



Approximation Algorithms for Large Scale Data Analysis

Barna Saha

barnas@berkeley.edu

University of California Berkeley

Berkeley, California, USA

ABSTRACT

One of the greatest successes of computational complexity theory is the classification of countless fundamental computational problems into polynomial-time and NP-hard ones, two classes that are often referred to as tractable and intractable, respectively. However, this crude distinction of algorithmic efficiency is clearly insufficient when handling today's large scale of data. We need a finer-grained design and analysis of algorithms that pinpoints the exact exponent of polynomial running time, and a better understanding of when a speed-up is not possible. Based on stronger complexity assumptions than P vs NP, like the Strong Exponential Time Hypothesis, recently conditional lower bounds for a variety of fundamental problems in P have been proposed. Unfortunately, these conditional lower bounds often break down when one may settle for a near-optimal solution. Indeed, approximation algorithms can play a significant role when designing fast algorithms not just for traditional NP Hard problems, but also for polynomial time problems.

For some applications arising in machine learning, the time complexity of the underlying algorithms is not sufficient to ensure a fast solution. It is often needed to collect side information about the data to ensure high accuracy. This requires low query complexity.

In this presentation, we will cover new facets of fast algorithm design for large scale data analysis that emphasizes on the role of developing approximation algorithms for better polynomial time/query complexity.

CCS CONCEPTS

• **Theory of computation** → **Approximation algorithms analysis**.

KEYWORDS

approximation algorithms, fine-grained complexity, dynamic algorithm, interactive algorithms

ACM Reference Format:

Barna Saha. 2021. Approximation Algorithms for Large Scale Data Analysis. In *Proceedings of the 40th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS '21)*, June 20–25, 2021, Virtual Event, China. ACM, New York, NY, USA, 3 pages. <https://doi.org/10.1145/3452021.3458813>



This work is licensed under a Creative Commons Attribution International 4.0 License.

PODS '21, June 20–25, 2021, Virtual Event, China.

© 2021 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-8381-3/21/06.

<https://doi.org/10.1145/3452021.3458813>

1 EXTENDED OVERVIEW

From its inception, complexity theory through concepts such as NP-Hardness has classified computational problems into those that have relatively efficient solutions versus those that are intractable. Any problem solvable in time polynomial in the input size falls in the first category. On the other hand, *approximation algorithms* are developed to obtain efficient solutions for NP-Hard problems. Trading accuracy for time, they return near-optimal solutions in polynomial time. Over the years, seminal works by researchers charting the landscape of approximability of NP-Hard problems have contributed to a significant growth of this field [20, 35, 40]. Many systematic techniques based on convex programming, such as linear and semi-definite programming relaxation [16, 27, 32, 33], metric embedding [11, 23], local search [5], probabilistically checkable proof [6], etc. have been developed which have enriched our discipline.

However, the crude distinction of algorithmic efficiency—polynomial vs NP-hard, is insufficient when handling today's large scale of data. We need a finer-grained design and analysis of algorithms that pinpoints the exact exponent of polynomial running time, and a better understanding of when a speed-up is not possible. The traditional theory of approximation algorithms often ignores the exact polynomial time complexity. Systematic techniques that emphasize on the trade-off between running time and approximation, especially for high-degree polynomial time problems are required. Often, there are nice structures in data or a problem that need to be exploited to accelerate algorithms. Time complexity is not always the right measure of efficiency. For core learning problems, query complexity, that quantifies the amount of labelled data acquired via active querying, is often more important. In this presentation, we will look into these new aspects of approximation algorithms design that are relevant for large scale data analysis.

1.1 Fine-Grained Approximation Algorithms

There are many important problems within P that can be solved in time $\tilde{O}(n^k)$ ¹ for some constant k , but for which even decades of attempts have not resulted into any $\tilde{O}(n^{k-\epsilon})$ time algorithms for any constant $\epsilon > 0$. Such problems pervade computer science. From graph algorithms, all-pairs shortest paths (APSP) on weighted dense graphs can be computed by the Floyd-Warshall's classical algorithm in $O(n^3)$ time, and no truly subcubic algorithms are known. From computational biology, the fastest known algorithm to compute the edit distance (minimum number of insertion, deletion and substitution edits) between two length n genome sequences dates back to 1974 by Wagner and Fisher that runs in $\tilde{O}(n^2)$ time. From computational geometry, given n weighted points in plane, one can determine an axis-parallel box that maximizes the total

¹Here $\tilde{O}$ hides $n^{o(1)}$ factors.

weight of the points it contains in $O(n^2)$ time, and this is the best known [10, 14].

NP-hardness itself seems to have little use in showing that for problems like above, their long-standing running time is probably optimal. Instead, a more fine-grained approach has been used in recent years. In this approach, one starts with a widely believed hypothesis about the time complexity of a key problem, e.g., Strong Exponential Time Hypothesis (SETH) that states CNF-SAT on n variables cannot be solved in time $2^{(1-\epsilon)n}$ for any $\epsilon > 0$ [21, 22], and then uses low-polynomial time reductions to reduce this key problem to other important problems, giving conditional lower bounds on how fast these problems can be solved. For example, assuming SETH, there is no $O(n^{2-\epsilon})$ algorithm to compute string edit distance [8]; if all-pairs shortest path does not have an $n^{3-\epsilon}$ time algorithm, then neither do a large collection of fundamental graph and matrix problems, and vice versa [34].

Unfortunately, these conditional lower bounds break down when one may settle for a near-optimal solution. Consider the following as illustrative examples: (1) for string edit distance, there is an $O(n^{1+\epsilon})$ -time algorithm that returns a solution within $O(f(\epsilon)OPT)$, where OPT is the minimum number of edits and $f(\epsilon)$ is some function of ϵ independent of n [4]; and (2) for weighted APSP, there is an $\tilde{O}(\frac{1}{\epsilon}n^\omega)$ -time algorithm which approximates every shortest path within a $(1 + c)$ -factor for any $c > 0$, where $\omega \leq 2.373$ is the exponent of fast matrix multiplication [41].

Can we design a sublinear time algorithm for string edit distance with proper trade-offs with accuracy? Are there special structures in a problem instance that help to design faster algorithms? E.g., when the weights are small integers, an exact $O(n^\omega)$ algorithm exists for APSP [3, 30]. We will look into new developments in fine-grained approximation algorithms that have attempted to design new techniques and models to exploit inherent structures in the data [12, 13, 18, 19, 24, 28, 39].

1.2 Query Complexity vs Accuracy

Time complexity may not always be the primary measure of efficiency. There are many core machine learning problems where query complexity, that quantifies the amount of labelled data acquired via active querying, is more important. Query complexity or the decision tree complexity is a classical model of computation [7, 29]. Let $f : \{0, 1\}^N \rightarrow B$ be a function we want to compute given an input $x \in \{0, 1\}^N$. The bits of the input can only be accessed by adaptively querying, and the objective is to minimize the number of queries made. Many modern machine learning applications (including image and video classification, protein sequence classification and entity resolution) have massive amounts of unannotated or unlabeled data. Active learning is an extensively used technique for these applications, where the algorithm is presented with a large pool of unlabeled examples (such as all images available on the web) and can interactively ask for the labels of examples of its own choosing from the pool, with the goal to drastically reduce labeling effort (e.g., see the surveys [9, 31]). This line of works are gaining even more momentum due to the wide-spread use of crowdsourcing (e.g., Amazon Mechanical Turk) to obtain labelled data [15, 17, 36–38]. Clearly in this setting, query complexity is the natural measure of “efficiency”.

Consider a clustering problem. We are given $V \equiv [n]$, a set of n points, that need to be clustered into k clusters $V_i, i = 1, \dots, k; V_i \cap V_j = \emptyset, \forall i \neq j$, along with a pair-wise similarity or distance matrix; the subsets $V_i \subset [n]$ and k are unknown to us. There is an oracle which can answer to any pair-wise queries $V \times V \rightarrow \{\pm 1\}$ where a query answer of $+1$ for $(u, v) \in V \times V$ indicates u and v belong to the same cluster, and -1 indicates they do not; how many minimum such queries are needed to reconstruct the clusters exactly or approximately? For document classification, the similarity matrix can store the Jaccard similarity between each pair of documents. For k -means clustering, the distance matrix stores pair-wise Euclidean distance. Clearly, the presence of this *side information* should affect the query complexity result. Moreover, it may also be possible that the oracle gives erroneous answers with some probability $p < \frac{1}{2}$. Clustering with querying is a well-motivated practical problem [15, 17, 25, 26, 36–38]. In this presentation, we will look into new developments in this area that allows for obtaining better query complexity results via approximation [1, 2].

REFERENCES

- [1] R. Addanki, S. Galhotra, and B. Saha. How to design robust algorithms using noisy comparison oracle. In *PVLDB*, 2021.
- [2] N. Ailon, A. Bhattacharya, R. Jaiswal, and A. Kumar. Approximate clustering with same-cluster queries. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11–14, 2018, Cambridge, MA, USA*, volume 94 of *LIPICs*, pages 40:1–40:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.
- [3] N. Alon, Z. Galil, and O. Margalit. On the exponent of the all pairs shortest path problem. *J. Comput. Syst. Sci.*, 54(2):255–262, Apr. 1997.
- [4] A. Andoni and N. S. Nosatzki. Edit distance in near-linear time: it’s a constant factor. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16–19, 2020*, pages 990–1001. IEEE, 2020.
- [5] E. Angel. A survey of approximation results for local search algorithms. In *Efficient Approximation and Online Algorithms*, pages 30–73. Springer, 2006.
- [6] S. Arora. *Probabilistic checking of proofs and hardness of approximation problems*. PhD thesis, 1994.
- [7] S. Arora and B. Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.
- [8] A. Backurs and P. Indyk. Edit distance cannot be computed in strongly sub-quadratic time (unless seth is false). In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 51–58. ACM, 2015.
- [9] M.-F. Balcan and R. Uner. Active learning—modern learning theory. *Encyclopedia of Algorithms*, pages 8–13, 2016.
- [10] J. Barbay, T. M. Chan, G. Navarro, and P. Pérez-Lantero. Maximum-weight planar boxes in $o(n^2)$ time (and better). *Information Processing Letters*, 114(8):437–445, 2014.
- [11] Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 184–193. IEEE, 1996.
- [12] K. Bringmann, F. Grandoni, B. Saha, and V. V. Williams. Truly subcubic algorithms for language edit distance and RNA folding via fast bounded-difference min-plus product. *SIAM J. Comput.*, 48(2):481–512, 2019.
- [13] D. Chakraborty, D. Das, E. Goldenberg, M. Koucký, and M. E. Saks. Approximating edit distance within constant factor in truly sub-quadratic time. *J. ACM*, 67(6):36:1–36:22, 2020.
- [14] T. M. Chan. Klee’s measure problem made easy. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 410–419. IEEE, 2013.
- [15] D. Firmani, B. Saha, and D. Srivastava. Online entity resolution using an oracle. *PVLDB*, 9(5):384–395, 2016.
- [16] M. X. Goemans. Semidefinite programming in combinatorial optimization. *Mathematical Programming*, 79(1-3):143–161, 1997.
- [17] C. Gokhale, S. Das, A. Doan, J. F. Naughton, N. Rampalli, J. Shavlik, and X. Zhu. Corleone: Hands-off crowdsourcing for entity matching. In *SIGMOD Conference*, pages 601–612, 2014.
- [18] E. Goldenberg, R. Krauthgamer, and B. Saha. Sublinear algorithms for gap edit distance. In D. Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9–12, 2019*, pages 1101–1120. IEEE Computer Society, 2019.
- [19] E. Goldenberg, A. Rubinfeld, and B. Saha. Does preprocessing help in fast sequence comparisons? In *Proceedings of the 52nd Annual ACM SIGACT Symposium*

- on *Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 657–670. ACM, 2020.
- [20] D. S. Hochbaum. *Approximation algorithms for NP-hard problems*. PWS Publishing Co., 1996.
 - [21] R. Impagliazzo and R. Paturi. Complexity of k -sat. In *Computational Complexity, 1999. Proceedings. Fourteenth Annual IEEE Conference on*, pages 237–240. IEEE, 1999.
 - [22] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? In *Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on*, pages 653–662. IEEE, 1998.
 - [23] P. Indyk. Algorithmic applications of low-distortion geometric embeddings.
 - [24] T. Kociumaka and B. Saha. Sublinear-time algorithms for computing & embedding gap edit distance. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 1168–1179. IEEE, 2020.
 - [25] A. Mazumdar and B. Saha. Clustering with noisy queries. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5788–5799, 2017.
 - [26] A. Mazumdar and B. Saha. Query complexity of clustering with side information. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4682–4693, 2017.
 - [27] T. Rothvoß. The lasserre hierarchy in approximation algorithms. *Lecture Notes for the MAPSP*, pages 1–25, 2013.
 - [28] B. Saha. Fast & space-efficient approximations of language edit distance and RNA folding: An amnesic dynamic programming approach. In C. Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 295–306. IEEE Computer Society, 2017.
 - [29] M. Saks and A. Wigderson. Probabilistic boolean decision trees and the complexity of evaluating game trees. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 29–38. IEEE, 1986.
 - [30] R. Seidel. On the all-pairs-shortest-path problem in unweighted undirected graphs. *J. Comput. Syst. Sci.*, 51(3):400–403, 1995.
 - [31] B. Settles. Active learning literature survey. *University of Wisconsin, Madison*, 52(55-66):11.
 - [32] D. B. Shmoys. Using linear programming in the design and analysis of approximation algorithms: two illustrative problems. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 15–32. Springer, 1998.
 - [33] A. Srinivasan. Approximation algorithms via randomized rounding: A survey. In *Series in Advanced Topics in Mathematics, Polish Scientific Publishers PWN*, 1999.
 - [34] V. Vassilevska Williams and R. Williams. Subcubic equivalences between path, matrix and triangle problems. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 645–654, 2010.
 - [35] V. V. Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013.
 - [36] V. Verroios and H. Garcia-Molina. Entity resolution with crowd errors. In *31st IEEE International Conference on Data Engineering, ICDE 2015, Seoul, South Korea, April 13-17, 2015*, pages 219–230, 2015.
 - [37] N. Vespapunt, K. Bellare, and N. Dalvi. Crowdsourcing algorithms for entity resolution. *PVLDB*, 7(12):1071–1082, 2014.
 - [38] J. Wang, G. Li, T. Kraska, M. J. Franklin, and J. Feng. Leveraging transitive relations for crowdsourced joins. In *SIGMOD Conference*, pages 229–240, 2013.
 - [39] V. V. Williams and Y. Xu. Truly subcubic min-plus product for less structured matrices, with applications. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 12–29. SIAM, 2020.
 - [40] D. P. Williamson and D. B. Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011.
 - [41] U. Zwick. All pairs shortest paths in weighted directed graphs—exact and almost exact algorithms. In *FOCS*, 1998.