Do NP-Hard Problems Require Exponential Time?

Andrew Drucker

IAS

April 8, 2014

NP-completeness

- NP-completeness theory gives **great guidance** about which problems are efficiently solvable.

NP-completeness

- NP-completeness theory gives **great guidance** about which problems are efficiently solvable.
- 2-SAT, 2-Coloring, Euler Tour $\in$ PTIME;

NP-completeness

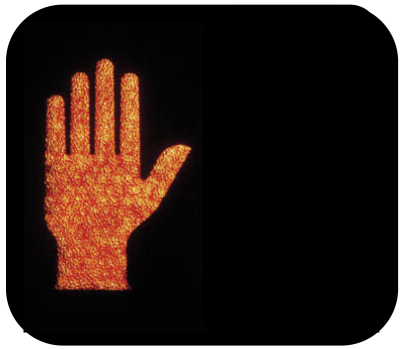
- NP-completeness theory gives **great guidance** about which problems are efficiently solvable.
- 2-SAT, 2-Coloring, Euler Tour $\in$ PTIME;

- 3-SAT, 3-Coloring, Hamilton Path

NP-completeness

- NP-completeness theory gives **great guidance** about which problems are efficiently solvable.
- 2-SAT, 2-Coloring, Euler Tour $\in$ PTIME;

- 3-SAT, 3-Coloring, Hamilton Path NP-complete.



NP-completeness

- Want to solve NP-complete problems $\Longrightarrow$ must accept **compromise!**
- Popular approach: find **approximately-optimal** solutions.

  (for optimization probs.)
- Here too, NP-completeness theory (+ PCPs) often provides great guidance!
  - .5-approx Max-LIN ($\mathbb{F}_2$) $\in$ PTIME;
  - $(.5+\varepsilon)$-approx Max-LIN ($\mathbb{F}_2$): NP-Complete. [Håstad'97]

NP-completeness

- Want to solve NP-complete problems $\Longrightarrow$ must accept **compromise!**
- Popular approach: find **approximately-optimal** solutions.

  (for optimization probs.)
- Here too, NP-completeness theory (+ PCPs) often provides great guidance!
  - .5-approx Max-LIN ($\mathbb{F}_2$) $\in$ PTIME;
  - $(.5+\varepsilon)$-approx Max-LIN ($\mathbb{F}_2$): NP-Complete. [Håstad'97]

NP-completeness

- Want to solve NP-complete problems $\Longrightarrow$ must accept **compromise!**
- Popular approach: find **approximately-optimal** solutions.

  (for optimization probs.)
- Here too, NP-completeness theory (+ PCPs) often provides great guidance!
  - .5-approx Max-LIN ($\mathbb{F}_2$) $\in$ PTIME;
  - $(.5+\varepsilon)$-approx Max-LIN ($\mathbb{F}_2$): NP-Complete. [Håstad'97]

NP-completeness

- Sometimes we need an **exact** solution to an NP-C problem.
- Then, compromise $\Longrightarrow$ must accept an **inefficient** algorithm!

  (at least, inefficient in the worst case—our focus)
- **Key question:** Can we at least beat brute-force search??

NP-completeness

- Sometimes we need an **exact** solution to an NP-C problem.
- Then, compromise $\Longrightarrow$ must accept an **inefficient** algorithm!

  (at least, inefficient in the worst case—our focus)
- **Key question:** Can we at least beat brute-force search??

NP-completeness

- Sometimes we need an **exact** solution to an NP-C problem.
- Then, compromise $\Longrightarrow$ must accept an **inefficient** algorithm!

  (at least, inefficient in the worst case—our focus)
- **Key question:** Can we at least beat brute-force search??

Time complexity

Known results for some popular NP-C problems:

Problem	**Parameter**	**Trivial**	**Improved**	**Ref.**
CNF-SAT	$n = \#\text{vars}$	2^n	1.99^n ??	
k-SAT			$2^{(1-1/k)n}$	[Paturi, Pudlák, Zane '97]
IND. SET	$n = \#\text{vertices}$	2^n	1.23^n	[Tarjan, Trojanowski'77; more...]
PLANAR IND. SET			$2^{O(\sqrt{n})}$	[Ungar '51; Lipton, Tarjan '79]
HAM. PATH		$n!$	2^n , 1.7^n	[Held, Karp '62; Bjorklund '10]

(Strictly: $F(n)$'s above should be $O^*(F(n)) \triangleq F(n) \cdot |\mathit{instance}|^{O(1)}$.)

Example: Schöning's alg.

- **Given:** a k-CNF $\mathcal{F} = C_1 \wedge C_2 \wedge \ldots \wedge C_m$.
  (each C_i an OR of $\leq k$ literals)
- **Goal:** find a satisfying solution to $\mathcal{F}$ if one exists.

Algorithm $A(\mathcal{F})$:

1. *Let $x \longleftarrow$ (random assignment).*
2. *Choose any unsat. clause C_i; flip a rand. variable in C_i.*
3. **Repeat** *Step 2 for $3n$ steps.*

Example: Schöning's alg.

- **Given:** a k-CNF $\mathcal{F} = C_1 \wedge C_2 \wedge \ldots \wedge C_m$.
  (each C_i an OR of $\leq k$ literals)
- **Goal:** find a satisfying solution to $\mathcal{F}$ if one exists.

Algorithm $A(\mathcal{F})$:

1. *Let* $x \longleftarrow$ *(random assignment).*
2. *Choose any unsat. clause* C_i*; flip a rand. variable in* C_i*.*
3. **Repeat** *Step 2 for* $3n$ *steps.*

Example: Schöning's alg.

Algorithm $A(\mathcal{F})$:

1. *Let* $x \longleftarrow$ *(random assignment).*
2. *Choose any unsat. clause* C_i*;* *flip a rand. variable in* C_i*.*
3. **Repeat** *Step 2 for* $3n$ *steps.*

- **Claim:** if $\mathcal{F} \in$ SAT, then

$$\Pr[A(\mathcal{F}) \text{finds a solution}] \geq 2^{-(1-c/k)n}.$$

$$\Longrightarrow \text{repeat for } 2^{(1-c/k)n} \text{ trials to find one w.h.p.!}$$

- Analysis idea is very simple!

Example: Schöning's alg.

Algorithm $A(\mathcal{F})$:

1. *Let* $x \longleftarrow$ *(random assignment).*
2. *Choose any unsat. clause* C_i*; flip a rand. variable in* C_i*.*
3. **Repeat** *Step 2 for* $3n$ *steps.*

- **Claim:** if $\mathcal{F} \in$ SAT, then

$$\Pr[A(\mathcal{F}) \text{finds a solution}] \geq 2^{-(1-c/k)n}.$$

$$\Longrightarrow \text{repeat for } 2^{(1-c/k)n} \text{ trials to find one w.h.p.!}$$

- Analysis idea is very simple!

Example: Schöning's alg.

Algorithm $A(\mathcal{F})$:

1. *Let* $x \longleftarrow$ *(random assignment).*
2. *Choose any unsat. clause* C_i*; flip a rand. variable in* C_i*.*
3. **Repeat** *Step 2 for* $3n$ *steps.*

- **Claim:** if $\mathcal{F} \in \mathrm{SAT}$, then

$$\Pr[A(\mathcal{F}) \text{finds a solution}] \geq 2^{-(1-c/k)n}.$$

$$\Longrightarrow \text{repeat for } 2^{(1-c/k)n} \text{ trials to find one w.h.p.!}$$

- Analysis idea is very simple!

Example: Schöning's alg.

Algorithm $A(\mathcal{F})$:

1. *Let* $x \longleftarrow$ *(random assignment).*
2. *Choose any unsat. clause* C_i*; flip a rand. variable in* C_i*.*
3. **Repeat** *Step 2 for* $3n$ *steps.*

- **Claim:** if $\mathcal{F} \in \mathrm{SAT}$, then

$$\Pr[A(\mathcal{F}) \text{finds a solution}] \geq 2^{-(1-c/k)n}.$$

$$\Longrightarrow \text{repeat for } 2^{(1-c/k)n} \text{ trials to find one w.h.p.!}$$

- Analysis idea is very simple!

Example: Schöning's alg.

Algorithm $A(\mathcal{F})$:

1. *Let* $x \longleftarrow$ *(random assignment).*
2. *Choose any unsat. clause* C_i*;* *flip a rand. variable in* C_i*.*
3. **Repeat** *Step 2 for* $3n$ *steps.*

- Suppose $\mathcal{F}(\mathbf{x}^*) = 1$. Let $x^t =$ state of x after t execs. of Step 2. Let

$$Y_t \quad \triangleq \quad ||x^t - \mathbf{x}^*||_1 \ .$$

- **Key fact:** if $Y_t > 0$, then $\Pr[Y_{t+1} = Y_t - 1] \ \geq \ 1/k$.
- Can lower-bound $\Pr[\min_t Y_t = 0]$ in terms of a biased random walk.

(biased against us, but not too badly!
Hope is that $Y_0 \ll n/2$.)

Example: Schöning's alg.

Algorithm $A(\mathcal{F})$:

1. *Let* $x \longleftarrow$ *(random assignment).*
2. *Choose any unsat. clause* C_i*; flip a rand. variable in* C_i*.*
3. **Repeat** *Step 2 for* $3n$ *steps.*

- Suppose $\mathcal{F}(\mathbf{x}^*) = 1$. Let $x^t =$ state of x after t execs. of Step 2. Let

$$Y_t \quad \triangleq \quad ||x^t - \mathbf{x}^*||_1 \ .$$

- **Key fact:** if $Y_t > 0$, then $\Pr[Y_{t+1} = Y_t - 1] \ \geq \ 1/k$.
- Can lower-bound $\Pr[\min_t Y_t = 0]$ in terms of a biased random walk.

(biased against us, but not too badly!
Hope is that $Y_0 \ll n/2$.)

Example: Schöning's alg.

Algorithm $A(\mathcal{F})$:

1. *Let* $x \longleftarrow$ *(random assignment).*
2. *Choose any unsat. clause* C_i*; flip a rand. variable in* C_i*.*
3. **Repeat** *Step 2 for* $3n$ *steps.*

- Suppose $\mathcal{F}(\mathbf{x}^*) = 1$. Let $x^t =$ state of x after t execs. of Step 2. Let

$$Y_t \quad \triangleq \quad ||x^t - \mathbf{x}^*||_1 \ .$$

- **Key fact:** if $Y_t > 0$, then $\Pr[Y_{t+1} = Y_t - 1] \ \geq \ 1/k$.
- Can lower-bound $\Pr[\min_t Y_t = 0]$ in terms of a biased random walk.

(biased against us, but not too badly!
Hope is that $Y_0 \ll n/2$.)

Example: Schöning's alg.

Algorithm $A(\mathcal{F})$:

1. *Let* $x \longleftarrow$ *(random assignment).*
2. *Choose any unsat. clause* C_i*; flip a rand. variable in* C_i*.*
3. **Repeat** *Step 2 for* $3n$ *steps.*

- Suppose $\mathcal{F}(\mathbf{x}^*) = 1$. Let $x^t =$ state of x after t execs. of Step 2. Let

$$Y_t \quad \triangleq \quad ||x^t - \mathbf{x}^*||_1 \ .$$

- **Key fact:** if $Y_t > 0$, then $\Pr[Y_{t+1} = Y_t - 1] \ \geq \ 1/k$.
- Can lower-bound $\Pr[\min_t Y_t = 0]$ in terms of a biased random walk.

(biased against us, but not too badly!
Hope is that $Y_0 \ll n/2$.)

Example: Schöning's alg.

Algorithm $A(\mathcal{F})$:

1. *Let* $x \longleftarrow$ *(random assignment).*
2. *Choose any unsat. clause* C_i*; flip a rand. variable in* C_i*.*
3. **Repeat** *Step 2 for* $3n$ *steps.*

- Suppose $\mathcal{F}(\mathbf{x}^*) = 1$. Let $x^t =$ state of x after t execs. of Step 2. Let

$$Y_t \quad \triangleq \quad ||x^t - \mathbf{x}^*||_1 \ .$$

- **Key fact:** if $Y_t > 0$, then $\Pr[Y_{t+1} = Y_t - 1] \ \geq \ 1/k$.
- Can lower-bound $\Pr[\min_t Y_t = 0]$ in terms of a biased random walk.

(biased against us, but not too badly!
Hope is that $Y_0 \ll n/2$.)

Time complexity

Known results for some popular NP-C problems:

Problem	Parameter	Trivial	Improved	Ref.
CNF-SAT	$n = \#\text{vars}$	2^n	1.99^n ??	
k-SAT			$2^{(1-1/k)n}$	[Paturi, Pudlák, Zane '97]
IND. SET	$n = \#\text{vertices}$	2^n	1.23^n	[Tarjan, Trojanowski'77; more...]
PLANAR IND. SET			$2^{O(\sqrt{n})}$	[Ungar '51; Lipton, Tarjan '79]
HAM. PATH		$n!$	2^n , 1.7^n	[Held, Karp '62; Bjorklund '10]

NP-C theory: no prediction about relative difficulty, best runtimes for these probs!

Time complexity

Known results for some popular NP-C problems:

Problem	Parameter	Trivial	Improved	Ref.
CNF-SAT	$n = \#\text{vars}$	2^n	1.99^n ??	
k-SAT			$2^{(1-1/k)n}$	[Paturi, Pudlák, Zane '97]
IND. SET	$n = \#\text{vertices}$	2^n	1.23^n	[Tarjan, Trojanowski'77; more...]
PLANAR IND. SET			$2^{O(\sqrt{n})}$	[Ungar '51; Lipton, Tarjan '79]
HAM. PATH		$n!$	2^n , 1.7^n	[Held, Karp '62; Bjorklund '10]

NP-C theory: no prediction about relative difficulty, best runtimes for these probs!

Time complexity

Known results for some popular NP-C problems:

Problem	**Parameter**	**Trivial**	**Improved**	**Ref.**
CNF-SAT	$n = \#\text{vars}$	2^n	1.99^n ??	
k-SAT			$2^{(1-1/k)n}$	[Paturi, Pudlák, Zane '97]
IND. SET	$n = \#\text{vertices}$	2^n	1.23^n	[Tarjan, Trojanowski'77; more...]
PLANAR IND. SET			$2^{O(\sqrt{n})}$	[Ungar '51; Lipton, Tarjan '79]
HAM. PATH		$n!$	2^n , 1.7^n	[Held, Karp '62; Bjorklund '10]

Challenge *for complexity theory:* **explain** *the seeming differences in difficulty!*
Identify barriers *to further progress!*
Guide *search for faster algorithms!*

Time complexity

- Could P $\neq$ NP conjecture imply that NP-C probs require exponential time?
- No idea. Seems hopeless!
- Influential approach: strengthen the conjecture!

Exponential Time Hypothesis—informal (Impagliazzo, Paturi, Zane '98)

No $2^{o(n)}$-time algorithm for n-variable 3-SAT.

Time complexity

- Could P $\neq$ NP conjecture imply that NP-C probs require exponential time?
- No idea. Seems hopeless!
- Influential approach: strengthen the conjecture!

Exponential Time Hypothesis—informal (Impagliazzo, Paturi, Zane '98)

No $2^{o(n)}$-time algorithm for n-variable 3-SAT.

The ETH

Exponential Time Hypothesis—informal (Impagliazzo, Paturi, Zane '98)

No $2^{o(n)}$-time algorithm for n-variable 3-SAT.

- Formally: for $k \geq 3$, define $s_k \in [0,1]$ by

$$s_k \triangleq \inf\{\varepsilon : k\text{-SAT decidable in time } O^*(2^{\varepsilon n})\} .$$

(allowing randomized algs with 1/3 error prob.)

Exponential Time Hypothesis—formal (IPZ)

$$s_3 > 0 .$$

The ETH

Exponential Time Hypothesis—informal (Impagliazzo, Paturi, Zane '98)

No $2^{o(n)}$-time algorithm for n-variable 3-SAT.

- Formally: for $k \geq 3$, define $s_k \in [0,1]$ by

$$s_k \triangleq \inf\{\varepsilon : k\text{-SAT decidable in time } O^*(2^{\varepsilon n})\} .$$

(allowing randomized algs with 1/3 error prob.)

Exponential Time Hypothesis—formal (IPZ)

$$s_3 > 0 .$$

The ETH

Exponential Time Hypothesis—informal (Impagliazzo, Paturi, Zane '98)

No $2^{o(n)}$-time algorithm for n-variable 3-SAT.

- Formally: for $k \geq 3$, define $s_k \in [0,1]$ by

$$s_k \triangleq \inf\{\varepsilon : k\text{-SAT decidable in time } O^*(2^{\varepsilon n})\} .$$

(allowing randomized algs with 1/3 error prob.)

Exponential Time Hypothesis—formal (IPZ)

$$s_3 > 0 .$$

The ETH

- Formally: for $k \geq 3$, define $s_k \in [0,1]$ by

$$s_k \triangleq \inf\{\varepsilon : k\text{-SAT decidable in time } O^*(2^{\varepsilon n})\} .$$

(allowing randomized algs with 1/3 error prob.)

Exponential Time Hypothesis—formal (IPZ)

$$s_3 > 0 .$$

- $s_3 \leq s_4 \leq s_5 \leq \ldots$
- Best known: $s_3 \leq .388$, $s_4 \leq .555$, $s_k \leq 1 - \Theta(1/k)$.

[Paturi, Pudlák, Saks, Zane '98; Hertli '11]

The ETH

- Formally: for $k \geq 3$, define $s_k \in [0,1]$ by

$$s_k \triangleq \inf\{\varepsilon : k\text{-SAT decidable in time } O^*(2^{\varepsilon n})\} .$$

(allowing randomized algs with 1/3 error prob.)

Exponential Time Hypothesis—formal (IPZ)

$$s_3 > 0 .$$

- $s_3 \leq s_4 \leq s_5 \leq \ldots$
- Best known: $s_3 \leq .388$, $s_4 \leq .555$, $s_k \leq 1 - \Theta(1/k)$.

[Paturi, Pudlák, Saks, Zane '98; Hertli '11]

The ETH

- Formally: for $k \geq 3$, define $s_k \in [0,1]$ by

$$s_k \triangleq \inf\{\varepsilon : k\text{-SAT decidable in time } O^*(2^{\varepsilon n})\} .$$

(allowing randomized algs with 1/3 error prob.)

Exponential Time Hypothesis—formal (IPZ)

$$s_3 > 0 .$$

- $s_3 \leq s_4 \leq s_5 \leq \ldots$
- Best known: $s_3 \leq .388$, $s_4 \leq .555$, $s_k \leq 1 - \Theta(1/k)$.

[Paturi, Pudlák, Saks, Zane '98; Hertli '11]

ETH

- Much stronger belief than $\mathsf{P} \neq \mathsf{NP}$.

- Payoff in explanatory power?

YES! *But, story is more complex than NP-completeness.*

- Issue: ETH studies dependence on key param.

$$n = \#\,\text{vars}(\mathcal{F}) \ll |\mathcal{F}|.$$

- Measures dimension of **search space**, not input size!

c.f. [Hunt, Stearns'90], "power index"

ETH

- Much stronger belief than $\mathsf{P} \neq \mathsf{NP}$.

- Payoff in explanatory power?

YES! *But, story is more complex than NP-completeness.*

- Issue: ETH studies dependence on key param.

$$n = \#\,\mathrm{vars}(\mathcal{F}) \ll |\mathcal{F}|.$$

- Measures dimension of **search space**, not input size!

c.f. [Hunt, Stearns'90], "power index"

ETH

- Much stronger belief than $\mathsf{P} \neq \mathsf{NP}$.

- Payoff in explanatory power?

YES! *But, story is more complex than NP-completeness.*

- Issue: ETH studies dependence on key param.

$$n = \#\,\mathrm{vars}(\mathcal{F}) \ll |\mathcal{F}|.$$

- Measures dimension of **search space**, not input size!

c.f. [Hunt, Stearns'90], "power index"

ETH

- Much stronger belief than $\mathsf{P} \neq \mathsf{NP}$.
- Payoff in explanatory power?

YES! *But, story is more complex than NP-completeness.*

- Issue: ETH studies dependence on key param.

$$n = \#\,\text{vars}(\mathcal{F}) \ll |\mathcal{F}|.$$

- Measures dimension of **search space**, not input size!

c.f. [Hunt, Stearns'90], "power index"

ETH

- Much stronger belief than $\mathsf{P} \neq \mathsf{NP}$.

- Payoff in explanatory power?

YES! *But, story is more complex than NP-completeness.*

- Issue: ETH studies dependence on key param.

$$n = \#\,\text{vars}(\mathcal{F}) \ll |\mathcal{F}|.$$

- Measures dimension of **search space**, not input size!

c.f. [Hunt, Stearns'90], "power index"

Consequences of ETH

- Consider **IND. SET** problem. Solvable in $O^*(1.23^N)$ time on N-vertex graphs.
- Can we hope for $2^{o(N)}$? Or, would that violate ETH?
- Given: $2^{o(N)}$-time alg for **IND. SET**; try to solve **3-SAT** instance $\mathcal{F}$. ($n \triangleq \#$ vars, $m \triangleq \#$clauses)
- Usual NP-C reduction: $\mathcal{F} \longrightarrow (G, k)$, where

$$|V(G)| \;=\; \Theta(n+m) \;=\; \Theta(m)\;.$$

- $\Longrightarrow$ We solve $\mathcal{F}$ in time $2^{o(m)}$. **WEAK!**

Consequences of ETH

- Consider **IND. SET** problem. Solvable in $O^*(1.23^N)$ time on N-vertex graphs.
- Can we hope for $2^{o(N)}$? Or, would that violate ETH?
- Given: $2^{o(N)}$-time alg for **IND. SET**; try to solve **3-SAT** instance $\mathcal{F}$. ($n \triangleq \#$ vars, $m \triangleq \#$clauses)
- Usual NP-C reduction: $\mathcal{F} \longrightarrow (G, k)$, where

$$|V(G)| \;=\; \Theta(n+m) \;=\; \Theta(m)\;.$$

- $\Longrightarrow$ We solve $\mathcal{F}$ in time $2^{o(m)}$. **WEAK!**

Consequences of ETH

- Consider **IND. SET** problem. Solvable in $O^*(1.23^N)$ time on N-vertex graphs.
- Can we hope for $2^{o(N)}$? Or, would that violate ETH?
- Given: $2^{o(N)}$-time alg for **IND. SET**; try to solve **3-SAT** instance $\mathcal{F}$. ($n \triangleq \#$ vars, $m \triangleq \#$clauses)
- Usual NP-C reduction: $\mathcal{F} \longrightarrow (G, k)$, where

$$|V(G)| = \Theta(n+m) = \Theta(m) .$$

- $\Longrightarrow$ We solve $\mathcal{F}$ in time $2^{o(m)}$. **WEAK!**

Consequences of ETH

- Consider **IND. SET** problem. Solvable in $O^*(1.23^N)$ time on N-vertex graphs.
- Can we hope for $2^{o(N)}$? Or, would that violate ETH?
- Given: $2^{o(N)}$-time alg for **IND. SET**; try to solve **3-SAT** instance $\mathcal{F}$. ($n \triangleq \#$ vars, $m \triangleq \#$clauses)
- Usual NP-C reduction: $\mathcal{F} \longrightarrow (G, k)$, where

$$|V(G)| = \Theta(n+m) = \Theta(m) .$$

- $\Longrightarrow$ We solve $\mathcal{F}$ in time $2^{o(m)}$. **WEAK!**

Consequences of ETH

- Consider **IND. SET** problem. Solvable in $O^*(1.23^N)$ time on N-vertex graphs.
- Can we hope for $2^{o(N)}$? Or, would that violate ETH?
- Given: $2^{o(N)}$-time alg for **IND. SET**; try to solve **3-SAT** instance $\mathcal{F}$. ($n \triangleq \#$ vars, $m \triangleq \#$clauses)
- Usual NP-C reduction: $\mathcal{F} \longrightarrow (G, k)$, where

$$|V(G)| = \Theta(n+m) = \Theta(m) .$$

- $\Longrightarrow$ We solve $\mathcal{F}$ in time $2^{o(m)}$. **WEAK!**

Consequences of ETH

- Consider **IND. SET** problem. Solvable in $O^*(1.23^N)$ time on N-vertex graphs.
- Can we hope for $2^{o(N)}$? Or, would that violate ETH?
- Given: $2^{o(N)}$-time alg for **IND. SET**; try to solve **3-SAT** instance $\mathcal{F}$. ($n \triangleq \#$ vars, $m \triangleq \#$clauses)
- Usual NP-C reduction: $\mathcal{F} \longrightarrow (G, k)$, where

$$|V(G)| \;=\; \Theta(n+m) \;=\; \Theta(m)\;.$$

- $\Longrightarrow$ We solve $\mathcal{F}$ in time $2^{o(m)}$. **WEAK!**

Consequences of ETH

- Consider **IND. SET** problem. Solvable in $O^*(1.23^N)$ time on N-vertex graphs.
- Can we hope for $2^{o(N)}$? Or, would that violate ETH?
- Given: $2^{o(N)}$-time alg for **IND. SET**; try to solve **3-SAT** instance $\mathcal{F}$. ($n \triangleq \#$ vars, $m \triangleq \#$clauses)
- Usual NP-C reduction: $\mathcal{F} \longrightarrow (G, k)$, where

$$|V(G)| \;=\; \Theta(n+m) \;=\; \Theta(m)\;.$$

- $\Longrightarrow$ We solve $\mathcal{F}$ in time $2^{o(m)}$. **WEAK!**

Consequences of ETH

- $2^{o(N)}$ time alg for **IND. SET** $\Longrightarrow$ Solve **3-SAT** in time $2^{o(m)}$.

Theorem (IPZ)

Solve **k-SAT** *in time* $2^{o(m)}$ $\Longrightarrow$ *Solve* **k-SAT** *in time* $2^{o(n)}$ *!!*

- So, $2^{o(N)}$ time alg for **IND. SET** violates **ETH**.

Consequences of ETH

- $2^{o(N)}$ time alg for **IND. SET** $\Longrightarrow$ Solve **3-SAT** in time $2^{o(m)}$.

Theorem (IPZ)

Solve **k-SAT** *in time* $2^{o(m)}$ $\Longrightarrow$ *Solve* **k-SAT** *in time* $2^{o(n)}$ *!!*

- So, $2^{o(N)}$ time alg for **IND. SET** violates **ETH**.

Consequences of ETH

- $2^{o(N)}$ time alg for **IND. SET** $\Longrightarrow$ Solve **3-SAT** in time $2^{o(m)}$.

Theorem (IPZ)

Solve **k-SAT** *in time* $2^{o(m)}$ $\Longrightarrow$ *Solve* **k-SAT** *in time* $2^{o(n)}$ *!!*

- So, $2^{o(N)}$ time alg for **IND. SET** violates **ETH**.

Consequences of ETH

- $2^{o(N)}$ time alg for **IND. SET** $\implies$ Solve **3-SAT** in time $2^{o(m)}$.

Theorem (IPZ)

k-SAT *in time* $O^*(2^{\varepsilon m})\ \forall \varepsilon > 0 \implies$ **k-SAT** *in time* $O^*(2^{\varepsilon n})\ \forall \varepsilon > 0$ *!!*

- So, $2^{o(N)}$ time alg for **IND. SET** violates **ETH**.

Consequences of ETH

- $2^{o(N)}$ time alg for **IND. SET** $\implies$ Solve **3-SAT** in time $2^{o(m)}$.

Theorem (IPZ)

k-SAT *in time* $O^*(2^{\varepsilon m})\ \forall \varepsilon > 0 \implies$ **k-SAT** *in time* $O^*(2^{\varepsilon n})\ \forall \varepsilon > 0$ *!!*

- So, $2^{o(N)}$ time alg for **IND. SET** violates **ETH**.
- Similar: **HAM PATH, DOMINATING SET, VERTEX COVER**.

Consequences of ETH

- **Planar IND. SET** problem: Solvable in $2^{O(\sqrt{N})}$ time on N-vertex planar graphs.
- Can we hope for $2^{o(\sqrt{N})}$?
- Usual NP-C reduction: 3-CNF $\mathcal{F} \longrightarrow$ (planar) (G, k), where
$$|V(G)| = \Theta(m^2) \ .$$
- Solve **Planar IND SET** in time $2^{o(\sqrt{N})} \Longrightarrow$ solve **3-SAT** in time $2^{o(m)}$.
Again, violates **ETH**!
- Similar: **(Planar) HAM PATH, DOM. SET, VERTEX COVER**.

Consequences of ETH

- **Planar IND. SET** problem: Solvable in $2^{O(\sqrt{N})}$ time on N-vertex planar graphs.
- Can we hope for $2^{o(\sqrt{N})}$?
- Usual NP-C reduction: 3-CNF $\mathcal{F} \longrightarrow$ (planar) (G, k), where
$$|V(G)| = \Theta(m^2) \ .$$
- Solve **Planar IND SET** in time $2^{o(\sqrt{N})} \Longrightarrow$ solve **3-SAT** in time $2^{o(m)}$.
Again, violates **ETH**!
- Similar: **(Planar) HAM PATH, DOM. SET, VERTEX COVER**.

Consequences of ETH

- **Planar IND. SET** problem: Solvable in $2^{O(\sqrt{N})}$ time on N-vertex planar graphs.

- Can we hope for $2^{o(\sqrt{N})}$?

- Usual NP-C reduction: 3-CNF $\mathcal{F} \longrightarrow$ (planar) (G, k), where
$$|V(G)| = \Theta(m^2) \ .$$

- Solve **Planar IND SET** in time $2^{o(\sqrt{N})} \implies$
solve **3-SAT** in time $2^{o(m)}$.
Again, violates **ETH**!

- Similar: **(Planar) HAM PATH, DOM. SET, VERTEX COVER**.

Consequences of ETH

- **Planar IND. SET** problem: Solvable in $2^{O(\sqrt{N})}$ time on N-vertex planar graphs.

- Can we hope for $2^{o(\sqrt{N})}$?

- Usual NP-C reduction: 3-CNF $\mathcal{F} \longrightarrow$ (planar) (G, k), where
$$|V(G)| = \Theta(m^2) \ .$$

- Solve **Planar IND SET** in time $2^{o(\sqrt{N})} \Longrightarrow$
solve **3-SAT** in time $2^{o(m)}$.
Again, violates **ETH**!

- Similar: **(Planar) HAM PATH, DOM. SET, VERTEX COVER**.

Consequences of ETH

- **Planar IND. SET** problem: Solvable in $2^{O(\sqrt{N})}$ time on N-vertex planar graphs.
- Can we hope for $2^{o(\sqrt{N})}$?
- Usual NP-C reduction: 3-CNF $\mathcal{F} \longrightarrow$ (planar) (G, k), where
$$|V(G)| = \Theta(m^2) \ .$$
- Solve **Planar IND SET** in time $2^{o(\sqrt{N})} \Longrightarrow$ solve **3-SAT** in time $2^{o(m)}$.
Again, violates **ETH**!
- Similar: **(Planar) HAM PATH, DOM. SET, VERTEX COVER**.

Consequences of ETH

- **Planar IND. SET** problem: Solvable in $2^{O(\sqrt{N})}$ time on N-vertex planar graphs.
- Can we hope for $2^{o(\sqrt{N})}$?
- Usual NP-C reduction: 3-CNF $\mathcal{F} \longrightarrow$ (planar) (G, k), where
$$|V(G)| = \Theta(m^2) \ .$$
- Solve **Planar IND SET** in time $2^{o(\sqrt{N})} \implies$ solve **3-SAT** in time $2^{o(m)}$.
Again, violates **ETH**!
- Similar: **(Planar) HAM PATH, DOM. SET, VERTEX COVER**.

Consequences of ETH

- **Planar IND. SET** problem: Solvable in $2^{O(\sqrt{N})}$ time on N-vertex planar graphs.
- Can we hope for $2^{o(\sqrt{N})}$?
- Usual NP-C reduction: 3-CNF $\mathcal{F} \longrightarrow$ (planar) (G, k), where
$$|V(G)| = \Theta(m^2) \ .$$
- Solve **Planar IND SET** in time $2^{o(\sqrt{N})} \implies$ solve **3-SAT** in time $2^{o(m)}$.
Again, violates **ETH**!
- Similar: **(Planar) HAM PATH, DOM. SET, VERTEX COVER**.

Robustness of ETH

- Why focus on **3-SAT**? Is it WLOG?

  Could **3-SAT** be much easier than **4-SAT**??

- Usual NP-C reduction maps $\mathcal{F}^{(4)} \longrightarrow \mathcal{G}^{(3)}$, where

$$\#\ \text{vars}(\mathcal{G}^{(3)}) \ = \ \Theta\left(\#\ \text{clauses}(\mathcal{F}^{(3)})\right) \ .$$

- $2^{o(n)}$ time alg for **3-SAT** $\implies$ Solve **4-SAT** in time $2^{o(m)}$.

  ([IPZ] result, again) $\implies$ Solve **4-SAT** in time $2^{o(n)}$!

Theorem (IPZ)

$$s_3 \ = \ 0 \quad \Longleftrightarrow \quad s_k \ = \ 0 \quad \forall k \geq 3 \ .$$

Robustness of ETH

- Why focus on **3-SAT**? Is it WLOG?

  Could **3-SAT** be much easier than **4-SAT**??

- Usual NP-C reduction maps $\mathcal{F}^{(4)} \longrightarrow \mathcal{G}^{(3)}$, where

$$\#\ \text{vars}(\mathcal{G}^{(3)}) \ = \ \Theta\left(\#\ \text{clauses}(\mathcal{F}^{(3)})\right) \ .$$

- $2^{o(n)}$ time alg for **3-SAT** $\implies$ Solve **4-SAT** in time $2^{o(m)}$.

  ([IPZ] result, again) $\implies$ Solve **4-SAT** in time $2^{o(n)}$!

Theorem (IPZ)

$$s_3 \ = \ 0 \quad \Longleftrightarrow \quad s_k \ = \ 0 \quad \forall k \geq 3 \ .$$

Robustness of ETH

- Why focus on **3-SAT**? Is it WLOG?

  Could **3-SAT** be much easier than **4-SAT**??

- Usual NP-C reduction maps $\mathcal{F}^{(4)} \longrightarrow \mathcal{G}^{(3)}$, where

$$\#\ \text{vars}(\mathcal{G}^{(3)}) \ = \ \Theta\left(\#\ \text{clauses}(\mathcal{F}^{(3)})\right) \ .$$

- $2^{o(n)}$ time alg for **3-SAT** $\implies$ Solve **4-SAT** in time $2^{o(m)}$.

  ([IPZ] result, again) $\implies$ Solve **4-SAT** in time $2^{o(n)}$!

Theorem (IPZ)

$$s_3 \ = \ 0 \quad \Longleftrightarrow \quad s_k \ = \ 0 \quad \forall k \geq 3 \ .$$

Robustness of ETH

- Why focus on **3-SAT**? Is it WLOG?

  Could **3-SAT** be much easier than **4-SAT**??

- Usual NP-C reduction maps $\mathcal{F}^{(4)} \longrightarrow \mathcal{G}^{(3)}$, where

$$\#\ \text{vars}(\mathcal{G}^{(3)}) \ = \ \Theta\left(\#\ \text{clauses}(\mathcal{F}^{(3)})\right) \ .$$

- $2^{o(n)}$ time alg for **3-SAT** $\Longrightarrow$ Solve **4-SAT** in time $2^{o(m)}$.

  ([IPZ] result, again) $\Longrightarrow$ Solve **4-SAT** in time $2^{o(n)}$!

Theorem (IPZ)

$$s_3 \ = \ 0 \quad \Longleftrightarrow \quad s_k \ = \ 0 \quad \forall k \geq 3 \ .$$

Robustness of ETH

- Why focus on **3-SAT**? Is it WLOG?

  Could **3-SAT** be much easier than **4-SAT**??

- Usual NP-C reduction maps $\mathcal{F}^{(4)} \longrightarrow \mathcal{G}^{(3)}$, where

$$\#\ \text{vars}(\mathcal{G}^{(3)}) \ = \ \Theta\left(\#\ \text{clauses}(\mathcal{F}^{(3)})\right) \ .$$

- $2^{o(n)}$ time alg for **3-SAT** $\implies$ Solve **4-SAT** in time $2^{o(m)}$.

  ([IPZ] result, again) $\implies$ Solve **4-SAT** in time $2^{o(n)}$!

Theorem (IPZ)

$$s_3 \ = \ 0 \quad \Longleftrightarrow \quad s_k \ = \ 0 \quad \forall k \geq 3 \ .$$

Robustness of ETH

- Why focus on **3-SAT**? Is it WLOG?

  Could **3-SAT** be much easier than **4-SAT**??

- Usual NP-C reduction maps $\mathcal{F}^{(4)} \longrightarrow \mathcal{G}^{(3)}$, where

$$\#\ \text{vars}(\mathcal{G}^{(3)}) \ = \ \Theta\left(\#\ \text{clauses}(\mathcal{F}^{(3)})\right) \ .$$

- $2^{o(n)}$ time alg for **3-SAT** $\Longrightarrow$ Solve **4-SAT** in time $2^{o(m)}$.

  ([IPZ] result, again) $\Longrightarrow$ Solve **4-SAT** in time $2^{o(n)}$!

Theorem (IPZ)

$$s_3 \ = \ 0 \quad \Longleftrightarrow \quad s_k \ = \ 0 \quad \forall k \geq 3 \ .$$

Robustness of ETH

- Why focus on **3-SAT**? Is it WLOG?

  Could **3-SAT** be much easier than **4-SAT**??

- Usual NP-C reduction maps $\mathcal{F}^{(4)} \longrightarrow \mathcal{G}^{(3)}$, where

$$\#\ \text{vars}(\mathcal{G}^{(3)}) \ = \ \Theta\left(\#\ \text{clauses}(\mathcal{F}^{(3)})\right) \ .$$

- $2^{o(n)}$ time alg for **3-SAT** $\implies$ Solve **4-SAT** in time $2^{o(m)}$.

  ([IPZ] result, again) $\implies$ Solve **4-SAT** in time $2^{o(n)}$!

Theorem (IPZ)

$$s_3 \ = \ 0 \quad \Longleftrightarrow \quad s_k \ = \ 0 \quad \forall k \geq 3 \ .$$

A stronger hypothesis

$$s_k \triangleq \inf\{\varepsilon : k\text{-SAT decidable in time } O^*(2^{\varepsilon n})\} .$$

Exponential Time Hypothesis (ETH) (IPZ'97)

$$s_3 > 0 .$$

Best known: $s_k \leq 1 - \Theta(1/k)$. Why not "go for broke?"

Strong Exp. Time Hypothesis (**SETH**) (IP'98)

$$\lim_k s_k = 1 .$$

- Note: SETH $\Rightarrow$ ETH.

A stronger hypothesis

$$s_k \triangleq \inf\{\varepsilon : k\text{-SAT decidable in time } O^*(2^{\varepsilon n})\} .$$

Exponential Time Hypothesis (ETH) (IPZ'97)

$$s_3 > 0 .$$

Best known: $s_k \leq 1 - \Theta(1/k)$. Why not "go for broke?"

Strong Exp. Time Hypothesis (**SETH**) (IP'98)

$$\lim_k s_k = 1 .$$

- Note: SETH $\Rightarrow$ ETH.

A stronger hypothesis

$$s_k \triangleq \inf\{\varepsilon : k\text{-SAT decidable in time } O^*(2^{\varepsilon n})\} .$$

Exponential Time Hypothesis (ETH) (IPZ'97)

$$s_3 > 0 .$$

Best known: $s_k \leq 1 - \Theta(1/k)$. Why not "go for broke?"

Strong Exp. Time Hypothesis (**SETH**) (IP'98)

$$\lim_k s_k = 1 .$$

- Note: SETH $\Rightarrow$ ETH.

A stronger hypothesis

$$s_k \triangleq \inf\{\varepsilon : k\text{-SAT decidable in time } O^*(2^{\varepsilon n})\} .$$

Exponential Time Hypothesis (ETH) (IPZ'97)

$$s_3 > 0 .$$

Best known: $s_k \leq 1 - \Theta(1/k)$. Why not "go for broke?"

Strong Exp. Time Hypothesis (SETH) (IP'98)

$$\lim_k s_k = 1 .$$

- Note: SETH $\Rightarrow$ ETH.

A stronger hypothesis

$$s_k \triangleq \inf\{\varepsilon : k\text{-SAT decidable in time } O^*(2^{\varepsilon n})\} .$$

Exponential Time Hypothesis (ETH) (IPZ'97)

$$s_3 > 0 .$$

Best known: $s_k \leq 1 - \Theta(1/k)$. Why not "go for broke?"

Strong Exp. Time Hypothesis (SETH) (IP'98)

$$\lim_k s_k = 1 .$$

- Note: SETH $\Rightarrow$ ETH.

A stronger hypothesis

$$s_k \triangleq \inf\{\varepsilon : k\text{-SAT decidable in time } O^*(2^{\varepsilon n})\} .$$

Exponential Time Hypothesis (ETH) (IPZ'97)

$$s_3 > 0 .$$

Best known: $s_k \leq 1 - \Theta(1/k)$. Why not "go for broke?"

Strong Exp. Time Hypothesis (SETH) (IP'98)

$$\lim_k s_k = 1 .$$

- Note: SETH $\Rightarrow$ ETH.

More consequences of ETH, SETH

- Many more runtime LBs shown under ETH, SETH.
- Strong power to explain dependence on natural input parameters.
- Major implications for **parametrized complexity theory**

[Downey, Fellows]; [Lokshtanov, Marx, Saurabh survey]

Parametrized problems

- Many problem instances have associated **integer parameter** — gives some indication of difficulty.

- E.g., VERTEX COVER:

  Given: (G, k)

  Decide: does G have a vertex cover of size k?

- Goal of "parametrized algorithm" design: design algs that are "fast when k is small."

Parametrized problems

- VERTEX COVER:

  Given: (G, k)

  Decide: does G have a vertex cover of size k?

- Known: VERTEX COVER solvable in time $2^k \cdot n^{O(1)}$ ($n = \#$ verts).

- **ETH implies**: Can't solve in time $2^{o(k)} \cdot n^{O(1)}$.

Parametrized problems

- VERTEX COVER:

  Given: (G, k)

  Decide: does G have a vertex cover of size k?

- Known: VERTEX COVER solvable in time $2^k \cdot n^{O(1)}$

  ($n = \#$ verts).

- **ETH implies**: Can't solve in time $2^{o(k)} \cdot n^{O(1)}$.

Parametrized problems

- VERTEX COVER:

  Given: (G, k)

  Decide: does G have a vertex cover of size k?

- Known: VERTEX COVER solvable in time $2^k \cdot n^{O(1)}$

  ($n = \#$ verts).

- **ETH implies**: Can't solve in time $2^{o(k)} \cdot n^{O(1)}$.

Parametrized problems

- CLIQUE:

  Given: (G, k)

  Decide: does G have a clique of size k?

- Known: CLIQUE solvable in time $\approx n^k/k!$ $(n = \#\text{ verts})$.
- Standard assumption (FPT $\neq$ W[1]) implies:

  can't solve in $F(k) \cdot n^{O(1)}$...

- **ETH implies**: Can't solve in time $F(k) \cdot n^{o(k)}$.

  [Chen, Huang, Kanj, Xia'06]

Parametrized problems

- CLIQUE:

  Given: (G, k)

  Decide: does G have a clique of size k?

- Known: CLIQUE solvable in time $\approx n^k/k!$ $(n = \#\text{ verts})$.
- Standard assumption (FPT $\neq$ W[1]) implies:

  can't solve in $F(k) \cdot n^{O(1)}$...

- **ETH implies**: Can't solve in time $F(k) \cdot n^{o(k)}$.

  [Chen, Huang, Kanj, Xia'06]

Parametrized problems

- CLIQUE:

  Given: (G, k)

  Decide: does G have a clique of size k?

- Known: CLIQUE solvable in time $\approx n^k/k!$ $(n = \#\text{ verts})$.
- Standard assumption (FPT $\neq$ W[1]) implies:

  can't solve in $F(k) \cdot n^{O(1)}$...

- **ETH implies**: Can't solve in time $F(k) \cdot n^{o(k)}$.

  [Chen, Huang, Kanj, Xia'06]

Parametrized problems

- k-DOMINATING SET:

  Given: graph G.

  Decide: does G have a dom. set of size k?

- Known: solvable in time $n^{k+o(1)}$.

  [Eisenbrand, Grandoni'04; Pătraşcu, Williams'10]

- **Strong ETH implies**:* Can't solve in time $n^{k-\varepsilon}$.

  [Pătraşcu, Williams'10]

*(Actually, a bit weaker hyp.)

Parametrized problems

- k-DOMINATING SET:

  Given: graph G.

  Decide: does G have a dom. set of size k?

- Known: solvable in time $n^{k+o(1)}$.

  [Eisenbrand, Grandoni'04; Pătraşcu, Williams'10]

- **Strong ETH implies**:* Can't solve in time $n^{k-\varepsilon}$.

  [Pătraşcu, Williams'10]

*(Actually, a bit weaker hyp.)

Parametrized problems

- k-DOMINATING SET:

  Given: graph G.

  Decide: does G have a dom. set of size k?

- Known: solvable in time $n^{k+o(1)}$.

  [Eisenbrand, Grandoni'04; Pătraşcu, Williams'10]

- **Strong ETH implies**:* Can't solve in time $n^{k-\varepsilon}$.

  [Pătraşcu, Williams'10]

*(Actually, a bit weaker hyp.)

Parametrized problems

- k-DOMINATING SET:

  Given: graph G.

  Decide: does G have a dom. set of size k?

- Known: solvable in time $n^{k+o(1)}$.

  [Eisenbrand, Grandoni'04; Pătraşcu, Williams'10]

- **Strong ETH implies**:* Can't solve in time $n^{k-\varepsilon}$.

  [Pătraşcu, Williams'10]

*(Actually, a bit weaker hyp.)

Treewidth

- **Treewidth** of a graph G: $tw(G) =$ measure of "fatness" of G.

- Known: many NP-C graph problems solvable fast on low-treewidth graphs (by dynamic prog.), in time*

$$c^{tw(G)} \cdot n^{O(1)} \quad \text{for some } c.$$

*(Given a tree decomposition.)

- [Lokshtanov, Marx, Saurabh '11]: **Strong ETH** $\Longrightarrow$ some of these algorithms are **optimal!**

(constant c can't be improved!)

E.g., IND SET, MAX-CUT: $c = 2$, DOM. SET: $c = 3$

Treewidth

- **Treewidth** of a graph G: $tw(G) =$ measure of "fatness" of G.

- Known: many NP-C graph problems solvable fast on low-treewidth graphs (by dynamic prog.), in time*

$$c^{tw(G)} \cdot n^{O(1)} \qquad \text{for some } c.$$

*(Given a tree decomposition.)

- [Lokshtanov, Marx, Saurabh '11]: **Strong ETH** $\Longrightarrow$ some of these algorithms are **optimal!**

(constant c can't be improved!)

E.g., IND SET, MAX-CUT: $c = 2$, DOM. SET: $c = 3$

Treewidth

- **Treewidth** of a graph G: $tw(G) =$ measure of "fatness" of G.

- Known: many NP-C graph problems solvable fast on low-treewidth graphs (by dynamic prog.), in time*

$$c^{tw(G)} \cdot n^{O(1)} \qquad \text{for some } c.$$

*(Given a tree decomposition.)

- [Lokshtanov, Marx, Saurabh '11]: **Strong ETH** $\Longrightarrow$ some of these algorithms are **optimal!**

(constant c can't be improved!)

E.g., IND SET, MAX-CUT: $c = 2$, DOM. SET: $c = 3$

Treewidth

- **Treewidth** of a graph G: $tw(G) =$ measure of "fatness" of G.

- Known: many NP-C graph problems solvable fast on low-treewidth graphs (by dynamic prog.), in time*

$$c^{tw(G)} \cdot n^{O(1)} \qquad \text{for some } c.$$

*(Given a tree decomposition.)

- [Lokshtanov, Marx, Saurabh '11]: **Strong ETH** $\implies$ some of these algorithms are **optimal!**

(constant c can't be improved!)

E.g., IND SET, MAX-CUT: $c = 2$, DOM. SET: $c = 3$

Hardness of subexponential-time approximation

How well can we approximate IND SET on n-vertex graphs in subexponential time?

Consider obtaining an r-approximation to max ind. set size, $r = r(n) = \omega(1)$.

Theorem (Chitniz, Hajiaghayi, Kortsarz'13)

Can get r-approximation in time $O^*(2^{n/r})$.

Theorem (Chalermsook, Laekhanukit, Nanongkai)

Under **ETH**, *no alg. for $r(n) < n^{.49}$ can have runtime $O^*(2^{n^{.99}/r^{1.01}})$.*

Hardness of subexponential-time approximation

How well can we approximate IND SET on n-vertex graphs in subexponential time?

Consider obtaining an r-approximation to max ind. set size, $r = r(n) = \omega(1)$.

Theorem (Chitniz, Hajiaghayi, Kortsarz'13)

Can get r-approximation in time $O^*(2^{n/r})$.

Theorem (Chalermsook, Laekhanukit, Nanongkai)

Under **ETH**, *no alg. for $r(n) < n^{.49}$ can have runtime $O^*(2^{n^{.99}/r^{1.01}})$.*

Hardness of subexponential-time approximation

How well can we approximate IND SET on n-vertex graphs in subexponential time?

Consider obtaining an r-approximation to max ind. set size, $r = r(n) = \omega(1)$.

Theorem (Chitniz, Hajiaghayi, Kortsarz'13)

Can get r-approximation in time $O^*(2^{n/r})$.

Theorem (Chalermsook, Laekhanukit, Nanongkai)

Under **ETH**, *no alg. for* $r(n) < n^{.49}$ *can have runtime* $O^*(2^{n^{.99}/r^{1.01}})$.

Hardness of subexponential-time approximation

How well can we approximate IND SET on n-vertex graphs in subexponential time?

Consider obtaining an r-approximation to max ind. set size, $r = r(n) = \omega(1)$.

Theorem (Chitniz, Hajiaghayi, Kortsarz'13)

Can get r-approximation in time $O^*(2^{n/r})$.

Theorem (Chalermsook, Laekhanukit, Nanongkai)

Under **ETH**, *no alg. for* $r(n) < n^{.49}$ *can have runtime* $O^*(2^{n^{.99}/r^{1.01}})$.

Graph diameter

$$\text{diameter}(G) \triangleq \max_{u,v} \text{dist}_G(u,v) .$$

Theorem (Aingworth, Chekuri, Indyk, Motwani'96; Roddity, Vassilevska Williams'13)

For a simple graph G on n verts, m edges, can compute 3/2-approximation to diameter(G) in (expected) time $\widetilde{O}(m\sqrt{n})$.

Theorem (Roddity, Vassilevska Williams'13)

If we can estimate diameter(G) to approx. factor $(3/2-\varepsilon)$ in time $O(m^{2-\delta})$, then **SETH** *fails.*

- **SETH** $\Longrightarrow$ Detailed info about complexity of a **poly-time** computation!

Graph diameter

$$\text{diameter}(G) \triangleq \max_{u,v} \text{dist}_G(u,v) .$$

Theorem (Aingworth, Chekuri, Indyk, Motwani'96; Roddity, Vassilevska Williams'13)

For a simple graph G on n verts, m edges, can compute $3/2$-approximation to $\text{diameter}(G)$ in (expected) time $\widetilde{O}(m\sqrt{n})$.

Theorem (Roddity, Vassilevska Williams'13)

If we can estimate $\text{diameter}(G)$ to approx. factor $(3/2-\varepsilon)$ in time $O(m^{2-\delta})$, then **SETH** *fails.*

- **SETH** $\Longrightarrow$ Detailed info about complexity of a **poly-time** computation!

Graph diameter

$$\text{diameter}(G) \triangleq \max_{u,v} \text{dist}_G(u,v) .$$

Theorem (Aingworth, Chekuri, Indyk, Motwani'96; Roddity, Vassilevska Williams'13)

For a simple graph G on n verts, m edges, can compute 3/2-approximation to diameter(G) in (expected) time $\widetilde{O}(m\sqrt{n})$.

Theorem (Roddity, Vassilevska Williams'13)

If we can estimate diameter(G) to approx. factor $(3/2-\varepsilon)$ in time $O(m^{2-\delta})$, then **SETH** *fails.*

- **SETH** $\Longrightarrow$ Detailed info about complexity of a **poly-time** computation!

Graph diameter

$$\text{diameter}(G) \triangleq \max_{u,v} \text{dist}_G(u,v) .$$

Theorem (Aingworth, Chekuri, Indyk, Motwani'96; Roddity, Vassilevska Williams'13)

For a simple graph G on n verts, m edges, can compute $3/2$-approximation to $\text{diameter}(G)$ in (expected) time $\widetilde{O}(m\sqrt{n})$.

Theorem (Roddity, Vassilevska Williams'13)

If we can estimate $\text{diameter}(G)$ to approx. factor $(3/2-\varepsilon)$ in time $O(m^{2-\delta})$, then **SETH** *fails.*

- **SETH** $\Longrightarrow$ Detailed info about complexity of a **poly-time** computation!

Graph diameter

$$\text{diameter}(G) \triangleq \max_{u,v} \text{dist}_G(u,v) .$$

Theorem (Aingworth, Chekuri, Indyk, Motwani'96; Roddity, Vassilevska Williams'13)

For a simple graph G on n verts, m edges, can compute $3/2$-approximation to $\text{diameter}(G)$ in (expected) time $\widetilde{O}(m\sqrt{n})$.

Theorem (Roddity, Vassilevska Williams'13)

If we can estimate $\text{diameter}(G)$ to approx. factor $(3/2-\varepsilon)$ in time $O(m^{2-\delta})$, then **SETH** *fails.*

- **SETH** $\Longrightarrow$ Detailed info about complexity of a **poly-time** computation!

Further afield

- [Abboud, Vassilevska Williams'14]: Improvements in certain **dynamic algorithms** for graph problems $\Rightarrow$ $\neg$SETH.
- [Bringmann, this morning]: Compute Fréchet distance in $n^{2-\varepsilon}$ time $\Rightarrow$ $\neg$SETH.
- Seems likely to see more results of this kind...

The key theorem

Theorem (IPZ)

k-SAT *in time* $O^*(2^{\varepsilon m})\ \forall \varepsilon > 0 \quad \Longrightarrow \quad$ **k-SAT** *in time* $O^*(2^{\varepsilon n})\ \forall \varepsilon > 0.$

$m = \#$ clauses$(\mathcal{F})$, $\qquad n = \#$ variables$(\mathcal{F})$.

- Let's see the proof ideas.
  Main challenge: for general "dense" $\mathcal{F}$, may have $m \gg n$.
- Ideal approach: give a "sparsification" reduction:

$$\mathcal{F} \longrightarrow^{ptime} \mathcal{F}' \qquad SAT(\mathcal{F}) = SAT(\mathcal{F}')$$

$$m' \ , \ n' \ \leq \ O(n) \ .$$

- Solve $\mathcal{F}'$ in time $2^{o(m')} = 2^{o(n)} \quad \Longrightarrow \quad$ solve $\mathcal{F}$. ???

The key theorem

Theorem (IPZ)

k-SAT *in time* $O^*(2^{\varepsilon m})\ \forall \varepsilon > 0 \quad \Longrightarrow \quad$ **k-SAT** *in time* $O^*(2^{\varepsilon n})\ \forall \varepsilon > 0.$

$m = \#$ clauses$(\mathcal{F})$, $\qquad n = \#$ variables$(\mathcal{F})$.

- Let's see the proof ideas.
  Main challenge: for general "dense" $\mathcal{F}$, may have $m \gg n$.
- Ideal approach: give a "sparsification" reduction:

$$\mathcal{F} \longrightarrow^{ptime} \mathcal{F}' \qquad SAT(\mathcal{F}) = SAT(\mathcal{F}')$$

$$m',\ n' \leq O(n)\ .$$

- Solve $\mathcal{F}'$ in time $2^{o(m')} = 2^{o(n)} \Longrightarrow$ solve $\mathcal{F}$. ???

The key theorem

Theorem (IPZ)

k-SAT *in time* $O^*(2^{\varepsilon m})\ \forall \varepsilon > 0 \quad \Longrightarrow \quad$ **k-SAT** *in time* $O^*(2^{\varepsilon n})\ \forall \varepsilon > 0.$

$m = \#$ clauses$(\mathcal{F})$, $\qquad n = \#$ variables$(\mathcal{F})$.

- Let's see the proof ideas.
  Main challenge: for general "dense" $\mathcal{F}$, may have $m \gg n$.
- Ideal approach: give a "sparsification" reduction:

$$\mathcal{F} \longrightarrow^{ptime} \mathcal{F}' \qquad SAT(\mathcal{F}) = SAT(\mathcal{F}')$$

$$m',\ n' \leq O(n)\ .$$

- Solve $\mathcal{F}'$ in time $2^{o(m')} = 2^{o(n)} \Longrightarrow$ solve $\mathcal{F}$. ???

The key theorem

Theorem (IPZ)

k-SAT *in time* $O^*(2^{\varepsilon m})\ \forall \varepsilon > 0 \quad \Longrightarrow \quad$ **k-SAT** *in time* $O^*(2^{\varepsilon n})\ \forall \varepsilon > 0.$

$m = \#$ clauses$(\mathcal{F})$, $\qquad n = \#$ variables$(\mathcal{F})$.

- Let's see the proof ideas.
  Main challenge: for general "dense" $\mathcal{F}$, may have $m \gg n$.
- Ideal approach: give a "sparsification" reduction:

$$\mathcal{F} \longrightarrow^{ptime} \mathcal{F}' \qquad SAT(\mathcal{F}) = SAT(\mathcal{F}')$$

$$m',\ n' \leq O(n)\ .$$

- Solve $\mathcal{F}'$ in time $2^{o(m')} = 2^{o(n)} \quad \Longrightarrow \quad$ solve $\mathcal{F}$. ???

The key theorem

Theorem (IPZ)

k-SAT *in time* $O^*(2^{\varepsilon m})\ \forall \varepsilon > 0 \quad \Longrightarrow \quad$ **k-SAT** *in time* $O^*(2^{\varepsilon n})\ \forall \varepsilon > 0.$

$m = \#$ clauses$(\mathcal{F})$, $\qquad n = \#$ variables$(\mathcal{F})$.

- Let's see the proof ideas.
  Main challenge: for general "dense" $\mathcal{F}$, may have $m \gg n$.
- Ideal approach: give a "sparsification" reduction:

$$\mathcal{F} \longrightarrow^{ptime} \mathcal{F}' \qquad SAT(\mathcal{F}) = SAT(\mathcal{F}')$$

$$m',\ n' \leq O(n)\ .$$

- Solve $\mathcal{F}'$ in time $2^{o(m')} = 2^{o(n)} \quad \Longrightarrow \quad$ solve $\mathcal{F}$. **???**

The key theorem

Theorem (IPZ)

k-SAT *in time* $O^*(2^{\varepsilon m})\ \forall \varepsilon > 0 \quad \Longrightarrow \quad$ **k-SAT** *in time* $O^*(2^{\varepsilon n})\ \forall \varepsilon > 0.$

$m = \#$ clauses$(\mathcal{F})$, $\qquad n = \#$ variables$(\mathcal{F})$.

- Let's see the proof ideas.
  Main challenge: for general "dense" $\mathcal{F}$, may have $m \gg n$.
- Ideal approach: give a "sparsification" reduction:

$$\mathcal{F} \quad \longrightarrow^{2^{o(n)}\ \textit{time}} \quad \mathcal{F}' \qquad SAT(\mathcal{F}) \;=\; SAT(\mathcal{F}')$$

$$m',\ n' \quad \leq \quad O(n)\ .$$

- Solve $\mathcal{F}'$ in time $2^{o(m')} \;=\; 2^{o(n)} \quad \Longrightarrow \quad$ solve $\mathcal{F}$. ???

The key theorem

Theorem (IPZ)

k-SAT *in time* $O^*(2^{\varepsilon m})\ \forall \varepsilon > 0 \quad \Longrightarrow \quad$ **k-SAT** *in time* $O^*(2^{\varepsilon n})\ \forall \varepsilon > 0.$

$m = \#$ clauses$(\mathcal{F})$, $\qquad n = \#$ variables$(\mathcal{F})$.

- Let's see the proof ideas.
  Main challenge: for general "dense" $\mathcal{F}$, may have $m \gg n$.
- Ideal approach: give a "sparsification" reduction:

$$\mathcal{F} \longrightarrow^{2^{o(n)}\ time} \mathcal{F}' \qquad SAT(\mathcal{F}) = SAT(\mathcal{F}')$$

$$m',\ n' \leq O(n)\ .$$

- Solve $\mathcal{F}'$ in time $2^{o(m')} = 2^{o(n)} \Longrightarrow$ solve $\mathcal{F}$. **???**

The key lemma

- Relax this idea further...

$$\mathcal{F} \quad \rightarrow^{2^{o(n)}\ time} \quad \mathcal{G}^1, \mathcal{G}^2, \ldots, \mathcal{G}^s \qquad s \;=\; 2^{o(n)}$$

$$SAT(\mathcal{F}) \;=\; \bigvee_i SAT(\mathcal{F}^i)$$

Sparsification Lemma (IPZ'97)

Fix $k \geq 3, \varepsilon > 0$.
There exists a reduction $\mathcal{F} \rightarrow \mathcal{G}^1, \ldots, \mathcal{G}^s$, computable in time $O^*(2^{\varepsilon n})$, such that

1. $\mathcal{F} \in SAT$ *iff* $\exists i : \mathcal{G}^i \in SAT$;
2. $s \leq 2^{\varepsilon n}$;
3. $\#vars(\mathcal{G}^i) \leq n$;
4. $\#clauses(\mathcal{G}^i) \leq O_{k,\varepsilon}(n)$.

The key lemma

- Relax this idea further...

$$\mathcal{F} \quad \rightarrow^{2^{o(n)}\ time} \quad \mathcal{G}^1, \mathcal{G}^2, \ldots, \mathcal{G}^s \qquad s \;=\; 2^{o(n)}$$

$$SAT(\mathcal{F}) \;=\; \bigvee_i SAT(\mathcal{F}^i)$$

Sparsification Lemma (IPZ'97)

Fix $k \geq 3, \varepsilon > 0$.
There exists a reduction $\mathcal{F} \rightarrow \mathcal{G}^1, \ldots, \mathcal{G}^s$, computable in time $O^*(2^{\varepsilon n})$, such that

1. $\mathcal{F} \in SAT$ *iff* $\exists i : \mathcal{G}^i \in SAT$;
2. $s \leq 2^{\varepsilon n}$;
3. $\#vars(\mathcal{G}^i) \leq n$;
4. $\#clauses(\mathcal{G}^i) \leq O_{k,\varepsilon}(n)$.

The key lemma

- Relax this idea further...

$$\mathcal{F} \quad \rightarrow^{2^{o(n)}\ time} \quad \mathcal{G}^1, \mathcal{G}^2, \ldots, \mathcal{G}^s \qquad s \;=\; 2^{o(n)}$$

$$SAT(\mathcal{F}) \;=\; \bigvee_i SAT(\mathcal{F}^i)$$

Sparsification Lemma (IPZ'97)

Fix $k \geq 3, \varepsilon > 0$.
There exists a reduction $\mathcal{F} \rightarrow \mathcal{G}^1, \ldots, \mathcal{G}^s$, *computable in time* $O^*(2^{\varepsilon n})$, *such that*

1. $\mathcal{F} \in SAT$ *iff* $\exists i : \; \mathcal{G}^i \in SAT$;
2. $s \leq 2^{\varepsilon n}$;
3. $\#vars(\mathcal{G}^i) \leq n$;
4. $\#clauses(\mathcal{G}^i) \leq O_{k,\varepsilon}(n)$.

The key lemma

Sparsification Lemma (IPZ'97)

Fix $k \geq 3, \varepsilon > 0$.
There exists a reduction $\mathcal{F} \to \mathcal{G}^1, \ldots, \mathcal{G}^s$, *computable in time* $O^*(2^{\varepsilon n})$, *such that*

1. $\mathcal{F} \in SAT$ *iff* $\exists i : \mathcal{G}^i \in SAT$;
2. $s \leq 2^{\varepsilon n}$;
3. $\#vars(\mathcal{G}^i) \leq n$;
4. $\#clauses(\mathcal{G}^i) \leq O_{k,\varepsilon}(n)$.

- Now suppose we could solve **k-SAT** in time $2^{\delta m}$ for small $\delta > 0$.
- Use Lemma to solve **k-SAT** in time $2^{\varepsilon n} \cdot 2^{\delta(C_{k,\varepsilon} n)}$. Take $\delta \ll C_{k,\varepsilon}^{-1} \varepsilon$.

The key lemma

Sparsification Lemma (IPZ'97)

Fix $k \geq 3, \varepsilon > 0$.
There exists a reduction $\mathcal{F} \to \mathcal{G}^1, \ldots, \mathcal{G}^s$, computable in time $O^*(2^{\varepsilon n})$, such that

1. $\mathcal{F} \in SAT$ *iff* $\exists i : \mathcal{G}^i \in SAT$;
2. $s \leq 2^{\varepsilon n}$;
3. $\#vars(\mathcal{G}^i) \leq n$;
4. $\#clauses(\mathcal{G}^i) \leq O_{k,\varepsilon}(n)$.

- Now suppose we could solve **k-SAT** in time $2^{\delta m}$ for small $\delta > 0$.
- Use Lemma to solve **k-SAT** in time $2^{\varepsilon n} \cdot 2^{\delta(C_{k,\varepsilon} n)}$. Take $\delta \ll C_{k,\varepsilon}^{-1} \varepsilon$.

The key lemma

Sparsification Lemma (IPZ'97)

Fix $k \geq 3, \varepsilon > 0$.
There exists a reduction $\mathcal{F} \to \mathcal{G}^1, \ldots, \mathcal{G}^s$, *computable in time* $O^*(2^{\varepsilon n})$, *such that*

1. $\mathcal{F} \in SAT$ *iff* $\exists i : \mathcal{G}^i \in SAT$;
2. $s \leq 2^{\varepsilon n}$;
3. $\#vars(\mathcal{G}^i) \leq n$;
4. $\#clauses(\mathcal{G}^i) \leq O_{k,\varepsilon}(n)$.

- Now suppose we could solve **k-SAT** in time $2^{\delta m}$ for small $\delta > 0$.
- Use Lemma to solve **k-SAT** in time $2^{\varepsilon n} \cdot 2^{\delta(C_{k,\varepsilon} n)}$. Take $\delta \ll C_{k,\varepsilon}^{-1} \varepsilon$.

The key lemma

Sparsification Lemma (IPZ'97)

Fix $k \geq 3, \varepsilon > 0$.
There exists a reduction $\mathcal{F} \to \mathcal{G}^1, \ldots, \mathcal{G}^s$, *computable in time* $O^*(2^{\varepsilon n})$, *such that*

1. $\mathcal{F} \in SAT$ *iff* $\exists i : \mathcal{G}^i \in SAT$;
2. $s \leq 2^{\varepsilon n}$;
3. $\#vars(\mathcal{G}^i) \leq n$;
4. $\#clauses(\mathcal{G}^i) \leq O_{k,\varepsilon}(n)$.

- Now suppose we could solve **k-SAT** in time $2^{\delta m}$ for small $\delta > 0$.
- Use Lemma to solve **k-SAT** in time $2^{\varepsilon n} \cdot 2^{\delta(C_{k,\varepsilon} n)}$. Take $\delta \ll C_{k,\varepsilon}^{-1} \varepsilon$.

Proof of sparsification lemma (debt to D. Scheder's notes!)

.....

Thanks!