Theoretical Foundations of Big Data Analytics for Trajectories

Sepideh Aghamolaei

February 18, 2025

Challenges in big data analytics are explained by 7 Vs: Volume, Velocity, Variety, Variability, Veracity, Value, and Visualization. From the theoretical point of view, we translate these as large input size, small processing time, high-dimensions, uncertainty in data, intelligence/optimization, and visualization. Big data analytics as opposed to big data analysis discusses the processing of large volumes of data from raw data to the final result, which cannot always be achieved by a series of data analyzes as their objective functions might be in trade-offs with each other. The main theoretical works in this field either provide proofs or theoretical frameworks for existing algorithms such as explainable AI for deep learning applications, or they provide customized algorithms from the theoretical computer science literature that has been customized to work in big data settings such as streaming and massively parallel computation. Possible applications include self-driving cars, market analysis, and bioinformatics.

MapReduce class (MRC) [1] is a model where data is distributed among a set of machines and the memory of each machine (m) and the number of machines (L) is both sublinear in the input size (n) and the number of rounds (R) where the machines run in parallel and communicate at the end is limited to polylogarithmic in n. Formally, there exist constants $\eta, \psi \in (0,1)$ such that $m = O(n^{\eta}), L = O(n^{\psi})$ and the conditions $Lm = o(n^2)$ and $R = \text{polylog}(n)$ hold.

Massively parallel computation (MPC) [2] is a more restricted model that requires the total amount of computations to be at most near-linear (that is, linear ignoring polylogarithmic factors). This means the number of machines times the memory of each machine is near-linear, i.e., $Lm = O(n \,\text{polylog}(n))$. This is different from the massively parallel model designed for graphical processing units (GPUs). Based on the definitions of MRC and MPC, all MPC algorithms are in MRC.

All the problems in class **NC** are in MRC by a simulation from PRAM to MRC [3, 4, 5]. The relation to classes **NL** and **L**, and also subclasses of **NC** have been discussed [6]. Using the current best simulation [4, 5], the MRC algorithm has a logarithmic factor better time complexity (round complexity) than the PRAM algorithm.

Some of the basic parallel computations that have been generalized to MRC are parallel semi-group and parallel prefix sum and run in $O(1)$ rounds in MRC [3]. For a set of items $x_1, x_2, \ldots, x_n$ and a binary associative operator $\oplus$, these computations are defined as follows:

- semi-group: $x_1 \oplus x_2 \oplus \cdots \oplus x_n$, and
- prefix-sum: $x_1 \oplus x_2 \oplus \cdots \oplus x_i$, for $i = 1, \ldots, n$. There is a similar computation called diminished prefix sum where the i-th output is $x_1 \oplus x_2 \oplus \cdots \oplus x_{i-1}$, for $i = 1, \ldots, n$.

Examples of semi-group computations are computing the maximum, minimum, and sum of a set of numbers, and examples of prefix-sum are sorting (element ranking) and taking the maximum of the prefixes of a set of numbers. Polynomial-time computations on data of size $O(m)$ can also be done in one machine locally, so, it can be performed in one round or during an existing round (as a pre-processing or post-processing step).

Trajectory analysis is the study of polygonal curves with timestamps and is mostly categorized in geoinformatics which is the development of infrastructures for information science on earth sciences including geographic information systems (GIS) and global positioning system (GPS). In the theoretical computer science literature, trajectory analysis is discussed in similarity measures between curves, sometimes even between strings, path planning for robots, especially multi-robot scenarios, and spatio-temporal trajectory clustering to provide semantics for trajectory data, for example map reconstruction.

Formally, the Fréchet distance is defined as follows. A general two-dimensional curve $P(t)$ for $t \in [0,1]$, is defined as the set of points $\{(x(t), y(t)) \mid t \in [0,1]\}$. A reparameterization τ is a continuous and bijective function $\tau : [0,1] \to [0,1]$. The *Fréchet distance* between two curves $P, Q : [0,1] \to \mathbb{R}^2$ is defined as

$$\mathrm{d_F}(P,Q) = \inf_{\alpha(\cdot),\beta(\cdot)} \max_{t\in[0,1]} \mathrm{dist}(P(\alpha(t)), Q(\beta(t))),$$

where α and β are reparameterizations of P and Q with $\alpha(0) = 0$, $\alpha(1) = 1$, $\beta(0) = 0$, $\beta(1) = 1$, and $\mathrm{dist}(.,.)$ is the Euclidean distance or 2-norm.

Traditional trajectory analytics faces extra challenges due to multiple processing steps performed on the data, with (sub)sampling, simplification/summarization, various machine learning algorithms, and uncertainties due to changes made to the curve during the time spent on the previous steps of processing. Some of the trajectory problems discussed in the deep learning literature are (sub)trajectory classification, anomaly detection, arrival time prediction, location classification, next location/destination prediction, synthetic data generation, traffic volume prediction, and trajectory prediction/imputation [7].

We use algorithmic tools from robotics/path planning, simplification, maximum network flow/ minimum graph cut, clustering, and windowing query data structures to address as many of these problems as possible. We also use mathematical concepts such as expander graphs, Fréchet distance, and provide computational complexity results. In trajectory analysis, the maximum flow problem appears when there are several trajectories and a type of aggregation is to be computed, while clustering is often used to get more accurate results. Path planning allows the space to be pruned in places where the series of GPS coordinates are not from a car or based on a road network. Range query data structures can be used to reduce the amount of time or memory required to solve the problems, which are currently mostly used as dynamic data structures to update the solutions computed so far by the algorithm (this is useful in cases where computing a solution takes a long time hence updating an existing solution makes more sense). Some of these results have been published [8, 9, 10, 11, 12, 13, 14, 15, 16].

The target conferences for such results includes data mining and machine learning conferences for experimental results, database conferences for the data structures and algorithms for big data, and more theoretical conferences for general methods and results.

I have already worked on most of these problems separately: trajectory analysis [10, 12] where we discussed how to identify cases for which the Fréchet distance can be approximated faster than the general case (c-packed curves), clustering [9, 16] for points in metric spaces which can be used to reduce the size of the data instead of subsampling, uncertain data [13] to handle the regions created by summarizing point sets, maximum-flow [14, 15] for predicting/suggesting which steps can be taken for example based on the traffic, data structures for massive data [11, 12] for computing packedness of curves, visualization [15, 17] and path planning [10, 18].

Some of the recent results that directly discuss trajectory analysis [19, 20] are focused on subquadratic-time algorithms for the Fréchet distance. Algorithms for problems discussed on objects that are more complicated than points, such as graphs, can also be useful in trajectory analysis, for example, a map is a graph, or a flooding algorithm from a single source is a directed acyclic graph (DAG) [21]. Approximate clustering based on the correlation graph in dynamic streaming model has also been discussed [22]. Minimum spanning tree in general metric spaces in the massively parallel computation model has been proven hard [23], however, for general-enough special cases with real-world applications (in our case, for trajectory analytics or data analytics in general), approximation algorithms with reasonable approximation factors might exist (see Table 1). Abstract or combinatorial maps such as subway maps can also be used by algorithms for large-scale routing, possibly using different means of transportation, which can be modeled as hypergraphs: each port or station is modeled as a vertex and each subset reachable by a vehicle limited to a region/block is a hyper-edge. The complexity of drawing such a map depends on the order dimension of the hypergraph poset [24] but there can be a non-planar efficient map for the problem.

Algorithm	**# Rounds**	**Approximation Ratio**	**Reference**
EMST (low dimension)	$O(1)$	$(1+\epsilon)$	Andoni et al., 2014
Geometric spanner	$O(1)$	$(1+\epsilon)$	Aghamolaei et al., 2018
EMST	$O(\log\Delta)$	Exact	Andoni et al., 2018
EMST (high dimension)	$O(1)$	$\tilde{O}(\log^{1.5} n)$	Ahanchi et al., 2023
EMST (high dimension)	$O(1)$	$O(\log n)$	Jayaram et al., 2023
EMST (high dimension)	$\tilde{O}(\log\log n)$	$O(1)$	Jayaram et al., 2025

Table 1: EMST and geometric spanners in MPC. Here Δ is the diameter of the graph.

References

[1] Howard Karloff, Siddharth Suri, and Sergei Vassilvitskii. A model of computation for MapReduce. In *Proceedings of the 21st annual ACM-SIAM symposium on Discrete Algorithms*, pages 938–948. SIAM, 2010.

[2] Paul Beame, Paraschos Koutris, and Dan Suciu. Communication steps for parallel query processing. In *Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGAI symposium on principles of database systems*, pages 273–284. ACM, 2013.

[3] Michael T Goodrich, Nodari Sitchinava, and Qin Zhang. Sorting, searching, and simulation in the MapReduce framework. In *Proceedings of the 22nd International Symposium on Algorithms and Computation*, volume 7074, page 374. Springer Science & Business Media, 2011.

[4] Fabian Frei and Koichi Wada. Efficient circuit simulation in MapReduce. In *Proceedings of the 30th International Symposium on Algorithms and Computation.* Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[5] Fabian Frei and Koichi Wada. Efficient deterministic MapReduce algorithms for parallelizable problems. *Journal of Parallel and Distributed Computing*, 177:28–38, 2023.

[6] Danupon Nanongkai and Michele Scquizzato. Equivalence classes and conditional hardness in massively parallel computations. *Distributed computing*, 35(2):165–183, 2022.

[7] Anita Graser, Anahid Jalali, Jasmin Lampert, Axel Weißenfeld, and Krzysztof Janowicz. Mobilitydl: a review of deep learning from trajectory data. *GeoInformatica*, pages 1–33, 2024.

[8] Sepideh Aghamolaei, Majid Farhadi, and Hamid Zarrabi-Zadeh. Diversity maximization via composable coresets. In *Proceedings of the 27th Canadian Conference on Computational Geometry*, pages 38–48, 2015.

[9] Sepideh Aghamolaei and Mohammad Ghodsi. A composable coreset for k-center in doubling metrics. In *Proceedings of the 30th Canadian Conference on Computational Geometry*, pages 165–171, 2018.

[10] Sepideh Aghamolaei, Vahideh Keikha, Mohammad Ghodsi, and Ali Mohades. Windowing queries using minkowski sum and their extension to mapreduce. *The Journal of Supercomputing*, 77:936–972, 2021.

[11] Sepideh Aghamolaei, Fatemeh Baharifard, and Mohammad Ghodsi. Geometric spanners in the mapreduce model. In *International Computing and Combinatorics Conference*, pages 675–687, 2018.

[12] Sepideh Aghamolaei, Vahideh Keikha, Mohammad Ghodsi, and Ali Mohades. Sampling and sparsification for approximating the packedness of trajectories and detecting gatherings. *International Journal of Data Science and Analytics*, 15(2):201–216, 2023.

[13] Vahideh Keikha, Sepideh Aghamolaei, Ali Mohades, and Mohammad Ghodsi. Clustering geometrically-modeled points in the aggregated uncertainty model. *Fundamenta Informaticae*, 184, 2022.

[14] Sepideh Aghamolaei, Mohammad Ghodsi, and S Miri. A mapreduce algorithm for metric anonymity problems. In *Proceedings of the 31st Canadian Conference on Computational Geometry*, 2019.

[15] Sepideh Aghamolaei and Mohammad Ghodsi. Explainable graph clustering via expanders in the massively parallel computation model. *Information Sciences*, 2024.

[16] Sepideh Aghamolaei and Mohammad Ghodsi. Density-based clustering in mapreduce with guarantees on parallel time, space, and solution quality. *Transactions on Combinatorics*, 2024.

[17] Sepideh Aghamolaei and Mohammad Ghodsi. Planar euclidean tsp via snowflake tree. In *The Third Iranian Conference on Computational Geometry*, page 25, 2020.

[18] Sepideh Aghamolaei and Mohammad Ghodsi. A theoretical proof of angular random walk. In *First Iranian Conference on Computational Geometry*, page 11, 2020.

[19] Siu-Wing Cheng and Haoqiang Huang. Solving Fréchet distance problems by algebraic geometric methods. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 4502–4513. SIAM, 2024.

[20] Siu-Wing Cheng and Haoqiang Huang. Fréchet distance in subquadratic time. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 5100–5113. SIAM, 2025.

[21] An T Le, Georgia Chalvatzaki, Armin Biess, and Jan R Peters. Accelerating motion planning via optimal transport. *Advances in Neural Information Processing Systems*, 36, 2024.

[22] Mélanie Cambus, Fabian Kuhn, Etna Lindy, Shreyas Pai, and Jara Uitto. A $(3+\epsilon)$-approximate correlation clustering algorithm in dynamic streams. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2861–2880. SIAM, 2024.

[23] Amir Azarmehr, Soheil Behnezhad, Rajesh Jayaram, Jakub Łącki, Vahab Mirrokni, and Peilin Zhong. Massively parallel minimum spanning tree in general metric spaces. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 143–174. SIAM, 2025.

[24] Uta Priss and Dominik Dürrschnabel. Rectangular Euler diagrams and order theory. In *International Conference on Theory and Application of Diagrams*, pages 165–181. Springer, 2024.