Additional Midterm Review Questions

April 2, 2012

1. For each of the following statements, state whether it is true or false and provide a brief explanation:
   (a) The intersection of finitely many regular languages is regular
   (b) The intersection of infinitely many regular languages is regular
   (c) The intersection of finitely many context-free languages is context-free
   (d) The intersection of infinitely many context-free languages is context-free

2. Let \( FRANKENSTEIN(L_1, L_2) = \{w = a \circ b | \exists x, y \in L_2 \text{ s.t. } a \circ x \in L_1, y \circ b \in L_2\} \).
   For example, if \( L_1 = \{0^*1^*\} \) and \( L_2 = \{1^*0^*\} \), then \( FRANKENSTEIN(L_1, L_2) = \{0^*1^*0^*\} \).
   Show that the regular languages are closed under \( FRANKENSTEIN \).

3. Let \( L = \{\langle M \rangle | M \text{ accepts at least one palindrome}\} \).
   Show that \( L \) is undecidable.

4. Let \( L = \{\langle M \rangle | M \text{ accepts only palindromes}\} \).
   Show that \( L \) is unrecognizable.

5. For each of the following languages, prove whether it is decidable, recognizable but not decidable, or not recognizable:
   (a) \( L_1 = \{\langle M \rangle | M \text{ does not accept any string ending in } 0\} \)
   (b) \( L_2 = \{\langle M, x \rangle | M \text{ halts on } x \text{ with the tape-head in the leftmost position}\} \).
   (This question refers to Turing machines with tape that is infinite only on the right side.)

6. For each of the following languages, determine whether the language is decidable and whether it is recognizable, and prove your answer.
   (a) \( L_3 = \{\langle M_1, M_2 \rangle | M_1() \text{ or } M_2() \text{ accepts (or both)}\} \)
   (b) \( L_4 = \{\langle M \rangle | L(M) \text{ is infinite and } \overline{L(M)} \text{ is also infinite}\} \)

7. Let \( L_1 \) and \( L_2 \) be two Turing-recognizable languages with the additional property that \( L_1 \cup L_2 = \{0, 1\}^* \) and \( L_1 \cap L_2 \neq \emptyset \).
   Show that \( L_1 \preceq_T (L_1 \cap L_2) \).

8. Prove that \( \mathbf{NP} \) is closed under intersection, that is, if \( L_1, L_2 \in \mathbf{NP} \) then \( L_1 \cap L_2 \in \mathbf{NP} \).
9. Given a Turing machine $M$, we define $\overline{M}$ to be the Turing machine obtained by switching every accepting state of $M$ with a rejecting state, and vice-versa. Prove or give a counter-example to the following statement:

“If $M$ is a nondeterministic TM that halts on all inputs, then $L(M) = L(\overline{M})$.”
Solutions

1. (a) **True.** Let \( L_1, \ldots, L_k \) be finite languages recognized by DFAs \( A_1, \ldots, A_k \) respectively. By De-Morgan’s laws, \( L_1 \cap \ldots \cap L_k = \overline{L_1 \cup \ldots \cup L_k} \). We know that regular languages are closed under complement, so each \( \overline{L_i} \) is regular for all \( i \), and we also know that regular languages are closed under union, so \( \overline{L_1 \cup \ldots \cup L_k} \) is also regular. (Strictly speaking, we only know that the union of two regular languages is regular. But that means \( \overline{L_1 \cup \overline{L_2}} \), which in turn means \( \overline{(L_1 \cup \overline{L_2}) \cup \overline{L_3}} \) is regular, and so on.) Finally we take the complement again, which shows that \( L_1 \cap \ldots \cap L_k = L_1 \cup \ldots \cup L_k \) is regular.

(b) **False.** For example, we can build the language \( L = \{0^k1^k \mid k \geq 0\} \) by taking the intersection of infinitely many languages \( L_{n,m} \) for \( n, m \in \mathbb{N} \) where \( n \neq m \) and \( L_{n,m} := \{w \in \{0, 1\}^* \mid w \neq 0^n1^m\} \). In other words, we rule out one by one each word that is not in the format \( 0^n1^m \).

Each \( L_{n,m} \) is regular, because we only need to check if the input matches one specific word \( 0^n1^m \) (there is only one value of \( n, m \) that we need to check!). To recognize \( L_{n,m} \) we can build a DFA that has \( n + m + 2 \) states: one special state \( q\text{good} \), plus \( n + m + 1 \) states \( q_0, q_00, \ldots, q_0^n, q_0^{n+1}, \ldots, q_0^{n+m-1}, \ldots, q_0^{m-1} \). The initial state is \( q_0 \), and all the states are accepting except for \( q_0^{n+m} \). The transitions are:

- From state \( q\text{good} \) we only have a self-loop back to \( q\text{good} \);
- From state \( q_{0j} \) for \( j < n \), when we read 0 we go to \( q_{0j+1} \) and when we read 1 we go to \( q\text{good} \);
- From state \( q_{0^{n+1}j} \) for \( j < i \), when we read 1 we go to \( q_{0^{n+1}j+1} \) and when we read 0 we go to \( q\text{good} \);
- From state \( q_{0^{n+m}} \) we go to \( q\text{good} \) on reading either 0 or 1.

When we take the intersection \( \bigcap_{n,m} L_{n,m} \) we get \( \{0^k1^k \mid k \geq 0\} \), because only words of the form \( 0^k1^k \) belong to all the languages \( L_{n,m} \).

Another way to show this: by De-Morgan’s laws, if regular languages are closed under the intersection of infinitely many languages, then they are also closed under the union of infinitely many languages. But \( \{0^k1^k \mid k \geq 0\} = \bigcup_{n \in \mathbb{N}} \{0^n1^n\} \), which is a contradiction because \( \{0^k1^k \mid k \geq 0\} \) is not regular but each languages \( \{0^n1^n\} \) (which just has one word) is regular.

(c) **False.** Context-free languages are not closed under intersection of even two languages, as we saw in problem set 2.

(d) **False,** as in the regular case. For example, \( \bigcap_{n,m,k: n \neq m \text{ or } m \neq k} \{w \in \{0, 1\}^* \mid w \neq 0^n1^m2^k\} = \{0^n1^s2^s \mid s \geq 0\} \) is not context-free, but each language in the intersection is regular and therefore also context-free.

2. Given two regular languages \( L_1 \) and \( L_2 \), let \( D_1 \) and \( D_2 \) be the respective DFAs that recognize them. Consider all the states in \( D_1 \) that lie on a path between its start state and one of its
accepting states, inclusive of the start and accepting states — call these the included states. The string that is read between the start state of $D_1$ and any of its included states is a prefix of some string in $L_1$ (i.e. it corresponds to the “a” in the FRANKENSTEIN definition). Meanwhile, the string that is read between any included state in $D_2$ and any of its accepting states is a suffix of some string in $L_2$ (i.e. it corresponds to the $b$ in the FRANKENSTEIN definition). So to create a recognizer for FRANKENSTEIN($L_1$, $L_2$) we simply need to draw $\varepsilon$-transitions between each of the included states in $D_1$ and each of the included states in $D_2$. This new NFA will recognize FRANKENSTEIN($L_1$, $L_2$).

3. To show that $L$ is undecidable, we give a reduction from $HALT$. Given $(\langle M \rangle, w)$ as input, we construct a new machine $M'(x)$ that first runs $M(w)$ and then if and when that has halted, accepts if $x$ is a palindrome. Thus, we query the oracle for $L$ about $M'$; $M' \in L$ if and only if $(\langle M \rangle, w) \in HALT$.

4. To show that $L$ is unrecognizable, we first consider $\overline{L}$. $\overline{L}$ is the language of all Turing machines that accept at least one string that is not a palindrome. $\overline{L}$ is undecidable — to see this, we can give the same reduction from $HALT$ as we did in problem #?? above, except that $M'$ will now accept if its input $x$ is not a palindrome, rather than if it is. A language is decidable if and only if its complement is recognizable; since $\overline{L}$ is undecidable, it follows that $L$ is unrecognizable.

5. (a) $L_1$ is not recognizable (so it is also not decidable). To show this, we give a reduction from $HALT$, which we know is not recognizable.

Given $(\langle M \rangle, x)$, we build a machine $M'$ that given input $y$, first runs $M$ on $x$; if $M$ halts on $x$, then $M'$ checks if $y$ ends with a 0, and accepts iff it does.

If $M$ halts on $x$, then $M'$ accepts all the strings ending in 0, so $\langle M' \rangle \not\in L_1$. On the other hand, if $M$ does not halt on $x$, then $M'$ doesn’t accept any strings, and in particular no strings ending with 0. Therefore $\langle M' \rangle \in L_1$.

(b) $L_2$ is recognizable — we can simply run $M(w)$ and if it accepts, check to see if the tape head is in the leftmost position, and accept if and only if it is. (We will loop if $M(w)$ loops, but that’s fine.)

$L_2$ is not decidable: we show this by giving a reduction from $HALT$. Given $(\langle M \rangle, x)$, we simply build a machine $M'$ that runs just like $M$, but if $M$ halts, then $M'$ moves the tape-head back to the leftmost position and halts. It’s easy to see that $M'$ halts on $x$ iff $M'$ halts on $x$ with the tape on the leftmost position.

6. (a) $L_3$ is not decidable: we prove it by showing that $HALT \leq_T L_3$. Here is a many-one reduction from $HALT$ to $L_3$: Given input $(\langle M \rangle, x)$, we output the pair $(\langle M' \rangle, \langle M' \rangle)$, where $M'$ is a TM that runs $M$ on $x$, and accepts if $M$ halts on $x$.

$L_3$ is recognizable: to test whether $(\langle M_1 \rangle, \langle M_2 \rangle) \in L_3$, we run $M_1$ and $M_2$ on empty input, alternating the steps of $M_1$ and $M_2$ to make sure we don’t get stuck running one of them if it doesn’t halt. If one of the TMs halts and accepts, we accept. If one of the TMs halts and rejects, we continue running the other one. If both TMs halt and reject, we reject.
(b) $L_4$ is neither decidable nor recognizable. To prove it we show that $\overline{\text{HALT}} \leq_T L_4$.

Given input $(\langle M \rangle, x)$, we output $(M')$, where $M'$ is a new TM we construct. $M'(y)$ runs as follows:

- If $|y|$ is even, $M'$ accepts.
- If $|y|$ is odd, $M'$ runs $M$ on $x$. If $M$ halts on $x$, $M'$ accepts $y$. (Otherwise $M'$ just continues running $M'$ on $x$ forever.)

Why is this reduction correct? First we have to convince ourselves that it’s computable, that is, given $(\langle M \rangle, x)$ we know how to compute $(M')$, but that part is easy to see. More interestingly, we have to convince ourselves that $M$ does not halt on $x$ iff $L(M')$ and $\overline{L(M')}$ are both infinite. By construction, $L(M')$ is always infinite, because $M'$ accepts all the even-length strings. So we just have to worry about $\overline{L(M')}$: we have to show that $M$ does not halt on $x$ iff $\overline{L(M')}$ is infinite. Here’s why: if $M$ does not halt on $x$, then on all the odd-length inputs, $M'$ is going to loop (it runs $M$ on $x$ forever); so $\overline{L(M')}$ contains all the odd-length inputs and $\overline{L(M')}$ is infinite. On the other hand, if $M$ does halt on $x$, then $M'$ is going to accept all the odd-length inputs. We already know that it accepts all the even-length inputs, so $M'$ just accepts everything. This means that $\overline{L(M')} = \emptyset$, which is not infinite.

7. Let $M_1$ and $M_2$ be Turing machines that recognize $L_1$ and $L_2$, respectively. Define $M^{L_1 \cap L_2}$ as follows:

$M^{L_1 \cap L_2}(x)$:

(a) Use the oracle to check if $x$ in $L_1 \cap L_2$. If it is, then accept (since it’s in the intersection, it’s also in $L_1$).

(b) Otherwise, run $M_1(x)$ and $M_2(x)$ side by side. Since their union is $\{0, 1\}^\ast$, eventually, either $M_1$ or $M_2$ will accept.

(c) If $M_1$ accepts, accept. If $M_2$ accepts, reject.

8. If $L_1, L_2 \in \text{NP}$ then there exist nondeterministic TMs $M_1, M_2$ that decide $L_1$ and $L_2$ in polynomial time. We can build a nondeterministic TM $M$ that decides membership in $L_1 \cap L_2$ in polynomial time: on input $x$, $M$ runs $M_1$ and $M_2$ side-by-side (alternating their steps), and accepts only if both $M_1$ and $M_2$ accept $x$.

9. The statement is false: since $M$ and $\overline{M}$ are both nondeterministic, $L(M)$ is the set of strings that some run of $M$ accepts (but maybe other runs of $M$ reject); in other words, it’s the set of strings that some run of $M$ rejects (but maybe other runs of $M$ accept). On the other hand, $\overline{L(M)}$ is the set of strings that all runs of $M$ reject. These languages are not the same.

Here is a counter-example: let $M$ be a TM that on input $n$, guesses two numbers $a, b > 1$ and accepts iff $n = a \cdot b$. The language of $M$ is the composite numbers; for example, $6 \in L(M)$, so $6 \notin \overline{L(M)}$. But $6 \in L(\overline{M})$: there are many runs of $M$ that reject 6 (for example, if we guess $a = b = 3$), and all of these are accepting runs of $\overline{M}$ on 6.