Lecture 18

- More on P, NP, coNP
- Hamiltonian Path
- Decision vs Search

Review of time complexity classes:

\[ \text{PATH} = \{ G, s, t \mid \text{Graph } G \text{ contains a path from } s \text{ to } t \} \]

we know the intersection of NP, co-NP that is not known to be NP is non-empty, so COMPOSITES WITH HINT (from HW 8) is in NP, coNP.

- zoom in
- green area: if non-empty, then P=NP
  - if empty, then P ≠ NP.

- grey area: if non-empty, then NP=coNP, because
  - suppose L in coNP is also NP-complete.
  - want to show that \( \forall A \in \text{NP}, A \in \text{coNP} \) and \( \forall B \in \text{NP}, B \in \text{coNP} \).
  - since \( A \in \text{NP}, A \notin L \), then \( A^c \notin L^c \). Since \( L \in \text{coNP} \), \( L^c \in \text{NP} \), so decide \( A^c \) in bounded poly-time, also:
    - on input \( \langle x \rangle \):
      - compute \( f(x) \), where \( f \) is the reduction from \( A \) to \( L \)
      - run \( N \), the NTM decoder (poly-time) for \( L^c \) on \( f(x) \)
      - agree with its decision.

you may refer to the solution of HW 8, LP3 for another description of this logic.

Hence \( A^c \in \text{NP} \Rightarrow A \in \text{coNP} \).
Given $B \in \text{CONP}$, $B^c \in \text{PL}$, so $B \in \text{PL}^c$. Hence $B$ is polytime reducible to an NP language $\Rightarrow B \in \text{NP}$.

Hence if the green region is not empty, then $\text{NP} = \text{CONP}$.

We also have NP-hard languages that are not $\in$ NP, one such language is known to be in $\text{MinColorability}$, which you will become more familiar with on the midterm! Yay!

Reminder; some NP-COMPLETE LANGUAGES are SAT, 3SAT, CLIQUE, 3-COLORABILITY, SUBSET-SUM, HAMILTONIAN PATH + friends.

Similarly to NP-COMPLETENESS, we have CONP-COMPLETE, and CONP-HARD languages. You will show that MIN-COLORABILITY is CONP-HARD.

It is conjectured that CONP $\neq$ NP. It follows that NP $\neq$ P. This belief is depicted in the graph:

- If $P \neq \text{NP}$, but $\text{NP} = \text{CONP}$.
- If $P = \text{NP}$.

Our favorite CONP-Complete Language is TAUTOLOGY.

* If you want to look at more complexity classes, there is a website called complexity.zoo.uwaterloo.ca!
Recall the language $\text{HAMILTONIAN PATH} = \{ G, s, t \mid G \text{ is a directed graph, and } \exists \text{ a path from } s \text{ to } t \text{ that with every vertex in } V(G) \text{ exactly once} \}$

It is easy to see that $\text{HAMILTONIAN PATH}$ is in $\text{NP}$ by using a verifier.

It is $\text{NP}$-hard by a reduction from $\text{3SAT}$.

Student question: Why do we use $\text{3SAT}$ so much?

Usually because it gives the cleanest reduction. In "real life", it is often hard to see which language is nice to use. Languages like $\text{3-SAT}$ and $\text{3-Colorability}$ are the three that gives a nice structure to the problems.

Here is a tree to explain the flow of reductions:

- SAT
  - $\text{3SAT}$
    - CLIQUE
      - Independent set
        - Vertex cover
        - $\text{3-Colorability}$
          - $\text{SUBSET-SUM}$
            - Bin packing
              - "McCoy's Problem"
                - $\text{PARTITION}$

If you are locked for $\text{NP}$-complete problems, there is a book by Garey and Johnson titled "NP-Completeness" from the 70's. They have a great introduction!
Back to HAMPATH!

Input $\phi$ with $n$ variables and $m$ clauses.

Idea: construct a graph with a peculiar structure

Imagine one chamber for each variable, clearly, $\exists a HAMPATH$ through this graph. There are actually $2^n$ of them, two options for each variable.

Add in, for each chamber

Imagine clause $\phi_i = (x_i \lor \overline{x_i} \lor \overline{x_i} \lor x_i)$. Then we have

Let going left correspond to being true for the variable.

Add edge out of left middle vertex of $x_i$ chamber if $x_i$ appears in $\phi_i$; and going back from node $\phi_i$ to the right middle $\phi_i'$. If $x_i$ appears as false, then have edge $(x_i \lor \text{right-middle}, \phi_i)$ and $(\phi_i, x_i \lor \text{left-middle})$.

Then, imagine $\phi_i$ as the only clause in $\phi$.

Then, we only visit all the nodes if we take the path to $\phi_i$ at some point, which we can only do by picking the correct direction to go through the graph, corresponding to the truth assignment of variable $x_i$ in clause $\phi_i$.

In general, we have more clauses:

- $V_i$, top
- $V_i$, middle
- $V_i$, bottom

If $x_i$ appears in clause $\phi_j$, add edges

- $(V_{ij}, V_{k,l})$, $(V_{kl}, V_{ij})$
- $(V_{ij}, \overline{x}_i \lor \overline{x}_j)$, $(V_{kl}, V_{ij})$
Reduction:

"On input a 3CNF \( \phi = \phi_1 \land \phi_2 \land \ldots \land \phi_m \) on \( n \)-variables \( x_1, \ldots, x_n \)

Compute directed graph \( G \),

\[
V(G) = \left\{ \begin{array}{l}
V_1, \topo, V_2, \topo, \ldots, V_n, \topo, V_{n+1}, \topo = t, \\
V_{ij} : 1 \leq i \leq n, 0 \leq j \leq m \}
\end{array} \right.
\]

\[
E(G) = \left\{ \begin{array}{l}
(V_{ij}, V_{i+1,j}) \text{ for } 0 \leq j \leq m-1, \\
(V_{ij}, V_{i,j+1}) \text{ for } 0 \leq j \leq m-1, \\
(V_{i,0}, V_{i+1,0}), \\
(V_{i,m}, V_{i+1,m}) \text{ for } 1 \leq i \leq n
\end{array} \right.
\]

Output \((G, s, t)\) .

Running time: \( O(nm) \) for vertices, \( O(n^2 m) \) edges \( \rightarrow \) polynomial

Corollary: Suppose \( \phi \in 3SAT \). Let us construct a path \( a_1, \ldots, a_n \) then \( a_1, \ldots, a_n \)
be the set assignment to \( \phi \) (which we have assumed exists). Path starts
at \( s = V_1, \topo \). For every \( i, \) go from \( V_i, \topo \) to \( V_{i,0} \) if \( a_i = T \),
and to \( V_{i,m} \) if \( a_i = F \).

If \( a_i = T \) and \( x_i \) is in clause \( \phi_j \), and \( V_{i,j} \) has not yet been
visited, go from \( V_{i,j-1} \) to \( V_{i,j} \). And \( V_{i,j} \) is \( V_{i,j} \). Else, go from \( V_{i,j-1} \) to \( V_{i,j} \).

Hence we can construct a path from \( s \) to \( t \). From the true assignment.
Suppose $G$ has a Hamilton path from $s$ to $t$. Then, by similar logic, it
has a Hamilton assignment. (Exercise for the reader!!)