CSE431: Complexity Theory February 10, 2023

Homework 3

Anup Rao Due: February 19, 2023

Read the fine print[1]. Each problem is worth 10 points:

1. Recall the following function:

$$\mathbf{2COL}(G) = \begin{cases} 1, & \text{if graph } G \text{ has a coloring with two colors} \\ 0, & \text{otherwise,} \end{cases}$$

where a coloring of G with c colors is an assignment of a number in $[c]$ to each vertex such that no adjacent vertices get the same number.

Prove that $2\mathbf{COL} \in \mathbf{NL}$. You can use the following fact: A graph G can be colored with two colors if and only if it contains no cycle of odd length.

Solutions: First observe that there is an NL algorithm to detect whether or not a graph has an odd cycle:

(a) Use non-determinism to guess a number $k > 1$. Abort if k is even.
(b) For $i = 1, \ldots, k+1$:
  i. Guess v_i. If $i > 1$ check that (v_{i-1}, v_i) is an edge of the graph, abort if not.
(c) If $v_{k+1} \neq v_1$, abort.
(d) Accept the input.

If the graph has an odd cycle, then some sequence of non-deterministic steps will make the algorithm accept. If the graph does not have an odd cycle, then the algorithm cannot accept, because if $v_1, \ldots, v_k$ is a walk in the graph with $v_{k+1} = v_k$, then this walk can be broken up into a collection of simple cycles, but they cannot all be of odd length, or k would also be odd.

This algorithm shows that checking for odd cycles is in NL. Since $NL = coNL$, the same problem must be in $coNL$, which gives an algorithm for 2 coloring.

2. Consider the following game between two players: Given a directed graph $G = (V, E)$, and a start vertex s, the players (starting with Player 1) alternately choose an outgoing edge incident to the current vertex to reach a vertex that was not previously visited. If one of the

[1] In solving the problem sets, you are allowed to collaborate with fellow students taking the class, but **each submission can have at most one author**. If you do collaborate in any way, you must acknowledge, for each problem, the people you worked with on that problem. The problems have been carefully chosen for their pedagogical value, and hence might be similar to those given in past offerings of this course at UW, or similar to other courses at other schools. Using any pre-existing solutions from these sources, for from the web, constitutes a violation of the academic integrity you are expected to exemplify, and is strictly prohibited. Most of the problems only require one or two key ideas for their solution. It will help you a lot to spell out these main ideas so that you can get most of the credit for a problem even if you err on the finer details. Please justify all answers. Some other guidelines for writing good solutions are here: `http://www.cs.washington.edu/education/courses/cse421/08wi/guidelines.pdf`.

players cannot choose a next vertex, he loses. Let **GAME**(G) be the function that is 1 if and only if Player 1 has a strategy that ensures that she always wins no matter what Player 2 does.

Show that **GAME** is in **PSPACE**.

Solution: We give a recursive algorithm that uses a polynomial amount of space:

(a) For each vertex v of the graph let G_v denote the graph obtained by deleting s from G and setting v to be the start vertex of the new graph.
(b) To compute $GAME(G)$, recursively compute whether or not Player 2 has a strategy for winning the game on G_v if they start the game.
(c) If there is a v where Player 2 has no such strategy for winning, then output 1.

This algorithm has a recursion depth that is equal to the number of vertices in the graph, and each level of the recursion uses polynomial space. So the overall algorithm also uses polynomial space.

3. Let **PATH** be the following function:

$$\mathbf{PATH}(G, s, t, k) = \begin{cases} 1, & \text{if the shortest path between vertices } s \text{ and } t \text{ has length exactly } k \\ 0, & \text{otherwise.} \end{cases}$$

Prove that $\mathbf{PATH} \in \mathbf{NL}$.

Solution: Define the function

$$f(G, s, t, k) = \begin{cases} 1 & \text{if there is a path of length} \leq k \text{ from } s \text{ to } t \\ 0 & \text{otherwise.} \end{cases}$$

We have already seen that f is in NL. Since NL = coNL, f is also in coNL, and so $1 - f$ is in NL.

To compute PATH, use the following NL algorithm:

(a) Compute $f(G, s, t, k)$.
(b) If that accepts, compute $1 - f(G, s, t, k - 1)$ using the NL algorithm for $1 - f$.

Since both computations are in NL, the overall computation is in NL. Moreover, if the shortest path has length exactly k, then both computations must have an accepting path. If the shortest path is of a different length, one of the computations will never accept.

4. Suppose **TQBF** is also **PSPACE**-complete under log-space reductions—meaning that for every $f \in \mathbf{PSPACE}$, there is a logspace computable function h such that $f(x) = \mathbf{TQBF}(h(x))$. Prove that this implies that $\mathbf{TQBF} \notin \mathbf{NL}$. Hint: Use Savitch's theorem and one of the hierarchy theorems.

Solution: Under the assumptions, if TQBF is in NL, we get that PSPACE is contained in NL. But Savitch's algorithm implies that NL is contained in L^2. This contradicts the space hierarchy theorem, since PSPACE must be strictly larger than L^2.