23/11/21

— Worst case Analysis.

Assume $P \neq NP$

$3SAT \notin P$ or any other NP-complete problems.

(1) Allow run-time larger than polynomial

For example, $3SAT \in TIME[1.304^n]$

$K\text{-}SAT \in TIME[2^{n(1-\frac{\epsilon}{K})}]$ for some constant $\epsilon > 0$.

"Exact- Algorithms"

(2) Still run in poly-time but only ask for "approximate" solutions.

Vertex - Cover. Solution size $\leq 2 \cdot$ Optimum size.

G,

3SAT — a satisfying assignment satisfies all clauses

approximate solution - an assignment that satisfies "most" of the clauses

$\geq 99\%$

"Approximation Algorithms".

(3) May be you want an algorithm that runs correctly in poly-time on "most" of the instance.
$\geq 99\%$ (SAT Solvers)

"Average Case Complexity".

Approximation - Algorithms "Hardness-of-Approximation"

want a solution that is close to the optimum solution.

minimization problem e.g. vertex cover.

optimum solution $\leq$ Solution given by the algorithm $\leq$ 5. Optimal solution

maximization problem; e.g. MaxCut, Max SAT

$$\frac{\text{optimal solution}}{5} \leq \text{solution given by the algorithm} \leq \text{optimal solution}$$

(Probabilistically Checkable proofs.)

PCP Thm: $P \neq NP$. Then,

$\nexists$ any poly-time algorithm that finds an assignment that satisfies $\geq 99.99\%$ of the clauses.

[Håstad '99] Assuming $P \neq NP$. $\nexists$ any poly-time algorithm that given 3-CNF formula finds an assignment that satisfies, $\forall \varepsilon > 0$, $\geq \left(\frac{7}{8} + \varepsilon\right)$ fraction of clauses.

optimal.

$\exists$ a poly-time algorithm that given 3-CNF formula outputs an assignment that satisfies at least $\frac{7}{8}$ fraction of clauses.

Proof: randomized algorithm that in expectation satifies $\geq \frac{7}{8}$ -fraction

will be working with Exact - 3 - CNF.

Given φ a Exact - 3 - CNF on n - variables.

Let $a \in \{0,1\}^n$ be a random assignment

$$I_j = \begin{cases} 1 & \text{if clause } j \text{ is Satisfied by } a \\ 0 & \text{otherwise} \end{cases}$$

$$\# \text{ Satisfied clauses} = I_1 + \cdots + I_m$$

$$E_a[\# \text{ Satisfied Clauses}] = \sum_{i=1}^{m} E[I_i]$$

$$= \sum_{i=1}^{m} \Pr[\,I_i \text{ evaluates to True}\,]$$

$(x_1 \vee x_3 \vee \bar{x}_5)$

$$= \frac{7}{8} m$$

Derandomization:- (Method of conditional Expectation)

$$E_a[\# \text{ Clauses}] = \frac{1}{2} \cdot E[\# \text{ Clauses} \mid x_1 = 1] + \frac{1}{2} \cdot E[\# \text{ Clauses} \mid x_1 = 0]$$

Vertex Cover

Solution from your algo. $\leq 2 \cdot$ Optimal Solution

Approximating Vertex-cover better than $\sqrt{2} - \varepsilon$ is NP-hard.

$\approx 1.414 - \varepsilon$

Minimum Bisection: Given a graph G you want to find a subset S of vertices of size $\frac{|V|}{2}$ s.t. the edges going across $S \Leftrightarrow \bar{S}$ is minimized.

approx. upper bound $\leq$ $(\log n)$

we don't have any approx. lower bound.

Problem:- Given a 3-CNF formula Output the max. no. of clauses that can be satisfied.

Output: some number $1 \le \text{output} \le m$

Suppose you have a polytime algorithm that gives an assignment that satisfies $> \frac{7}{8}$ fraction of clauses.
$\frac{7}{8} + 0.00000001$

Then we can use this promised algo to solve 3-SAT in poly-time.

Exact-Algorithms:

$$k\text{SAT} \in \text{TIME}\left[2^{n(1-\frac{c}{k})}\right]$$

for some constant $c > 0$.

3-SAT runs in time $1.304\ldots^{n}$

$\downarrow$ maybe.

$n^{\text{poly}(\log n)}$

$\downarrow$

$n^{\log \lg n}$

[Impagliazzo-Paturi-Zane '01]

Strong-Exponential-Time-Hypothesis (SETH)

$\forall \varepsilon > 0$, $\exists k$ s.t. k-SAT requires

$2^{(1-\varepsilon)n}$ time.

[2010s]

LCS
$O(n^2)$ based on Dynamic programming

Edit Distance.
$O(n^2)$

Orthogonal Vectors.
$O(n^2)$

given $S \subseteq \{0,1\}^d$
$|S| = n$
Does $\exists$ two vector $u, v \in S$
s.t.
$\langle u, v \rangle = 0$?

Conditioned on SETH
you can show that
the above runtime
is optimal.

Contrapositive

if LCS has $O(n^{1.9999})$-time
algo then SETH false.

Goal: understand the exact complexity

③ "Average Case Complexity"

Applications in Cryptography.

- Average Case Hardness

- One-way functions.

$f : \{0,1\}^* \to \{0,1\}^*$

compute $f(x)$ easily but given y

computing $f^{-1}(y)$ is hard.

- pseudo-random generators

— × — × —

↦ Lower bounds.

SAT requires linear time

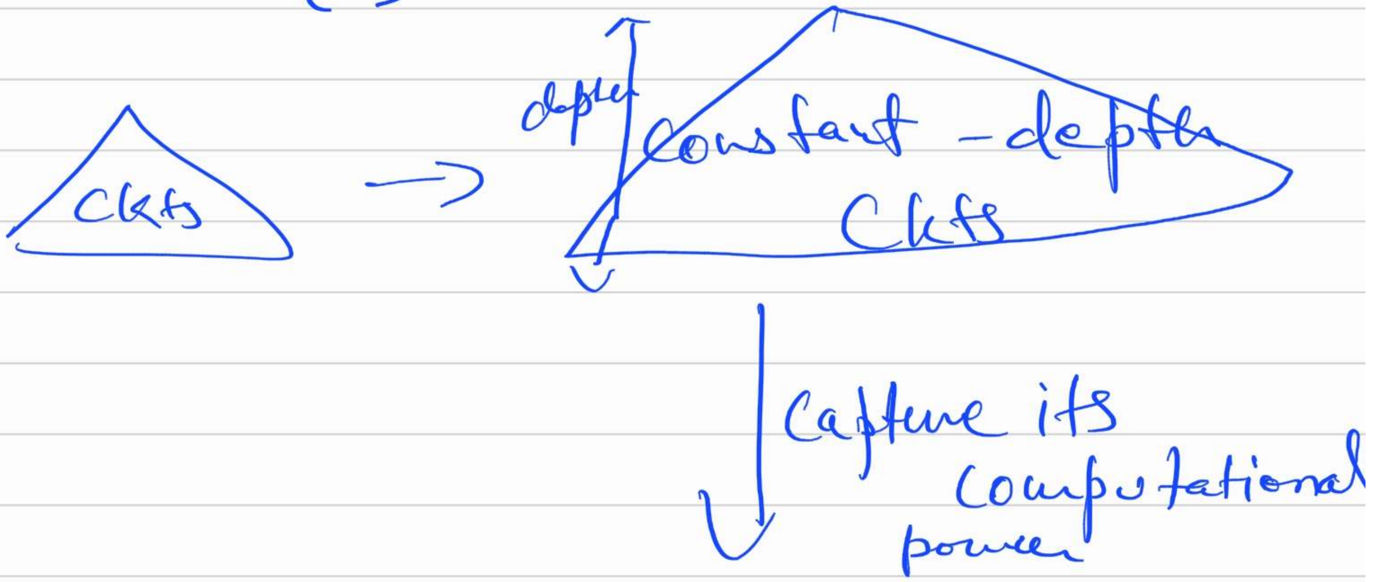
linear size

ckts..

↦ Circuit Complexity

→ Communication Complexity.

→ Polynomial methods in CS.



multivariate polynomials.

[Razborov–Smolensky]
Constant-depth ckts can be approximated by a low-degree polynomials

e.g. $\oplus_n$ = Parity over n-bits