CSE 200
Computability and Complexity
Homework 3
NP, Completeness, and Reductions, part 2: Beyond NP
Space complexity
Due Monday, June 2

May 5, 2008

Give proofs for each problem. Proofs can be high-level, but be precise. You may use without giving a proof any result proved in class or in the textbook. In particular, to prove NP-completeness, it suffices to give a reduction from any of the NP-complete problems from the text or from class. However, you must show your reduction is valid, by showing the equivalence of the constructed instance and the original.

Quadratic programming: Prove that the following problem is NP-hard: Quadratic programming: Given a set of equations $E_1, ..E_t$ in n variables $x_1, ..x_n$ of the form: E_k is $\sum_{i=1}^{i=n} a_{i,k}{x_i}^2 + \sum_{1 \leq i \leq j \leq n} b_{i,j,k} x_i x_j + \sum 1 \leq i \leq n d_{i,k} x_i = c_k$ where $a_{i,k}, b_{i,j,k}, d_{i,k}, c_k$ are given integers. Decide if there is a solution, i.e., an assinment of real numbers $v_1, ..v_n$ to the variables making all equations true. Is this problem in NP? (Don't give a formal proof for whether the problem is in NP) (Hint: consider the equation: ${x_i}^2 = x_i$.)

PSPACE Prove that $NP^{PSPACE} = PSPACE$.

Polynomial-time hierarchy Prove that the polynomial-time hierarchy is contained in $PSPACE$.

2-SAT: The 2-SAT problem is to decide Satisfiability for CNF formulas with at most 2 literals per clause. Prove that $2 - SAT$ is $co - NL$-complete (under deterministic logspace many-one reductions). You can use without proof that $PATH$ is NL-complete.

Counting classes Consider the problem of *counting* the number of accepting runs of a non-deterministic Turing machine N on input x. Show that if N uses $O(\log n)$ space, then the associated counting problem is in FP.

Sudoku We started looking at reducing sudoku problems to SAT last assignment.

The *sudoku* problem of size n is as follows. The input is an $n^2 \times n^2$ matrix M whose entries are either "blank" or an integer between 1 and n^2. A solution fills in the blank spaces with integers between 1 and n^2. The following constraints must be met: Each integer from 1 to n^2 appears exactly once in each row, in each column, and in each $n \times n$ sub-matrix of the form $M[jn + 1...(j + 1)n][in + 1..(i + 1)n]$ for each $0 \leq i, j \leq n - 1$. The problem is to find any solution meeting the constraints, or return "no solution possible" if there is no such solution.

Last assignment, you gave at least two different ways to reduce the Sudoku problem to $CNF-SAT$. (See the answer key for some last minute hints.)

Try solving sudoku problems by combining the above reductions with a complete SAT solver, such as Zchaff (download page, http://www.princeton.edu/ chaff/zchaff/index2.html). Use the following test distribution:

Repeat k times: Pick a random unfilled position in the matrix. Pick a random value v. If there are no v's in the block, row, or column, assign the matrix element value v. Otherwise, leave it blank.

For each reduction, and as many values of n and k, as you can get to run in a reasonable amount of time, and averaging over as many runs as is reasonable, give the probability of a solution for n and k, and the average time your algorithm took to run. What seem to be the hardest random instances of sodoku?

(Note: Be careful not to use up too much computer time. Don't leave programs running unsupervised too long. Depending on your algorithm, you may find even very small sizes take huge amounts of time. However, some credit will be based on getting results for larger n.)