HW10: out later today (due next Tue 8pm)

HW11

--- So FAR ---

Part 1: Models of Computation

Part 2: Design Efficient Algorithms

Part 3: Problems for which no algorithms exist

Problems for which no efficient algorithms exist.

How to Argue that solutions are unlikely to exist?

Template:

* Suppose you're trying to figure out if there exists an efficient algorithm for problem Y.

* You have a channel to God.

* God will only tell you whether a different problem X is hard.

If Y has a solution, then so does X.

God told you X is hard.

⇒ Y is also hard.
**CONDITIONAL RESULTS**

**DECISION PROBLEMS**

**Problem.** \( \Pi \) : Collection of instances (strings).

For each instance, answer is YES or NO.

Answer function \( f_\Pi : \Sigma^* \rightarrow \{0, 1\} \) where

\[
\begin{align*}
  f_\Pi(I) &= 1 \quad \text{iff } I \text{ is YES instance} \\
  f_\Pi(I) &= 0 \quad \text{iff } I \text{ is NO instance}
\end{align*}
\]

\[ L_\Pi = \{ I \mid f_\Pi(I) = 1 \} \]

\(<x>\) refers to an encoding of \( x \) in some format.

Graph \( G \), \(<G>\) is an encoding of the graph as a string.

\( G = (V, E), s, t, B \) length of shortest path from \( s \) to \( t \) in \( G \).

Instance \( = <G, s, t, B> \).

**REDUCTION BETWEEN LANGUAGES.**

For two languages \( L_x, L_y \).

A reduction FROM \( L_x \) TO \( L_y \) is an algorithm:

input : \( w \in \Sigma^* \)

output : \( w' \in \Sigma^* \)

such that \( w \in L_y \iff w \in L_x \).
**R**: Reduction (from) \( X \rightarrow Y \) \( X \leq Y \)

Given \( A_y \): Algorithm for \( Y \),
Build \( A_x \): Algorithm for \( X \) (that uses \( A_y \))

\[ R \] has running time \( R(n) \) where \( n \) is size of input to \( R \)
\( A_y \) has running time \( O(n) \) where \( n \) is size of input to \( A_y \)
\( A_x \) has running time \( \frac{R(n) + 0(R(n))}{n^2 + n^3} \)

Suppose \( |I_x| = n \). First run \( R \), takes \( R(n) \).
Next run \( A_y \), which takes \( O(|I_x|) \leq O(R(n)) \)

If \( R \) is polynomial-time and \( A_y \) is also polynomial time,
then \( A_x \) is polynomial time.

If \( R \) is polynomial time and makes polynomially many accesses to \( A_y \), and \( A_y \) is also polynomial time,
then \( A_x \) is polynomial time.
Lemma (1) If \( X \leq Y \) and \( Y \) has an algorithm,
then \( X \) has an algorithm.

(2) If \( X \leq_Y \) and \( Y \) has a polynomial-time algorithm,
poly-time reduction then \( X \) has a polynomial-time algorithm.

(3) If \( X \leq Y \) and \( X \) does not have an algorithm
\( Y \) does not have an algorithm.

(4) If \( X \leq_{p} Y \) and \( X \) does not have a poly-time algorithm
\( Y \) does not have a polynomial-time algorithm.

\[
\begin{align*}
X \leq Y, & \quad Y \leq Z \implies X \leq Z \\
X \leq_{(p)} Y, & \quad Y_{(p)} \leq Z \implies X \leq_{(p)} Z
\end{align*}
\]

\( X \leq Y \implies Y \leq X \)

Prove hardness of new problem \( Y \),
based on known hardness of
well-known problem \( X \).

\[
\begin{align*}
X \leq Y & \quad \text{You know: } X \text{ is hard.} \\
\text{and } X \leq Y & \quad \text{and } X \leq Y \\
\implies Y \text{ is hard.}
\end{align*}
\]
How to prove $X \leq Y$.

Give $R(x) \rightarrow y$

Such that $I_x$ is YES instance of $x$  

$\iff I_y$ is YES instance of $y$

$I_x \text{ is YES}_x \Rightarrow I_y \text{ is YES}_y$

$I_y \text{ is YES}_y \Rightarrow I_x \text{ is YES}_x$

How to prove $X \leq_P Y$?

In addition to proving that

$I_x \text{ is YES}_x \iff I_y \text{ is YES}_y$

Also prove that $R$ is polynomial time.
EXAMPLES OF REDUCTIONS

\[ G = (V, E) \]

INDEPENDENT SET: \( S \subseteq V \) such that no 2 vertices in \( S \) are connected by an edge.

\[ \langle G, k \rangle \]

Does \( G \) have an INDEPENDENT SET of size \( \geq k \)?

CLIQUE: Set \( S \subseteq V \) s.t. every pair of vertices in \( S \) is connected by an edge.

\[ \langle G, k \rangle \]

\( G \) has a clique of size \( \geq k \)?
\[ G = (G, k) \]

Does \( G \) have an independent set of size \( \geq k \)?

\( R \): Given \( G \), computes \( \overline{G} \)

- If \( (u, v) \in \overline{G} \) \( \Leftrightarrow \) \( (u, v) \) is not an edge in \( G \).

To prove: \( I_x \) is a YES instance of INDEPENDENT SET \( \Leftrightarrow I_y \) is a YES instance of CLIQUE.
\( G \) has an independent set of size \( \geq k \)

\( \implies \) \( \overline{G} \) has a clique of size \( \geq k \).

A set \( S \) is an independent set in \( G \)

\( \implies \) no 2 vertices in \( S \) have an edge between them in \( G \).

\( \implies \) every pair of vertices in \( S \) have an edge between them in \( \overline{G} \).

\( \implies \) \( S \) is a clique in \( \overline{G} \).

---

**VERTEX COVER**

Let \( G = (V, E) \) be a graph.

\( S \) is an independent set \( \iff \forall S \) is a vertex cover.
INDEPENDENT SET $\leq$ VERTEX COVER

Consider any $uv \in E$

$u \notin S$ or $v \notin S$ \hspace{1cm} (S is indep set)

$\Rightarrow u \in V \setminus S$ or $v \in V \setminus S$

$\Rightarrow V \setminus S$ is a vertex cover.

$V \setminus S$ is V.C.

Consider any $u, v \in S$.

$\Rightarrow uv$ is not an edge of $G$ else $V \setminus S$ does not cover $uv$

$\Rightarrow S$ is an independent set.

$(G, k) \in$ INDSET $\iff (G', n-k) \in_{yes} VCOV$

INDSET $\leq_p$ VCOV

VCOV $\leq_p$ INDSET