CS5230
Tutorial 3: Answer sketches

Q1 (sketch): Suppose M witnesses that $L \in NSPACE(S(n))$. Without loss of generality assume there is only one accepting ID of M on input x. We will then show below that $\overline{L}$ is in $NSPACE(S(n))$. This would imply $NSPACE(S(n)) \subseteq coNSPACE(S(n))$, and thus, $NSPACE(S(n)) = coNSPACE(S(n))$.

For any input x, construct a graph G as follows. The vertices of G are all the possible IDs of M (where for the input tape, we only consider head location). There is an edge from ID_1 to ID_2 iff there is a one step transition from ID_1 to ID_2 (for M). Thus, in the graph, we can test whether there is an edge between two vertices (using space proportional to the space used by two vertices).

Now, $x \notin L$ iff there is no path from startID to AcceptingID. As done in Immerman-Szelepscenyi result, this can be determined in nondeterministic space s, where s is the space for representing each vertex (each ID).

Each vertex can be represented using space:

- $O(S(n))$ for contents of the working tapes and head location
- $O(\log n)$ for head location on the input tape
- $O(1)$ for state of the machine M

Thus, we have that $\overline{L}$ is in $NSPACE(O(S(n))) = NSPACE(S(n))$.

Q2. (sketch) Follows from Q1 as context sensitive languages are exactly the languages in $NSPACE(n)$.

To show that a context sensitive language is in $NSPACE(n)$ do as follows. On input w, first lay down $|w|$ space. Then start with the start symbol S, of the context sensitive grammar, and do a derivation guessing the productions used in the derivation. If w can be derived in this fashion then accept.

To show that a language L in $NSPACE(n)$ is a CSL, construct a grammar as follows. Suppose M witnesses that L is in $NSPACE(n)$. Since we are considering linear space, we may assume that the input tape is read/write and the only tape used by M. Furthermore, assume that M, on any input of length n, has unique accepting ID $q_f(\#)^n$, where q_f is the only accepting state. For ease of presentation, we give a grammar to generate strings of the form $\$_l w \$_r$, where w is in L; here $\$_l$ and $\$_r$ stand for left-end and right-end markers respectively (one can get rid of $\$_l$ and $\$_r$ by using appropriate coding). We also assume without loss of generality that M does not move to the left of $\$_l$ or right of $\$_r$ during its computation.

Besides the terminals (which are symbols used by M), in the grammar we have nonterminals S and p^a for each state p and symbol a used by M. Intuitively, p^a means that the machine is in state p and reading the symbol a. The IDs of M are represented in the form $\Gamma^* p^a \Gamma^*$. We intend to do the "reverse" simulation of M to derive the original string from S.

The following productions are used by the grammar:

- S is the starting symbol which derives $q_f^{\#}\#^*$ (via productions of the form $S \rightarrow S\#$, $S \rightarrow q_f^{\#}$). Note that $q_f^{\#}\#^*$ is accepting ID.

- For transitions $\delta(q,b) = (p, b', R)$, in M, we have a production of the form $b'p^c \to q^b c$, for all c in the alphabet.
- For transitions $\delta(q,b) = (p, b', L)$, in M, we have a production of the form $p^a b' \to a q^b$, for all a in the alphabet.
- We also have a production of the form $q_0^{\$_l} \to \$_l$.

It can now be easily verified that M accepts w iff the above grammar has a derivation of the form $S \Rightarrow^* q_f^{\#} \#^{|w|+1} \Rightarrow^* q_0^{\$_l} w \$_r \Rightarrow \$_l w \$_r$.

Remark: If one wants to generate w rather than $\$_l w \$_r$ as above, one could code $\$_l$ with the first symbol of w and $\$_r$ with the last symbol of w and get rid of them at the end using special production rules.

In above, we will only generate strings of length at least 2 in the language. Other strings of length ≤ 1 can be generated using separate rules (of the form $S' \to S$ and $S' \to x$, where x denotes string of length one in the language).

Q3 (sketch): Basically the same proof as done in class for non-deterministic space works. For deterministic cases, use i as $f(n) - n$.

Q4: $NSPACE(n^2) \subseteq DSPACE(n^4)$ by Savitch's theorem.

$DSPACE(n^4) \subset DSPACE(n^5)$ by space hierarchy theorem.

$DSPACE(n^5) \subseteq NSPACE(n^5)$ by definition.

Thus, $NSPACE(n^2) \subset NSPACE(n^5)$.

Q5. Clearly, NSPACE(n^3) $\subseteq$ NSPACE(n^5). Suppose by way of contradiction that

NSPACE(n^5) $\subseteq$ NSPACE(n^3) — (2)

Let $f(n) = \lfloor n^{1.5} \rfloor$. Note that $f(n)$ is fully space constructible. Thus, using translation lemma and (2), we get

NSPACE($(\lfloor n^{1.5} \rfloor)^5$) $\subseteq$ NSPACE($(\lfloor n^{1.5} \rfloor)^3$).

But then, using (2) we have

DSPACE(n^7) $\subseteq$ NSPACE(n^7) $\subseteq$ NSPACE($(\lfloor n^{1.5} \rfloor)^5$) $\subseteq$ NSPACE($(\lfloor n^{1.5} \rfloor)^3$) $\subseteq$ NSPACE(n^5) $\subseteq$ NSPACE(n^3) $\subseteq DSPACE(n^6)$.

(where the last $\subseteq$ is due to Savitch's theorem).

But, this contradicts space hierarchy theorem as $\lim_{n\to\infty} \frac{n^6}{n^7} = 0$.