The Origins of Fine-Grained Complexity

Massimo Equi

Foundation Friday

26 April 2024

What is Fine-Grained Complexity?

Lower Bounds

Unconditional lower bounds

- E.g. number of comparisons for sorting

Lower Bounds

Unconditional lower bounds

- E.g. number of comparisons for sorting
- May be hard to prove

Lower Bounds

Unconditional lower bounds

- E.g. number of comparisons for sorting
- May be hard to prove

Conditional lower bounds

- E.g. showing NP hardness

Lower Bounds

Unconditional lower bounds

- E.g. number of comparisons for sorting
- May be hard to prove

Conditional lower bounds

- E.g. showing NP hardness
- Conditioned on $P \neq NP$

Lower Bounds

Unconditional lower bounds

- E.g. number of comparisons for sorting
- May be hard to prove

Conditional lower bounds

- E.g. showing NP hardness
- Conditioned on $P \neq NP$
- Very "coarse" grain: only polynomial vs exponential complexities

Lower Bounds

Unconditional lower bounds

- E.g. number of comparisons for sorting
- May be hard to prove

Conditional lower bounds

- E.g. showing NP hardness
- Conditioned on $P \neq NP$
- Very “coarse” grain: only polynomial vs exponential complexities

Let’s say you can solve problem A in time $\boldsymbol{O(n^2)}$

Lower Bounds

Unconditional lower bounds

- E.g. number of comparisons for sorting
- May be hard to prove

Conditional lower bounds

- E.g. showing *NP* hardness
- Conditioned on $P \neq NP$
- Very "coarse" grain: only polynomial vs exponential complexities

Let's say you can solve problem A in time $\boldsymbol{O(n^2)}$
How to prove that A cannot be solved in $\boldsymbol{O(n^{2-\epsilon})}$?
What can we condition on?

ETH and SETH

Fine-grained complexity provides the hypotheses for that [Impagliazzo and Paturi, 2001]

Exponential Time Hypothesis (ETH)

There exists a constant $\bar{\alpha}$ s.t. no algorithm solves CNF-SAT with n variables in time $O(2^{\alpha n})$, $\alpha < \bar{\alpha}$

ETH and SETH

Fine-grained complexity provides the hypotheses for that [Impagliazzo and Paturi, 2001]

Exponential Time Hypothesis (ETH)

There exists a constant $\bar{\alpha}$ s.t. no algorithm solves CNF-SAT with n variables in time $O(2^{\alpha n})$, $\alpha < \bar{\alpha}$

Strong Exponential Time Hypothesis (SETH)

No algorithm solves CNF-SAT with n variables in time $O(2^{\alpha n})$, $\alpha < 1$

ETH and SETH

Fine-grained complexity provides the hypotheses for that [Impagliazzo and Paturi, 2001]

Exponential Time Hypothesis (ETH)

There exists a constant $\bar{\alpha}$ s.t. no algorithm solves CNF-SAT with n variables in time $O(2^{\alpha n})$, $\alpha < \bar{\alpha}$

Strong Exponential Time Hypothesis (SETH)

No algorithm solves CNF-SAT with n variables in time $O(2^{\alpha n})$, $\alpha < 1$

Idea: if we solve problem A in $\boldsymbol{O(n^{2-\epsilon})}$, then we solve CNF-SAT in $\boldsymbol{O(2^{(1-\epsilon')n})}$, contradicting SETH

Some notation

CNF-SAT: SAT in conjunctive normal form (CNF)

Some notation

CNF-SAT: SAT in conjunctive normal form (CNF)

input: a boolean formula F in CNF
n variables $v_1, \ldots, v_n$, k clauses $c_1, \ldots, c_q$

$$F(v_1, \ldots, v_n) = (v_1 \vee \bar{v_2}) \wedge (\bar{v_1} \vee v_2 \vee \bar{v_3}) \wedge (v_1 \vee v_3 \vee \bar{v_4}) \wedge \cdots$$

Some notation

CNF-SAT: SAT in conjunctive normal form (CNF)

input: a boolean formula F in CNF
n variables $v_1, \ldots, v_n$, k clauses $c_1, \ldots, c_q$

$$F(v_1, \ldots, v_n) = (v_1 \vee \bar{v_2}) \wedge (\bar{v_1} \vee v_2 \vee \bar{v_3}) \wedge (v_1 \vee v_3 \vee \bar{v_4}) \wedge \cdots$$

output: Yes/No - is there $a \in \{0,1\}^n$ s.t. $F(a) = 1$?

Some notation

CNF-SAT: SAT in conjunctive normal form (CNF)

input: a boolean formula F in CNF
n variables $v_1, \ldots, v_n$, k clauses $c_1, \ldots, c_q$

$$F(v_1, \ldots, v_n) = (v_1 \vee \bar{v_2}) \wedge (\bar{v_1} \vee v_2 \vee \bar{v_3}) \wedge (v_1 \vee v_3 \vee \bar{v_4}) \wedge \cdots$$

output: Yes/No - is there $a \in \{0,1\}^n$ s.t. $F(a) = 1$?

k-**SAT** : as CNF-SAT but with at most k literals per clause.

k-**SAT** : as CNF-SAT but with at most k literals per clause.

ETH and SETH in more detail

k-**SAT** : as CNF-SAT but with at most k literals per clause.

Let $s_k = inf \; \{\alpha \; : \; \text{There is an } O(2^{\alpha n})\text{-time algorithm for } k\text{-SAT}\}$

ETH and SETH in more detail

k-**SAT** : as CNF-SAT but with at most k literals per clause.

Let $s_k = inf \{\alpha \ : \ \text{There is an } O(2^{\alpha n})\text{-time algorithm for } k\text{-SAT}\}$

Exponential Time Hypothesis (ETH)

For all $k \geq 3$, $s_k > 0$

Strong Exponential Time Hypothesis (SETH)

$$s_\infty = \lim_{k \to \infty} s_k = 1$$

Arguments in favour of ETH

Theorem

The following statements are equivalent:

For every $k \geq 3$, $s_k > 0$ (ETH)

For some k, $s_k > 0$

$s_3 > 0$

SNP $\not\subseteq$ ***SUBEXP***

Satisfiability of linear-sized circuits cannot be solved in subexponential time

[Impagliazzo, Paturi and Zane, 1998]

Arguments in favour of ETH

Theorem

The following statements are equivalent:

For every $k \geq 3$, $s_k > 0$ (ETH)

For some k, $s_k > 0$

$s_3 > 0$

$\boldsymbol{SNP \not\subseteq SUBEXP}$

Satisfiability of linear-sized circuits cannot be solved in subexponential time

[Impagliazzo, Paturi and Zane, 1998]

SNP: the class of properties expressible by a series of second order existential quantifiers, followed by a series of first order universal quantifiers, followed by a basic formula

Note:

- $P \neq NP$ allows e.g. $O(2^{\sqrt{n}})$ for k-SAT, ETH **does not**
- ETH $\Rightarrow P \neq NP$

Arguments in favour of SETH

SETH : $$s_\infty = \lim_{k\to\infty} s_k = 1$$

Arguments in favour of SETH

SETH : $$s_\infty = \lim_{k \to \infty} s_k = 1$$

Main points in favour of it:

(a) Upper bound $O(2^{\alpha n}), \alpha = 1 - d/O(k)$, with d constant
$\Rightarrow\ s_\infty \leq 1$

Arguments in favour of SETH

SETH : $$s_\infty = \lim_{k\to\infty} s_k = 1$$

Main points in favour of it:

(a) Upper bound $O(2^{\alpha n}), \alpha = 1 - d/O(k)$, with d constant $\Rightarrow\ s_\infty \leq 1$

(b) Assuming ETH $\Rightarrow$ every $s_k < s_{k'}$, for $k < k'$, i.e. sequence $\{s_k\}$ is increasing infinitely often
[Impagliazzo and Paturi, 2001]

Arguments in favour of SETH

SETH : $$s_\infty = \lim_{k \to \infty} s_k = 1$$

Main points in favour of it:

(a) Upper bound $O(2^{\alpha n}), \alpha = 1 - d/O(k)$, with d constant
$\Rightarrow\ s_\infty \leq 1$

(b) Assuming ETH $\Rightarrow$ every $s_k < s_{k'}$, for $k < k'$, i.e. sequence $\{s_k\}$ is increasing infinitely often
[Impagliazzo and Paturi, 2001]

SETH states $s_\infty = 1$, but it might be $s_\infty < 1$

Let's look at (b) more closely

Arguments in favour of SETH

Theorem

For every k, $s_k \leq (1 - d/k)s_\infty$

Arguments in favour of SETH

Theorem

For every $k, \quad s_k \leq (1 - d/k)s_\infty$

ETH: for $\boldsymbol{k \geq 3}$, $\boldsymbol{s_k > 0}$

Arguments in favour of SETH

Theorem

For every k, $\quad s_k \leq (1 - d/k)s_\infty$

ETH: for $\boldsymbol{k \geq 3}$, $\boldsymbol{s_k > 0}$

Consider $\boldsymbol{k}$ and $\boldsymbol{k'}$ such that $\boldsymbol{k < k'}$

- Then $\boldsymbol{(1 - d/k)s_\infty < (1 - d/k')s_\infty}$

Arguments in favour of SETH

Theorem

For every k, $\quad s_k \leq (1 - d/k)s_\infty$

ETH: for $\boldsymbol{k \geq 3}$, $\boldsymbol{s_k > 0}$

Consider $\boldsymbol{k}$ and $\boldsymbol{k'}$ such that $\boldsymbol{k < k'}$

- Then $\boldsymbol{(1 - d/k)s_\infty < (1 - d/k')s_\infty}$
- Moreover $\exists \boldsymbol{k'}$ s.t. $\boldsymbol{s_k \leq (1 - d/k)s_{k'}}$

Arguments in favour of SETH

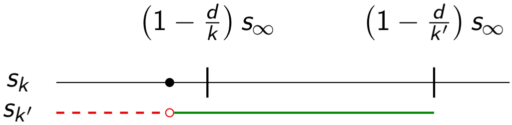
Theorem

For every k, $s_k \le (1 - d/k)s_\infty$

ETH: for $\boldsymbol{k \ge 3}$, $\boldsymbol{s_k > 0}$

Consider $\boldsymbol{k}$ and $\boldsymbol{k'}$ such that $\boldsymbol{k < k'}$

- Then $\boldsymbol{(1 - d/k)s_\infty < (1 - d/k')s_\infty}$
- Moreover $\exists \boldsymbol{k'}$ s.t. $\boldsymbol{s_k \le (1 - d/k)s_{k'}}$



Arguments in favour of SETH

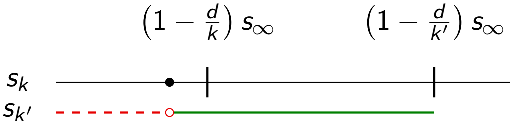
Theorem

For every k, $s_k \leq (1 - d/k)s_\infty$

ETH: for $\boldsymbol{k \geq 3}$, $\boldsymbol{s_k > 0}$

Consider $\boldsymbol{k}$ and $\boldsymbol{k'}$ such that $\boldsymbol{k < k'}$

- Then $\boldsymbol{(1 - d/k)s_\infty < (1 - d/k')s_\infty}$
- Moreover $\exists \boldsymbol{k'}$ s.t. $\boldsymbol{s_k \leq (1 - d/k)s_{k'}}$



Then $\boldsymbol{s_k < s_{k'}}$, i.e. sequence $\boldsymbol{\{s_k\}}$ is strictly increasing

SETH

Summing up, we know that

- $s_\infty \leq 1$
- $s_k \leq (1 - d/k)s_\infty$, for every k
- ETH $\Rightarrow s_k < s_{k'}$, for some $k < k'$

SETH

Summing up, we know that

- $s_\infty \leq 1$
- $s_k \leq (1 - d/k)s_\infty$, for every k
- ETH $\Rightarrow s_k < s_{k'}$, for some $k < k'$

Hence the claim

$$\text{SETH}: \quad s_\infty = 1$$

Conditional Lower Bounds

Orthogonal Vectors

Why is SETH useful?

It can be used to prove **conditional lower bounds** for polynomial problems.

Orthogonal Vectors

Why is SETH useful?

It can be used to prove **conditional lower bounds** for polynomial problems.

SETH can be used to prove a conditional lower bound for the **Orthogonal Vectors** (OV) problem

Orthogonal Vectors

Why is SETH useful?

It can be used to prove **conditional lower bounds** for polynomial problems.

SETH can be used to prove a conditional lower bound for the **Orthogonal Vectors** (OV) problem

Doing so allows us to use OV instead of CNF-SAT for proving polynomial conditional lower bounds, making the reductions easier

Quick Complexity Background

Orthogonal Vectors (OV) Problem

Let $X, Y \subseteq \{0,1\}^d$ be two sets of n binary vectors of length d.

Determine whether there exist $x \in X, y \in Y$ such that $x \cdot y = 0$

Quick Complexity Background

Orthogonal Vectors (OV) Problem

Let $X, Y \subseteq \{0,1\}^d$ be two sets of n binary vectors of length d.

Determine whether there exist $x \in X, y \in Y$ such that $x \cdot y = 0$

$$X = \begin{pmatrix} 001 \\ \mathbf{010} \\ 011 \\ 101 \end{pmatrix} \qquad \begin{pmatrix} 010 \\ 011 \\ \mathbf{100} \\ 111 \end{pmatrix} = Y$$

Quick Complexity Background

Orthogonal Vectors (OV) Problem

Let $X, Y \subseteq \{0,1\}^d$ be two sets of n binary vectors of length d.

Determine whether there exist $x \in X, y \in Y$ such that $x \cdot y = 0$

$$X = \begin{pmatrix} 001 \\ \mathbf{010} \\ 011 \\ 101 \end{pmatrix} \qquad \begin{pmatrix} 010 \\ 011 \\ \mathbf{100} \\ 111 \end{pmatrix} = Y$$

Orthogonal Vectors Hypothesis (OVH)

No algorithm can solve *Orthogonal Vectors* in time $O(n^{\alpha}\,\mathrm{poly}(d))$, $\alpha < 2$

Quick Complexity Background

Orthogonal Vectors (OV) Problem

Let $X, Y \subseteq \{0,1\}^d$ be two sets of n binary vectors of length d.

Determine whether there exist $x \in X, y \in Y$ such that $x \cdot y = 0$

$$X = \begin{pmatrix} 001 \\ \mathbf{010} \\ 011 \\ 101 \end{pmatrix} \qquad \begin{pmatrix} 010 \\ 011 \\ \mathbf{100} \\ 111 \end{pmatrix} = Y$$

Orthogonal Vectors Hypothesis (OVH)

No algorithm can solve *Orthogonal Vectors* in time $O(n^\alpha \operatorname{poly}(d))$, $\alpha < 2$

CNF-SAT can be reduced to OV, thus SETH $\Rightarrow$ OVH

Orthogonal Vectors

Reduction from CNF-SAT to OV

- start from a CNF-SAT formula F with n variables and q clauses

Orthogonal Vectors

Reduction from CNF-SAT to OV

- start from a CNF-SAT formula F with n variables and q clauses
- separate the variables in two groups $v_1, \ldots, v_{\frac{n}{2}}$ and $v_{\frac{n}{2}+1}, \ldots, v_n$

Orthogonal Vectors

Reduction from CNF-SAT to OV

- start from a CNF-SAT formula F with n variables and q clauses
- separate the variables in two groups $v_1, \ldots, v_{\frac{n}{2}}$ and $v_{\frac{n}{2}+1}, \ldots, v_n$

$$2^{\frac{n}{2}} \left\{ \begin{array}{lll} x_1 = & (0\ \ 0) & (0\ \ 0\ \ 1\ \ 1\ \ 1) & = u_1 \\ x_2 = & (0\ \ 1) & (0\ \ 1\ \ 1\ \ 0\ \ 0) & = u_2 \\ x_3 = & (1\ \ 0) & (1\ \ 0\ \ 0\ \ 0\ \ 1) & = u_3 \\ x_4 = & (1\ \ 1) & (1\ \ 0\ \ 1\ \ 0\ \ 0) & = u_4 \end{array} \right\} m \qquad \begin{array}{c} u_i[h] = 0 \\ \Leftrightarrow \\ x_i \models c_h \end{array}$$

$$2^{\frac{n}{2}} \left\{ \begin{array}{llll} y_1 = & (0\ \ 0) & (0\ \ 0\ \ 1\ \ 1\ \ 0) & = w_1 \\ y_2 = & (0\ \ 1) & (1\ \ 1\ \ 0\ \ 0\ \ 1) & = w_2 \\ y_3 = & (1\ \ 0) & (0\ \ 1\ \ 0\ \ 1\ \ 1) & = w_3 \\ y_4 = & \underbrace{(1\ \ 1)}_{\frac{n}{2}} & \underbrace{(0\ \ 1\ \ 0\ \ 1\ \ 0)}_{k} & = w_4 \end{array} \right\} m \qquad \begin{array}{c} w_j[h] = 0 \\ \Leftrightarrow \\ y_j \models c_h \end{array}$$

Orthogonal Vectors

Contradiction with SETH

- this reductions takes $O(2^{\frac{n}{2}}\text{poly}(q))$

Orthogonal Vectors

Contradiction with SETH

- this reductions takes $O(2^{\frac{n}{2}}\text{poly}(q))$
- $m = 2^{\frac{n}{2}}$ vectors

Orthogonal Vectors

Contradiction with SETH

- this reductions takes $O(2^{\frac{n}{2}}\text{poly}(q))$
- $m = 2^{\frac{n}{2}}$ vectors
- each vector is of size $d = q$

Orthogonal Vectors

Contradiction with SETH

- this reductions takes $O(2^{\frac{n}{2}}\text{poly}(q))$
- $m = 2^{\frac{n}{2}}$ vectors
- each vector is of size $d = q$

Solving OV in time $O(m^{2-\epsilon}\text{poly}(d))$ implies the following for CNF-SAT:

$$O(2^{\frac{n}{2}(2-\epsilon)}\text{poly}(q)) = O(2^{n(1-\frac{\epsilon}{2})}\text{poly}(q))$$

Orthogonal Vectors

Contradiction with SETH

- this reductions takes $O(2^{\frac{n}{2}}\text{poly}(q))$
- $m = 2^{\frac{n}{2}}$ vectors
- each vector is of size $d = q$

Solving OV in time $O(m^{2-\epsilon}\text{poly}(d))$ implies the following for CNF-SAT:

$$O(2^{\frac{n}{2}(2-\epsilon)}\text{poly}(q)) = O(2^{n(1-\frac{\epsilon}{2})}\text{poly}(q))$$

This contradicts SETH

Orthogonal Vectors

Contradiction with SETH

- this reductions takes $O(2^{\frac{n}{2}}\text{poly}(q))$
- $m = 2^{\frac{n}{2}}$ vectors
- each vector is of size $d = q$

Solving OV in time $O(m^{2-\epsilon}\text{poly}(d))$ implies the following for CNF-SAT:

$$O(2^{\frac{n}{2}(2-\epsilon)}\text{poly}(q)) = O(2^{n(1-\frac{\epsilon}{2})}\text{poly}(q))$$

This contradicts SETH

We conclude that OVH is true unless SETH is false (SETH $\Rightarrow$ OVH)

Problems with lower bounds

Problems with a reduction from **OV** (or k-OV) that implies a lower bound, unless SETH is false [Williams, 2018]:

- Graph Diameter for unweighted, undirected graphs with n nodes and $O(n)$ edges
- Edit Distance, Longest Common Subsequence, Dynamic Time Warping Distance, Fréchet Distance
- Subtree Isomorphism
- All Pairs Max Flow
- Subset Sum on n integers and target T

Final remarks

No known connections with **APSP** or **3-SUM**

- Often used as different source problems for fine-grained reductions

Final remarks

No known connections with **APSP** or **3-SUM**

- Often used as different source problems for fine-grained reductions

Reductions from **Formula-SAT** instead of CNF-SAT
[Abboud, Hansen, Williams and Williams, 2016]

- For a problem with upper bound $O(n^d / \log^c n)$, you can show that $c < \bar{c}$, for some constant $\bar{c}$.

Thank you!

References I

[Impagliazzo, Paturi and Zane, 1998]
Russell Impagliazzo, Ramamohan Paturi, Fancis Zane
Which Problems Have Strongly Exponential Complexity?
FOCS 1998

[Impagliazzo and Paturi, 2001]
Russell Impagliazzo, Ramamohan Paturi
On the Complexity of k-SAT
Journal of Computer and System Sciences, 2001

References II

[Abboud, Hansen, Williams and Williams, 2016]
Amir Abboud, Thomas Dueholm Hansen, Virginia Vassilevska Williams, Ryan Williams
Simulating branching programs with edit distance and friends: or: a polylog shaved is a lower bound made
STOC 2016

[Williams, 2018]
Virginia Vassilevska Williams
On some fine-grained questions in algorithms and complexity
ICM 2018