Solutions of homework 9

1. Show that NL is closed under star.

**Solution Outline:** (25 points)

Consider any language $L \in NL$ and the corresponding nondeterministic log-space machine $M$ that decides $L$. Now, consider the following nondeterministic log-space machine that decides $L^*$: on input $w = w_1 \ldots w_n$ of length $n$, set $i = 1$ and nondeterministically choose $j \in \{1, 2, \ldots, n\}$. If $j < i$, reject. Write both $i$ and $j$ on the work tape. Simulate $M$ on $w_i \ldots w_j$ (while making nondeterministic guesses for the simulation). If $j < n$, replace $i$ on the work tape with $j + 1$ and repeat with a new value of $j$, etc. Accept if there are nondeterministic choices for the values of $j$ and for the nondeterministic choices of $M$ such that each invocation of $M$ accepts.

2. Sipser problem 8.8. Prove that the Acceptance problem for NFA is NL-complete. In addition, prove that the Acceptance problem for DFA is in L.

**Solution Outline:** (30 points)

To see that $A_{DFA} \in L$, observe that in the simulation of the DFA on the input, we only have to store the current state of the DFA on the work tape. To see that $A_{NFA} \in NL$, we still merely simulate the NFA on the input, having only to store the current state of the NFA on the work tape, and using the nondeterminism of the log-space machine to make nondeterministic choices for the NFA.

To prove that $A_{NFA}$ is NL-complete, it suffices to prove $PATH \leq_L A_{NFA}$. Given an instance $(G, s, t)$ for $PATH$, the reduction produces the instance $(N, \varepsilon)$ for $A_{NFA}$, where $N$ is an NFA defined as follows: start space are the vertices of $G$, start state is $s$, accept state is $\{t\}$ and alphabet is $\{0\}$. For each directed edge $(u, v)$ in $G$, there is an $\varepsilon$-transition from $u$ to $v$ in $N$. It is easy to see that this reduction can be computed using no more than a logarithmic amount of space.

(7 points for $A_{DFA} \in L$, 8 points for $A_{NFA} \in NL$, 15 points for $A_{NFA}$ being NL-hard.)

3. (a) Show that MAX-CLIQUE $\in$ PSPACE.

(b) Explain why the following argument fails to show that MAX-CLIQUE $\in$ coNP: To show that $(G, k) \notin$ MAX-CLIQUE, it suffices to demonstrate the existence of a larger clique in $G$ of size greater than $k$, so the NP algorithm for MAX-CLIQUE just guesses the larger clique.

**Solution Outline:** (15,5 points)

(a) In polynomial space, we can enumerate over all subsets of vertices of the graph and check whether it forms a clique. This allows us to figure out the size of the largest clique.

(b) The clause “it suffices to demonstrate the existence of a larger clique in $G$ of size greater than $k$” is incorrect. It could be that $(G, k) \notin$ MAX-CLIQUE because the largest clique in $G$ has size $k - 1$, say.
4. **Sipser problem 8.18 (DFACHAIN ∈ PSPACE).**

**Solution Outline:** (25 points)

It suffices to show that $DFACHAIN \in \text{NPSPACE}$. The idea is as follows: given $(M, s, t)$, reject if $|s| \neq |t|$. Otherwise, consider a graph $G$ of exponential size whose vertices are indexed by strings in $\Sigma^{|s|}$, and there is a directed edge from $w_1$ to $w_2$ iff $w_1$ and $w_2$ differ in exactly one character, and $w_1, w_2 \in L(M)$. Then, $(M, s, t) \in DFACHAIN$ iff there is a path from $s$ to $t$ in $G$. This we can check in $\text{NPSPACE}$ by guessing the path (akin to the $\text{NL}$ algorithm for $PATH$), and at each step, storing only the name of current vertex (which is a string in $\Sigma^{|s|}$). To guess the path, at vertex $w_1$, we will nondeterministic select a new vertex $w_2$ that differs from $w_1$ in exactly one character, and verify that $M$ accepts $w_2$. (We can ensure that the machine always halts by keeping a counter and incrementing it with each guess, and rejecting when the counter hits $|\Sigma^{|s|}$. This is because if there exists a chain from $s$ to $t$, then there exists one of length at most $|\Sigma^{|s|}$ by removing loops.)