Towards Parallel Detection of Moving Flock Patterns in Large Spatiotemporal Datasets

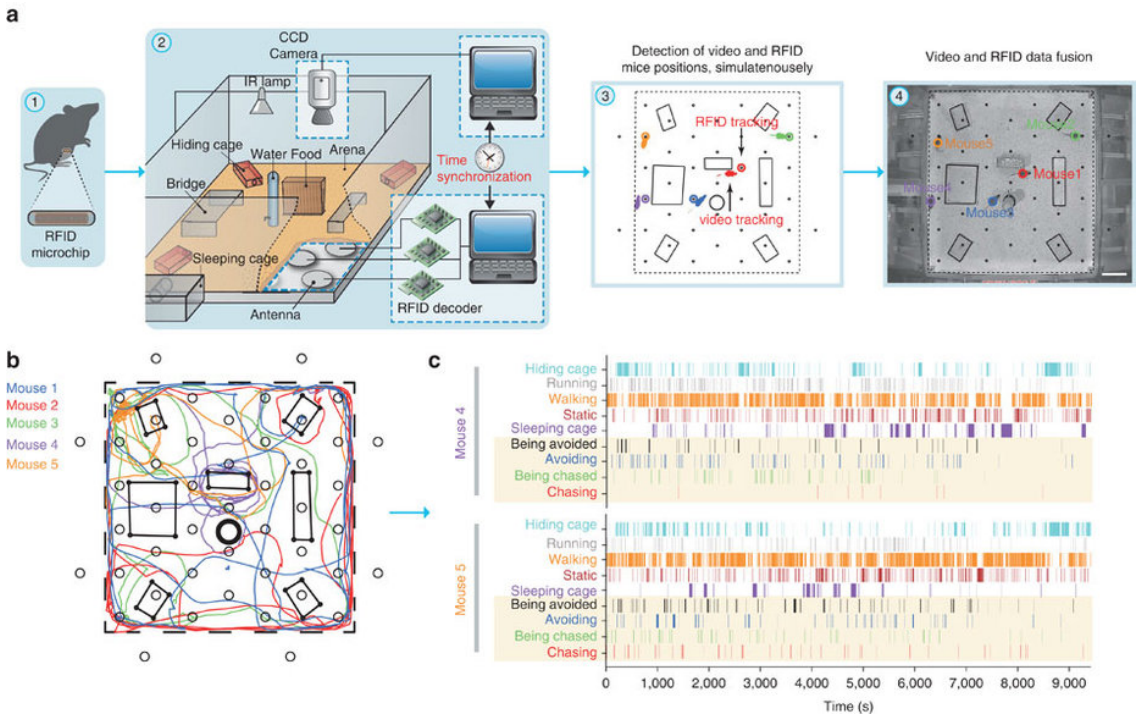
Final Report

Andres Calderon

December 4, 2016

Trajectory Datasets

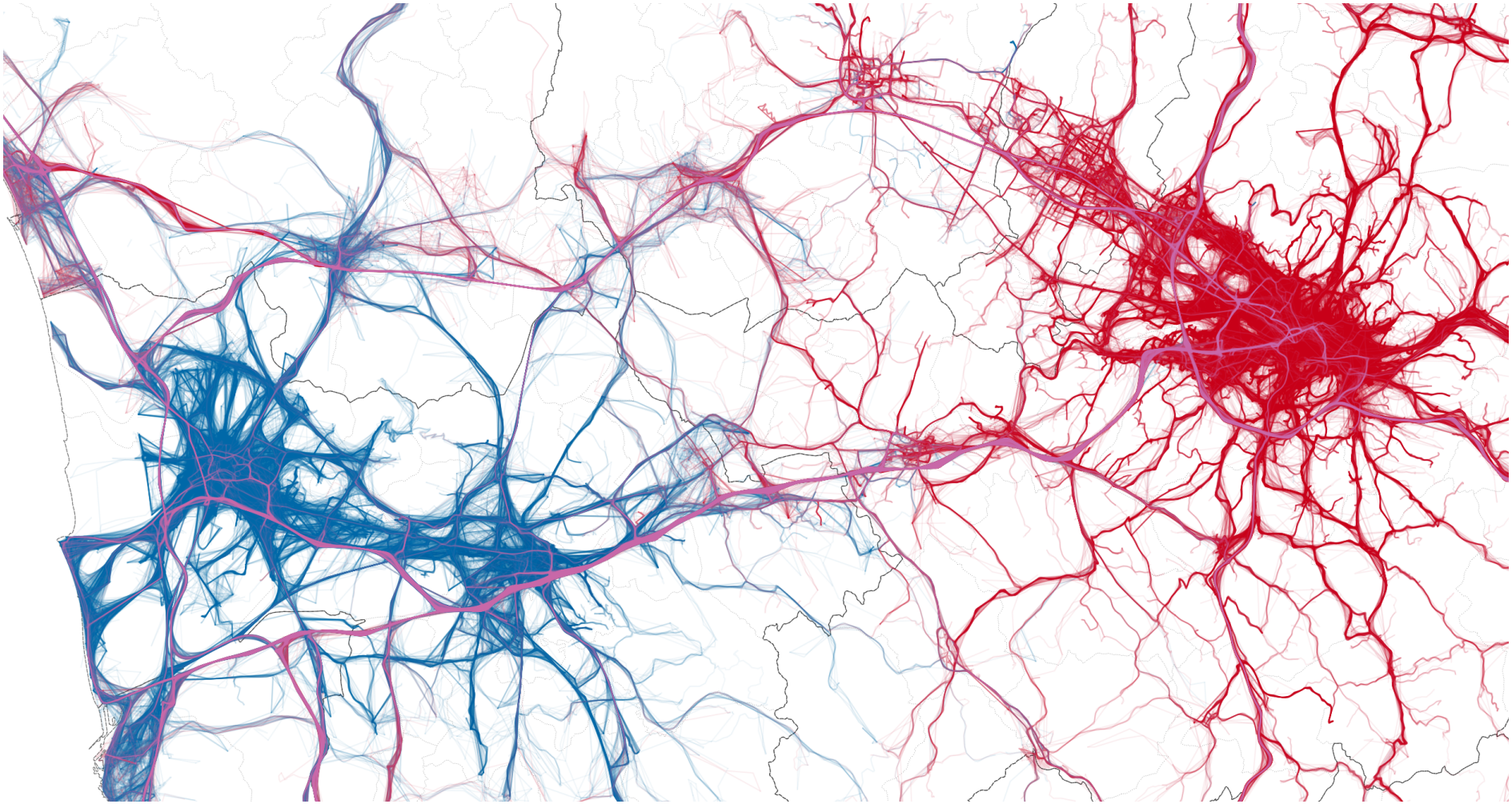
- Anything that could move (or could be moved), will be tracked...
  - Smart phones, GPS, RFID, WiFi, Bluetooth, IoT, Remote sensing...



http://tinyurl.com/jjfqmhh

Applications

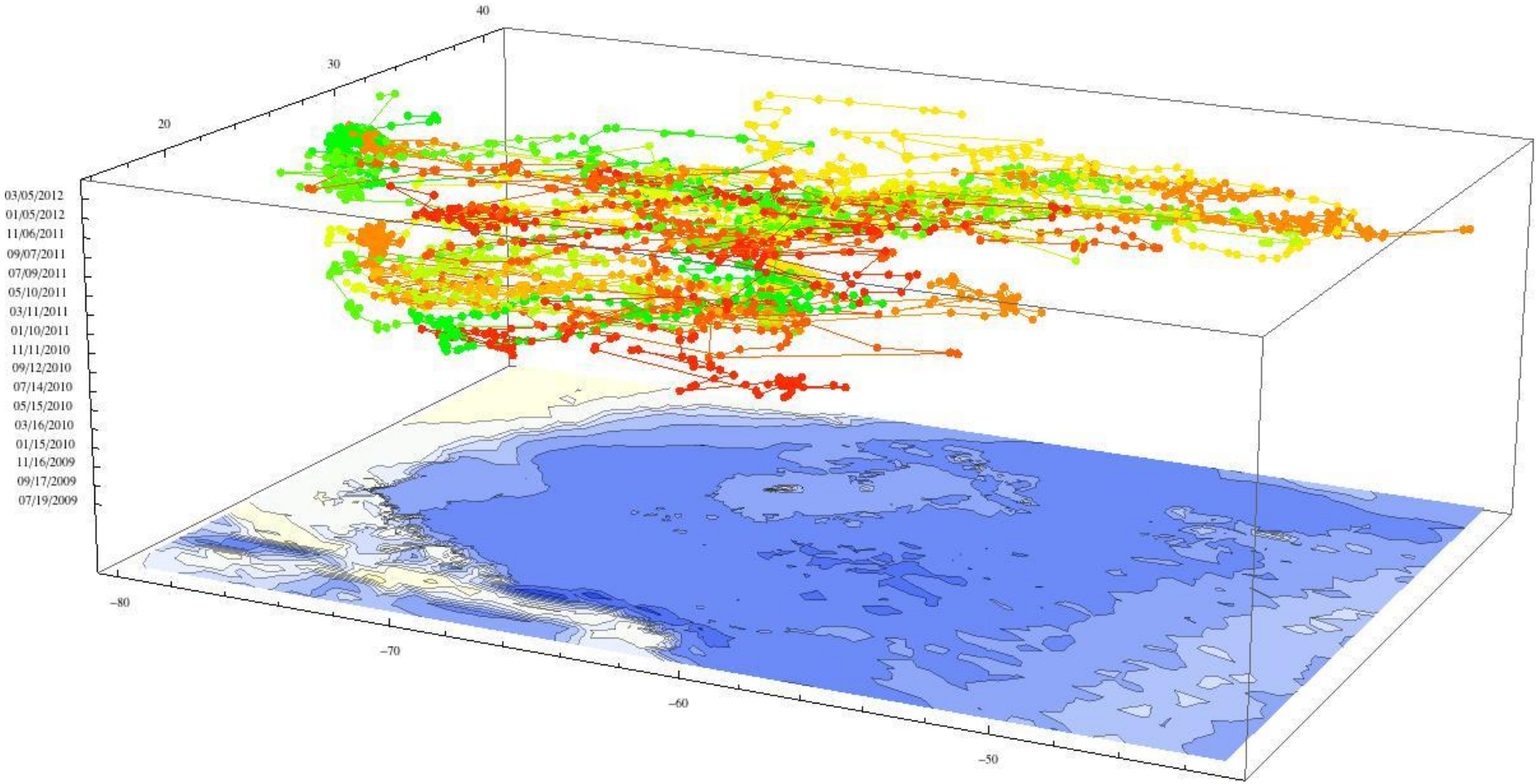
Are you a Returner or an Explorer? Ask Big Data.



http://tinyurl.com/hl4e9u8

Applications

Tiger Shark Data Analysis.



http://tinyurl.com/gtqjh4o

Outline

What is a flock???

Definition $((\mu, \epsilon, \delta) - \textit{flock})$

Sets of at least μ objects moving close enough (ε) for at least δ time intervals (Benkert et al, 2008).

Motivation

Why is important of focus on flocks and finding disks???

- 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.
  - It has a high complexity ($\mathcal{O}(2n^2)$).
  - It is no trivial, disks can be at any location.

Outline

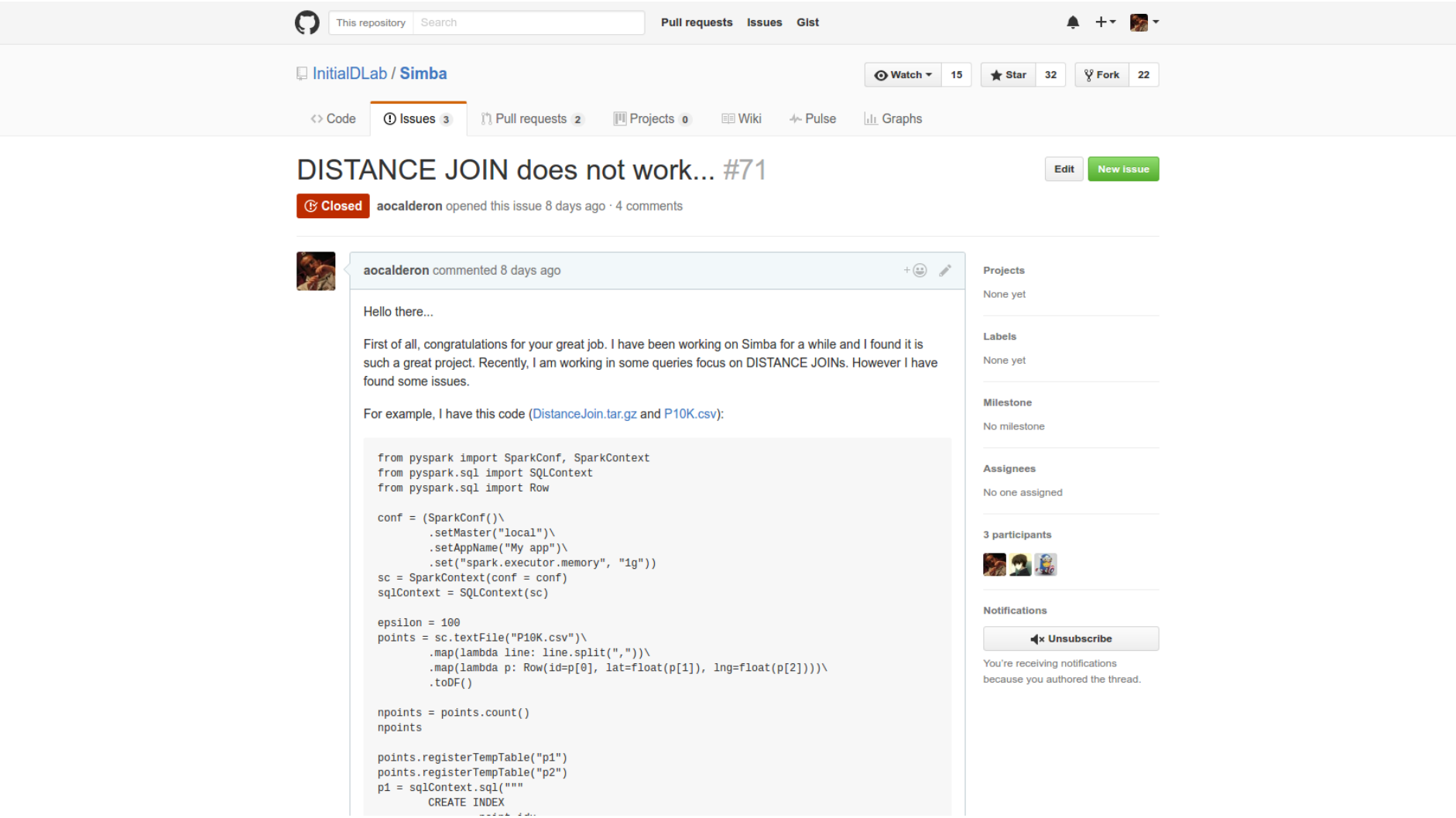
BFE Implementation

- On-Line Discovery of Flock Patterns in Spatio-Temporal Data (Vieira et al, SIGSPATIAL'09).
- Implementation available at `https://github.com/poldrosky/FPFlock`.

PBFE Implementation

- Done using Simba 1.6.0.
  - `DISTANCE JOIN` and `CIRCLERANGE` operators were the key!!!
- Demo:
  - `http://tinyurl.com/jl55849`.
  - Password is 'nancy' (behave yourself).
- Code available at `https://github.com/aocalderon/PhD/tree/master/Y2Q1/SDB/Project/Code/Scripts/pbfe2`.

Bug report



This repository Search Pull requests Issues Gist

InitialDLab / Simba Watch 15 Star 32 Fork 22

Code Issues 3 Pull requests 2 Projects 0 Wiki Pulse Graphs

DISTANCE JOIN does not work... #71

Edit New issue

Closed aocalderon opened this issue 8 days ago · 4 comments

aocalderon commented 8 days ago

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.tar.gz and P10K.csv):

```
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(id=p[0], lat=float(p[1]), lng=float(p[2])))\
        .toDF()

npoints = points.count()
npoints

points.registerTempTable("p1")
points.registerTempTable("p2")
p1 = sqlContext.sql("""
        CREATE INDEX
```

Projects: None yet

Labels: None yet

Milestone: No milestone

Assignees: No one assigned

3 participants

Notifications: Unsubscribe

You're receiving notifications because you authored the thread.

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

Bug report

- A `SQLParser` error in the DISTANCE JOIN examples.
- A requirement failure when JOIN's were combined with filters and inequations.
- Caused by the Simba Optimizer:
  - `DistanceJoin` node does not support inequations inside (another Filter node is needed).
- Quickly fixed at RP #72.

Bug report

- Both were been throwing a 'requirement failure' error:

```python
# Code for Spark-SQL (Python)
sql = """
    SELECT *
    FROM p1
    DISTANCE JOIN p2
    ON POINT(p2.lng, p2.lat) IN CIRCLERANGE(POINT(p1.lng, p1.lat), {0})
    WHERE p1.id < p2.id""".format(epsilon)
pairs = sqlContext.sql(sql)
```

```scala
// Code for Spark DataFrame API (Scala)
val pairs = p1.distanceJoin(p2, Point(p1("x"), p1("y")), Point(p2("x2"), p2("y2")), epsilon)
 ↪    .filter("id < id2")
```

Bug report

- Found a workaround using the DataFrame API:

```
// Code for Spark DataFrame API (Scala)
val pairs = p1.distanceJoin(p2, Point(p1("x"), p1("y")), Point(p2("x2"), p2("y2")), epsilon)
val disks = pairs.rdd.filter( (x:Row) => x.getInt(0) > x.getInt(3) )
```

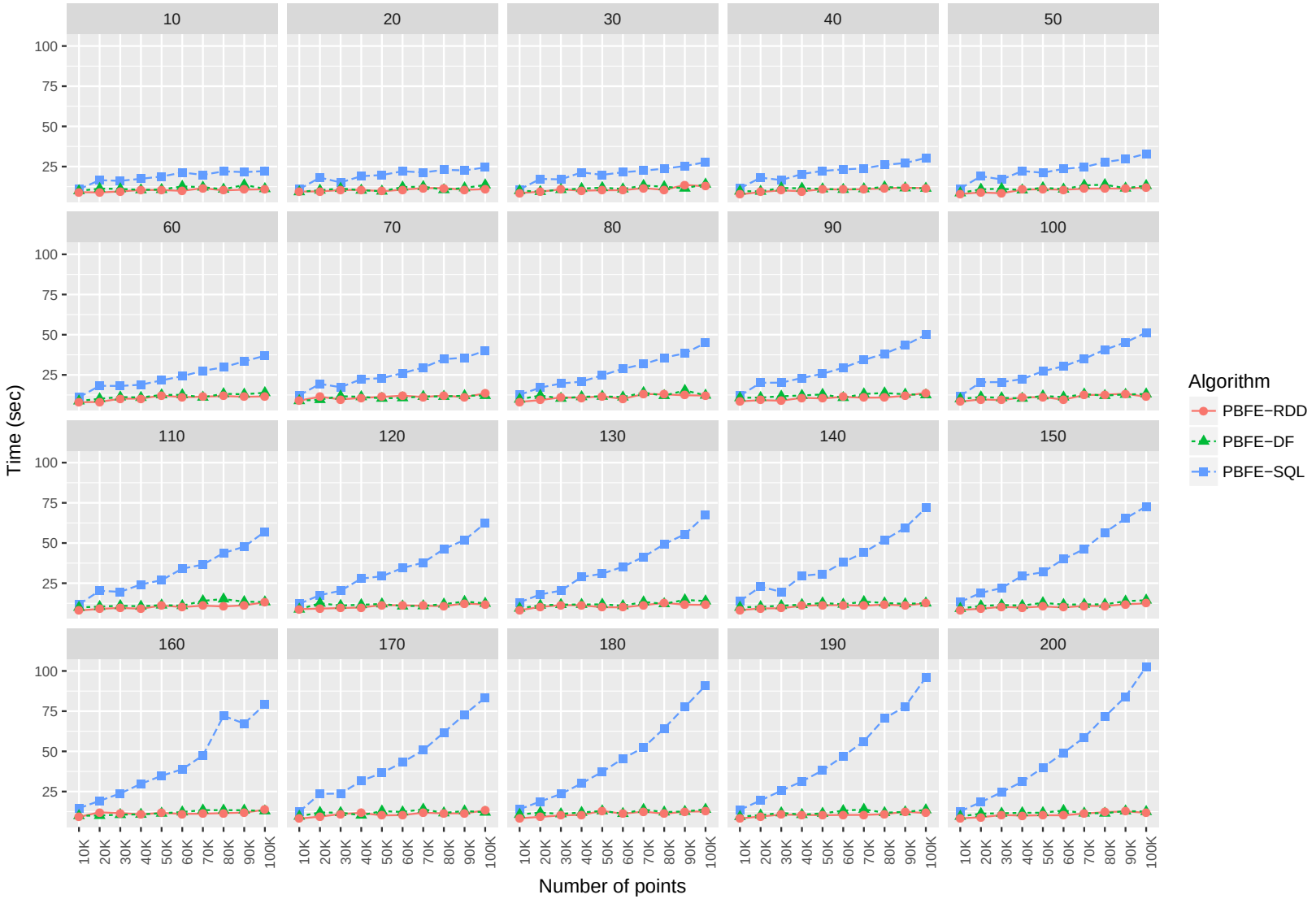
Outline

RDD vs DF vs SQL Filters

- Which have the best performance???
  - Spark SQL `WHERE` clause.
  - Spark DataFrame `filter` function.
  - Spark RDD `filter` function.

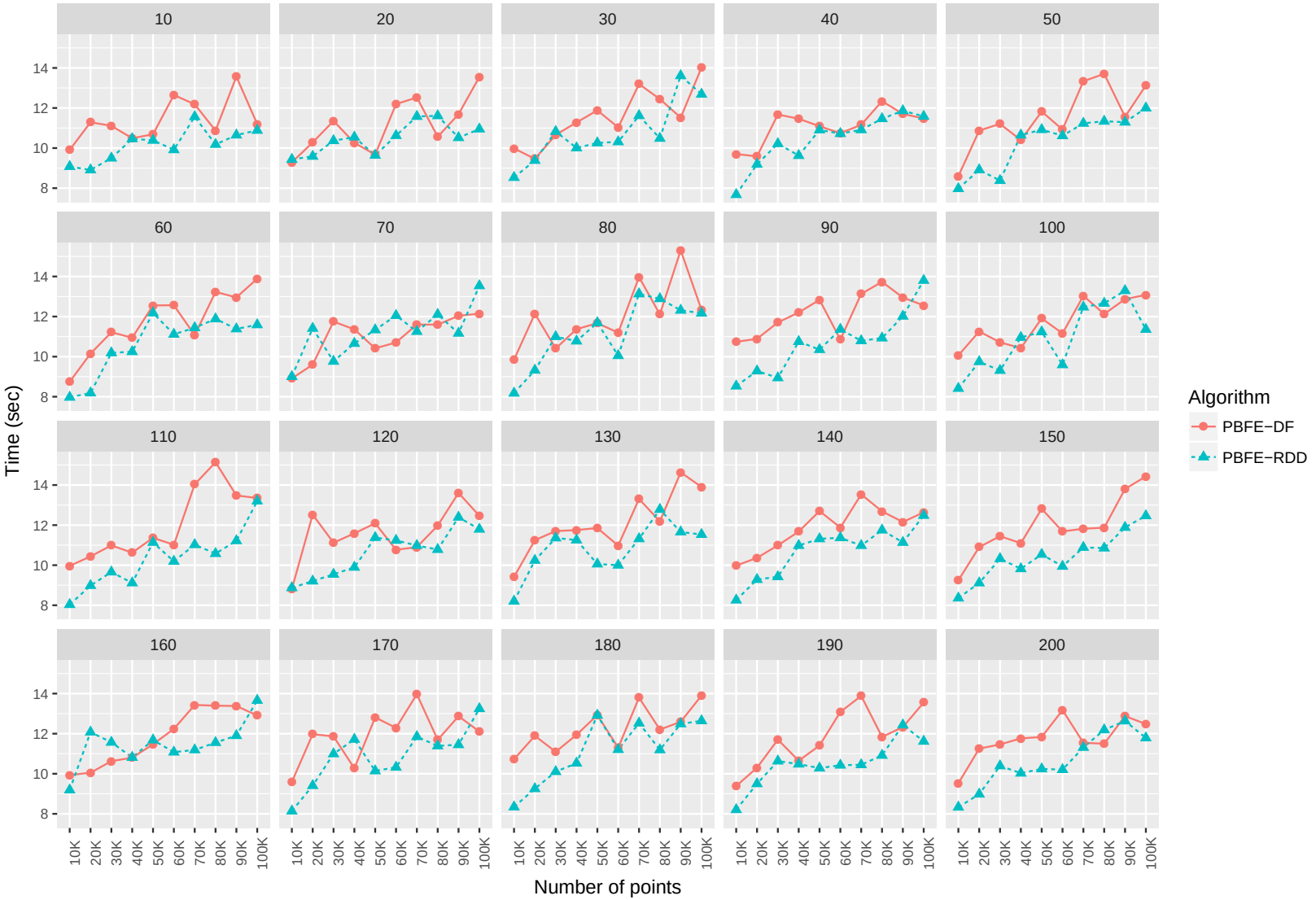
Benchmarking on Beijing Dataset

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



Benchmarking on Beijing Dataset

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

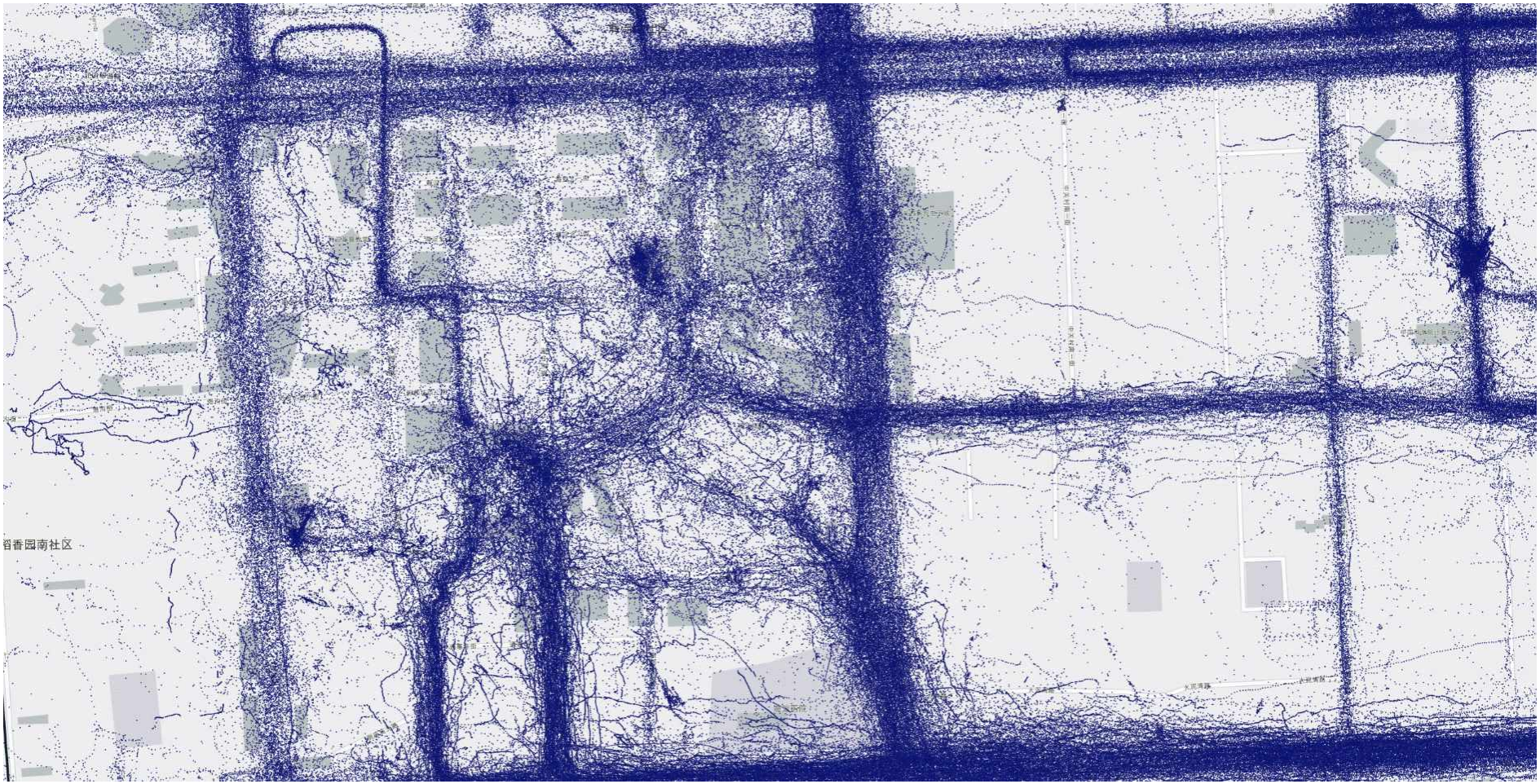


Dataset

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

[1] http://tinyurl.com/j7t2cao

Beijing Dataset

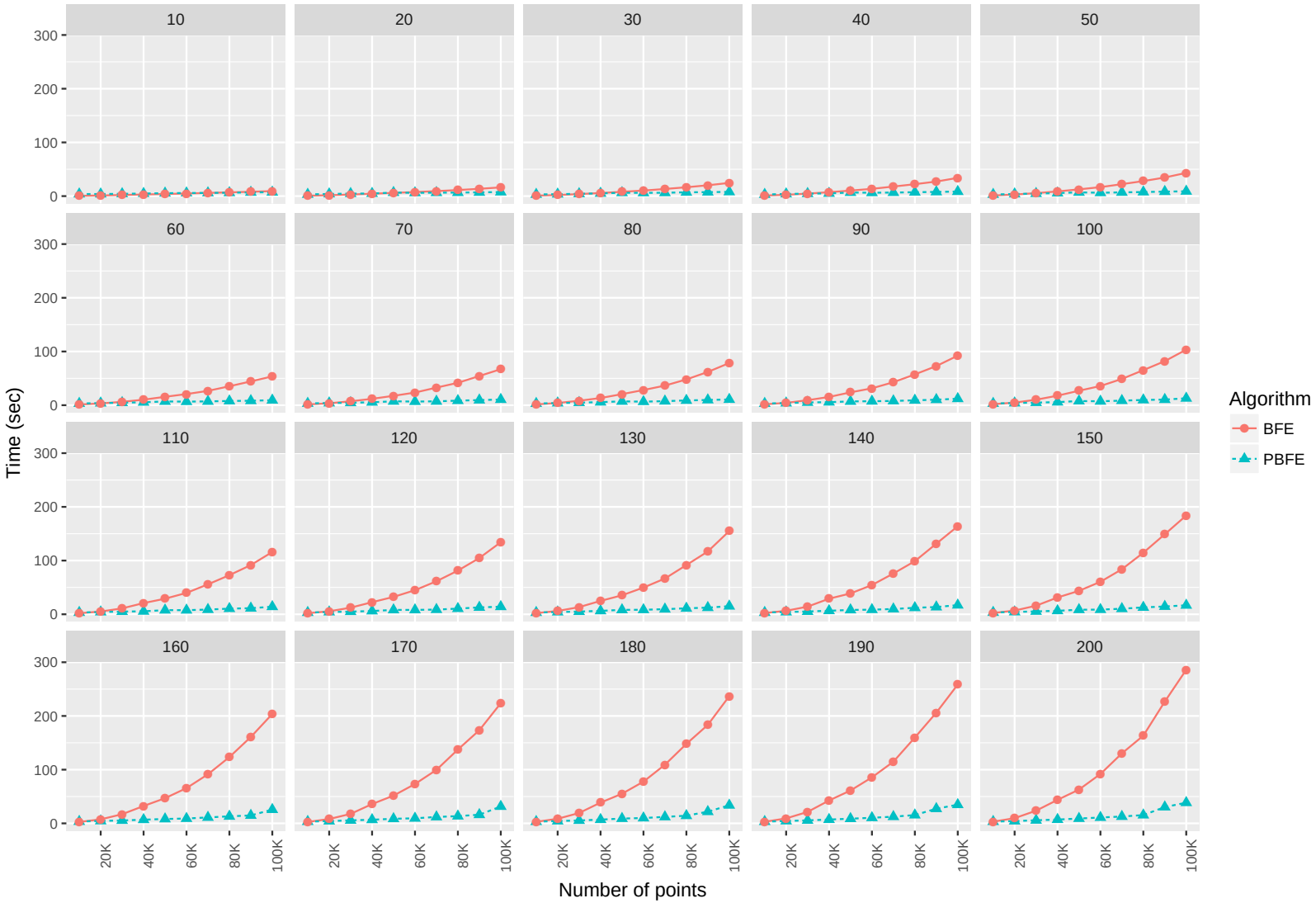


Setup

- 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.

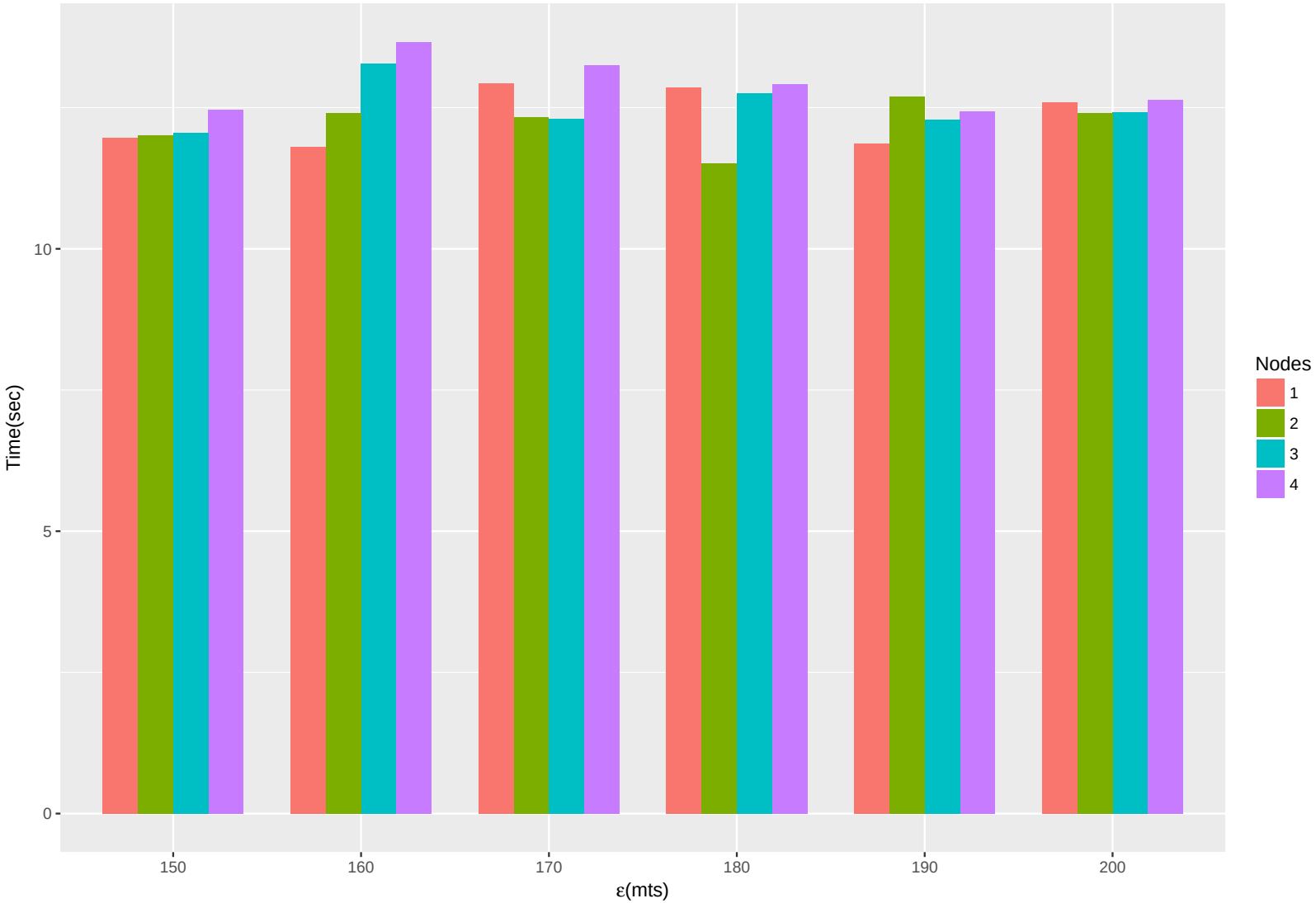
Execution time:

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

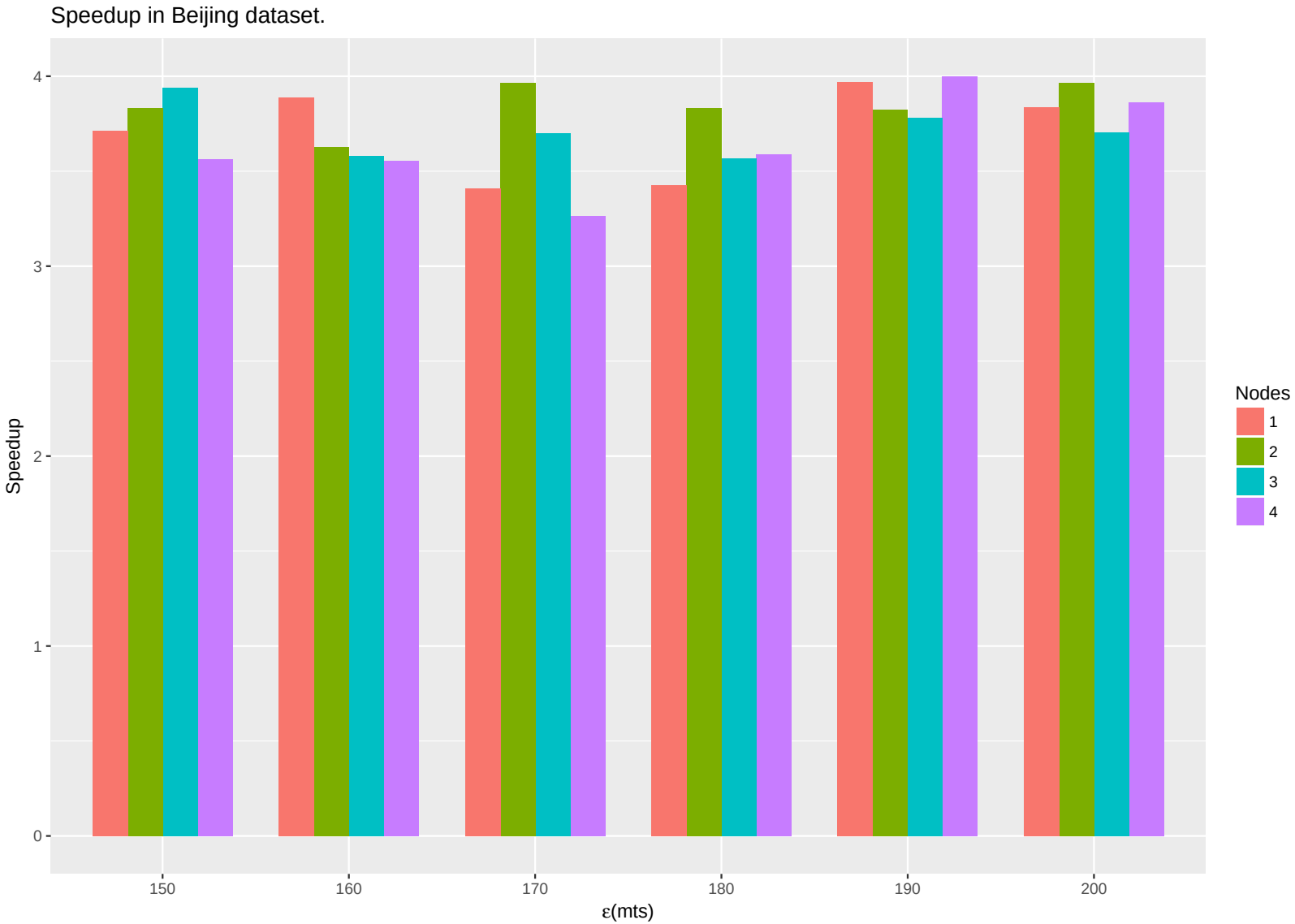


Scaleup

Scaleup in Beijing dataset.



Speedup

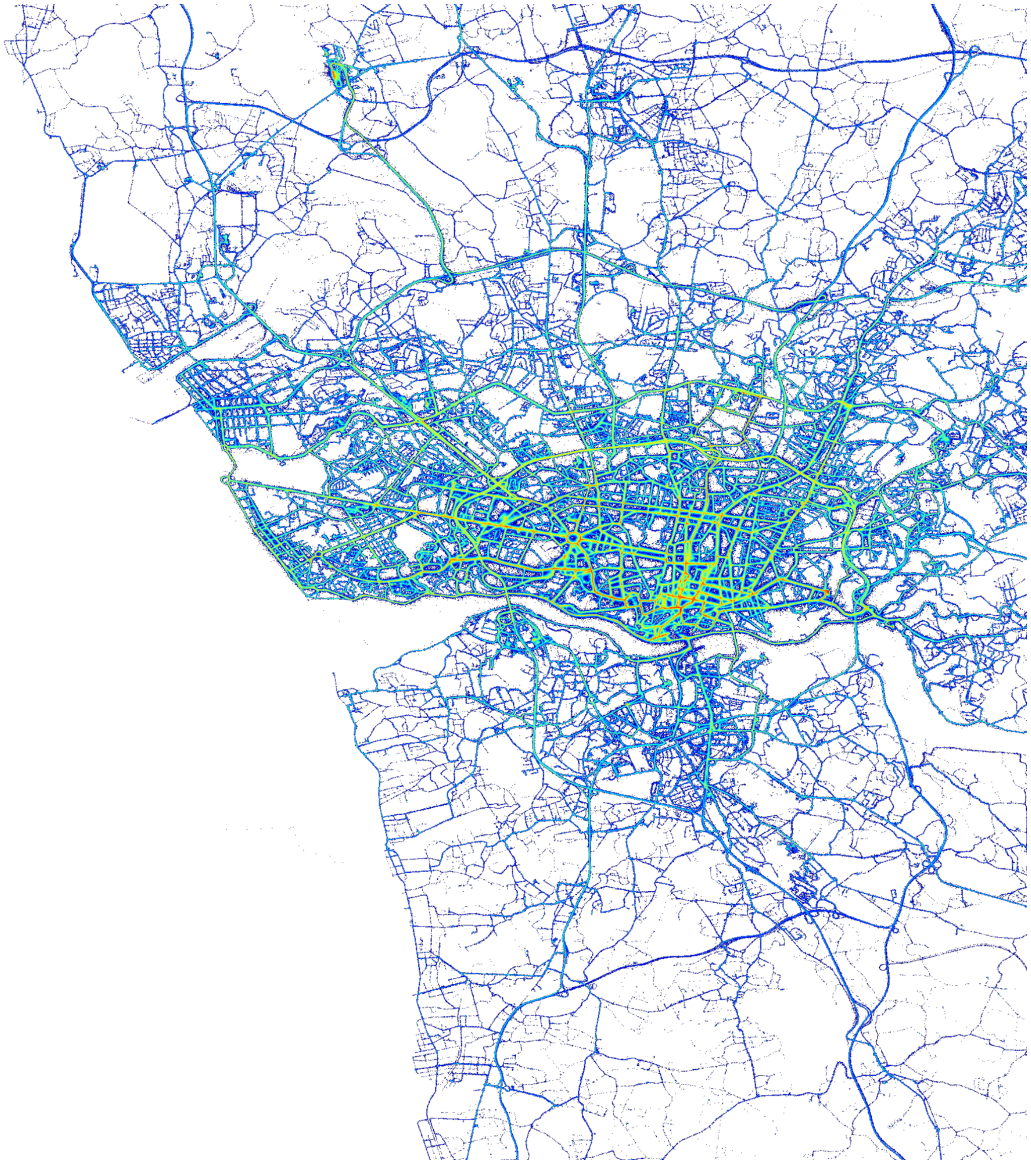

Speedup in Beijing dataset.
Speedup
0
1
2
3
4
150
160
170
180
190
200
ε(mts)
Nodes
1
2
3
4

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.
  - ≈17.7 million points (no duplicates).

[2] http://tinyurl.com/zzbtlt9

Porto Dataset

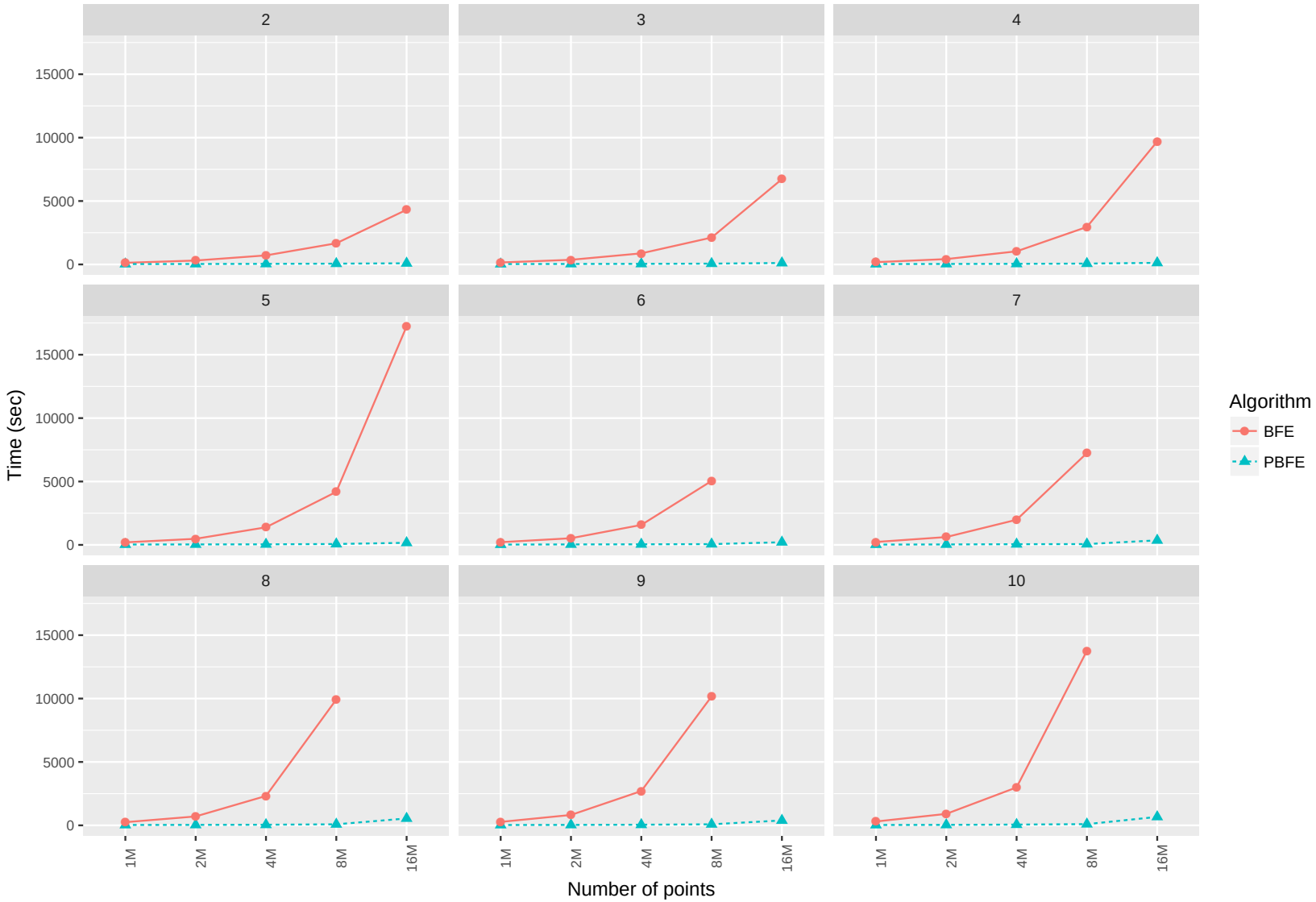


Setup

- 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.

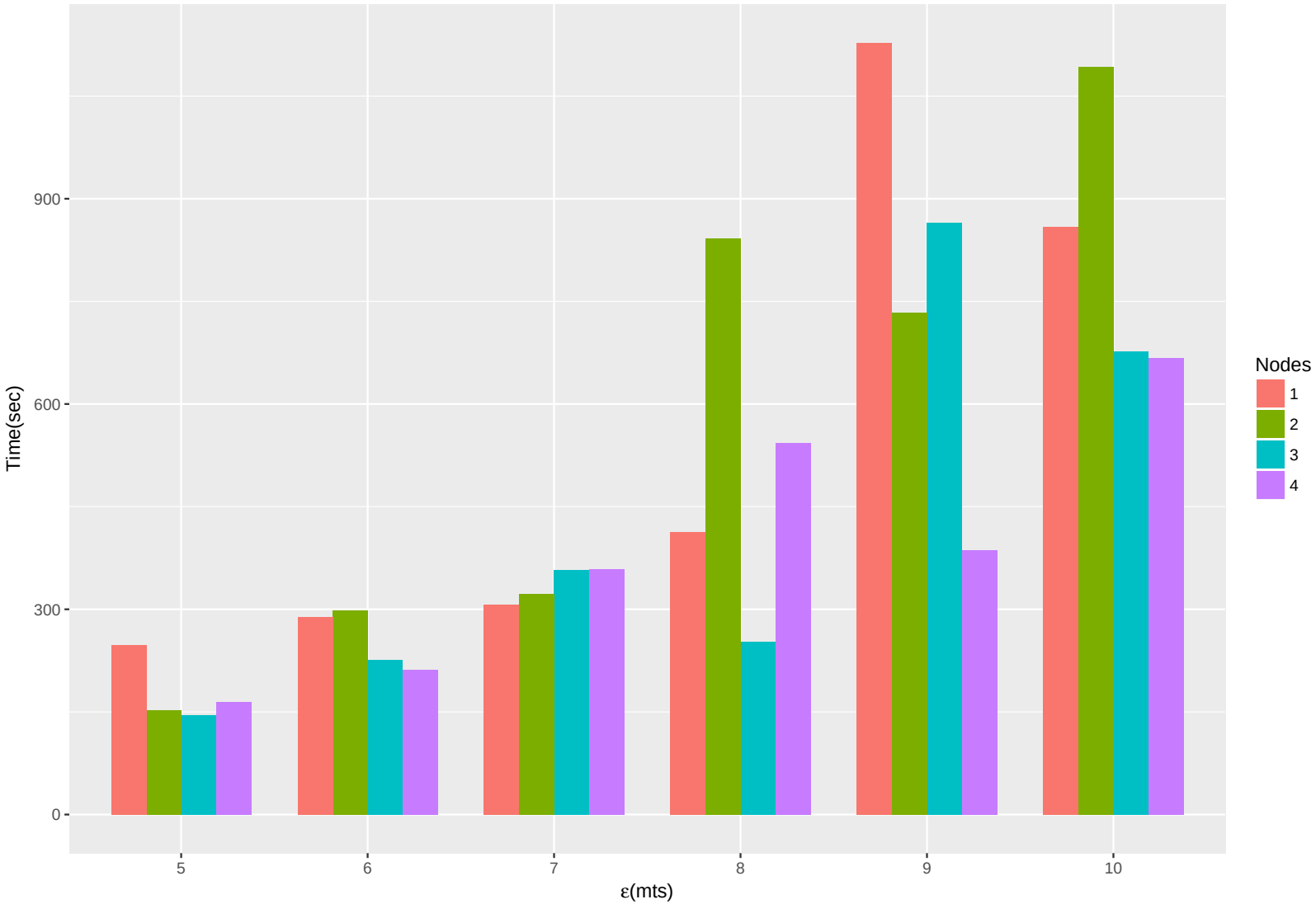
Execution time

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

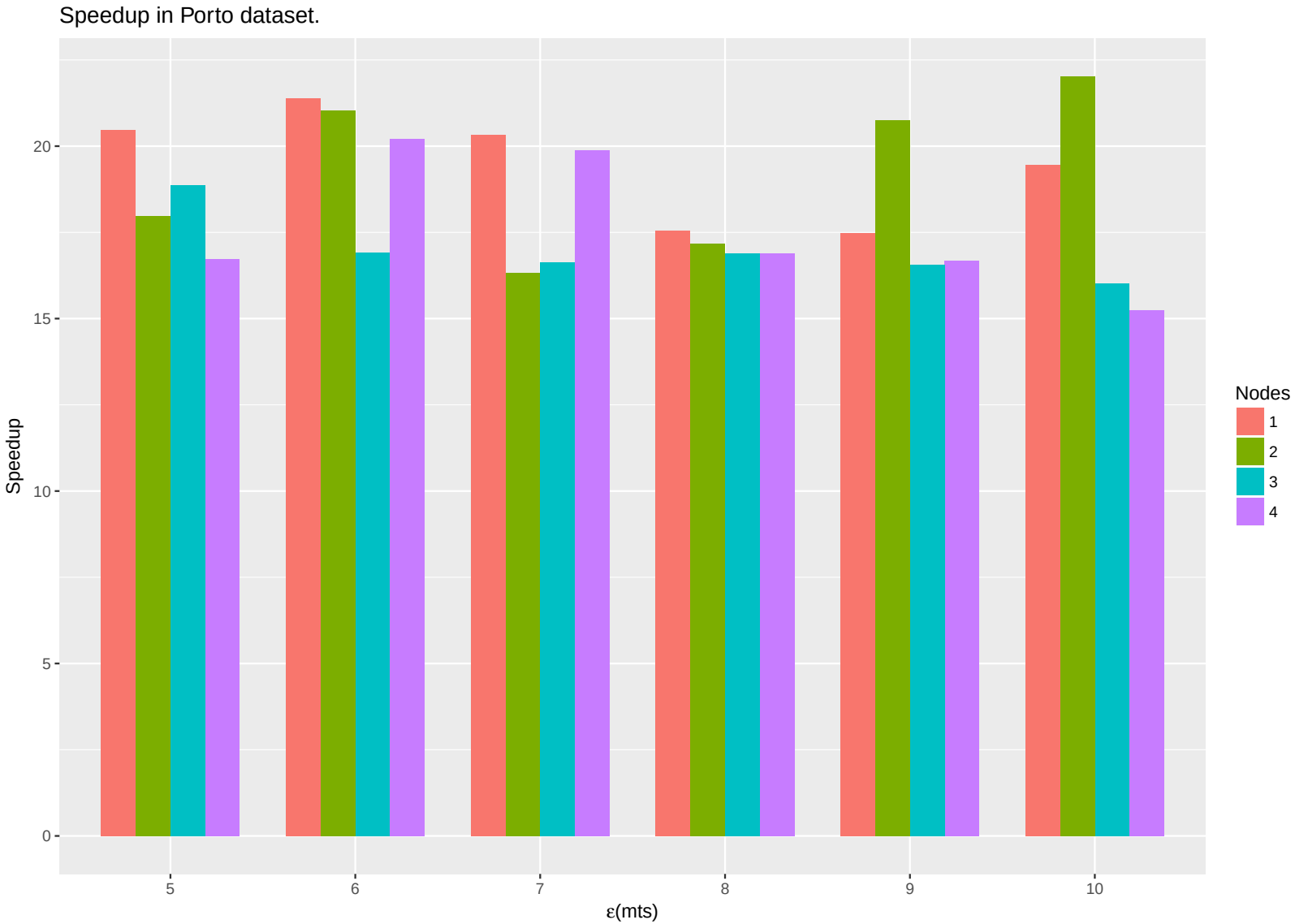


Scaleup

Scaleup in Porto dataset.



Speedup


Speedup in Porto dataset.
Speedup
0
5
10
15
20
5
6
7
8
9
10
ε(mts)
Nodes
1
2
3
4

Outline

Conclusions

Coming soon...

Thank you!!!

Do you have any question?