

CSE 200
Computability and Complexity
Homework 2
NP, Completeness, and Reductions
Due Monday, May 10

April 26, 2010

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.

Restricted 3-SAT Show that the 3-SAT problem remains *NP*-complete when restricted to formulas where each variable appears at most 3 times. (Hint: remember that you can use smaller clauses than size 3.)

Independent Set Maximality (ISM) Prove that the following problem is *NP*-complete. Given a graph G and an independent set I in G , is there a larger independent set I' , $|I| < |I'|$?

Abstract tiling Consider the abstract tiling problem shown non-computable in class, but where in addition to the set of tiles T , you have an additional input 1^B representing the dimension of the square to be tiled, i.e., is there a $B \times B$ matrix so that the borders are the special border symbols and all 2×3 consecutive submatrices are in T ? Prove that this problem is *NP*-complete. (Hint: it may be easier to reduce directly from a Turing Machine than to reduce a known *NP*-complete problem to it.)

Abstract Sudoku Consider the *abstract sudoku* problem. An instance is a number k , n variables $x_1 \dots x_n$ taking on values $1..k$, and a family of subsets S_1, \dots, S_m , each $S_j \subset \{1, \dots, n\}$ and has size $|S_j| = k$. A solution assigns each variable x_i a value a_i in $\{1, \dots, k\}$. It is feasible if every value appears exactly once in each subset, i.e., for each S_j , and each $1 \leq v \leq k$, there is an $i \in S_j$ with $a_i = v$. The abstract sudoku problem is, given k , n and the family S_1, \dots, S_m , decide whether there is a feasible solution.

Prove that abstract sudoku is NP -complete. You can use without proof the NP -completeness of any problem proved NP -complete in class, in the text, or on the homeworks. (However, although it has been proved that non-abstract sudoku (see below) is also NP -complete, you should not use that fact for this problem.)

Sudoku 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 $M[jn+1 \dots (j+1)n][in+1 \dots (i+1)n]$ for each $0 \leq i, j \leq n-1$. The problem is to decide whether there is a solution meeting the constraints.

Give at least two distinct reductions from Sudoku to CNF SAT. For each, analyze the number of variables in the resulting CNF , and the number of clauses of different sizes. In the next homework, you will be asked to run experiments combining these reductions with a SAT solver to solve Sudoku. Which reduction do you expect to have the best results in the experiment, and why?