CSE 200 - Computability and Complexity
Homework 3
Due Wednesday, November 15th, 11:59pm

Instructions: You may work individually or in a team of 2 people. You may switch teams for different assignments. Please ensure your name(s) and PID(s) are clearly visible on the first page of your submission, and then upload the PDF to Gradescope. If working in a group, submit only one submission per group: one partner uploads the submission through their Gradescope account and then adds the other group member to the Gradescope submission by selecting their name in the "Add Group Members" dialog box. You will need to re-add your group member every time you resubmit a new version of your assignment.

It is highly recommended (though not required) that you type your answers. It is your responsibility to make any handwriting clear and legible for grading. For ease of grading, please start each new problem on a separate page.

Problems:

1. **Graph problems in small space**

   For directed graph G, let $\text{dist}_G(s,t)$ be the length of the shortest path from s to t in G; or if no such path exists, then $\text{dist}_G(s,t) = \infty$. Consider the following variant of the st-connectivity problem where the path must be of some minimal length $d \in \mathbb{N}$:

   $$\text{MIN-ST-CON} := \{(G, s, t, d) \mid \text{dist}_G(s,t) = d\}.$$

   (a) Prove that MIN-ST-CON is in NL.

   (b) Prove that MIN-ST-CON is NL-hard. Hence MIN-ST-CON is NL-complete.

2. **Games are hard**

   In this problem, we will reason about the computational complexity of determining the winner of a game. Specifically, we will consider games that have the following properties:

   - *2-Player:* There are two players who alternate taking turns.
   - *Perfect information:* There is no concealed information, each player knows the set of moves of the other player.
   - *Finite:* Each player has a finite number of moves, and the game is gauranteed to end after a finitely many turns. Specifically, if the game has input description of size n, then the players have $\mathsf{poly}(n)$ options for each move, and the game ends after $\mathsf{poly}(n)$ moves.

   Furthermore, we assume that moves in the game can be computed in polynomial time, and when the game ends, it is easy to tell who won the game. That is, from the ending position, there is a polynomial time algorithm that computes who won (either the player who went first, or the one who went second).

   (a) Show that it is possible to determine the winner of such games in PSPACE.

It will turn out that most natural games of this kind are, in fact, PSPACE-complete.

Let's consider one such game called "Cartography". In this game, the input is a directed graph G and a starting vertex s. The game begins with a marker at vertex s. Each turn, the current player moves the marker along one of the outgoing edges from the current vertex. Importantly, each directed edge can be used at most once for the entire game. The first player unable to move loses.

Define the language CARTOGRAPHY to be the initial positions (G, s) such that Cartography played on graph G with starting marker at vertex s is a win for the first player.

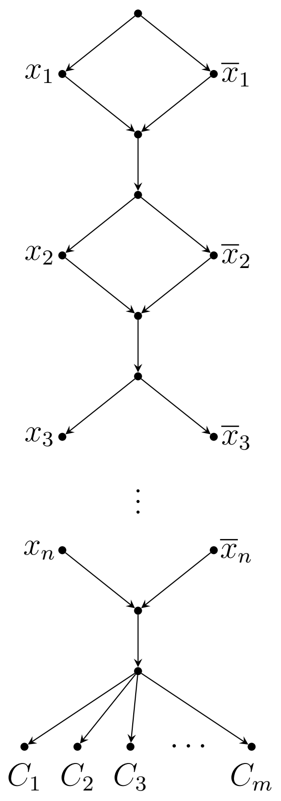
You will show that CARTOGRAPHY is PSPACE-hard. In particular, you will prove that

$$\text{TQBF} \leq_L \text{CARTOGRAPHY}.$$

Since this reduction is somewhat tricky, I will get you started. Start with TQBF instance

$$\exists x_1 \forall x_2 \ldots \exists x_n \; \varphi(x_1, x_2, \ldots, x_n),$$

where φ is a CNF formula with clauses $C_1, C_2, \ldots, C_m$. We map this TQBF instance to an instance of CARTOGRAPHY (G, s) where the first player wins iff the TQBF instance is satisfied. The graph G will have the form of the graph on the right, and some of the vertices are labeled suggestively. The vertex set is exactly correct, but some edges are missing. Your goal is the finish the proof.



(b) Complete the proof that TQBF $\leq_L$ CARTOGRAPHY. You should not only describe the edges missing in the graph above, but also explain why the reduction is correct and in log-space.

3. **PSPACE does not have fixed poly-size circuits**

It is widely believed that polynomial-size circuits cannot compute all functions requiring polynomial space, i.e., PSPACE $\not\subseteq$ P/poly. However, this is not known. In this problem, you will prove a weaker version of this statement: PSPACE $\not\subseteq$ SIZE(n^k) for any fixed k.

For every k, we will construct a language $L_k \subseteq \{0,1\}^*$ with the following two properties:

- $L_k \in$ PSPACE.
- There exists n_k such that for all $n > n_k$, the language $L_k \cap \{0,1\}^n \not\subseteq$ SIZE(n^k).

We break the proof into the following pieces:

(a) For every n, let $\mathcal{F}_n$ be the class of functions $f\colon \{0,1\}^n \to \{0,1\}$ which can be computed by circuits of size n^k. Prove that $|\mathcal{F}_n| \leq 2^m$ for $m = O(n^{k+1})$.

(b) Let $t \geq 1$ and fix distinct inputs $x_1, \ldots, x_t \in \{0,1\}^n$. Prove that there exists outputs $y_1, \ldots, y_t \in \{0,1\}$ such that the number of functions $f \in \mathcal{F}_n$ that satisfy $f(x_i) = y_i$ for all $i \in [t]$ is at most 2^{m-t}.

(c) Prove that for $t = m+1$, there are inputs $x_1, \ldots, x_t \in \{0,1\}^n$ and outputs $y_1, \ldots, y_t \in \{0,1\}$ such that any function $f\colon \{0,1\}^n \to \{0,1\}$ which satisfies $f(x_i) = y_i$ for all $i \in [t]$ must be outside of $\mathcal{F}_n$.

(d) Prove there exists an algorithm that on input 1^n *finds* such inputs and outputs using $\mathsf{poly}(m)$ space.

(e) Complete the proof. That is, describe the language L_k and show why it has the desired properties.