CS 294-114 Fine-Grained Complexity and Algorithms **Fall 2015**

Lecture 12: Algorithms for NP-Complete Problems

Lecturer: Russell Impagliazzo *Scribe: Tyler Linderoth*

Last class:

If there is any improved circuit-SAT algorithm, then NEXP $\not\subseteq$ P/poly.
How close are we to getting this type of algorithm?
Special cases of SAT?
Improved SAT for super linear size circuits provide super linear lower bound for E^{NP} (JMV).

Today:

What does the complexity of satisfiability tell us about the complexity of other problems (e.g. hard problems)? Accordingly, how much can we improve NP-complete problems?

Consider a circuit with n inputs, $X_1 = g_1$,...,$X_n = g_n$, and m gates, $g_{n+1} = op_{n+1}(i_{n+1}, j_{n+1})$,...,$g_m$.

To reduce to 3-SAT, ask: $\exists$ X_1,...,X_n, g_1,...,g_m so that $g_i = op_i(g_{ji}, g_{ki})$, where $g_m = 1$? Each g_i involves 3 variables so that it can be written as a 3-CNF.

If 3-SAT can be solved in TIME($2^{\epsilon n}$), we can solve circuit-SAT in TIME($2^{\epsilon m}$).

If we can find an algorithm for 3-SAT that is time $2^{o(n)}$, then we prove a circuit lower bound.

ETH (Exponential Time Hypothesis) states that no such algorithm exists: $\exists\epsilon$ so that no $2^{\epsilon n}$ time algorithm can solve 3-SAT.

Best known 3-SAT (K-SAT) algorithms:
1) Algorithm based on the switching lemma; probability zero error, ran in time O($2^{n(1-\frac{1}{ck})}$) to solve K-SAT. Note that as K gets larger, our savings get smaller. Note also that by SETH (Strong Exponential Time Hypothesis), $\forall\epsilon > 0$ $\exists$K so that K-SAT is not solvable in $2^{(1-\epsilon)n}$ time.

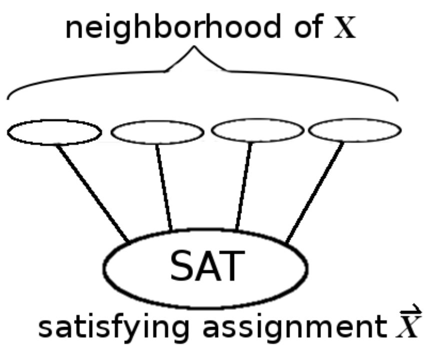
2) Algorithm by Peturi, Pudlàk, and Zane that uses compression method: randomly permute the variables and for each variable set it to a random value UNLESS it is forced from previous choices (i.e. there is a clause, $\mathscr{C}$, $X_i\vee$,...,$\vee X_{i_{k-1}}$ so that we've already set X_{i_1},X_{i_2},...,$X_{i_{k-1}}$ to FALSE). We hope that we set a satisfying assignment. IF $\mathscr{C}$ is satisfiable, we find a set assignment.
$X \vee Y \vee Z \wedge Z \vee W \vee U$
set U=FALSE:
$X \vee Y \vee Z \wedge Z \vee W \vee \cancel{U}$
set W=TRUE:
$X \vee Y \vee Z \wedge \cancel{Z \vee W \vee U}$
set X=FALSE:
$\cancel{X} \vee Y \vee Z \wedge \cancel{Z \vee W \vee U}$
set Y=FALSE:
$\cancel{X \vee Y \vee Z} \wedge \cancel{Z \vee W \vee U}$
Z is forced to TRUE

What if the formula had just one assignment that satisfied it?
X_1,...,X_n
$X_i \rightarrow \neg X_i \Longleftarrow$ flip X_i from satisfying to not satisfying
$\mathscr{C} = C_1 \wedge ... \wedge C_m \Longleftarrow$ all were true
Now the flip made at least one of $\mathscr{C}$'s C_is false.
$\forall i, \exists\, X_i \vee \overline{X}, ..., \vee \overline{X_k}$
C_{j_i} : X_i is the only literal satisfying C_j
C_{j_i} is called the "critical clause" for X_i.

Prob(PPZ algorithm outputs X)
= Prob(all random decisions are equal to X)
$= 2^{-(\text{\# random decisions on the path consistent with X})}$
$= 2^{-n+\text{\# forced decisions on the path consistent with X}}$

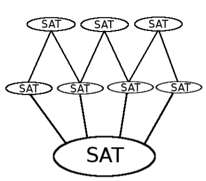
When does C_{j_i} force X_i?
IF X_i is the last variable branched on in C_{j_i}
Prob(X_i is the last) $\leq \frac{1}{k}$

Therefore,
Expected # of forced decisions on the path consistent with $X \geq \frac{n}{k}$
Expectation(PPZ algorithm outputs X) $\geq 2^{-n+\frac{n}{k}} = 2^{-n(1-\frac{1}{k})}$



ith neighbor $X_i \rightarrow \overline{X_i}$ (differs in 1 bit)
D(X) = # neighbores that also satisfy $\mathscr{C}$
i.e. D(X) is the degree of X.
n − D(X) variables with critical clauses.
Expected(# of forced moves) $\geq \frac{n-\text{D}(X)}{k}$

We want the algorithm to return X, but we are just as happy if a neighbor of X is returned if the neighbors are SAT:



$\forall X \in$ SAT assignment,
$\mathrm{P}(\text{PPZ returns } X) \geq 2^{-n+\frac{n-\mathrm{D}(X)}{k}} \geq 2^{-n(1-\frac{1}{k})-\frac{\mathrm{D}(X)}{k}}$
$\mathrm{P}(\text{PPZ returns some satisfying assignment}) \geq 2^{-n(1-\frac{1}{k})} \sum_{X} 2^{\frac{-\mathrm{D}(X)}{k}}$

Now let's say we have a graph where the nodes represent satisfying assignments and the edges represent the index of variables.
$E_i = \#$ of pairs that differ only in X_i

Harper's Lemma: For any set of size S, the average degree is $\leq \log|S|$.

Entropy, H: measures randomness of a distribution.
$\mathrm{H}(D) = -\sum_{X,\ \mathrm{P}(X)\geq 0} \mathrm{P}(X)\log(\mathrm{P}(X))$
$\mathrm{H}(X, Y) = (\text{expected Y of entropy of X}|\text{this value of } Y)$
$\mathrm{H}(< X_i, ..., X_n >) = \sum_{i=1}^{n} \mathrm{H}(X_i|X_1, ..., X_{i-1})$

Pick $X \in S$ at random and let X_i be the ith bit of X.
$\log|S| = \mathrm{H}(X) = \sum_{i=1}^{n} \mathrm{H}(X_i|X_1, ..., X_{i-1})$
$\geq \sum_{i=1}^{n} \mathrm{H}(X_i|X_1, ..., X_{i-1}, X_{i+1}, ..., X_n)$
$= \text{Expected}(\mathrm{D}(X))$

Then,
$\sum_{X} 2^{\frac{-\mathrm{D}(X)}{k}} \geq S2^{\frac{-\log|S|}{k}}$ (increase with k).