1. Moving Flock Patterns
2. Implementation
3. Experiments
4. Conclusions
What is a flock???

**Definition**\(\left(\mu, \epsilon, \delta\right) \rightarrow \text{flock}\)

Sets of at least \(\mu\) objects moving close enough (\(\epsilon\)) for at least \(\delta\) time intervals \((\text{Benkert et al, 2008})\).
BFE algorithm (Vieira et al, 2009)
BFE algorithm  (Vieira et al, 2009)
BFE algorithm (Vieira et al, 2009)
BFE algorithm (Vieira et al., 2009)
BFE algorithm  (Vieira et al, 2009)
BFE algorithm (Vieira et al, 2009)
BFE algorithm  (Vieira et al, 2009)
BFE algorithm (Vieira et al, 2009)
Why am I doing this???

- Why are moving flock patterns important?
  - They capture the collective behavior of trajectories as groups.

- Why is the finding of disks important?
  - It is the base of the algorithm but it has a high complexity ($O(2n^2)$).
  - It is no trivial, disks can be at any location.
Outline

1. Moving Flock Patterns
2. Implementation
3. Experiments
4. Conclusions
Demo time:
  http://tinyurl.com/jl55849.
DISTANCE JOIN does not work... #71

Hello there...

First of all, congratulations for your great job. I have been working on Simba for a while and I found it is such a great project. Recently, I am working in some queries focus on DISTANCE JOINS. However I have found some issues.

For example, I have this code (DistanceJoin.tgz and P10K.csv):

```python
from pyspark import SparkConf, SparkContext
from pyspark.sql import SQLContext
from pyspark.sql import Row

conf = (SparkConf()
    .setMaster("local")
    .setAppName("My app")
    .set("spark.executor.memory", "1g")
)
sc = SparkContext(conf=conf)
sqlContext = SQLContext(sc)

epsilon = 100
points = sc.textFile("P10K.csv")
    .map(lambda line: line.split(","))
    .map(lambda p: Row(10p[0], lat=1.0f(float(p[1]), lng=1.0f(float(p[2]))))
    .toDF()

npoints = points.count()
npoints

points.registerTempTable("pt")
points.registerTempTable("pt2")
pt = sqlContext.sql(""
    CREATE INDEX
    "

https://github.com/InitialDLab/Simba/issues/71
Dataset

- **Beijing** from Geolife project\(^1\).
  - 182 users in a period of over three years (from April 2007 to August 2012).
  - 17,621 trajectories.
  - \(\approx\)18 million points (no duplicates).

\(^1\) [http://tinyurl.com/j7t2cao](http://tinyurl.com/j7t2cao)
Setup

- Single node.
- Processor: 4-core Intel(R) Core(TM) i5-2400S CPU @ 2.50GHz
- RAM: 8 GB.
- Ubuntu 16.04 LTS, Simba/Spark 1.6.0.
Beijing \([N = 10K - 100K; \varepsilon = 10 - 200 \text{ (mts)}]\)

Execution time by \(\varepsilon\) (radius of disk in mts) in Beijing dataset.

http://tinyurl.com/js6us8g
Dataset

- **Porto** from ECML/PKDD 15 Taxi Trajectory Prediction Challenge\(^2\).
  - A complete year (from 01/07/2013 to 30/06/2014).
  - Trajectories for all the 442 taxis running in the city of Porto, in Portugal.
  - \(\approx 17.7\) million points (no duplicates).

\(^2\)http://tinyurl.com/zzbt1t9
4-node cluster at DBLab.
Processors: 8-core Intel(R) Xeon(R) CPU E3-1230 V2 @ 3.30GHz
RAM: 15.5 GB.
Centos 6.8, Simba/Spark 1.6.0.
Experiments

Porto \([N = 1M - 16M; \varepsilon = 2 - 10 \text{ (mts)}]\)

Execution time by \(\varepsilon\) (radius of disk in mts) in Porto dataset.

![Graphs showing execution time by \(\varepsilon\) for different numbers of points and algorithms BFE and PBFE.](http://tinyurl.com/j9u9c7h)
Outline

1. Moving Flock Patterns
2. Implementation
3. Experiments
4. Conclusions
Cooming soon...
Thank you!!!

Do you have any question?