Homework 5: Solutions

Problem 1.

Show that if $NP \neq coNP$ then $P \neq NP$.

Answer

We can prove that given $NP \neq coNP$, $P \neq NP$ using the contrapositive: if $P = NP$ then $NP = coNP$.

Assume $P = NP$, then if $A \in NP$, A is also in P. Since A is in P, we can decide in polynomial time any member of A. We can reverse the output of the decider to decide the compliment (swith accept with reject). This gives up $\overline{A}$. $\overline{A} \in P$ and therefore $\overline{A} \in NP$. If $\overline{A} \in NP$, $A \in coNP$. Therefore, $NP = coNP$.

Problem 2.

Sipser 7.16

Answer

1. We modify the solution of PATH from Sipser to verify the path stopping after k iterations.

   On input $\langle G, s, t, k\rangle$ where G is an undirected graph with nodes s and t and a path at most length k.

   a. Place a mark on s.
   b. Repeat the following for at most k repetitions or until no additional nodes are marked.
   c. Scan all the edges of G. If an edge (s, t) is found going from a marked node s to an unmarked node t, mark t.
   d. If b is marked, *accept*. Otherwise *reject*.

   This takes polynomial time because it only needs to visit each node once (we have a simple path).

2. There are two parts to showing that $LPATH$ is NP-complete. First we must show that $LPATH \in NP$. Then we show a reduction from $UHAMPATH$ to $LPATH$.

   We can show $LPATH \in NP$ by giving a polynomial-time verifier. If we are given a solution (a path), we can verify this path in polynomial time. We do this by checking that our solution starts with a, ends with b, and traverses valid edges that connect nodes in our graph.

   Next, we do the reduction. We are given that $UHAMPATH$ is in NP. Taking an instace of $UMAMPATH$, $\langle G, s, t\rangle$, we construct an instance of $LPATH$: $\langle G, s, t, k = |G_{\text{nodes}}| - 1\rangle$. We set k to be the number of nodes in

Problem 3.

Sipser 7.17

Answer

1. Because A is in P and P = NP, A is in NP.
2. Because A is in P, we know there is a TM M_1 deciding A in polynomial time. Let L be an arbitrary language in NP. Because P = NP, there is a TM M_2 deciding L in polynomial time. We construct a machine M that, on input string w, runs M_2 on w. If M_2 accepts, M runs M_1 on a string w_1 in A and outputs the result, which is, of course, accept. If M_2 rejects, M runs M_1 on a string w_2 not in A and outputs the result, which is, of course, reject. We know that w_1 and w_2 exist because A is neither the empty set nor the set of all strings. M accepts iff w is in L and requires only polynomial time before running M_1, so M reduces L to A in polynomial time. Therefore A is NP-hard.

It follows from 1. and 2. that A is NP-complete.

Problem 4.

Sipser 7.26

Answer

We can place each card in the box in two possible orientations, the *normal* or the *reverse* orientation. We don't care about the relative positions of the cards in the box, that is we don't care if card A is on top of card B or the opposite, since in both cases the holes are covered in the same way.

First, we show that $PUZZLE$ is in NP. We construct a nondeterministc polynomial time Turing machine M that decides $PUZZLE$ as follows. Machine M on input $\langle c_1, \ldots, c_k \rangle$ chooses nondeterministically an orientation for each of the cards c_i and places the cards in the box. Then it checks to see if all the holes are covered, and if they are covered it accepts, othewise it rejects. The running time of machine M is polynomial in the number of holes and the number of cards.

Second, we show that every language in NP is polynomial time reducible to $PUZZLE$. To do that, we show that $3SAT$ is polynomial time reducible to $PUZZLE$. That is, for each 3cnf-formula ϕ, we show how to construct, in polynomial time, specific cards for the $PUZZLE$ problem, such that the $PUZZLE$ has a solution iff ϕ is satisfiable.

We construct the cards from the 3cnf-formula ϕ in the following way.

- For each variable x_i of ϕ we create a new card c_i.
- In each card, we divide the two columns in rows, such that each row corresponds to a clause of ϕ. (The number of rows is equal to the number of clauses of ϕ.)

- In card c_i, if literal x_i appears in clause j then we don't punch a hole in the left position of row j. Otherwise, if x_i doesn't appear in clause j then we punch a hole in the left position of row j.
- In card c_i, if literal $\neg x_i$ appears in clause j then we don't punch a hole in the right position of row j. Otherwise, if $\neg x_i$ doesn't appear in clause j then we punch a hole in the right position of row j.
- We also create a special card that has holes in all its left column, and no holes in its right column.

To show that this construction works, we argue that if ϕ is satisfiable then the $PUZZLE$ has a solution, and conversely, if $PUZZLE$ has a solution then ϕ is satisfiable.

Suppose that ϕ is satisfiable. Then we can solve the $PUZZLE$ as follows. Since ϕ is satisfiable, there exists an assignment to its variables that satisfies ϕ. For each variable x_i of ϕ, if x_i is 1, in the satisfying assignment, we put the corresponding card c_i in the box in its normal orientation, otherwise, if x_i is 0, we put the card in its reverse orientation. We also put the special card in its normal orientation. This placement of the cards covers completely the bottom of the box. Assume, for contadiction that this is not true, and therefore there exists a row j, that corresponds to clause j, through which you can view the left bottom of the box. For each card c_i with normal orientation, we have that its corresponding variable x_i has a hole in the left position of row j, which means that the variable appears either as $\neg x_i$, or not at all in clause j. Since c_i has normal orientation, we have that $\neg x_i = 0$ and therefore variable x_i does not satisfy clause j. Thus the variables of the normal oriented cards do not satisfy clause j. Using a similar reasoning we have that the variables of the cards with the reverse orientation do not satisfy the clause j either. Thus the initial assignment of the variables does not satisfy the formula, which is a contradiction, since we have that the assignment is satisfying.

Suppose now that the $PUZZLE$ has a solution. Then we show that ϕ is satisfiable, by constructing a satisfying assignment as follows. The special card, according to its orientation, does not cover one of the two columns. If it does not cover the left column, then for each row j there exists at least one card c_k that has its left position of row j with no hole. If the card c_k has normal orientation then we assign the value 1 to its variable x_k, otherwise if the orientation is reverse we assign the value 0. We repeat the same procedure for all the rows. If by the above procedure some of the variables have not been assigned values, we assign to them an arbitrary value 0 or 1. The procedure is similar when the special card has reverse orientation. This assignment is a satisfying one, since for each clause j (that corresponds to row j) there exists at least one variable whose literal x_i or $\neg x_i$ in the clause j has value 1. If the literal is x_i then its card c_i has normal orientation, otherwise if the literal is $\neg x_i$ then the its card c_i has reverse orientation, when the special card has normal orientation.

The above reduction takes polynomial time in the number of clauses and variables of a 3cnf-formula ϕ. This completes the proof.

Problem 5.

Sipser 7.34

Answer

See Theorem 9.3 (page 187), and Theorem 9.8 (page 198), in the Complexity book of Papadimitriou.