Precept Topics

- Intractability and Algorithm Design
- NP-completeness
- Jeopardy!

A. RECAP: Intractability and Algorithm Design

Your preceptor will give an overview of the content of this week's lectures.

Feel free to use this space for notes or as scratch paper.

B. EXERCISE: NP-completeness of Independent Set

In this problem, we will prove a well-known intractability result: that INDEPENDENT-SET (IND-SET hereafter) is NP-complete. Recall that, to do so, we need to – besides understanding what the IND-SET problem is – prove two separate things:

1. IND-SET is in NP (i.e., there is a polynomial-time algorithm for *verifying* a candidate solution);
2. Some NP-complete problem poly-time reduces to IND-SET.

We will pick SAT as the NP-complete problem to reduce from.

An instance of the IND-SET problem has two components: a graph G and a positive integer k. A solution to the instance (G, k) is a set S of vertices such that none of the edges of G have both endpoints in S.

a) Let's start with the first (and easier) step: prove that IND-SET is in NP. That is, describe an algorithm that, given a purported solution S to an instance (G, k), verifies whether the solution is valid or not in polynomial time.

b) In order to reduce from SAT to IND-SET, we must construct an instance of IND-SET from an instance of SAT. Here is one way to do so: if the system of boolean equations has m equations and n variables, set $k = m$ and create a graph G with one vertex for each appearance of a literal (i.e. a variable or a negation of a variable) in an equation. Then place an edge between a pair of vertices if the variables they represent are:

- in the same equation; or
- the negation of one another.

Apply this transformation to the SAT instances below and list all of the independent sets of size at least m in the graphs you obtain.

x_1	or	$\neg x_2$	or	$\neg x_3$	or	x_4	=	true
$\neg x_1$	or	$\neg x_2$			or	x_4	=	true
		x_2	or	x_3	or	$\neg x_4$	=	true

$\neg x_1$	or	x_2			=	true
		$\neg x_2$	or	x_3	=	true
x_1			or	$\neg x_3$	=	true
$\neg x_1$	or	$\neg x_2$	or	$\neg x_3$	=	true
x_1	or	x_2	or	x_3	=	true

c) We're still missing one step in the reduction: post-processing the solution of the IND-SET instance to obtain a solution to the SAT instance. Describe an algorithm that does so.

d) Explain why the reduction is correct; that is, show that

1. it runs in polynomial time;
2. it generates an unsatisfiable IND-SET instance when the SAT instance is unsatisfiable; and
3. it generates a satisfiable instance when the SAT instance is satisfiable, and the solution obtained from the post-processing step is a satisfying truth assignment.

C. JEOPARDY!

Your preceptor will lead a *Jeopardy!* round with categories from topics of the course. Have fun!
(And if you don't know the rules, make sure to ask before starting.)

EXERCISE (<u>optional</u>): VERTEX-COVER is the problem whose instances are graph-integer pairs (G, k) and a solution is a set S of vertices such that every edge has at least one endpoint in S. Prove that VERTEX-COVER is NP-complete.