Towards Parallel Detection of Movement Patterns in Large Spatio-temporal Datasets

Andres Oswaldo Calderon Romero

April 23, 2019

1 Introduction

Nowadays, spatio-temporal data is ubiquitous. Thanks to new technologies and the proliferation of location devices (such as Internet of Things, Remote Sensing, Smart phones, GPS, RFID, etc.), the collection of huge amount of spatio-temporal data is now possible. With the appearance of this datasets also appears the need of new techniques which allow the analysis and detection of useful patterns in large spatio-temporal databases.

Applications for this kind of information are diverse and interesting, in particular if they come in the way of trajectory datasets [Jeung et al., 2011; Huang and Yuan, 2015]. Case of studies range from transportation system management [Di Lorenzo et al., 2016; Johansson and Terelius, 2015] to Ecology [Johnston et al., 2015; La Sorte et al., 2016]. For instance, Turdukulov et al. [2014] explore the finding of complex motion patterns to discover similarities between tropical cyclone paths. Similarly, Amor et al. [2016] use eye trajectories to understand which strategies people use during a visual search. Also, Holland et al. [1999] track the behavior of tiger sharks in the coasts of Hawaii in order to understand their migration patters.

Recently, there has been an increasing interest in exploiting more complex movement patterns in spatio-temporal datasets. Traditional range and nearest neighbor queries do not capture the collective behavior of moving objects. Moving cluster [Kalnis et al., 2005], convoys [Jeung et al., 2008] and flock patterns [Benkert et al., 2008; Gudmundsson and van Kreveld, 2006] are new movement patterns which unveil how entities move together during a minimum time interval.

In particular, a moving flock pattern show how objects move close enough during a given period of time. A better understanding on how entities move in space is of special interest in areas such as sports [Iwase and Saito, 2002], surveillance and security [Makris and Ellis, 2002; Piciarelli et al., 2005], urban development [Huang et al., 2016; Long and Thill, 2015] and socio-economic geography [Frank et al., 2000].

Despite the fact that much more data become available, state-of-the-art techniques to mine complex movement patterns still depict low scalability and poor performance in big spatial data. The present work aims to find an initial solution to implement a parallel method to discover moving flock patterns in large spatio-temporal datasets. It is thought that new trends in distributed in-memory framework for spatial operations could help to speed up the detection of this kind of patterns.

The following section will state the related work in the area. Section 3 will explain the details of the implementation of the proposed solution while section 3.3 will present their experimental results. Finally, section 5 will discuss some conclusions and future work.

2 Related work

Recently increase use of location-aware devices (such as GPS, Smart phones and RFID tags) has allowed the collection of a vast amount of data with a spatial and temporal component linked to them. Different studies have focused in analyzing and mining this kind of collections [Leung, 2010; Miller and Han, 2001]. In this area, trajectory datasets have emerged as an interesting field where diverse kind of patterns can be identified [Zheng and Zhou, 2011; Vieira and Tsotras, 2013]. For instance, authors have proposed techniques to discover motion spatial patterns such as moving clusters [Kalnis et al., 2005], convoys [Jeung et al., 2008] and flocks [Benkert et al., 2008; Gudmundsson and van Kreveld, 2006]. In particular, Vieira et al. [2009] proposed BFE (Basic Flock Evaluation), a novel algorithm to find moving flock patterns in polynomial time over large spatio-temporal datasets.

A flock pattern is defined as a group of entities which move together for a defined lapse of time [Benkert et al., 2008] (figure 1). Applications to this kind of patterns are rich and diverse. For example, [Calderon, 2011] finds moving flock patterns in iceberg trajectories to understand their movement behavior and how they related to changes in ocean's currents.

The BFE algorithm presents an initial strategy in order to detect flock patterns. In that, first it finds disks with a predefined diameter (ε) where moving entities could be close enough at a given time interval. This is a costly operation due to the large number of points and intervals to be analyzed ($\mathcal{O}(2n^2)$ per time interval). The technique uses a grid-based index and a stencil (see figure 2) to speed up the process, but the complexity is still high.

Calderon [2011] and Turdukulov et al. [2014] use a frequent pattern mining approach to improve performance during the combination of disks between time intervals. Similarly, Tanaka et al. [2016] introduce the use of plane sweeping along with binary signatures and inverted indexes to speedup the same process. However, the above-mentioned methods still keep the same strategy as BFE to find the disks at each interval.

Arimura et al. [2014] and Geng et al. [2014] use depth-first algorithms to analyze the time intervals of each trajectory to report maximal duration flocks. However, these techniques are not suitable to find patterns in an on-line fashion.

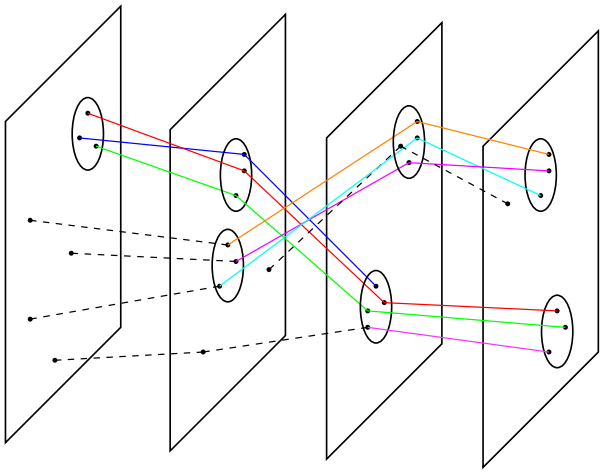

Figure 1: Moving flock pattern example.

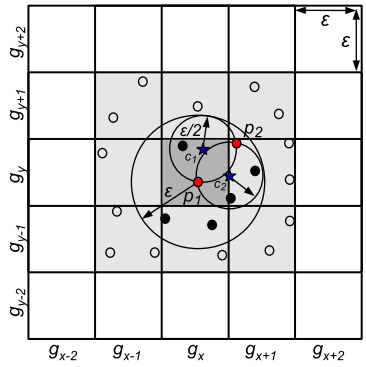

Figure 2: Grid-based index used in Vieira et al. [2009].

Given the high complexity of the task, it should not be surprising the use of parallelism to increase performance. Fort et al. [2014] use extreme and intersection sets to report maximal, longest and largest flocks on the GPU with the limitations of its memory model.

Indeed, despite the popularity of cluster computing frameworks (in particular whose supporting spatial capabilities [Eldawy, 2014; Yu et al., 2016; Hughes et al., 2015; Xie et al., 2016]) there are not significant advances in this area. At the best of our knowledge, this work is the first to explore in-memory distributed systems towards the detection of moving flock patterns.

3 Parallelizing candidate disk detection

Given that the finding of disks at each time interval is one of the most costly operations towards the detection of moving flock patterns, the main goal of this work is to implement a parallel method to detect that set of disks. In order to do that, we will use the spatial operations offered by Simba [Xie et al., 2016], a distributed in-memory spatial

```
SELECT
        *
FROM
        points p1
DISTANCE JOIN
        points p2
ON
        POINT(p2.x, p2.y) IN CIRCLERANGE(POINT(p1.x, p1.y), ε)
WHERE
        p1.id < p2.id
```

Figure 3: SQL statement on Simba to find points lying inside an ε distance.

analytic engine based on Apache Spark. This section explains the details of the algorithm implemented in Simba and how some spatial predicates introduced by it can leverage the finding of disks.

3.1 Spatial operations on Simba

Simba (Spatial In-Memory Big data Analytics) extends the Spark SQL engine to provide rich spatial operations through both SQL and the DataFrame API. Besides, it introduces two-layer spatial indexing and cost-based optimizations to support efficient spatial queries in parallel. Simba is open source and public available at the project's website[1].

In particular, `DISTANCE JOIN` and `CIRCLERANGE` operators were used in order to find groups of points lying close enough each other. The algorithm uses a user-defined distance (ε) to define the diameter of the disk. Figure 3 shows the SQL statement used to find pairs of points inside an ε distance in parallel.

3.2 Finding disks for pairs of points

Once the set of pairs of points has been found, a map function computes the center of the two possible disks per each pair according to the BFE algorithm. Vieira et al. [2009] states that: "For each such pair there are exactly two disks with radius $\frac{\varepsilon}{2}$ that have those points on their circumference". Figure 4 illustrates how to find the center of those disks.

After that, the complete set of candidate disks is collected and ready to be processed by the following phases of the BFE algorithm.

The implementation of the proposed method was written in Scala 2.10.6 and tested in Simba/Spark 1.6.0. The current code can be accessed at the authors' repository[2].

[1] `http://www.cs.utah.edu/~dongx/simba/`

[2] `https://github.com/aocalderon/PhD/tree/master/Y2Q1/SDB/Project/Code/Scripts/pbfe2`

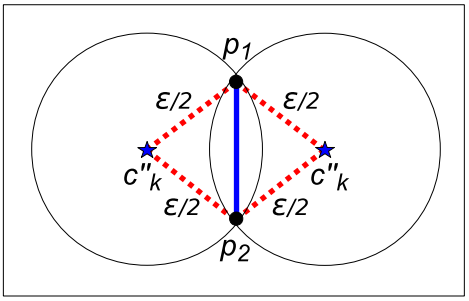

Figure 4: Disks for $\{p_1, p_2\}$, $d(p_1, p_2) \leq \varepsilon$ [Vieira et al., 2009].

3.3 Experiments

The main idea of the experiments is to assess the performance of the parallel method against a sequential version of the BFE algorithm proposed by Vieira et al. [2009]. A public available implementation written in Python can be accessed at this repository[3]. The source code was modified to stop after the finding of the disks and report the total number of computed disks. Same configuration was followed by the parallel implementation.

Next, a set of experiments evaluates the execution time of both algorithms on two real datasets. Further details of the settings and datasets are discussed below.

3.4 Beijing dataset

This dataset was extracted from the Geolife project[4] [Zheng et al., 2008, 2009, 2010]. It collects GPS trajectories of 182 users in a period of over three years (from April 2007 to August 2012) for an overall total of 17,621 trajectories. The timestamp field was ignored and duplicate locations were removed to simulate an unique and large time interval. In total, the point dataset contains ≈18 million points.

An initial set of experiments takes relatively small samples of the data and runs the algorithms under different values of ε. Experiments were deployed in a single-node machine with a 4-core Intel(R) Core(TM) i5-2400S CPU @ 2.50GHz processor, 8 GB of RAM running Ubuntu 16.04 LTS, Python 3.5 and Simba/Spark 1.6.0. Figure 5 show the results of these experiments.

3.5 Porto dataset

This dataset was extracted from the ECML/PKDD'15 Taxi Trajectory Prediction Challenge[5] [Lam et al., 2015; Moreira-Matias et al., 2013]. It collects a complete year (from 01/07/2013 to 30/06/2014) of trajectories for all the 442 taxis running in the city of

[3] `https://github.com/poldrosky/FPFlock`
[4] `https://www.microsoft.com/en-us/download/details.aspx?id=52367`
[5] `https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i/data`

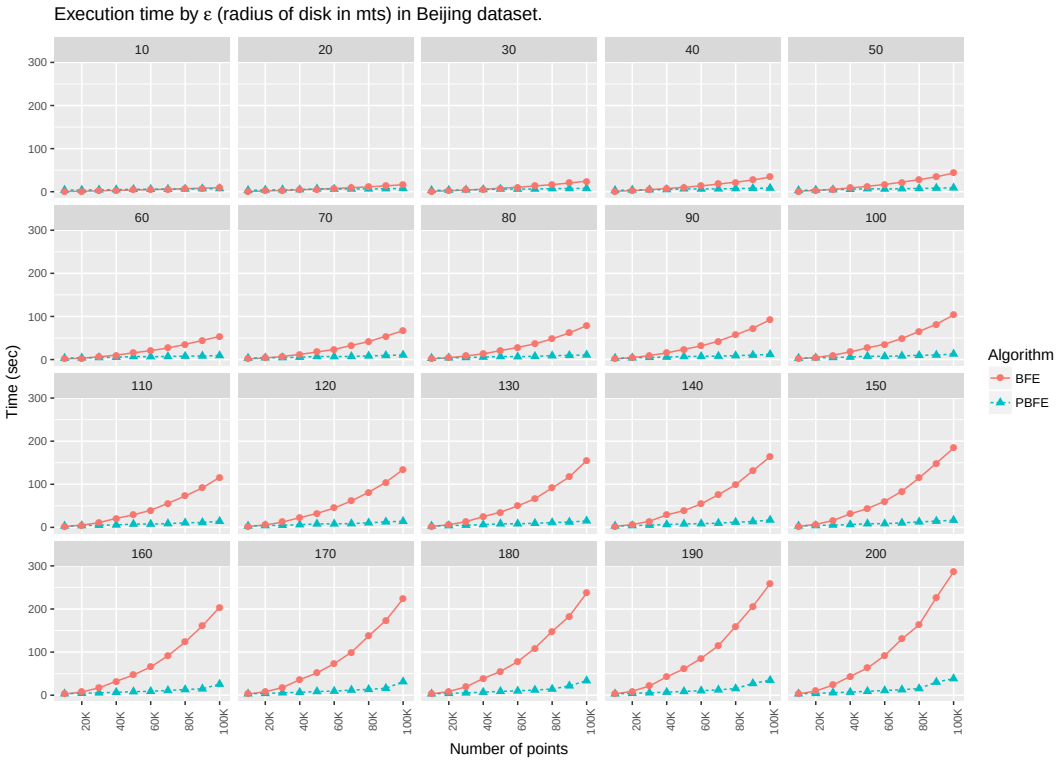

Figure 5: Execution time for Beijing dataset.

Porto, in Portugal. After pre-processing and duplicate removal the collection had ≈17.7 million points.

This set of experiments takes data samples of 1, 2, 4, 8 and 16 million of points. Similarly, it runs the algorithms under different values of ε. This time, experiments were deployed in a 4-node academic cluster with the following setup: an 8-core Intel(R) Xeon(R) CPU E3-1230 V2 @ 3.30GHz processor and 15.5 GB of RAM per node. The systems run Centos 6.8, Python 3.5 and Simba/Spark 1.6.0. Figure 6 show the results of these experiments.

3.6 Cologne dataset

The SUMO (Simulation of Urbar MObility) project [Krajzewicz et al., 2012] is an open source highly configurable road traffic simulator created by the Institute of Transportation Systems in the German Aerospace Center. The project had designed a simulation scenario describing a whole-day traffic in the city of Cologne (Germany). The data demand comes from TAPAS, a system which compute mobility wished based on information about traveling habits of Germans and the infrastructure in the area where they live[6].

The scenario was used to built a dataset collecting timestamps and location of the

[6] `http://sumo.dlr.de/wiki/Data/Scenarios/TAPASCologne`

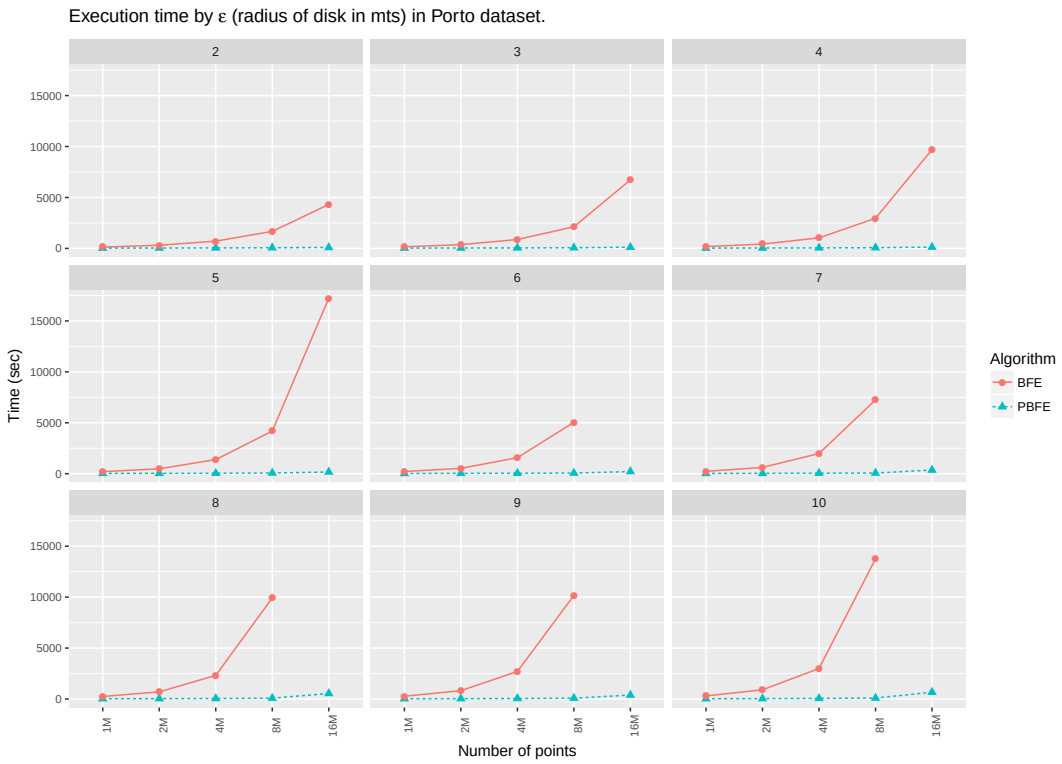

Figure 6: Execution time for Porto dataset.

involved vehicles. Data samples of 8, 10, 12, 14 and 16 million points were taken to run similar experiments as described in the previous section. Figure 7 show the results of these experiments.

3.7 Speedup and Scaleup

In order to test the scalability and parallel performance of our implementation, the speedup and scaleup metrics were applied to measure the improvement of the algorithm when more resources are added to the cluster. The following experiments show speedup and scaleup of the implementation comparing the execution using just one core and after the addition of more nodes to the cluster (each node add 8 new cores). Figures 8, 9 and 10 show the results for each of the studied datasets using different values for ε.

4 Finding maximal set of disks

Section 3 explains how to find a valid set of candidates disks but it is not a final set. Many of the disks are redundant (they are contained by others) or they do not contain the minimum number of requested objects (μ parameter). The final set of valid disks is known as the set of maximal disks. BFE faces the issue through an iterative filter over all the candidate disks discarding those whose do not fulfill the requirements by direct

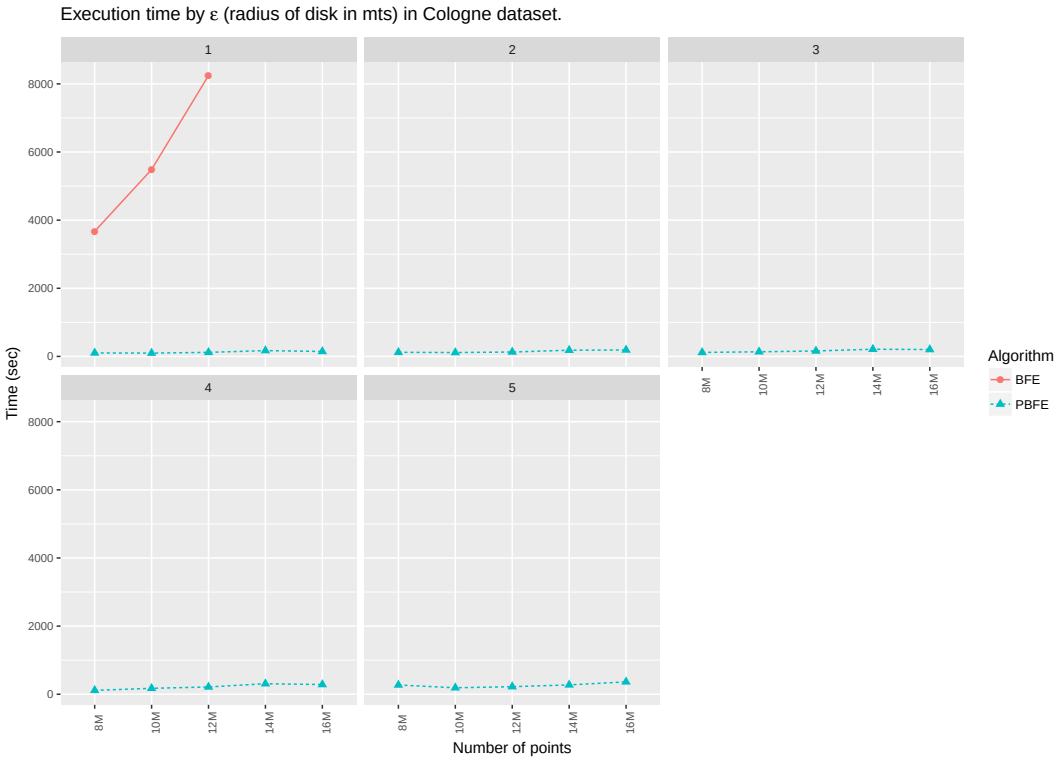

Figure 7: Execution time for Cologne dataset.

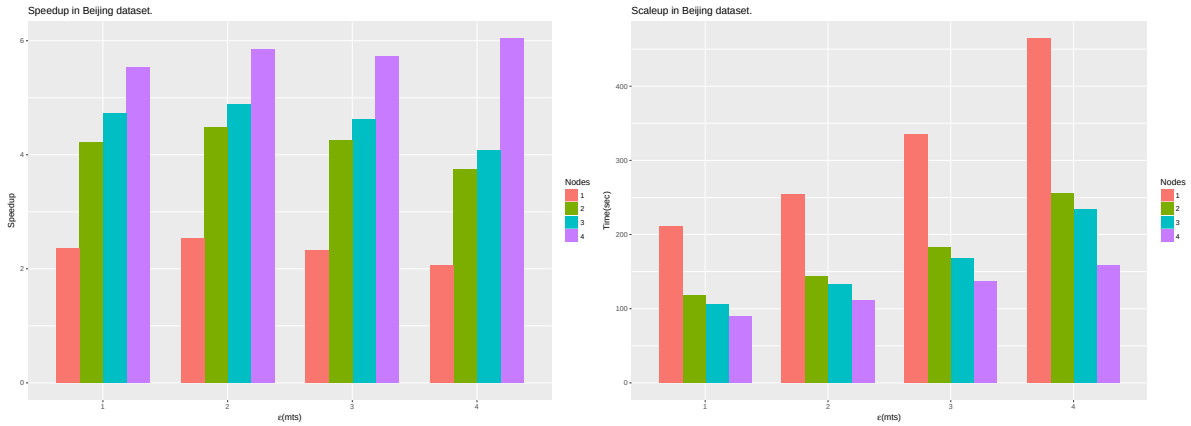

Figure 8: Speedup and Scaleup in Beijing dataset.

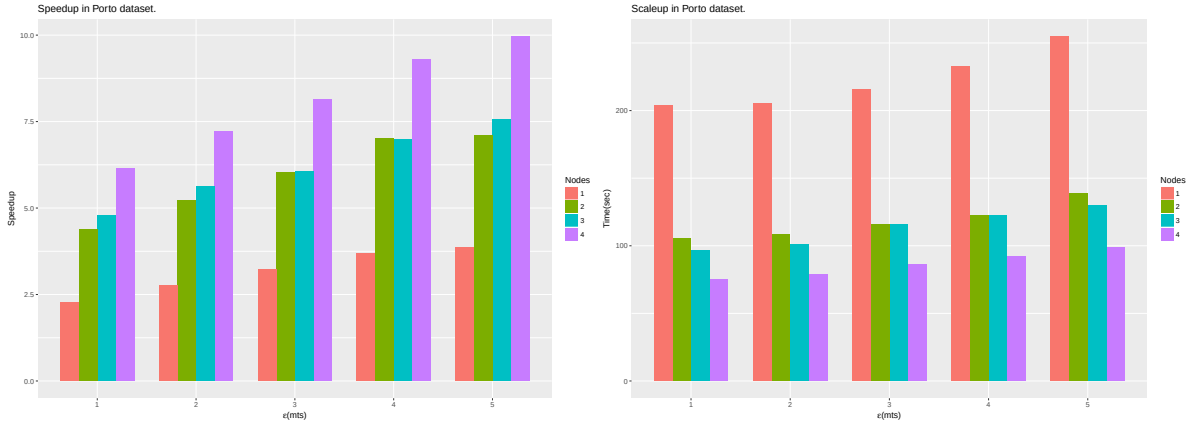

Figure 9: Speedup and Scaleup in Porto dataset.

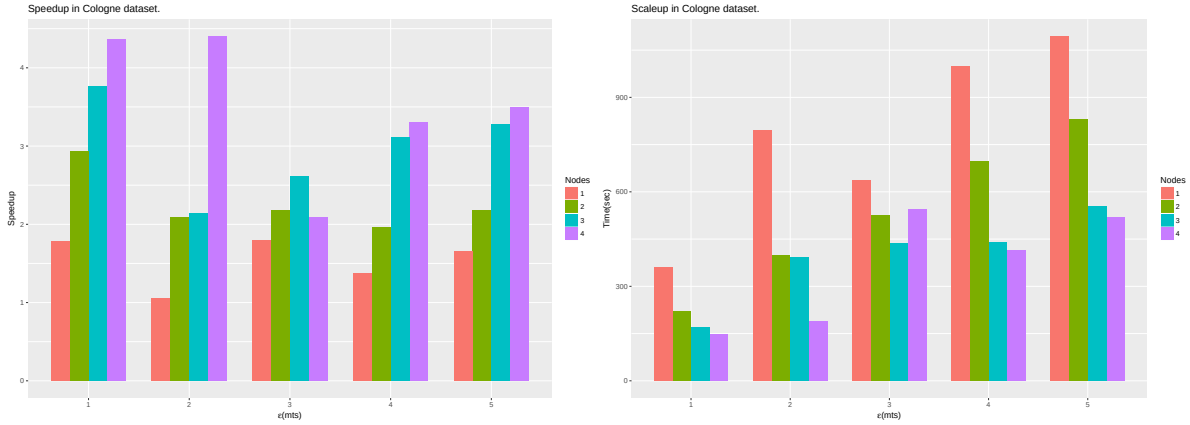

Figure 10: Speedup and Scaleup in Cologne dataset.

comparison. However, it can be a costly operation according with the number of disks.

For this stage, we propose apply a frequent pattern mining approach to filter the candidate disks. The main idea is to filter redundant disks by applying well-known frequent pattern algorithm for maximal pattern detection [Han, 2005].

Maximal patterns are valid representation of the complete set of frequent patterns for a dataset because they contains all the subsets whose overpass a minimum threshold. They are important because all the remaining frequent patterns can be computed from them. Many efforts and techniques have been developed in order to find efficient methods to find this particular kind of patterns.

Goethals and Zaki [2004] and Fournier-Viger et al. [2016] have explored and collected a significant number of implementations and details about the available algorithms in the topic. Among the available techniques, Uno et al. [2004] propose the LCM algorithm which stands out as one of the most efficient implementation. It is able to find maximal sets in linear time according with the number of patterns.

Given that the maximal set of disks are those that contain redundant disks, maximal frequent patterns algorithms can be used to detect those disks. If the objects enclosed by the disks are seen as transactions, it results simple to apply any technique to discover maximal patterns.

4.1 Experiments

This section shows a set of preliminary experiments to test the suitability of using a frequent pattern mining approach to find the set of maximal disks. The main idea is to collect the objects enclosed by the candidate disks as transactions and built a dataset which will be evaluated by a maximal frequent pattern algorithm (LCM in this case).

To parallelize the algorithm, a local-global approach was used. In this case, the transaction version of the candidate set was distributed among the available nodes and LCM was run over the local datasets. In a global step, the results were merged in a unique dataset and a final run on LCM was executed on it. The main advantage was that each local run of LCM return a much smaller dataset which can be processed in linear time.

The experiments use reduced samples of Beijing and Porto datasets as described in section 3.3. The setup is also similar to that used in the Porto and Cologne experiments. Figures 11 and 12 shows the results.

5 Conclusions and future work

An implementation of a parallel method to detect disks for the BFE algorithm has been presented. The proposed method proves to be scalable and reliable. Experiments shows that execution time improves up to 3 orders of magnitude compared to the sequential implementation of the BFE algorithm for the same step.

Parallel implementation for the remaining steps of the BFE algorithm are part of the future work. It is expected to work on a parallel strategy to join the sets of valid disks

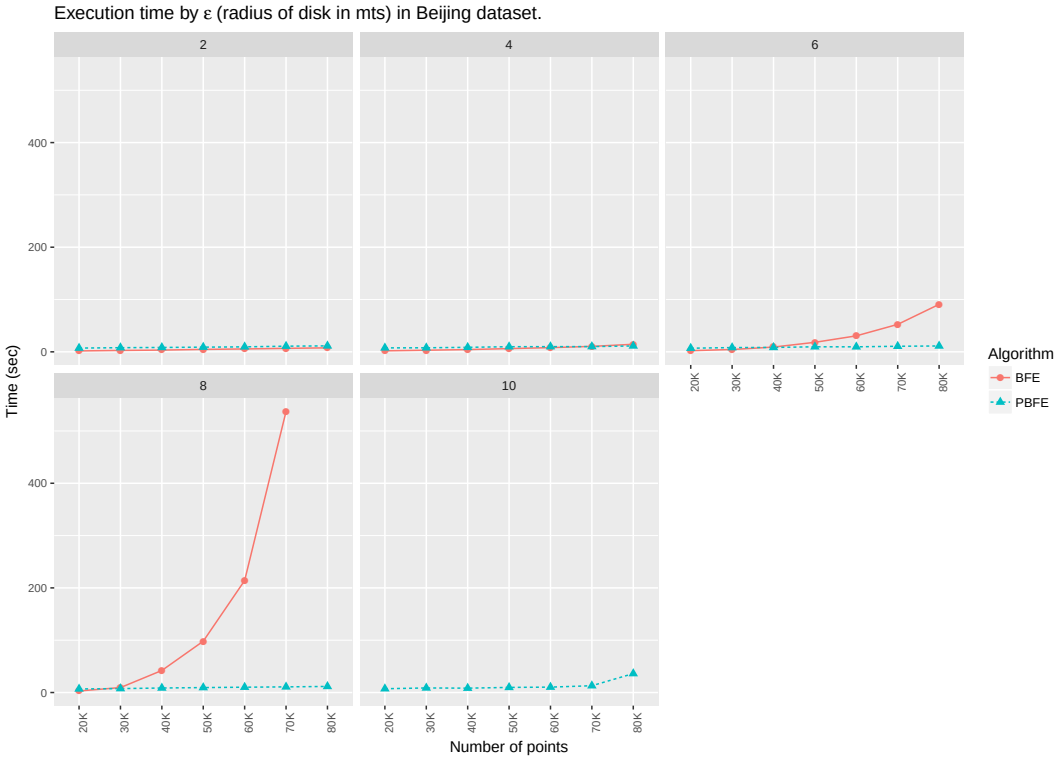

Figure 11: Execution time for Beijing dataset.

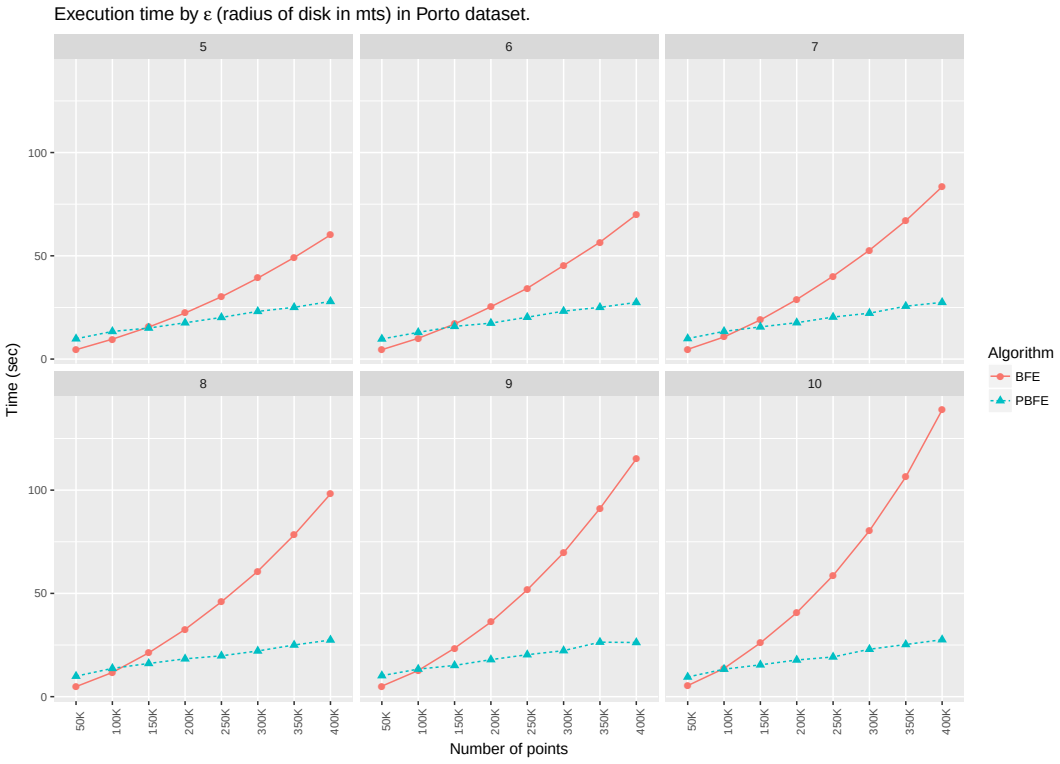

Figure 12: Execution time for Porto dataset.

between time intervals. Also, others approaches for partitioning should be explored, especially in the case of finding maximal disks. The local-global approach is just a preliminary prototype. Certainly, a proper parallel implementation of LCM algorithm should be an interesting idea. In addition, data pre-processing and result visualization are still open issues.

References

Tatiana A. Amor, Saulo D. S. Reis, Daniel Campos, Hans J. Herrmann, and José S. Andrade. Persistence in eye movement during visual search. *Scientific Reports*, 6: 20815, February 2016. ISSN 2045-2322. doi: 10.1038/srep20815. 00000.

Hiroki Arimura, Takuya Takagi, Xiaoliang Geng, and Takeaki Uno. Finding All Maximal Duration Flock Patterns in High-dimensional Trajectories. 00000, 2014.

Marc Benkert, Joachim Gudmundsson, Florian Hübner, and Thomas Wolle. Reporting flock patterns. *Computational Geometry*, 41(3):111–125, November 2008. ISSN 09257721. doi: 10.1016/j.comgeo.2007.10.003. 00000.

Andres Calderon. Mining moving flock patterns in large spatio-temporal datasets using a frequent pattern mining approach. Master's thesis, University of Twente, 2011. 00000.

G. Di Lorenzo, M. Sbodio, F. Calabrese, M. Berlingerio, F. Pinelli, and R. Nair. AllAboard: Visual Exploration of Cellphone Mobility Data to Optimise Public Transport. *IEEE Transactions on Visualization and Computer Graphics*, 22(2):1036–1050, February 2016. ISSN 1077-2626. doi: 10.1109/TVCG.2015.2440259. 00004.

Ahmed Eldawy. SpatialHadoop: Towards flexible and scalable spatial processing using mapreduce. pages 46–50. ACM Press, 2014. ISBN 978-1-4503-2924-8. doi: 10.1145/2602622.2602625. 00020.

Marta Fort, J. Antoni Sellares, and Nacho Valladares. A parallel GPU-based approach for reporting flock patterns. *International Journal of Geographical Information Science*, 28(9):1877–1903, September 2014. ISSN 1365-8816, 1362-3087. doi: 10.1080/13658816.2014.902949. 00002.

Philippe Fournier-Viger, Jerry Chun-Wei Lin, Antonio Gomariz, Ted Gueniche, Azadeh Soltani, Zhihong Deng, and Hoang Thanh Lam. *The SPMF Open-Source Data Mining Library Version 2*, pages 36–40. Springer International Publishing, Cham, 2016. ISBN 978-3-319-46131-1. doi: 10.1007/978-3-319-46131-1_8. URL `http://dx.doi.org/10.1007/978-3-319-46131-1_8`.

Andrew Frank, Jonathan Raper, and J. P. Cheylan, editors. *Life and Motion of Socio-Economic Units.* CRC Press, London ; New York, 1 edition edition, December 2000. ISBN 978-0-7484-0845-0. 00071.

Xiaoliang Geng, Takuya Takagi, Hiroki Arimura, and Takeaki Uno. Enumeration of complete set of flock patterns in trajectories. pages 53–61. ACM Press, 2014. ISBN 978-1-4503-3139-5. doi: 10.1145/2676552.2676560. 00004.

B. Goethals and M. J. Zaki. Advances in frequent itemset mining implementations: Report of fimi'03. *ACM SIGKDD Explorations*, 6(1):109–117, June 2004.

Joachim Gudmundsson and Marc van Kreveld. Computing Longest Duration Flocks in Trajectory Data. In *Proceedings of the 14th Annual ACM International Symposium on Advances in Geographic Information Systems*, GIS '06, pages 35–42, New York, NY, USA, 2006. ACM. ISBN 1-59593-529-0. doi: 10.1145/1183471.1183479. 00210.

Jiawei Han. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005. ISBN 1558609016.

K. N. Holland, B. M. Wetherbee, C. G. Lowe, and C. G. Meyer. Movements of tiger sharks (Galeocerdo cuvier) in coastal Hawaiian waters. *Marine Biology*, 134(4):665–673, 1999. 00000.

Pengtao Huang and Bo Yuan. Mining Massive-Scale Spatiotemporal Trajectories in Parallel: A Survey. In *Trends and Applications in Knowledge Discovery and Data Mining*, volume 9441 of *Lecture Notes in Computer Science*. Springer International Publishing, Cham, 2015. ISBN 978-3-319-25659-7 978-3-319-25660-3. 00001.

Xiaoke Huang, Ye Zhao, Chao Ma, Jing Yang, Xinyue Ye, and Chong Zhang. TrajGraph: A Graph-Based Visual Analytics Approach to Studying Urban Network Centralities Using Taxi Trajectory Data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):160–169, January 2016. ISSN 1077-2626. doi: 10.1109/TVCG.2015.2467771. 00000.

James N. Hughes, Andrew Annex, Christopher N. Eichelberger, Anthony Fox, Andrew Hulbert, and Michael Ronquest. GeoMesa: A distributed architecture for spatio-temporal fusion. page 94730F, May 2015. doi: 10.1117/12.2177233. 00000.

Sachiko Iwase and Hideo Saito. Tracking Soccer Player Using Multiple Views. In *MVA*, pages 102–105, 2002. 00000.

Hoyoung Jeung, Man Lung Yiu, Xiaofang Zhou, Christian S. Jensen, and Heng Tao Shen. Discovery of Convoys in Trajectory Databases. *Proc. VLDB Endow.*, 1(1): 1068–1080, August 2008. ISSN 2150-8097. doi: 10.14778/1453856.1453971. 00306.

Hoyoung Jeung, Man Lung Yiu, and Christian S. Jensen. Trajectory pattern mining. In *Computing with Spatial Trajectories*, pages 143–177. Springer, 2011. 00038.

Karl Johansson and Hakan Terelius. An efficiency measure for road transportation networks with application to two case studies. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 5149–5155. IEEE, 2015. 00000.

Alison Johnston, Daniel Fink, Mark D. Reynolds, Wesley M. Hochachka, Brian L. Sullivan, Nicholas E. Bruns, Eric Hallstein, Matt S. Merrifield, Sandi Matsumoto, and Steve Kelling. Abundance models improve spatial and temporal prioritization of conservation resources. *Ecological Applications*, 25(7):1749–1756, 2015. 00007.

Panos Kalnis, Nikos Mamoulis, and Spiridon Bakiras. On Discovering Moving Clusters in Spatio-temporal Data. In Claudia Bauzer Medeiros, Max J. Egenhofer, and Elisa Bertino, editors, *Advances in Spatial and Temporal Databases*, Lecture Notes in Computer Science, pages 364–381. Springer Berlin Heidelberg, August 2005. ISBN 978-3-540-28127-6 978-3-540-31904-7. doi: 10.1007/11535331_21. 00000.

Daniel Krajzewicz, Jakob Erdmann, Michael Behrisch, and Laura Bieker. Recent development and applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements*, 5(3&4):128–138, December 2012.

Frank A. La Sorte, Daniel Fink, Wesley M. Hochachka, and Steve Kelling. Convergence of broad-scale migration strategies in terrestrial birds. *Proceedings of the Royal Society B: Biological Sciences*, 283(1823):20152588, January 2016. ISSN 0962-8452, 1471-2954. doi: 10.1098/rspb.2015.2588. 00000.

Hoang Thanh Lam, Ernesto Diaz-Aviles, Alessandra Pascale, Yiannis Gkoufas, and Bei Chen. (Blue) Taxi Destination and Trip Time Prediction from Partial Trajectories. *ECML/PKDD Discovery Challenge 2015*, 2015. 00000.

Yee Leung. *Knowledge Discovery in Spatial Data*. Springer Science & Business Media, March 2010. ISBN 978-3-642-02664-5. 00032.

Ying Long and Jean-Claude Thill. Combining smart card data and household travel survey to analyze jobs–housing relationships in Beijing. *Computers, Environment and Urban Systems*, 53:19–35, September 2015. ISSN 0198-9715. doi: 10.1016/j.compenvurbsys.2015.02.005. 00000.

Dimitrios Makris and Tim Ellis. Path detection in video surveillance. *Image and Vision Computing*, 20(12):895–903, October 2002. ISSN 0262-8856. doi: 10.1016/S0262-8856(02)00098-7. 00165.

Harvey J. Miller and Jiawei Han. *Geographic Data Mining and Knowledge Discovery*. Taylor & Francis, Inc., Bristol, PA, USA, 2001. ISBN 0-415-23369-0. 00000.

Luis Moreira-Matias, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas. Predicting Taxi-Passenger Demand Using Streaming Data. *Trans. Intell. Transport. Sys.*, 14(3):1393–1402, September 2013. ISSN 1524-9050. doi: 10.1109/TITS.2013.2262376. 00048.

C. Piciarelli, G. L. Foresti, and L. Snidaro. Trajectory clustering and its applications for video surveillance. In *IEEE Conference on Advanced Video and Signal Based Surveillance, 2005.*, pages 40–45, September 2005. doi: 10.1109/AVSS.2005.1577240. 00000.

Pedro Sena Tanaka, Marcos R. Vieira, and Daniel S. Kaster. An Improved Base Algorithm for Online Discovery of Flock Patterns in Trajectories. *Journal of Information and Data Management*, 7(1):52, 2016. 00000.

Ulanbek Turdukulov, Andres Calderon, Otto Huisman, and Vasilios Retsios. Visual mining of moving flock patterns in large spatio-temporal data sets using a frequent pattern approach. *International Journal of Geographical Information Science*, 28(10):2013–2029, October 2014. ISSN 1365-8816. doi: 10.1080/13658816.2014.889834. 00000.

Takeaki Uno, Masashi Kiyomi, and Hiroki Arimura. LCM ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. In *Fimi*, volume 126, 2004. 00355.

Marcos R. Vieira and Vassilis Tsotras. *Spatio-Temporal Databases: Complex Motion Pattern Queries*. Springer Science & Business Media, October 2013. ISBN 978-3-319-02408-0. 00000.

Marcos R. Vieira, Petko Bakalov, and Vassilis J. Tsotras. On-line Discovery of Flock Patterns in Spatio-temporal Data. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '09, pages 286–295, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-649-6. doi: 10.1145/1653771.1653812. 00000.

Dong Xie, Feifei Li, Bin Yao, Gefei Li, Liang Zhou, and Minyi Guo. Simba: Efficient In-Memory Spatial Analytics. pages 1071–1085. ACM Press, 2016. ISBN 978-1-4503-3531-7. doi: 10.1145/2882903.2915237. 00000.

J. Yu, J. Wu, and M. Sarwat. A demonstration of GeoSpark: A cluster computing framework for processing big spatial data. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 1410–1413, May 2016. doi: 10.1109/ICDE.2016.7498357. 00000.

Yu Zheng and Xiaofang Zhou. *Computing with Spatial Trajectories*. Springer Science & Business Media, October 2011. ISBN 978-1-4614-1629-6. 00000.

Yu Zheng, Quannan Li, Yukun Chen, Xing Xie, and Wei-Ying Ma. Understanding mobility based on GPS data. In *Proceedings of the 10th International Conference on Ubiquitous Computing*, pages 312–321. ACM, 2008. 00462.

Yu Zheng, Lizhu Zhang, Xing Xie, and Wei-Ying Ma. Mining interesting locations and travel sequences from GPS trajectories. In *Proceedings of the 18th International Conference on World Wide Web*, pages 791–800. ACM, 2009. 00959.

Yu Zheng, Xing Xie, and Wei-Ying Ma. GeoLife: A Collaborative Social Networking Service among User, Location and Trajectory. *IEEE Data Eng. Bull.*, 33(2):32–39, 2010. 00388.