- You know the drill now: Find students around you to form a ***small group***; use ***all resources*** to help to solve the problems; ***discuss*** your idea with other group member and ***write down*** your own solutions; raise your hand and pull the ***course staffs*** to help; ***submit*** your writeup through Gradescope in *24 hours*.

Our topic for this working session is on *exponential time hypotheses*.

In class we introduced the exponential time hypothesis. There are actually a few variants:

- ***Exponential Time Hypothesis (ETH)*** says CNFSAT cannot be solved in time $m^{O(1)} \cdot 2^{o(n)}$. In other words, no *sub-exponential time* (of the form $2^{n^{0.99}}$) algorithm for CNFSAT.
- ***Strong Exponential Time Hypothesis (SETH)*** assumes no algorithm can solve CNFSAT in time $m^{O(1)} \cdot 2^{\delta n}$ for any $\delta < 1$. In other words, the best constant for any exponential algorithm for CNFSAT has to be 2 (i.e. no algorithm with time 1.99^n).
- ***Super Strong Exponential Time Hypothesis (SSETH)*** assumes the best algorithm that solve k-SAT runs in time $m^{O(1)} \cdot 2^{\left(1-\frac{1}{\Theta(k)}\right)n}$. That is, the best tradeoff for k is a linear function.

We have shown that if we can make the reduction from CNFSAT to problem R efficient by imposing size restriction on the new instance for R to be linear in the original CNFSAT instance size, then we can prove that R cannot be solved in subexponential time under ETH.

When Impagliazzo, Paturi, and Zane introduced the exponential time hypotheses in 2001, they were uncertain if these assumptions actually hold (especially SSETH which might be too strong to be true), and thus the term "hypothesis" instead of "conjecture". They didn't find much use for SETH because the reduction has to be *ultra* efficient, as in the instance size cannot even blowup by a constant (it has to be $n + o(n)$). Fortunately, Williams and Williams realized that we can instead make the instance *sub-exponenially bigger*, and thus establishing polynomial lower bounds for problems in P! This gives rise to the blooming field of *fine-grained complexity*, which is the topic of today.

To this date some TCS researchers still doubt the validity of SETH and SSETH (most of other NP-complete problems have algorithms with running time c^n for some $c < 2$; why should CNFSAT be different?). Funny enough, every algorithm we know for k-SAT has the running time precisely of the form $m^{O(1)} \cdot 2^{\left(1-\frac{1}{\Theta(k)}\right)n}$.

A small detail we didn't bring up: we actually need to assume the CNFSAT is still hard when the number of clauses m is not much more than the number of variables n, otherwise saying the problem cannot be solved in $2^{o(m)}$ is impossible when the brute-force algorithm runs in 2^n time. Fortunately, the *sparsification lemma* by Impagliazzo and Paturi reduces any n-variable m-clause k-SAT instance to $2^{\varepsilon n}$ many ***sparse*** k-SAT instances where each has number of clauses at most linear in the number of variables, for any $\varepsilon > 0$. So we can safely assume we work with sparse instances at all times.

A ***dominating set*** in graph G is set of vertices S so that every vertex is in S or a neighbor of S.

DOMSET
- ***Input:*** *An undirected graph G, a parameter q*
- ***Output:*** *Does G have a dominating set of size at most q?*

We can also separate the parameter q out from the definition of DOMSET:

q-DOMSET
- ***Input:*** *An undirected graph G*
- ***Output:*** *Does G have a dominating set of size at most q?*

Obviously for each *fixed* q, q-DOMSET can be solved in $O(n^q)$ time. (Why?)

1. Prove that DOMSET does not have sub-exponential time algorithm under ETH.
2. Prove that q-DOMSET cannot be solved in $O(n^{q-\delta})$ time for any $\delta > 0$ under SETH, by reducing an arbitrary n-variable sparse ($m = O(n)$) k-SAT instance to a q-DOMSET instance with $O(q2^{n/q} + m)$ nodes.

To think about later: (No submissions needed)

3. We are given a system of m linear inequalities on n variables (A_{ij} and b_j are integers):

$$\begin{cases} A_{11}x_1 + A_{12}x_2 + \cdots + A_{1n}x_n & \leq & b_1 \\ A_{21}x_1 + A_{22}x_2 + \cdots + A_{2n}x_n & \leq & b_2 \\ \vdots & & \vdots \\ A_{m1}x_1 + A_{m2}x_2 + \cdots + A_{mn}x_n & \leq & b_m \end{cases}$$

   The ***01 integer programming problem*** (01-IP) asks: *Is there a vector $(x_1, \ldots, x_n) \in \{0,1\}^n$ satisfying all inequalities?*

   Assuming ETH is true, show that no algorithm can solve 01-IP in time $2^{o(n)}$.

4. Let A and B be two sets of n $\{0,1\}$-vectors in dimension d. The ***orthogonal vector problem*** (OV) asks whether there are vectors $\alpha \in A$ and $\beta \in B$ such that α and β are orthogonal: $\langle \alpha, \beta \rangle = \sum_i \alpha_i \beta_i = 0$

   Reduce from CNF-SAT to show that no algorithm can solve OV in $O(n^{2-\varepsilon})$ time for any $\varepsilon > 0$ under SETH.

Conceptual question: I don't know if P≠NP, ETH, or SETH is true or not. But through the lens of reductions, we can safely say that every algorithm designed can be viewed as a pursuit to answer these big questions, as in an algorithmic breakthrough may potentially falsify a hypothesis and lead to better algorithms for SAT. This is not a math question, but *do you believe in SETH?*