CS 254 Homework 3

SJSU Students

November 13, 2006

Problem 1

For this problem, define the following two classes for any proper function f:
$H_f^T = \{\langle M, x\rangle | M \text{ accepts input } x \text{ after at most } f(|x|) \text{ steps}\}$
$H_f^S = \{\langle M, x\rangle | M \text{ accepts input } x \text{ after using at most } f(|x|) \text{ space}\}$
I will give a specific f for each part of the problem.

(a) Let $f(n) = 2^{2^n}$. H_f^T is decidable, since a Universal Turing machine with no time bounds can simulate any machine-input pair for 2^{2^n} steps to decide whether or not it is in H_f^T. The proof of the time hierarchy theorem tells us that $H_f^T \notin \text{TIME}(f(\frac{n}{2})) = \text{TIME}(2^{2^{\frac{n}{2}}})$. This class contains EXP (since the exponential term in the exponent grows faster than any polynomial term in the exponent), and so $H_f^T \notin$ EXP.

(b) Let $f(n) = 2^{n^2}$. From the proof of the time hierarchy theorem, we know that $H_f^T \in \text{TIME}(f(n)^3) = \text{TIME}(2^{3n^2})$. We can see that $\text{TIME}(2^{3n^2}) \subseteq \text{TIME}(2^{n^3}) \subseteq$ EXP, and so $H_f^T \in$ EXP. Furthermore, the proof of the time hierarchy theorem tells us that $H_f^T \notin \text{TIME}(f(\frac{n}{2})) = \text{TIME}(2^{\frac{n^2}{4}})$. This class contains E (since the quadratic term in the exponent grows faster than any linear term in the exponent), and so $H_f^T \notin$ E.

(c) Let $f(n) = 2^n$. From the proof of the time hierarchy theorem, we know that $H_f^T \in \text{TIME}(f(n)^3) = \text{TIME}(2^{3n})$. Since $\text{TIME}(2^{3n})$ is clearly contained in E, we know that $H_f^T \in$ E. Furthermore, the proof of the time hierarchy theorem tells us that $H_f^T \notin \text{TIME}(f(\frac{n}{2})) = \text{TIME}(2^{\frac{n}{2}})$. This class contains P (since an exponential term grows faster than any polynomial term), so we know that $H_f^T \notin$ P.

(d) Let $f(n) = n$. Notice that the first lemma in the proof of the time hierarchy theorem (from the lecture notes) can be modified to show that $H_f^S \in \text{SPACE}(f(n)) = \text{SPACE}(n)$. To do this, we construct the same Universal Turing machine described in the lemma, and perform the same simulation. The only difference is that we use the "alarm clock" tape to keep track of how many tape squares the simulation has used, rather than how many steps the simulation has taken. Notice that the initial setup of the simulation takes O($f(n)$) time, and thus it must use O($f(n)$) space, since a Turing machine cannot use more space than time. We also need to be careful of how many steps our 1-tape simulation of a k-tape machine takes; however, it is clear from the theorem which defines such a simulation that the space complexity is roughly $k(f(|x|))$. Thus, the total space used is $O(f(n))$, which gives $H_f^S \in$ SPACE$(n) \subseteq$ PSPACE as desired.

To show that $H_f^S \notin$ L, we use a proof similar to that of the second lemma in the proof of the time hierarchy theorem. Specifically, we suppose that $M_{H_f^S}$ decides H_f^S while using only $f(\frac{n}{2})$ space, and we construct a diagonalizing machine D_M that computes $D_M(\langle M_0 \rangle)$*: if* $M_{H_f^S}(\langle M_0, M_0 \rangle)$ *accepts then reject; else accept.* By the same argument in the proof from the lecture notes, the computation $D_M(\langle D_M \rangle)$ gives a contradiction, so $M_{H_f^S}$ cannot exist, and so $H_f^S \notin$ SPACE$(f(\frac{n}{2}))$ = SPACE$(\frac{n}{2})$. This class contains L (since a linear term grows faster than a logarithmic term), and so $H_f^S \notin$ L.

Problem 2

The proof is by contradiction. Let k be some number greater than 0 such that NP = SPACE(n^k). Since we know that NP is closed under polynomial-time reductions, it must be the case that SPACE(n^k) is as well. Let $L \in$ SPACE(n^{2k}) be some language that is not in SPACE(n^k); we know such a language exists because of the space hierarchy theorem. Now let $L' = \{(x, \alpha^{|x|^2}) | x \in L\}$, where α is just some arbitrary symbol. Notice that $L' \in$ SPACE(n^k), since the machine that decides L in SPACE(n^{2k}) will decide L' in SPACE(n^k) as long as it ignores the αs. However, there is a trivial polynomial-time reduction from L to L' (on input x, just output $(x, \alpha^{|x|^2})$), which implies that L is in SPACE(n^k) under the assumption that SPACE(n^k) is closed under polynomial-time reductions. Since we know $L \notin$ SPACE(n^k) from the construction of L, we have a contradiction, and thus there is no k greater than 0 such that NP = SPACE(n^k).

Problem 3

As pointed out on the discussion board, there is a bug in the solution below. There is a 1pt bonus for saying what the bug is and correcting it.

First, define the following four classes. Note that in each, M is a Turing machine, x is an arbitrary input string, and k is a natural number.
$H_P = \{\langle M, x, k \rangle | M$ *is a DTM accepts* x *in at most* $|x|^k$ *steps* $\}$
$H_{NP} = \{\langle M, x, k \rangle | M$ *is an NTM that accepts* x *in at most* $|x|^k$ *steps* $\}$
$H_{PSPACE} = \{\langle M, x, k \rangle | M$ *is a DTM accepts* x *after using at most* $|x|^k$ *space* $\}$
$H_{NL} = \{\langle M, x, k \rangle | M$ *is an NTM that accepts* x *after using at most log*$(|x|^k)$ *space* $\}$

Claim 1: H_P is P-complete under log-space reductions.
Proof: First, we must show that $H_P \in$ P. To see this, notice that n^k is a proper complexity function, so can be computed on input x in time $O(|x|^k)$. We can use this function as a clock on a Universal Turing machine that simulates M on x for $|x|^k$ steps. We do this using the same simulation used in the proof of the first lemma from the proof of the time hierarchy theorem from the lecture notes. This can be done from the notes in polynomial time in $|x|$, and so $H_P \in$ P. Now let L be any language in P. To complete the proof, we must show that L is log-space reducible to H_P. Let M_L be the machine that decides L in time n^k for some natural number k. Define R_L, the machine that performs

the reduction, as follows:

On input x:

1. Write an encoding of M_L on the output tape.
2. Write x on the output tape.
3. Write the binary representation of k on the output tape.

Notice that steps 1 and 2 require no space on R_L's work tape, because we can "hard wire" R_L to write down the encodings for these steps. Similarly, step 2 is just copying from the input tape to the output tape, which also requires no work tape space. So, R_L actually performs a constant-space reduction (which is still a log-space reduction). Now, for any $x \in$ L, M_L will accept x in at most $|x|^k$ steps (by definition), and so $R_L(x) \in H_P$. Likewise, if $R_L(x) \in H_P$, this means that M_L will accept x in at most $|x|^k$ steps, and so $x \in L$. Thus, the reduction is sound and complete, and so H_P is P-complete.$\square$

Claim 2: H_{NP} is NP-complete under log-space reductions.
Proof: $H_{NP} \in$ NP for the same reasoning as in Claim 1. The only difference is that the UTM must take non-deterministic steps when simulating one step of M, but this does not add any time complexity. Also, the log-space reducer R_L is exactly the same, and the reduction is valid for the same reasons; note that R_L does *not* need to be non-deterministic, and is still constant space.$\square$

Claim 3: H_{PSPACE} is PSPACE-complete under log-space reductions.
Proof: $H_{PSPACE} \in$ PSPACE for the same reasoning as in Claim 1. The only difference is that we're interested in the space complexity of the UTM rather than the time complexity, but this is still polynomial. We have the UTM mark off $|x|^k$ many tapes on its main tape (this could be done, for instance, by replacing spaces on the main tape with a new kind of space symbol with a dot under it). If the simulation ever goes off this marked off area the machine rejects, we now no longer use the "alarm clock" that we did in the first two parts above. Again, the log-space reducer is the same as in Claim 1, and the reduction is valid for the same reasons.$\square$

Claim 4: H_{NL} is NL-complete under log-space reductions.
Proof: For this proof, the reduction is the same as in the previous proofs, but we need to be a little more careful with showing that $H_{NL} \in$ NL. The non-deterministic Universal Turing machine that decides if $\langle M, x, k\rangle \in H_{NL}$ has 4 tapes. The input and output tapes are exactly the same as M's. One tape is used to keep track of M's state, and this is clearly logarithmic in $|\langle M\rangle|$. The final tape keeps track of M's work tape (we are assume without loss of generality that M is a one tape machine). Each of these tapes has maximum length at most $\log(|x|^k)$, and so the simulator's final tape has length O($c\log(|x|^k)$) = O($\log(|x|^{ck})$). Thus, H_{NL} is definitely in NL, and so this class is NL-complete.$\square$