FINE-GRAINED COMPLEXITY COURSE
LECTURE 7: SETH HARDNESS

AMIR ABBOUD

1. This Lecture

So far the pattern of the lectures in this course has been: one lecture for introducing a conjecture and then one lecture for surveying its consequences. After having introduced our third main conjecture last week, the Strong Exponential Time Hypothesis (SETH), today we will talk about its consequences. But since the conditional lower bounds that are based on SETH are so numerous and so diverse, we will be discussing them for more than one week. In this lecture we will see the simplest (but still pretty interesting) reductions, and later on we will see more complicated ones.

2. OV

All SETH-based lower bounds for problems in P implicitly or explicitly start out with a reduction from k-SAT to a problem called Orthogonal Vectors (OV) or one of its variants (that we will also see today). Since the name of the problem (and its more standard definition) might sound scary for people (like me) who are more comfortable with combinatorics than algebra, I will also give an equivalent formulation called *Graph OV* that may be easier to think about.

In the following definitions one should think of d as being small compared to n, e.g. $\Theta(\log n)$.

Definition 1 (OV). *Given two sets $A, B \subseteq \{0,1\}^d$ of binary vectors, where $|A| = |B| = n$, is there a pair $a \in A, b \in B$ such that $\langle a, b\rangle = \sum_{j=1}^{d} a[j] \cdot b[j] = 0$?*

Note that since the vectors are binary, the orthogonality condition can be thought of as a *disjointness* condition (of the 1 coordinates). The following formulation is just a recasting of OV as a graph problem (and it is the more convenient starting point for reductions to graph problems).

Definition 2 (Graph OV). *Given a graph $G = (V, E)$ on three layers $V = A \cup D \cup B$, $E \subseteq A \times D \cup D \times B$, where $|A| = |B| = n$ and $|D| = d$, is there a pair of nodes $a \in A, b \in B$ at distance > 2?*

2.1. **Algorithms for OV.** The trivial algorithm solves OV in $O(n^2 d)$ time by trying all pairs. Another simple algorithm checks, for each $a \in A$ whether any of the $O(2^d)$ vectors that are orthogonal to a exist in the set B; the latter check can be implemented efficiently if we first hash the vectors in B into a dictionary. The time for this algorithm is $O(2^d \cdot n)$

and is subquadratic for $d < \log n$. The fastest algorithm known to date [2] has a time bound of $n^{2-\Theta(\log n/d)} \cdot d^{O(1)}$ which is subquadratic for $d = c \log n$ for any constant c. This algorithm is more complicated and it uses an algebraic encoding of the OV problem as a polynomial to reduce the problem into a matrix multiplication computation (over matrices of certain small dimensions). It is an example of a technique called *polynomial method* and its first introduction into fine-grained complexity was the algorithm that shaves a $2^{\sqrt{\log n}}$ factor for APSP [8].

2.2. **SAT to OV.** The following theorem shows that this last algorithm is the best possible, up to the constant in the Θ, assuming SETH. Let us remark that for most conditional lower bounds for polynomial time problems, the fact that d can be so small and still get an $\Omega(n^{2-\varepsilon})$ lower bound is not important; it is enough to know that OV requires quadratic time when $d = n^{o(1)}$, and for that we do not need to use the Sparsification Lemma at all in the proof.

Theorem 1 ([9]). *SETH implies that for all $\varepsilon > 0$ there is a constant $c_\varepsilon > 0$ such that OV on $d = c_\varepsilon \log n$ dimensions cannot be solved in $O(n^{2-\varepsilon})$ time.*

Proof. Given a k-CNF formula ϕ on N variables $x_1, \ldots, x_N$ and M clauses $C_1, \ldots, C_M$ we partition the variables into two sets $U_1 = \{x_1, \ldots, x_{N/2}\}$ and $U_2 = \{x_{N/2+1}, \ldots, x_N\}$ and define vectors as follows.

- For each $\alpha \in \{0,1\}^{N/2}$ that is an assignment to the variables in U_1 we create a vector $a_\alpha \in \{0,1\}^M$ and add it to the set A. This vectors is defined such that $\forall j \in [M] : a_\alpha[j] = 0$ if α (by itself) satisfies the clause C_j, and $a_\alpha[j] = 1$ otherwise. Note that the set A contains $n = 2^{N/2}$ vectors since this is the number of assignments to $N/2$ variables.
- In a similar way, for each $\beta \in \{0,1\}^{N/2}$ that is an assignment to the variables in U_2 we create a vector $b_\beta \in \{0,1\}^M$ and add it to the set B. This vector is defined such that $\forall j \in [M] : b_\beta[j] = 0$ if β (by itself) satisfies the clause C_j, and $a_\beta[j] = 1$ otherwise.

This completes the description of the reduction. The correctness follows from the following claim.

Claim 1. *For any assignment α to U_1 and any assignment β to U_2 we have that: $\langle a_\alpha, b_\beta \rangle = 0$ iff the full assignment to $U_1 \cup U_2$ composed of α and β satisfies ϕ.*

Proof. For the first direction observe that $\langle a_\alpha, b_\beta \rangle = 0$ implies that $\forall j \in [M] a_\alpha[j] \cdot b_\beta[j] = 0$ which implies that $\forall j \in [M]$ at least one of α or β satisfies the clause C_j which implies that the full assignment composed of α and β satisfies ϕ.

For the reverse direction, note that all the implications above are true in the other direction as well. □

Finally, consider the efficiency of the reduction. The number of vectors is $n = 2^{N/2}$. The dimension $d = M$. Without any sparsification we can upper bound $d = M = O(n^k) =$

$O((\log n)^k)$ which, as remarked above, suffices for most applications. To get the tighter bound, however, we can apply the Sparsification Lemma.

In more detail, let $\varepsilon > 0$ and our goal is to obtain a constant $c_\varepsilon > 0$ for which an $O(n^{2-\varepsilon})$ time algorithm for OV on $c_\varepsilon \log n$ dimensions refutes SETH. Let $\varepsilon' := \varepsilon/4$; by SETH there exists a constant $k = k_\varepsilon \geq 3$ such that k-SAT cannot be solved in $O(2^{(1-\varepsilon')N})$ time. Apply the Sparsification Lemma on ϕ with $\varepsilon' = \varepsilon/4$ and get a set of $t \leq 2^{\varepsilon' N}$ k-CNF formulas on N variables and $M \leq C(k,\varepsilon) \cdot N$ clauses each. Apply the reduction to each one of them, obtaining t instances of OV each with $n = 2^{N/2}$ vectors and dimension $d = M \leq C(k,\varepsilon) \cdot N = c_\varepsilon \log n$ for some fixed c_ε. Now, if we could indeed solve each instance in $O(n^{2-\varepsilon})$ time, we would solve SAT on ϕ in time:

$$t \cdot (2^{N/2})^{2-\varepsilon} = 2^{(\varepsilon/4)\cdot N} \cdot 2^{(1-\varepsilon/2)\cdot N} = 2^{(1-\varepsilon/4)\cdot N}$$

which is a contradiction to the above choice of ε' (and therefore to SETH). □

It may be helpful to try to execute the above reduction with Graph OV instead. A node $a_\alpha \in A$ gets connected to a node $j \in D$ iff α does not satisfy C_j, and similarly for $b_\beta \in B$. A two path from a_α to b_β implies that there is a clause that is not satisfied by the pair of partial assignments. A far pair indicates a pair α, β that make up a satisfying assignment.

3. Graph-OV to Diameter

In the Diameter problem we are given a graph and want to know the largest distance.

Theorem 2 ([7]). *Assuming SETH, for all $\varepsilon > 0$, no algorithm can given an unweighted undirected graph on n nodes and $O(n \log n)$ edges distinguish between diameter ≤ 2 or ≥ 3 in time $O(n^{2-\varepsilon})$.*

Proof. The reason that this theorem doesn't follow immediately from the lower bound we just saw for Graph OV is that a node $a \in A$ and a node $d \in D$ could have distance > 2 (affecting the diameter) even though all pairs $a \in A, b \in B$ are within distance 2.[1] To fix this, we simply add two auxiliary nodes x, y and connect: all nodes in $A \cup D$ to x, all nodes in $B \cup D$ to y, and finally connect x and y with an edge. It is now easy to verify that the distance between all pairs of nodes, except pairs in $A \times B$, is always 2. Thus, the diameter is 2 if the Graph OV instance is "no" and the diameter is 3 if it is a "yes". □

This simple reduction is surprisingly tight for the diameter problem even if we just want an approximation algorithm. Consider the case of sparse graphs on $m = \tilde{O}(n)$ edges. We can compute the diameter exactly in $O(n^2)$ time, and the above theorem shows that we cannot get a $1.5 - \varepsilon$ approximation in subqudratic time. Meanwhile, a folklore algorithm gets a 2-approximation in $\tilde{O}(n)$ time: pick any node and report the largest distance from that node. It turns out there's even a 1.5-approximation in subquadratic time; by a simple but clever algorithm that only looks at the distances from $\sqrt{n}$ nodes [4]. And the lower

[1] In other words, we can only conclude a lower bound for the ST-Diameter problem where we ask for the largest distance in $S \times T$ for a given pair of subsets $S, T \subseteq V$; note that in this case, we can even increase the gap between the yes-case and no-case from 2 vs. 3 to 2 vs. 4.

bound shows that this 1.5 is best-possible if we want subquadratic time. Very recently, it was also shown that 2 is the best-possible for near-linear time algorithm [5]. The precise time vs. approximation tradeoff is not fully understood yet.

4. Graph OV to Single-Source Max-Flow

Recall that the maximum flow from s to t is the largest amount of flow one can push while respecting the edge capacities, and such that the incoming flow equals the outgoing flow for each node except s and t. In the single-source version we want to know the maximum flow from s to v for all other nodes v.

Theorem 3 ([3])**.** *Assuming SETH, for all $\varepsilon > 0$, no algorithm can given a directed graph on n nodes and $O(n \log n)$ edges, and a given source s compute the maximum flow from s to all other nodes in time $O(n^{2-\varepsilon})$.*

Proof. Take the Graph OV instance and direct all edges from A to D to B, and also add a source node s and connect it to all nodes in A. Set the capacity on all the edges to 1 except for the edges between D and B whose capacity is set to n (or any large number).

The efficiency of the reduction is clear. The correctness is based on the claim that the maximum flow from s to a node $b \in B$ is n if and only if all nodes $a \in A$ have a two-path to b. Therefore, if there is an "orthogonal pair" a, b without a two-path then the flow from s to that b will be at most $n - 1$. □

Next we will see how to prove a higher lower bound of $\Omega(n^{3-\varepsilon})$ for the All-Pairs problem [6]. Notably, these lower bounds only apply for directed graphs; in undirected graphs the problem can be solved in $\tilde{O}(n^2)$ time (even if the graph is dense) [1, 10].

Please see the handwritten notes for the rest of the lecture.

References

[1] Amir Abboud, Robert Krauthgamer, Jason Li, Debmalya Panigrahi, Thatchaphol Saranurak, and Ohad Trabelsi. Gomory-hu tree in subcubic time. *CoRR*, abs/2111.04958, 2021.

[2] Amir Abboud, Richard Ryan Williams, and Huacheng Yu. More applications of the polynomial method to algorithm design. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 218–230. SIAM, 2015.

[3] Amir Abboud, Virginia Vassilevska Williams, and Huacheng Yu. Matching triangles and basing hardness on an extremely popular conjecture. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 41–50. ACM, 2015.

[4] Shiri Chechik, Daniel H. Larkin, Liam Roditty, Grant Schoenebeck, Robert Endre Tarjan, and Virginia Vassilevska Williams. Better approximation algorithms for the graph diameter. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1041–1052. SIAM, 2014.

[5] Mina Dalirrooyfard, Ray Li, and Virginia Vassilevska Williams. Hardness of approximate diameter: Now for undirected graphs. *CoRR*, abs/2106.06026, 2021.

[6] Robert Krauthgamer and Ohad Trabelsi. Conditional lower bounds for all-pairs max-flow. *ACM Trans. Algorithms*, 14(4):42:1–42:15, 2018.

[7] Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 515–524. ACM, 2013.

[8] R. Ryan Williams. Faster all-pairs shortest paths via circuit complexity. *SIAM J. Comput.*, 47(5):1965–1985, 2018.

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

[10] Tianyi Zhang. Gomory-hu trees in quadratic time. *arXiv preprint arXiv:2112.01042*, 2021.