Computational Models - Exercise #5 solution sketch

1. (a) Correct. Let f be the reduction from A to B and let g be the reduction from B to C. Define $h : \Sigma^\star \to \Sigma^\star$ such that $h(x) = g(f(x))$. As f and g are computable, h is computable as well (how?). Also, $x \in A$ iff $f(x) \in B$ iff $g(f(x)) \in C$ iff $h(x) \in C$, as desired.

   (b) Incorrect. Let $A = H_{TM}$ and $B = H_{TM,\varepsilon}$. Clearly, $A \neq B$. Prove yourself that indeed $A \leq_m B$ and $B \leq_m A$.

   (c) Incorrect. Let $A = H_{TM,\varepsilon}$ and $B = \Sigma^\star$. Obviously, $A \subseteq B$, but it is not the case that $A \leq_m B$, as $B \in \mathsf{R}$ and $A \in \mathsf{RE}$.

   (d) Incorrect. Let $A = A_{TM}$ and $B = \overline{A_{TM}}$. It is not the case that $A \leq_m B$ since $B \in \mathsf{coRE}$ but $A \notin \mathsf{coRE}$. Similarly, it is not the case that $B \leq_m A$.

   (e) Correct. A is context-free, so $A \in \mathsf{R}$. Let M_A be the TM that decides A.

   Let $f(x) = \langle M, x\rangle$ be such that M on input x: Run M_A on x and answer the same. Clearly, f is computable. Now, obviously, $x \in A$ iff M accepts x.

2. (a) $L_1 \in \mathsf{RE}$: A TM M' that accepts L_1, on input $\langle M\rangle$: Use a controlled execution (specify exactly how) of M over all words in $\Sigma^\star$ and every time M accepts, M' increases a counter and do not go over this word again. M' accepts if the counter reaches 2016. The correctness is clear.

   $L_1 \notin \mathsf{R}$: Use Rice's theorem. Define the non-trivial property $C = \{L \in \mathsf{RE} \mid |L| \geq 2016\}$, so $L_C = L_1$.

   (b) $L_2 \notin \mathsf{RE}$: We will show $\overline{H_{TM,\varepsilon}} \leq_m L_2$. The reduction is $f(\langle M\rangle)$ such that if $\langle M\rangle$ is an illegal encoding then f returns an arbitrary YES instance of L_2, and otherwise returns $(\langle M'\rangle, 0, 1)$, whereas M' on input x: If $x = 0$ it runs M on ε and answers accordingly. Otherwise, it rejects.

   The reduction is computable. If M does not halt on ε then M' halts only on 1 and $(\langle M\rangle, 0, 1) \in L_2$ (the case of an illegal encoding is handled in this direction as well). If M does halt on ε then M' halts on both 1 and 0, so $(\langle M\rangle, 0, 1) \notin L_2$.

   $L_2 \notin \mathsf{coRE}$: We will show $H_{TM,\varepsilon} \leq_m L_2$. The reduction is $f(\langle M\rangle)$ such that if $\langle M\rangle$ is an illegal encoding then f returns an arbitrary NO instance of L_2, and otherwise returns $(\langle M'\rangle, 0, 1)$, whereas M' on input x: If $x = 0$ it runs M on ε and answers accordingly. Otherwise, it enters an infinite loop.

   The reduction is computable. If M halts on ε then M' halts only on 0 and $(\langle M\rangle, 0, 1) \in L_2$. If M does not halt on ε then M' halts neither on 1 nor on 0, so $(\langle M\rangle, 0, 1) \notin L_2$ (the case of an illegal encoding is handled in this direction as well).

   (c) $L_3 \in \mathsf{R}$. Note that L_3 is finite, and all finite languages are in R.

(d) $L_4 \in \mathsf{coRE}$: A TM M' that accepts $\overline{L_4}$, on input $\langle M \rangle$: Use a controlled execution of M over all words in $\Sigma^\star$ where the simulation is done on an extra tape. Whenever, some simulation reaches position $|x| + 20$, accept. The correctness is clear.

$L_4 \notin \mathsf{R}$: We will show $\overline{A_{TM}} \leq_m L_4$. The reduction is $f(\langle M \rangle, w)$ such that if $\langle M \rangle$ is an illegal encoding then f returns an arbitrary YES instance of L_4, and otherwise returns $\langle M' \rangle$, whereas M' on input x: Check if x is an accepting computational history of the run of M on w. If it is, keep going right forever. Otherwise, reject.

The reduction is computable. If M does not accept w then there is no accepting computational history and for every x we reject. We saw that we don't need any space to the right of x to check it, so $\langle M' \rangle \in L_5$ (the case of an illegal encoding is handled in this direction as well). If M accepts w then there exists x that is an accepting computational history and on that x, M' runs forever. Thus, $\langle M' \rangle \notin L_5$.

(e) $L_5 \notin \mathsf{RE}$: We will show $\overline{H_{TM,\varepsilon}} \leq_m L_5$. Let M_ε be the TM that on input $\langle M \rangle$, M_ε runs M on ε and answers accordingly. The reduction is $f(\langle M \rangle)$ such that if $\langle M \rangle$ is an illegal encoding then f returns an arbitrary YES instance of L_5, and otherwise returns $\langle M' \rangle$, whereas M' on input x: Run M on ε for $|x|$ steps. If within the $|x|$ steps M halted, M' rejects. Otherwise, M' simulates M_ε on x and answers accordingly.

The reduction is computable. If M does not halt on ε then $L(M') = L(M_\varepsilon) = A_{TM,\varepsilon}$ and indeed $A_{TM} \leq_m A_{TM,\varepsilon}$ (the case of an illegal encoding is handled in this direction as well). If M does halt on ε then M' necessarily accepts only a finite number of words, so $L(M') \in \mathsf{R}$ so $\langle M' \rangle \notin L_5$.

$L_5 \notin \mathsf{coRE}$: We will show $H_{TM,\varepsilon} \leq_m L_5$. The reduction is $f(\langle M \rangle)$ such that if $\langle M \rangle$ is an illegal encoding then f returns an arbitrary NO instance of L_5, and otherwise returns $\langle M' \rangle$, whereas M' on input x: M' simulates M on ε and then simulates M_ε on x and answers according to the second simulation.

The reduction is computable. If M halts on ε then $L(M') = L(M_\varepsilon) = A_{TM,\varepsilon}$ and indeed $A_{TM} \leq_m A_{TM,\varepsilon}$. If M does not halt on ε then $L(M') = \emptyset$ so $\langle M' \rangle \notin L_5$ (the case of an illegal encoding is handled in this direction as well).

3. (a) Correct. M_6 on input $\langle M \rangle$:

- Run M on all inputs of length at most 50 for at most 50 steps.
- Accept if at least one string was accepted.

Obviously, M_6 always halts. Since we bound the number of steps that M runs on an input, then there is no point on looking at any strings that are longer than that number, since if a TM is allowed to run for at most c steps, it is not possible for that TM to "process" any input symbol beyond the c-th symbol.

(b) Correct. If $\langle M \rangle$ is indeed a TM then $L(M) \in \mathsf{RE}$ by definition, and RE is closed under union. Thus, a simple TM that decides L_7: Accept iff the input is a legal encoding.

(c) Correct. M_8 on input $\langle D, R \rangle$:

- Check that $\langle D \rangle$ and $\langle R \rangle$ are legal encodings. If not, reject.
- Compute the encoding of D' – the DFA that accepts $L(R)$.
- Compute the encoding of $D'' = L(D) \Delta L(D')$ (how?).
- Run emptiness check algorithm on D''. If $L(D'') = \emptyset$ accept and otherwise reject.

The correctness is clear, as we saw algorithms for constructing a DFA from a regular expression, constructing the intersection/union/complement DFA and checking if a DFA accepts the empty language. Note that $L_1 = L_2$ iff $L_1 \Delta L_2 = \emptyset$.

(d) Incorrect. We will show $\overline{H_{TM,\varepsilon}} \leq_m L_9$. The reduction is $f(\langle M \rangle)$ such that if $\langle M \rangle$ is an illegal encoding then f returns an arbitrary YES instance of L_9, and otherwise returns $\langle M' \rangle$, whereas M' on input x: M' simulates M on ε and if M halts, M' rejects.

The reduction is computable. If M halts on ε then M' always rejects (the case of an illegal encoding is handled in this direction as well). If M does not halt on ε then M' runs forever and never rejects.

4. (a) Incorrect. We saw in class that we cannot decide the problem for $A = \Sigma^\star$.

   (b) Incorrect. Let C be the class of all finite context-free languages. C is non-trivial and we saw an algorithm that decides whether a given CFG generates a finite language.

5. (a) Assume to the contrary that B is computable, and let M_B be a TM that computes it. Consider the TM M', that on input $\langle M \rangle$:

   - If $\langle M \rangle$ is an illegal encoding, reject.
   - Compute q – the number of states in M.
   - Run M_B on q. Let n be the output of M_B.
   - Simulate M on ε for n steps.
   - If M halted, accept. Otherwise, reject.

   We first argue that $L(M') = H_{TM,\varepsilon}$. If the input TM M halts on ε, by the correctness of M_B, it does it within n steps and M' will accept. Otherwise, M' will obviously reject. Also, note that M' always halts, so $L(M') \in \mathsf{R}$, contradicting the fact that $H_{TM,\varepsilon} \notin \mathsf{R}$. Hence, B is not computable.

   (b) Assume to the contrary that f is computable, and let M_f be a TM that computes it. Consider the TM M', that on input $\langle M \rangle$:

   - If $\langle M \rangle$ is an illegal encoding, reject.
   - Compute ℓ, the lexicographical order of $\langle M \rangle$ in $\Sigma^\star$.
   - For every $i \leq \ell$,
     - Run M_f on i. Let $\langle M'' \rangle$ be the result.
     - If $\langle M'' \rangle = \langle M \rangle$, accept.
   - Reject.

   Note that M' always halt. We argue that $L(M') = \overline{H_{TM,\varepsilon}}$, which leads to a contradiction. Note that if M is the k-th TM and does not halt on ε, then it must appear within $f(1), \ldots, f(k)$ and we will accept. Otherwise, it will not appear in the image of f and we will reject.

   Also, we proved in the recitation that a function has a monotone enumerator iff it is in R. What is given to us is exactly a monotone enumerator, and we know that $\overline{H_{TM,\varepsilon}} \notin \mathsf{R}$.

6. Assume that L has a verifier M_V. Let M be the TM such that on input x:

   - For every $c \in \Sigma^\star$ in lexicographical order:
     - Run $M_V(x, c)$. If M_V accepted, accept.

If $x \in L$ then there exists c such that $M_V(x, c)$ accepts. M will eventually reach it (within a finite number of steps) and will accept. If $x \notin L$ then for every c, $M_V(x, c)$ rejects. Thus, M will never halt. Overall, $L(M) = L$ so $L \in \mathsf{RE}$.

Assume that $L \in \mathsf{RE}$, so there exists a TM M such that $L(M) = L$. Let M_V be a TM such that on input $\langle x, c \rangle$:

- Treat c as an integer, and run M on x for c steps.
- If M halted then answer the same.
- Otherwise, reject.

First, note that M_V always halts. If $x \in L$ then M halts on x and accepts within a finite number of steps, say n. Thus, for c that is the binary representation of n, $M_V(x, c)$ accepts. If $x \notin L$ then M will not accept regardless of the number of steps, so M_V will always reject. Overall, M_V is indeed a verifier for L.

7. Fix such S and first assume that the constant function 0, f_0, is in S. We will prove that $H_{TM} \leq_m L_S$. $f(\langle M \rangle, w) = \langle M' \rangle$ such that M' on input x simulate M on w and outputs 0.

   The reduction is computable (note that we need to take care of illegal encodings). If M halts on w then $f_{M'}$ is f_0, which is in S. Otherwise, M' computes the empty function, which is not total and thus not in S.

   In the case where $f_0 \notin S$, consider $\bar{S} = F \setminus S$. By the previous argument, $L_{\bar{S}} \notin \mathsf{R}$. Hence, $\overline{L_S} \notin \mathsf{R}$ and by the fact that R is closed under complement, it must be the case that $L_S \notin \mathsf{R}$ as well.

8. First, if L is finite the the claim obviously holds, as there are infinitely many distinct languages in R. We now assume that L is infinite, and as $L \in \mathsf{RE}$, it has an enumerator $f : \mathbb{N} \to L$. Consider the TM M', that on input x:

   - Compute i, the lexicographical order of x.
   - Compute $\langle M \rangle = f_L(i)$.
   - Run M on x.
   - If M accepted, reject. If M rejected, accept.

   M' always halt, so $L(M') \in \mathsf{R}$. Thus, there exists a k such that $L(M') = L_k$.

   Assume to the contrary that $L \cap A_i \neq \emptyset$ for every i, specifically for $i = k$. As f_L is an enumerator, there exists a j for which $f_L(j) = \langle M \rangle \in A_k$. Run M' on the j-th string in lexicographical order. M' then outputs the opposite of M on the j-th string in lexicographical order, contradicting the fact that $L(M') = L_k$.