Tutorial part 1

A. Show that the following problems are in NP. Show the polytime verifiers.

1. COMPOSITE: Problem of determining whether a given number is composite.
2. CIRCUIT INEQUIVALENCE: Problem of determining whether two given circuits having a single output and same number of inputs behave differently.
3. SET COVER: Given a collection of sets S1,..,Sm of a ground set and an integet k, determine if there exists a subcollection of size k such that the union of the sets in the subcollection is the same as the union of the sets in the collection.
4. SAT
5. MULTIPROCESSOR SCHEDULING. Given tasks with durations t1..tn, p the number of processors, and T an integer target time, determine if it is possible to assign the tasks to processors such that each processor gets tasks of total duration at most T.

6. Given x, is x a perfect square.

7. Given a polynomial P(x) in x (e.g. P(x)=x^3-3x+1) and an interval [a,b], does P have a root in [a,b], i.e. does there exist an integer x s.t. a <= x <= b s.t. P(x)=0
(What if x is not required to be an integer?)

B. OFFICIAL DEFINITION OF NP COMPLETENESS: Q is said to be NP-complete if Q is in NP and every problem in NP can be reduced to Q.

Show that this definition is equivalent to the definition we have earlier, i.e. Q is NPC if Q is in NP and Q is NP-hard.

The phrase "x-complete" usually refers to something that contains the essence of "x" whatever x is. Or the "hardest"/"most interesting" elements of x. The definition states that Q is as hard as any problem in NP (for the purpose of finding polytime algorithms).

C. Show that if Q is in NP, then there exists an algorithm for solving Q that runs in time 2^m where m is some poly(|x|).

D. Use the above to give an example of a problem which is not in NP.

Examples of problems suspected to not be in NP:

1. Complements of SAT. Key observation: while it is possible to give a short proof to the statement "this circuit is satisfiable" by giving a satisfying assignment, this does not seem possible for the negation of this. If you want to prove that "this circuit is not satisfiable" in general it would seem that you must check all

possibilities. Similarly for problems 2,3,5.

Decision problems which are complements of NP problems are said to constitute the class Co-NP.

2. Decision problems of the form: "The minimum number of colours needed for this graph is 5". There is a short proof to show that 5 colours suffice (give the colouring), however, it seems that to prove that 4 colours are not sufficient, it would be necessary to give a complicated argument which would somehow endup considering all possibilities.

How to prove a problem Q to be NPC:

1. Prove it is in NP -- this can be done by showing a polytime verifier for it.

2. Prove it is NP-hard, i.e. that SAT can be reduced to it. But instead of SAT we can use another problem that has been proved NP-hard earlier.

1. is usually fairly easy -- but make sure you understand it. 2. can be very hard -- but often some simple ideas work. We will see this in the tutorial below.

Tutorial part 2.

Show that the following problems are NP-hard. You may assume that the following problems have already been shown to be NP-hard: Independent set, Clique, subgraph isomorphism, vertex cover, subset sum, hamiltonian circuit

1. Set cover
2. Travelling salesman
3. Multiprocessor scheduling
4. Facility location. There are a set of n customers and m possible locations for hospitals. For the ith location we are given the set si of customers that can be served if a hospital were actually opened in this location. Also given as input is an integer F. It is required to determine if all customers can be served by opening hospitals at some F of the m locations.
5. Circuit inequivalence

B. Show that Subset sum has a pseudopolynomial time algorithm (i.e. an algorithm which runs in polynomial time if the numbers are expressed in unary). Note that all NP-complete problems need not have pseudopolynomial time algorithms. Some such as TSP are NP-complete even if the input is expressed in unary. Such problems are called

strongly NP-complete.
