Matching Affinity Clustering: Improved Hierarchical Clustering at Scale with Guarantees

Extended Abstract

MohammadTaghi Hajiaghayi
University of Maryland, College Park
College Park, Maryland
hajiagha@cs.umd.edu

Marina Knittel
University of Maryland, College Park
College Park, Maryland
mknittel@cs.umd.edu

ABSTRACT

Hierarchical clustering is a stronger extension of one of today's most influential unsupervised learning methods: clustering. The goal of this method is to create a hierarchy of clusters, thus constructing cluster evolutionary history and simultaneously finding clusterings at all resolutions. We propose four traits of interest for hierarchical clustering algorithms: (1) empirical performance, (2) theoretical guarantees, (3) balance (the minimum ratio between cluster sizes), and (4) scalability. While a number of algorithms are designed to achieve one to two of these traits at a time, there exist none that achieve all four.

Inspired by Bateni et al.'s scalable and empirically successful Affinity Clustering [NeurIPs 2017], we introduce Affinity's successor, *Matching Affinity Clustering*. Like its predecessor, Matching Affinity Clustering maintains strong empirical performance, even outperforming Affinity when the dataset is size 2^n and clusters are balanced, and uses *Massively Parallel Communication* as its distributed model. Designed to maintain provably balanced clusters, we show that our algorithm achieves a $(1/3-\epsilon)$-approximation for Moseley and Wang's revenue (the dual to Dasgupta's cost) when the data set is of size 2^n, and a $(1/9-\epsilon)$-approximation in general. We prove the former approximation is tight, and also that Affinity Clustering cannot do better than a $1/O(n)$-approximation. In addition, we see that our algorithm empirically performs similarly to Affinity Clustering and k-Means, outperforming many state-of-the-art serial algorithms. Along the way, we also introduce an efficient k-sized maximum matching algorithm in the MPC model.

ACM Reference Format:
MohammadTaghi Hajiaghayi and Marina Knittel. 2020. Matching Affinity Clustering: Improved Hierarchical Clustering at Scale with Guarantees. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 3 pages.

1 INTRODUCTION

Clustering is one of the most prominent methods to provide structure, in this case clusters, to unlabeled data. Hierarchical clustering elaborates on this structure by adding a hierarchy of clusters contained within superclusters. The input is a graph whose edge weights represent the similarity (or dissimilarity) between two data points. A hierarchical clustering algorithm outputs a tree T, whose leaves represent the input data, and internal nodes represent the merging of data and clusters. This process must meet at a single root vertex, representing the cluster of all data.

Obviously it is more computationally intensive to find T than a simple clustering. However, having access to such a structure provides two main advantages: (1) it allows a user to observe the data at different granularities, effectively querying T for clusterings of different sizes without recomputation, and (2) it constructs a history of relationships between data. The latter is most readily applied to phylogenetics, where dendrograms depict the evolutionary history of genes and species [17]. Hierarchical clustering in general has been used in a number of other unsupervised applications. In this paper, we explore four important qualities of a strong and efficient hierarchical clustering algorithm:

(1) **Theoretical guarantees**. Previously, analysis for hierarchical clustering performance has relied on experimental evaluation. While experimental performance is one indicator for success, relying completely on it is problematic in that it cannot ensure performance across a wider range of datasets. Researchers combat this by considering optimization functions to evaluate broader guarantees [6, 18]. One function that has received significant attention recently [5, 9] is a cost function proposed by Dasgupta [10] to evaluate clusterings on graphs where edge weights represent data similarity. Cost is intuitively defined, encouraging any pair of highly similar points to be clustered more tightly. However, Charikar and Chatziafratis [4] showed that it is likely NP-hard to approximate cost within a constant factor. To overcome this we examine its dual, revenue [20]. We are interested in constant factor approximation algorithms for this function.
(2) **Empirical performance**. As theoretical guarantees are often only intuitive proxies for broader evaluation, it is still important to evaluate the empirical performance of algorithms on real datasets. Currently, Bateni et al. [3]'s Affinity Clustering remains the state-of-the-art for scalable hierarchical clustering algorithms with strong empirical results. To compete, we must ensure our algorithm exhibits similar empirical performance to Affinity Clustering. This is why we use Affinity Clustering as an inspirational foundation for our algorithm: so that we can maintain or exceed its experimental success.
(3) **Balance**. One downside of algorithms like Affinity Clustering is that they can create extremely unbalanced clusters. In many natural clustering problems, balanced clusters can be

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand. © 2020 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

preferable or more accurate. A simple example is a clustering of a population into genders, where the gender ratio is likely balanced. Some more specific applications include image collection clustering, where balanced clusters can make the database more easily navigable [12], and wireless sensor networks, where balancing clusters of sensor nodes ensures no cluster head gets overloaded [1]. Here, we define balance as the minimum ratio between cluster sizes.

(4) **Scalability**. Most current approximations for revenue are serial and do not ensure performance at scale. We approach this problem by utilizing the modularity of distributed algorithms for hierarchical clustering. Clustering itself, as well as other common big data problems, has been a topic of interest in the distributed community in recent years [2, 7, 8]. In particular, hierarchical clustering has been studied by Jin et al. [15, 16], but only Bateni et al. [3] has attempted to ensure theoretical guarantees. In their paper, Bateni et al. introduce their Affinity Clustering algorithm and briefly propose a cost metric for analysis. However, little motivation is provided for this metric. Therefore, we are interested in evaluating distributed algorithms with respect to a more well-founded cost function, such as revenue.

For our distributed model, we look to *Massively Parallel Communication (MPC)*, which was used to design Affinity Clustering. MPC is a theoretical abstraction of *MapReduce*, a popular programming framework famous for its ease of use, fault tolerance, and scalability [11]. In MPC, individual machines carry only a fraction of the data and execute individual computations in rounds. At the end of each round, machines can send limited messages to each other. Complexities of interest are the number of rounds and the individual machine space. This framework has been used in the analysis for many large-scale problems in big data, including clustering [14, 19]. It is a natural selection for this work.

There exist algorithms that can achieve some of these qualities. Affinity Clustering, notably, exhibits good empirical performance and was implemented efficiently in the MPC model. While Bateni et al. [3] describe some minor theoretical guarantees for Affinity Clustering, we believe that proving an algorithm's ability to optimize for revenue is a stronger and more well-founded result due to its popularity and relation to Dasgupta's cost function. A simple random divisive algorithm proposed by Charikar and Chatziafratis [4] was shown to achieve a $1/3$-approximation for revenue and can be efficiently implemented using MPC (simply allocate each cluster in the current clustering to its own machine). However, we show that it does not perform well empirically. Similarly, balanced partitioning may achieve balanced clusters, but it is unclear whether it is scalable, and it has not been shown to achieve strong theoretical guarantees.

In this paper, we are interested in finding an algorithm that, unlike the current state-of-the-art, can achieve all four strengths simultaneously.

1.1 Our contributions

In this work, we propose a new algorithm, *Matching Affinity Clustering*, for hierarchical clustering in the distributed setting. It is inspired by Affinity Clustering's reliance on the minimum spanning tree in order to greedily select edges to merge across [3]. Instead of using the MST, we iteratively merge clusters based off a maximum matching. We show that it achieves state-of-the-art results in all four qualities.

First, we prove that Matching Affinity Clustering achieves a good approximation for revenue to **theoretically motivate** our findings. Relative to Affinity Clustering, this a significantly stronger guarantee, as revenue is a well-studied and motivated dual to Dasgupta's popular cost function. These results are presented in Theorem 1.1.

Theorem 1.1. *There is a hierarchical clustering algorithm in MPC that, on graphs with $n = 2^N$ vertices, achieves a $(1/3 - \epsilon)$-approximation for revenue in $O\left(\log(n) \log\log(n) \cdot (1/\epsilon)^{O(1/\epsilon)}\right)$ rounds with $O(n)$ machine space. This approximation becomes $1/9 - \epsilon$ in $O\left(\log(nW) \log\log(n) \cdot (1/\epsilon)^{O(1/\epsilon)}\right)$ rounds with $O(n)$ machine space for general graphs with max edge weight W.*

We also show the $1/3 - \epsilon$ bound is tight. Furthermore, in Theorem 1.2, we prove that Matching Affinity Clustering approximates the dual better than Affinity Clustering by a factor of $O(n)$ for some graphs, and thus Affinity Clustering cannot approximate revenue.

Theorem 1.2. *There is a family of graphs on which Affinity Clustering can achieve at best a $1/O(n)$-factor approximation for revenue.*

As a side result, we present an efficient and near-optimal MPC algorithm for k-sized maximum matching.

Theorem 1.3. *There exists an MPC algorithm for k-sized maximum matching with nonnegative edge weights and max edge weight W for $n/2 > k$ and $k = O(n)$ that achieves a $(1 - \epsilon)$-approximation in $O(\log(nW) \log\log(n) \cdot (1/\epsilon)^{1/\epsilon})$ rounds and $O(n/polylog(n))$ machine space.*

To evaluate the **empirical performance** of our algorithm, we run Bateni et al. [3]'s experiments used for Affinity Clustering on small-scale datasets. We find Matching Affinity Clustering performs as well as Affinity Clustering and k-Means and outperforms other serial algorithms on raw data. Additionally, on randomly filtered datasets of size 2^n which have balanced ground truth clusters, we find that Matching Affinity Clustering consistently outperforms Affinity Clustering (and other tested algorithms) by at least a small but clear margin. This implies Matching Affinity Clustering may be more useful on balanced datasets than Affinity Clustering.

To confirm the **balance** of our algorithm, we are able to prove that Matching Affinity Clustering achieves perfectly balanced clusters on datasets of size 2^n, and otherwise guarantee near balance (a cluster size ratio of at most 2). This was also confirmed in our empirical evaluation.

Finally, Matching Affinity Clustering is highly **scalable** because it was designed in the same MPC framework as Affinity Clustering. We provide similar complexity guarantees to Affinity Clustering, which implies that it is also significantly more scalable than popular serial algorithms.

Matching Affinity Clustering is ultimately a nice, simply motivated successor to Affinity Clustering that achieves all four desired qualities: empirical performance, theoretical guarantees, balance, and scalability. No other algorithms that we know of can achieve more than two of these qualities.

REFERENCES

[1] Tarachand Amgoth and Prasanta K. Jana. 2014. Energy efficient and load balanced clustering algorithms for wireless sensor networks. *IJICT* 6, 3/4 (2014), 272–291. https://doi.org/10.1504/IJICT.2014.063216

[2] Maria Fiorina Balcan, Steven Ehrlich, and Yingyu Liang. 2013. Distributed K-Means and k-Median Clustering on General Topologies. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13)*. Curran Associates Inc., Red Hook, NY, USA, 1995–2003.

[3] MohammadHossein Bateni, Soheil Behnezhad, Mahsa Derakhshan, Mohammad-Taghi Hajiaghayi, Raimondas Kiveris, Silvio Lattanzi, and Vahab S. Mirrokni. 2017. Affinity Clustering: Hierarchical Clustering at Scale, See [13], 6867–6877.

[4] Moses Charikar and Vaggos Chatziafratis. 2017. Approximate Hierarchical Clustering via Sparsest Cut and Spreading Metrics. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '17)*. Society for Industrial and Applied Mathematics, USA, 841–854.

[5] Moses Charikar, Vaggos Chatziafratis, Rad Niazadeh, and Grigory Yaroslavtsev. 2019. Hierarchical Clustering for Euclidean Data. In *Proceedings of Machine Learning Research (Proceedings of Machine Learning Research)*, Kamalika Chaudhuri and Masashi Sugiyama (Eds.), Vol. 89. PMLR, 2721–2730.

[6] Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. 2004. Incremental Clustering and Dynamic Information Retrieval. *SIAM J. Comput.* 33, 6 (2004), 1417–1440. https://doi.org/10.1137/S0097539702418498

[7] Rajesh Chitnis, Graham Cormode, Hossein Esfandiari, MohammadTaghi Hajiaghayi, Andrew McGregor, Morteza Monemizadeh, and Sofya Vorotnikova. 2016. Kernelization via Sampling with Applications to Finding Matchings and Related Problems in Dynamic Graph Streams. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '16)*. Society for Industrial and Applied Mathematics, USA, 1326–1344.

[8] Rajesh Chitnis, Graham Cormode, MohammadTaghi Hajiaghayi, and Morteza Monemizadeh. 2015. Parameterized Streaming: Maximal Matching and Vertex Cover. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '15)*. Society for Industrial and Applied Mathematics, USA, 1234–1251.

[9] Vincent Cohen-addad, Varun Kanade, Frederik Mallmann-trenn, and Claire Mathieu. 2019. Hierarchical Clustering: Objective Functions and Algorithms. *J. ACM* 66, 4, Article Article 26 (June 2019), 42 pages. https://doi.org/10.1145/3321386

[10] Sanjoy Dasgupta. 2016. A Cost Function for Similarity-Based Hierarchical Clustering. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing (STOC '16)*. Association for Computing Machinery, New York, NY, USA, 118–127. https://doi.org/10.1145/2897518.2897527

[11] Jeffrey Dean and Sanjay Ghemawat. 2008. MapReduce: simplified data processing on large clusters. *Commun. ACM* 51, 1 (2008), 107–113. https://doi.org/10.1145/1327452.1327492

[12] Andreas Dengel, Tim Althoff, and A Ulges. 2011. Balanced Clustering for Content-Based Image Browsing. In *German Computer Science Society, Informatiktage.*

[13] Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 2017. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA.*

[14] Sungjin Im, Benjamin Moseley, and Xiaorui Sun. 2017. Efficient Massively Parallel Methods for Dynamic Programming. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2017)*. Association for Computing Machinery, New York, NY, USA, 798–811. https://doi.org/10.1145/3055399.3055460

[15] Chen Jin, Ruoqian Liu, Zhengzhang Chen, William Hendrix, Ankit Agrawal, and Alok N. Choudhary. 2015. A Scalable Hierarchical Clustering Algorithm Using Spark. In *First IEEE International Conference on Big Data Computing Service and Applications, BigDataService 2015, Redwood City, CA, USA, March 30 - April 2, 2015.* IEEE Computer Society, 418–426. https://doi.org/10.1109/BigDataService.2015.67

[16] Chen Jin, Md. Mostofa Ali Patwary, William Hendrix, Ankit Agrawal, Wei-keng Liao, and Alok Choudhary. 2013. DiSC: A Distributed Single-linkage Hierarchical Clustering Algorithm using MapReduce. *International Workshop on Data Intensive Computing in the Clouds (DataCloud)* (11 2013).

[17] Alexander Kraskov, Harald Stögbauer, Ralph G. Andrzejak, and Peter Grassberger. 2003. Hierarchical Clustering Using Mutual Information. *CoRR* q-bio.QM/0311037 (2003).

[18] Guolong Lin, Chandrashekhar Nagarajan, Rajmohan Rajaraman, and David P. Williamson. 2006. A general approach for incremental approximation and hierarchical clustering. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2006, Miami, Florida, USA, January 22-26, 2006.* ACM Press, 1147–1156. http://dl.acm.org/citation.cfm?id=1109557

[19] Simone A. Ludwig. 2015. MapReduce-based fuzzy c-means clustering algorithm: implementation and scalability. *Int. J. Machine Learning & Cybernetics* 6, 6 (2015), 923–934. https://doi.org/10.1007/s13042-015-0367-0

[20] Benjamin Moseley and Joshua Wang. 2017. Approximation Bounds for Hierarchical Clustering: Average Linkage, Bisecting K-means, and Local Search, See [13], 3097–3106.