CS–172 Computability & Complexity, Spring 2021

Homework 11 Solutions

Note: These solutions are not necessarily model answers. Rather, they are designed to be tutorial in nature, and sometimes contain more explanation than is required for full points. Also, bear in mind that there may be more than one correct solution. The maximum total number of points available is 33.

1. (a) Let $f(n)$ and $g(n)$ be any two functions such that $f(n) \geq (\log n)^2$ and $g(n) = o(\sqrt{f(n)})$. Then we have the following chain of inclusions: *5pts*

$$\text{NSPACE}(g(n)) \subseteq \text{SPACE}(g(n)^2) \subset \text{SPACE}(f(n)) \subseteq \text{NSPACE}(f(n)).$$

Here the symbol $\subset$ denotes *strict* inclusion, i.e., $A \subset B$ means that B is strictly larger than A. Thus we see that $\text{NSPACE}(f(n))$ contains a language that is not in $\text{NSPACE}(g(n))$, which is what we wanted.

We can justify the above chain of inclusions as follows. The first inclusion is by Savitch's Theorem. The second (strict) inclusion is by the Space Hierarchy Theorem. And the third inclusion is trivial.

[*Some students missed the (obvious) fact that* $\text{SPACE}(f(n)) \subseteq \text{NSPACE}(f(n))$. *Instead, they used Savitch's theorem to deduce that* $\text{NSPACE}(g(n)) \subseteq \text{SPACE}(g(n)^2)$ *and* $\text{NSPACE}(f(n)) \subseteq \text{SPACE}(f^2(n))$, *and the Space Hierarchy Theorem to deduce that* $\text{SPACE}(g(n)^2) \subset \text{SPACE}(f(n)^2)$. *But this doesn't imply anything about the relationship between* $\text{NSPACE}(g(n))$ *and* $\text{NSPACE}(f(n))$*: for sets* A, B, C, D *with* $A \subseteq B$ *and* $C \subseteq D$, *it is not the case that* $B \subset D$ *implies* $A \subset C$.]

(b) The key point you must identify is that at the heart of the Space Heirarchy theorem is a space efficient diagonalization where we build a language different from anything a machine with $g(n)$ space can recognize, by ensuring that for every such machine M, there is some string (namely $\langle M \rangle 10^k$, for some k) which is in the language iff it is not in the language of M. This is done by essentially watching how that machine behaves on the string and then doing the opposite of what the machine does. This "complementation" is possible so long as the machine being simulated is deterministic, because then its fine to just do the opposite of what the machine does. If however the machine is nondeterministic, then we can't just "flip" the output – as to know if the machine rejects we must simulate it along all possible paths and its not clear how to do that within the given space bound ($f(n)$). Note though, that it is possible to know when the machine accepts because the simulator itself is nondeterministic – so if the machine its simulating has an accepting path, then the simulator can guess that path as well. *3pts*

(c) The way to adapt the proof is to first build a language that agrees with every $g(n)$ space machine on some particular input, namely the string representation of the $g(n)$ space machine. This is done by simulating that machine using the extra $f(n)$ space (mimicking its nondeterminism) – and then doing exactly what that machine did – if the machine accepts along its guessed path, then the simulator also accepts – and if along the guessed path the machine rejects then reject. This way, we build a nondeterministic machine B using $f(n)$ space whose language $L(B)$ is such that for every machine M in $\text{NSPACE}(g(n))$ we have some string (namely $\langle M \rangle 10^k$, for some k) on which B and M agree. Now we can appeal to the Immerman-Szelepcsényi theorem which tells us that $\overline{L(B)}$ is also in $\text{NSPACE}(f(n))$. However, this language $\overline{L(B)}$ is such that for every machine M in $\text{NSPACE}(g(n))$ there is a string (namely the same $\langle M \rangle 10^k$) that is in $\overline{L(B)}$ iff the string is *not* in $L(M)$. Thus for every nondeterministic $g(n)$ space bounded machine M, we have that $L(M) \neq \overline{L(B)}$. Hence $\overline{L(B)}$ is a language that's in $\text{NSPACE}(f(n))$ but not in $\text{NSPACE}(g(n))$. *5pts*

2. (a) Following the hint, we consider any language $L \in$ PSPACE. By definition, there exists a TM M that decides L in n^k space.[1] Now consider the language L_{pad} as defined. We can build a TM M' for L_{pad} that runs in linear space as follows: M' first verifies that the input is in the correct format $x\$^{|x|^k}$. If so, then M' runs M on x (else M rejects). M will take space $|x|^k$, but this is linear in the length of the input to M' and hence $L_{\text{pad}} \in \text{SPACE}(n)$. *6pts*

Now suppose $\text{SPACE}(n) \subseteq \text{P}$. We show that L is in P and hence $\text{P} = \text{PSPACE}$. Since $L_{\text{pad}} \in \text{SPACE}(n)$, by hypothesis we have a polynomial time algorithm for L_{pad}. From this we can build a polynomial time algorithm for L as follows: On input x, construct the string $x\$^{|x|^k}$ (taking time n^k), and feed this into the polynomial time algorithm for L_{pad}. This gives us what we need.

(b) Suppose $\text{P} = \text{SPACE}(n)$. Then in particular $\text{SPACE}(n) \subseteq \text{P}$, and from part (a), $\text{P} = \text{PSPACE}$. But since we've assumed that $\text{SPACE}(n) = \text{P}$, we get that $\text{SPACE}(n) = \text{PSPACE}$. But the space hierarchy theorem tells us this is false ($\text{SPACE}(n)$ is *strictly* contained in (say) $\text{SPACE}(n^2)$, which is within PSPACE) so we have a contradiction and thus $\text{P} \neq \text{SPACE}(n)$. *4pts*

3. (a) The fallacious line in this "proof" is the following: "Because every language in NP is poly time reducible to SAT, this implies that $\text{NP} \subseteq \text{TIME}(n^k)$". The fly in the ointment is the fact that there could be languages in NP where the reduction itself takes time greater than $O(n^k)$ and so the time taken to decide that language would be greater than $O(n^k)$ (since it's the sum of the time for the reduction and the time to solve the resulting SAT instance), so we can't conclude that NP is contained in $\text{TIME}(n^k)$, which breaks the proof. *5pts*

(b) The problem with this "proof" is the second part of the statement "... any language L in PSPACE can be reduced in polynomial time to TQBF, and hence to TQBF_k for some k". Although it is true that any language in PSPACE can be reduced to TQBF, it is *not* true that the number of alternating quantifiers in this reduction can be bounded by any fixed k: indeed, you can check that the boolean formula produced in the reduction proving that TQBF is PSPACE-complete has a number of quantifier alternations that is polynomial in the input size for the language L, and hence unbounded. *5pts*

[*Other than the error pointed out in the solution to Q1(a) above, there were no other common errors in this HW. Please consult comments on your individual solutions for explanations of specific errors you may have made.*]

[1] Strictly speaking M requires $O(n^k)$ space; but by compressing multiple tape squares into one, we can assume that the leading constant in the big-O expression is 1.