Fine-Grained Complexity Theory

Karl Bringmann
Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany
kbringma@mpi-inf.mpg.de

Abstract

Suppose the fastest algorithm that we can design for some problem runs in time $O(n^2)$. However, we want to solve the problem on big data inputs, for which quadratic time is impractically slow. We can keep searching for a faster algorithm, but maybe none exists. Is there any reasoning that provides evidence against significantly faster algorithms, and thus allows us to *stop searching*? In other words, is there an *analogue of NP-hardness for polynomial-time problems*?

In this tutorial, we will give an introduction to *fine-grained complexity theory*, which allows to rule out faster algorithms by proving *conditional lower bounds* via *fine-grained reductions* from certain *key conjectures*. We will define these terms and show exemplary lower bounds.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases Hardness in P, conditional lower bound, fine-grained reduction

Digital Object Identifier 10.4230/LIPIcs.STACS.2019.4

Category Tutorial

1 Introduction

The traditional way of establishing a problem as intractable is to prove it to be NP-hard. This makes a polynomial-time algorithm unlikely, so even for medium-size instances we cannot expect to solve the problem in reasonable time, at least on worst-case instances.

However, in a modern big data world with inputs such as DNA sequences, social network graphs, or sensor network measurings, even a quadratic-time algorithm can be too slow and essentially only near-linear-time algorithms are feasible. This shift necessitates changes in intractability theory. In order to avoid searching until eternity for a faster algorithm that does not exist, we need intractability tools that establish far-from-linear lower bounds. In other words, we need an analogue of NP-hardness for polynomial-time problems. Unfortunately, P vs. NP is too coarse to even differentiate between running time $O(n)$ and $O(n^{100})$, and no techniques for proving unconditional lower bounds higher than $\Theta(n \log n)$ are known.

Therefore, the modern approach is to prove *conditional lower bounds*. To this end, we start from a widely believed *conjecture*[1] about the time complexity of a key problem, and transfer the conjectured intractability to another problem via a *fine-grained reduction*, yielding a conditional lower bound on how fast the other problem can be solved. An exemplary conjecture is the Strong Exponential Time Hypothesis, which essentially states that any algorithm for Satisfiability requires time $2^{(1-o(1))n}$ in the worst case [31], see Section 2 for details. The resulting area of *fine-grained complexity theory*, also sometimes called *hardness in P*, had initial results in the early 90s [29], was heavily influenced by developments in the fixed-parameter tractability community [23, 27, 33], and started to mature in the last 5 years, with a wealth of publications appearing every year at the topmost theory conferences, see [1–4, 6–15, 17, 18, 20, 21, 25, 30, 34, 35, 39].

[1] Some authors prefer the terminology *hypothesis*.

The development of fine-grained complexity theory in particular enabled the *design of conditionally best-possible algorithms*: On the one hand, we design an efficient algorithm running in some time bound $T(n)$, on the other hand, we prove a conditional lower bound ruling out time $T(n)^{1-\delta}$ for any $\delta > 0$ using fine-grained complexity theory. Together, we determined $T(n)$ as the *best-possible* time complexity of our problem (up to lower order factors and conditional on a plausible complexity-theoretic assumption). This provides strong indication to stop searching for a faster algorithm.

Despite fine-grained complexity theory being a young field of research, conditionally best-possible algorithms have already been found for various problems. The fine-grained approach has been particularly successful for dynamic programming problems. For instance, assuming the Strong Exponential Time Hypothesis, quadratic running time is essentially optimal for similarity measures such as Edit distance [13], Longest Common Subsequence [3, 21], and Fréchet distance [18]. These developments have also fueled algorithmic improvements, e.g., the classic $O(nt)$-time algorithm for Subset Sum[2] from 1957 [16] has been improved to time $O(t \operatorname{polylog}(t))$ [19], which matches a SETH-based lower bound [5].

In this tutorial, we give an introduction to fine-grained complexity theory. An overview of the key conjectures is presented in Section 2. We introduce the notions of fine-grained reduction and conditional lower bound in Section 3. Then, in Sections 4 and 5 we show basic examples of fine-grained reductions. Finally, we conclude in Section 6.

2 Key Conjectures

The four most central conjectures of fine-grained complexity theory are as follows.

Satisfiability

Recall the standard k-SAT problem: We are given a formula on n Boolean variables in k-CNF (i.e., the formula is a conjunction of clauses, where each clause is a disjunction of at most k literals, and each literal is a negated or unnegated variable). The task is to decide whether there is a satisfying assignment (i.e., an assignment of each variable to true or false for which the formula evaluates to true). This problem can be solved naively in time $O(2^n n^k)$, by enumerating all 2^n assignments and checking each of at most $O(n^k)$ clauses. The *Strong Exponential Time Hypothesis* (SETH) states that the naive running time is essentially optimal when k tends to infinity. More formally, for any $\delta > 0$ there is a $k \geq 3$ such that k-SAT has no $O(2^{(1-\delta)n})$-time algorithm [31].

Orthogonal Vectors

Given sets A, B consisting of n vectors in $\{0,1\}^d$, decide whether there are vectors $a \in A, b \in B$ that are orthogonal (i.e., for any $1 \leq i \leq d$ we have $a[i] \cdot b[i] = 0$). This problem can be solved naively in time $O(n^2 d)$, by enumerating all pairs of vectors and checking orthogonality in time $O(d)$. It is conjectured that Orthogonal Vectors has no $O(n^{2-\delta} d^c)$-time algorithm for any $\delta, c > 0$. It is well-known that SETH implies the Orthogonal Vectors conjecture [41].

[2] Given a set X of n positive integers and a target t, does any subset of X sum to exactly t?

All Pairs Shortest Path

Given a graph G with n nodes and positive edge weights, compute for all pairs of nodes their shortest path distance. The classic Floyd-Warshall algorithm solves this problem in time $O(n^3)$ [28, 40]. It is conjectured that All Pairs Shortest Path has no $O(n^{3-\delta})$-time algorithm for any $\delta > 0$.

3SUM

Given a set X of n integers, decide whether there are $a, b, c \in X$ with $a+b+c = 0$. A folklore algorithm solves this problem in time $O(n^2)$. It is conjectured that there is no $O(n^{2-\delta})$-time algorithm for any $\delta > 0$.

We remark that for all of these problems, lower order improvements beyond the stated running times are known. For instance, All Pairs Shortest Path can be solved in time $n^3/2^{\Omega(\sqrt{\log n})}$ [42]. However, these improvements are not enough to falsify the conjectures.

3 Fine-Grained Reductions and Conditional Lower Bounds

For simplicity and since they form the majority of reductions in the literature, here we only consider many-one reductions, also known as Karp reductions. For a fine-grained variant of Turing reductions, see e.g. [37].

▶ **Definition 1.** *For problems P, Q and time bounds t_P, t_Q, a* fine-grained reduction *from (P, t_P) to (Q, t_Q) is an algorithm that, given an instance I of P, computes an instance J of Q such that:*

1. *I is a YES-instance of P if and only if J is a YES-instance of Q,*
2. *for any $\varepsilon > 0$ there is a $\delta > 0$ such that $t_Q(|J|)^{1-\varepsilon} = O(t_P(|I|)^{1-\delta})$, and*
3. *the running time of the reduction is $O(t_P(|I|)^{1-\gamma})$ for some $\gamma > 0$.*

In particular, if there is a fine-grained reduction from (P, t_P) to (Q, t_Q) and there is an algorithm for Q running in time $O(t_Q(n)^{1-\varepsilon})$ for some $\varepsilon > 0$, then by combining the two we can solve any instance I of P in time

$$O(t_P(|I|)^{1-\gamma} + t_Q(|J|)^{1-\varepsilon}) = O(t_P(|I|)^{1-\gamma} + t_P(|I|)^{1-\delta}) = O(t_P(|I|)^{1-\delta'}),$$

for some $\delta' > 0$. In other words, any significant improvement over time $t_Q(n)$ for Q yields a significant improvement over time $t_P(n)$ for P. Or, equivalently, if P cannot be solved significantly faster than in time $t_P(n)$, then Q cannot be solved significantly faster than in time $t_Q(n)$.

Suppose we have a fine-grained reduction from (All Pairs Shortest Path, n^3) to (Q, t_Q). Then problem Q cannot be solved in time $O(t_Q(n)^{1-\varepsilon})$ for any $\varepsilon > 0$ unless All Pairs Shortest Path can be solved in time $O(n^{3-\delta})$ for some $\delta > 0$, meaning that the All Pairs Shortest Path conjecture fails. In this situation, we say that we have proven a *conditional lower bound* of $t_Q(n)^{1-o(1)}$ for problem Q, assuming the All Pairs Shortest Path conjecture.

4 Example I: SETH-Hardness of Orthogonal Vectors

As a first example for a fine-grained reduction, we present the following by-now classic result.

▶ **Theorem 2** ([41]). *The Strong Exponential Time Hypothesis implies the Orthogonal Vectors conjecture.*

Proof. We show a fine-grained reduction from Satisfiability to Orthogonal Vectors. Specifically, given a k-CNF formula ϕ on n variables and m clauses, we will construct sets A, B of $N = 2^{n/2}$ vectors in dimension $D = m$.

Denote the clauses of ϕ by $C_1, \ldots, C_m$. We split the n variables into sets X, Y of size $n/2$. For any assigment α of X, we construct a corresponding vector $a(\alpha) \in A$, by setting its i-th coordinate to 0 if α satisfies clause C_i (i.e., if there is a literal in C_i that is set to true by applying assignment α to X), and setting it to 1 otherwise. Similarly, for any assigment β of Y, we construct a corresponding vector $b(\beta) \in B$, by setting its i-th coordinate to 0 if β satisfies clause C_i, and setting it to 1 otherwise. Observe that $a(\alpha)$ and $b(\beta)$ are orthogonal if and only if (α, β) forms a satisfying assignment of ϕ, since orthogonality means that each clause is satisfied by at least one of the two half-assignments α, β. Thus, we constructed an equivalent Orthogonal Vectors instance.

Note that we indeed constructed sets A, B consisting of $N = 2^{n/2}$ vectors in $\{0,1\}^D$ for $D = m$. Also note that we can construct A, B in time $O(ND)$. Now suppose that the Orthogonal Vectors conjecture fails, i.e., Orthogonal Vectors has an $O(N^{2-\varepsilon}D^c)$-time algorithm for some $\varepsilon, c > 0$. Then by combining this algorithm with our reduction, we can solve k-SAT on n variables and m clauses in time $O(2^{n/2}m + 2^{(2-\varepsilon)n/2}m^c) = O(2^{(1-\delta)n}m^{c'})$ for some $\delta > 0, c' \geq 1$. Since in k-CNF there are $O(n^k)$ different clauses, and we can bound the polynomial $O(n^k)$ by the exponential function $O(2^{\delta n/(2c')})$, we may estimate the factor $m^{c'}$ by $O(2^{\delta n/2})$. This yields a final time bound of $O(2^{(1-\delta/2)n})$ for k-SAT, which contradicts SETH. As contraposition, we obtain that SETH implies the Orthogonal Vectors conjecture. ◀

5 Example II: Regular Expression Pattern Matching

Let us recall the basics of regular expressions. A regular expression R is a search pattern that matches any string in the corresponding language $L(R)$. Here we only consider the following operations over a fixed alphabet Σ. For any symbol $c \in \Sigma$, the regular expression $R = c$ matches the length-1 string c, i.e., $L(R) = \{c\}$. For any regular expressions R_1, R_2, the regular expression $R = R_1|R_2$ matches any string matched by R_1 or R_2, i.e., $L(R) = L(R_1) \cup L(R_2)$. For any regular expressions R_1, R_2, the regular expression $R = R_1 \circ R_2$ matches any string that is a concatenation of a string matched by R_1 with a string matched by R_2, i.e., $L(R) = \{ab \mid a \in L(R_1), b \in L(R_2)\}$.

In the Regular Expression Pattern Matching problem, given a regular expression R and a text string T, the task is to decide whether any substring of T matches R. Denoting the length of T by n and the number of operations that define R by m, there is a classic $O(nm)$-time algorithm for this problem [36]. Here we show the following tight lower bound.

▶ **Theorem 3** ([14]). *Regular Expression Pattern Matching has no $O((n+m)^{2-\varepsilon})$-time algorithm for any $\varepsilon > 0$, unless the Orthogonal Vectors conjecture fails.*

Proof. For a fine-grained reduction from Orthogonal Vectors to Regular Expression Pattern Matching, consider an Orthogonal Vectors instance $A, B \subseteq \{0,1\}^D$ of size N. We construct a regular expression R and a text string T as follows.

On the coordinate level, we define two regular expressions

$$CR(1) := 0 \qquad \text{and} \qquad CR(0) := (0|1).$$

Note that for coordinates $x, y \in \{0,1\}$, regular expression $CR(x)$ matches string y if and only if $x \cdot y = 0$.

On the vector level, for any vector $b \in B$ we construct the regular expression

$$VR(b) := CR(b[1]) \circ CR(b[2]) \circ \ldots \circ CR(b[D]).$$

Note that for vectors $a, b \in \{0,1\}^D$, the regular expression $VR(b)$ matches the string $a[1]a[2]\ldots a[D]$ if and only if a and b are orthogonal.

On the level of sets of vectors, for the set $B = \{b_1, \ldots, b_N\}$ we construct the regular expression

$$R := VR(b_1)|VR(b_2)|\ldots|VR(b_N).$$

Note that for a vector $a \in \{0,1\}^D$, the regular expression R matches the string $a[1]a[2]\ldots a[D]$ if and only if a is orthogonal to some $b \in B$.

Finally, we define the text string T to be the concatenation of all vectors in A, padded by a dummy symbol '#', i.e., for $A = \{a_1, \ldots, a_N\}$ we set

$$T := a_1[1]\ldots a_1[D]\#a_2[1]\ldots a_2[D]\#\ldots\#a_N[1]\ldots a_N[D].$$

Since the dummy symbol does not appear in R, the regular expression R matches text T if and only if there is a vector $a \in A$ such that R matches $a[1]a[2]\ldots a[D]$. Hence, (R, T) is a YES-instance of Regular Expression Pattern Matching if and only if (A, B) is a YES-instance of Orthogonal Vectors.

Note that R and T have size $O(ND)$ and can be constructed in time $O(ND)$. Thus, any $O((n+m)^{2-\varepsilon})$-time algorithm for Regular Expression Pattern Matching for some $\varepsilon > 0$ would yield an algorithm for Orthogonal Vectors in time $O((ND)^{2-\varepsilon} + ND) = O(N^{2-\delta}D^c)$ for some $\delta, c > 0$, which contradicts the Orthogonal Vectors conjecture. Hence, Regular Expression Pattern Matching is not in time $O((n+m)^{2-\varepsilon})$ for any $\varepsilon > 0$ unless the Orthogonal Vectors conjecture fails. ◀

6 Conclusion

In this short introduction to fine-grained complexity theory we focused on the main conjectures and two basic examples. Over the last years, fine-grained complexity theory developed into a widely applicable tool with many interesting directions including extensions to dynamic graph algorithms [9], hardness of approximation [8], compressed data [1], external memory algorithms [26], and many more. Besides the four main conjectures presented here, several other conjectures have been used and relations among conjectures have been explored [4, 22, 24, 32].

For further reading, we refer to the surveys [37, 38]. Lecture material including slides can can be found at `https://www.mpi-inf.mpg.de/departments/algorithms-complexity/teaching/summer16/poly-complexity/` (or linked on the author's homepage).

References

1 Amir Abboud, Arturs Backurs, Karl Bringmann, and Marvin Künnemann. Fine-Grained Complexity of Analyzing Compressed Data: Quantifying Improvements over Decompress-and-Solve. In *FOCS*, pages 192–203, 2017.

2 Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. If the Current Clique Algorithms are Optimal, So is Valiant's Parser. In *FOCS*, pages 98–117, 2015.

3 Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. Tight Hardness Results for LCS and Other Sequence Similarity Measures. In *FOCS*, pages 59–78, 2015.

4 Amir Abboud, Karl Bringmann, Holger Dell, and Jesper Nederlof. More consequences of falsifying SETH and the orthogonal vectors conjecture. In *STOC*, pages 253–266. ACM, 2018.

5 Amir Abboud, Karl Bringmann, Danny Hermelin, and Dvir Shabtay. SETH-based lower bounds for Subset Sum and Bicriteria Path. In *SODA*, 2019. To appear. See `https://arxiv.org/abs/1704.04546`.

6 Amir Abboud and Søren Dahlgaard. Popular Conjectures as a Barrier for Dynamic Planar Graph Algorithms. In *FOCS*, pages 477–486, 2016.

7 Amir Abboud, Thomas Dueholm Hansen, Virginia Vassilevska Williams, and Ryan Williams. Simulating branching programs with edit distance and friends, or: a polylog shaved is a lower bound made. In *STOC*, pages 375–388, 2016.

8 Amir Abboud, Aviad Rubinstein, and R. Ryan Williams. Distributed PCP Theorems for Hardness of Approximation in P. In *FOCS*, pages 25–36, 2017.

9 Amir Abboud and Virginia Vassilevska Williams. Popular Conjectures Imply Strong Lower Bounds for Dynamic Problems. In *FOCS*, pages 434–443, 2014.

10 Amir Abboud, Virginia Vassilevska Williams, and Huacheng Yu. Matching Triangles and Basing Hardness on an Extremely Popular Conjecture. In *STOC*, pages 41–50, 2015.

11 Udit Agarwal and Vijaya Ramachandran. Fine-grained complexity for sparse graphs. In *STOC*, pages 239–252, 2018.

12 Divesh Aggarwal and Noah Stephens-Davidowitz. (Gap/S)ETH hardness of SVP. In *STOC*, pages 228–238, 2018.

13 Arturs Backurs and Piotr Indyk. Edit Distance Cannot Be Computed in Strongly Subquadratic Time (unless SETH is false). In *STOC*, pages 51–58, 2015.

14 Arturs Backurs and Piotr Indyk. Which Regular Expression Patterns Are Hard to Match? In *FOCS*, pages 457–466, 2016.

15 Arturs Backurs, Liam Roditty, Gilad Segal, Virginia Vassilevska Williams, and Nicole Wein. Towards tight approximation bounds for graph diameter and eccentricities. In *STOC*, pages 267–280, 2018.

16 R.E. Bellman. *Dynamic programming*. Princeton University Press, 1957.

17 Huck Bennett, Alexander Golovnev, and Noah Stephens-Davidowitz. On the Quantitative Hardness of CVP. In *FOCS*, pages 13–24, 2017.

18 Karl Bringmann. Why Walking the Dog Takes Time: Frechet Distance Has No Strongly Subquadratic Algorithms Unless SETH Fails. In *FOCS*, pages 661–670, 2014.

19 Karl Bringmann. A Near-Linear Pseudopolynomial Time Algorithm for Subset Sum. In *SODA*, pages 1073–1084. SIAM, 2017.

20 Karl Bringmann, Allan Grønlund, and Kasper Green Larsen. A Dichotomy for Regular Expression Membership Testing. In *FOCS*, pages 307–318, 2017.

21 Karl Bringmann and Marvin Künnemann. Quadratic Conditional Lower Bounds for String Problems and Dynamic Time Warping. In *FOCS*, pages 79–97, 2015.

22 Marco L. Carmosino, Jiawei Gao, Russell Impagliazzo, Ivan Mihajlin, Ramamohan Paturi, and Stefan Schneider. Nondeterministic Extensions of the Strong Exponential Time Hypothesis and Consequences for Non-reducibility. In *ITCS*, pages 261–270, 2016.

23 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

24 Marek Cygan, Marcin Mucha, Karol Wegrzycki, and Michał Włodarczyk. On Problems Equivalent to (min, +)-Convolution. In *ICALP*, volume 80 of *LIPIcs*, pages 22:1–22:15, 2017.

25 Holger Dell and John Lapinskas. Fine-grained reductions from approximate counting to decision. In *STOC*, pages 281–288, 2018.

26 Erik D. Demaine, Andrea Lincoln, Quanquan C. Liu, Jayson Lynch, and Virginia Vassilevska Williams. Fine-grained I/O Complexity via Reductions: New Lower Bounds, Faster Algorithms, and a Time Hierarchy. In *ITCS*, volume 94 of *LIPIcs*, pages 34:1–34:23, 2018.

27 Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999.

28 Robert W. Floyd. Algorithm 97: Shortest path. *Commun. ACM*, 5(6):345, 1962.

29 Anka Gajentaan and Mark H. Overmars. On a Class of $O(n^2)$ Problems in Computational Geometry. *Comput. Geom.*, 5:165–185, 1995.

30 Monika Henzinger, Sebastian Krinninger, Danupon Nanongkai, and Thatchaphol Saranurak. Unifying and Strengthening Hardness for Dynamic Problems via the Online Matrix-Vector Multiplication Conjecture. In *STOC*, pages 21–30, 2015.

31 Russell Impagliazzo and Ramamohan Paturi. On the Complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

32 Marvin Künnemann, Ramamohan Paturi, and Stefan Schneider. On the Fine-Grained Complexity of One-Dimensional Dynamic Programming. In *ICALP*, volume 80 of *LIPIcs*, pages 21:1–21:15, 2017.

33 Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Lower bounds based on the Exponential Time Hypothesis. *Bulletin of the EATCS*, 105:41–72, 2011.

34 Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In *STOC*, pages 515–524, 2013.

35 Barna Saha. Language Edit Distance and Maximum Likelihood Parsing of Stochastic Grammars: Faster Algorithms and Connection to Fundamental Graph Problems. In *FOCS*, pages 118–135, 2015.

36 Ken Thompson. Regular Expression Search Algorithm. *Commun. ACM*, 11(6):419–422, 1968.

37 Virginia Vassilevska Williams. Hardness of Easy Problems: Basing Hardness on Popular Conjectures such as the Strong Exponential Time Hypothesis (Invited Talk). In *IPEC*, volume 43 of *LIPIcs*, pages 17–29, 2015.

38 Virginia Vassilevska Williams. On some fine-grained questions in algorithms and complexity. In *ICM*, 2018. To appear. See `https://people.csail.mit.edu/virgi/eccentri.pdf`.

39 Virginia Vassilevska Williams and Ryan Williams. Subcubic Equivalences between Path, Matrix and Triangle Problems. In *FOCS*, pages 645–654, 2010.

40 Stephen Warshall. A Theorem on Boolean Matrices. *J. ACM*, 9(1):11–12, 1962.

41 Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005.

42 Ryan Williams. Faster all-pairs shortest paths via circuit complexity. In *STOC*, pages 664–673, 2014.