CSE 431
Introduction to Theory of Computation
Homework #7
Due: Friday, May 28, 2010

W. L. Ruzzo 15 May 2010

Homework Assignment:

1. 7.20
2. 7.23
3. 7.24
4. 7.25
5. 7.41. (For comparison, 7.40 shows that minimization of DFAs *is* doable in polynomial time.)
6. Well folks, here's the event you've all been waiting for: 4, count 'em *four*, proofs that P = NP. Only you can stop Ruzzo from becoming world-famous! Find and explain the flaw in each of the "proof" sketches below. Try to be explicit about which hypotheses of critical theorems are being violated or misused. Give simple, concrete counterexamples where possible.

   Let *SUBSET-SUM*, also know as *KNAP*, be the set $\{a_1\#a_2\#\ldots\#a_n\#C \mid a_i$ and C are integers coded in binary, and there is a set $I \subseteq \{1,\ldots,n\}$ such that $\sum_{i\in I} a_i = C\}$. Let *UKNAP* be the same, except that the integers are coded in unary, i.e., a is represented by the string 1^a. It is known that *UKNAP* is in *P*, but *KNAP* is *NP*-complete.

   (a) For any string u in $\{1,\#\}^*$ we can easily produce a string v in $\{0,1,\#\}^*$ such that $u \in UKNAP \Leftrightarrow v \in KNAP$. (E.g., if $u = 11\#1\#11111$ then $v = 10\#1\#101$.) Further, the transformation can be done in time bounded by a polynomial in the length of u. Thus, $P = NP$.

   (b) For any string v in $\{0,1,\#\}^*$ we can easily produce a string u in $\{1,\#\}^*$ such that $v \in KNAP \Leftrightarrow u \in UKNAP$. Further, the transformation can be done in time bounded by a polynomial in the length of u. Thus, $P = NP$.

   (c) *1-of-3-SAT* is the set of 3-CNF Boolean formulas that are satisfiable by truth assignments making exactly *one* of the three literals in each clause true. Like 3-SAT, this problem is known to be *NP*-complete.

   Let f be a formula in conjunctive normal form with exactly 3 literals per clause (3CNF). Suppose it has variables $x_1,\ldots,x_m$, and clauses $c_1,\ldots,c_q$. Suppose "x_i" occurs in clauses numbered $i_1,\ldots,i_j$ and "$\overline{x}_i$" occurs in clauses numbered $i'_1,\ldots,i'_{j'}$. Let $a_i = \sum_{k=1}^{j} i_k$, and $\overline{a}_i = \sum_{k=1}^{j'} i'_k$. Calculate $a_r, \overline{a}_r$ for the other variables x_r similarly. Let $s = \sum_{i=1}^{q} i$. Generate the string:

   $$u = 1^{a_1}\#1^{\overline{a}_1}\#\ldots\#1^{a_m}\#1^{\overline{a}_m}\#1^s$$

   Now if f is satisfiable by an assignment that makes exactly one literal per clause true, i.e., if f is in *1-of-3-SAT*, then u is in *UKNAP*: Pick a_i or $\overline{a}_i$ depending on whether x_i is true or false respectively in the 1-of-3 satisfying assignment. Every clause is satisfied by exactly one literal, so the sum of the chosen $a, \overline{a}$'s is exactly s. Thus $u \in UKNAP$.

   Furthermore, the reduction can be done in time polynomial in the length of f; e.g., note that the numbers $a_i, \overline{a}_i$, and s are all of magnitude at most q^2, since each is the sum of at most q distinct numbers between 1 and q, so the length of u is $O(q^3) = O(|f|^3)$.

Thus $P = NP$.

(d) Proceed just as in part 6c, but if the formula is unsatisfiable, then output the fixed string 11#111, which is not in *UKNAP*. Again, we have *KNAP* $\leq_p$ *UKNAP*, thus $P = NP$.

(e) [**Extra Credit:**] Prove that *UKNAP* is in *P*.

(f) [**Extra Credit:**] Prove that *1-of-3-SAT* is *NP*-complete.

7. [**Extra Credit:]** 7.49