15-251: Great Theoretical Ideas In Computer Science

Recitation 13

Review

- The set **P** is the set of all languages L such that there exists a program P and a constant $c \in \mathbb{R}$ such that P decides L and $P(x)$ runs in at most $|x|^c$ steps for all x. Less formally, **P** is the set of all languages that are decidable in polynomial time.

- The set **NP** is the set of all languages L such that there exists a program V and a constant $c \in \mathbb{R}$ such that for all x,

  - If $x \in L$ then there exists a string y such that $V(x, y)$ returns "yes" in at most $|x|^c$ steps.
  - If $x \notin L$ then for all strings y, $V(x, y)$ returns "no" in at most $|x|^c$ steps.

  Less formally, **NP** is the set of all languages for which a proof of membership can be checked in polynomial time.

- A (many-one) reduction from a language L to a language L' is a function $f\colon L \to L'$ such that $x \in L \Leftrightarrow f(x) \in L'$. In this class we are interested only in polytime reductions, i.e., reductions that can be computed in polynomial time. Whenever we say reduction, we mean polytime reduction.

- A language L is **NP**-hard if for all languages $L' \in$ **NP**, there exists a polytime reduction from L' to L.

- A language L is **NP**-complete if $L \in$ **NP** and L is **NP**-hard.

- For your homework, you may assume without further proof (we've already shown this) that $\mathit{CIRCUITSAT}, \mathit{3SAT}, \mathit{3COLOR}, \mathit{CLIQUE}, \mathit{INDSET}$ are all **NP**-complete.

Stupid Reductions...

Recall the definition of the halting problem:

HALT: Given a machine M and an input w, does M halt on w?

And the definition of the three-coloring problem:

3COLOR: Given a graph G, does G have a valid 3-coloring?

Now, define a new language ISSORTED, in the expected way:

ISSORTED: Given a list l of positive integers, are they sorted in non-decreasing order?

Show that 3COLOR reduces to HALT.

Show that ISSORTED reduces to 3COLOR.

What can you conclude from these two reductions?

How about you PROVE these are NP-complete instead of just taking our word for it?

Recall two problems from lecture:

CLIQUE: Given a graph G and a positive integer k, does G have a k-clique?
INDSET: Given a graph G and a positive integer k, does G have an independent set of size k?

Show that CLIQUE and INDSET are NP-complete by reducing 3SAT to one of these problems. (Since we've already shown that CLIQUE and INDSET reduce to one another, it will suffice to show that either one is NP-complete.)

SUBSETSUM

Recall the following definition of problems:
SUBSETSUM: For a given finite $A \subseteq \mathbb{Z}$ and $k \in \mathbb{Z}$, determine if there is a set $S \subseteq A$ such that

$$\sum_{x \in S} x = k$$

.
PARTITION: For a given finite $A \subseteq \mathbb{Z}$ determine if there is a set $S \subseteq A$ such that

$$\sum_{x \in S} x = \sum_{k \in A \setminus S} x$$

Assume that SUBSETSUM is NP-complete. Prove that PARTITION is NP-complete.

Exactly 33% gives you an A+. All other scores get an F.

Recall the definition of 3-SAT: "Determine if a formula of the form $(\neg x_3 \vee x_2 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee x_3) \dots$ is satisfiable." Note that this is equivalent to each clause having at least one literal evaluate to true. We define another problem, 1in3-SAT as "Is there a truth assignment where exactly one literal in each clause evaluates to true."
Assume 1in3SAT is NP-complete. Prove that 3-SAT is NP-complete.

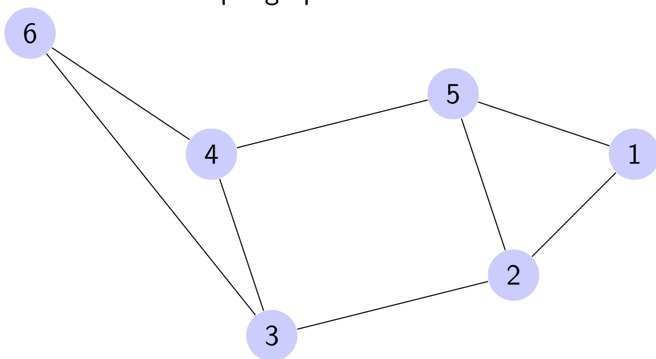
Do not visit xkcd.com/230

We define two very similar problems:

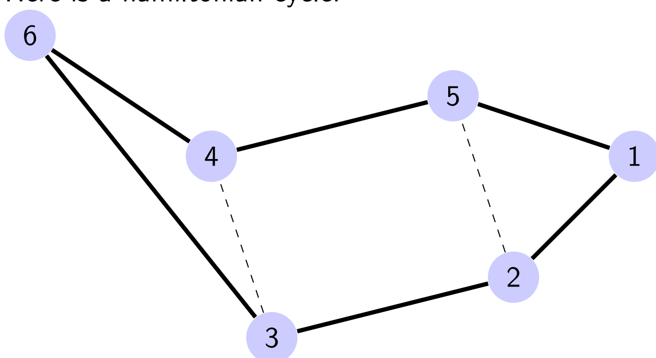
HAMILTONIAN-CYCLE: for a given graph G, is there a **cycle** that visits every vertex exactly once?

HAMILTONIAN-PATH: for a given graph G, is there a **path** that visits every vertex exactly once?

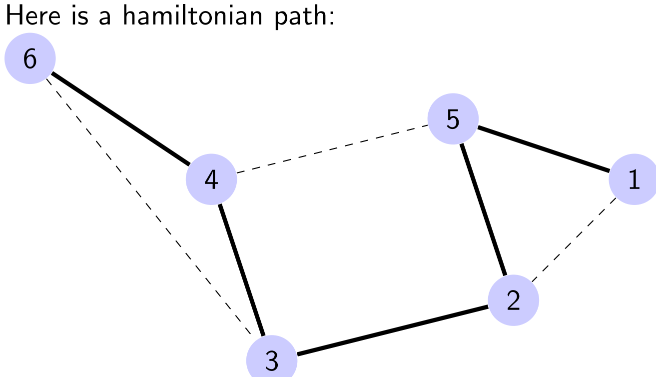
Consider an example graph:



Here is a hamiltonian cycle:



Here is a hamiltonian path:



Assume HAMILTONIAN-PATH is NP-complete. Prove that HAMILTONIAN-CYCLE is NP-complete.
Assume HAMILTONIAN-CYCLE is NP-complete. Prove that HAMILTONIAN-PATH is NP-complete.