Theory of Computation: Assignment 7

Arjun Chandrasekhar

1. (a) Because A and B are decidable, there are machines M_A and M_B that that decide A and B, respectively. We'll design a machine to decide $A \circ B$.

   1. M takes a string w as input
   2. For all possible ways of splitting up $w = xy$ do the following:
   a. Run M_A on x and M_B on y
   b. If both machines accept, then M accepts w
   c. If either machine rejects, move on to the next way of splitting up the string
   3. If all possible splits were rejected, M rejects w

   Because M_A and M_B is decidable, they will always halt; thus, M will halt as well.

   (b) Because A and B are decidable, there are machines M_A and M_B that that decide A and B, respectively. We'll design a machine to decide $A \circ B$. We will try every way of splitting up the string, and see if M_A and M_B accept the two substrings; however, we have to be careful. If we try each split one at a time, we may get stuck on one looping split and never get to try out an accepting split.

   1. M takes a string w as input
   2. Nondeterministically guess how to split up $w = xy$
   3. Run M_A on x and M_B on y
   4. If both machines accept, then M accepts w
   5. If either machine accepts, then M will not accept w (it may loop)

   Because A and B are recognizable, they halt and accept on all strings that are in the language. Thus if $w \in A \circ B$ then at least one split will be accepted by both machines, so at least one computation path will accept.

   Every nondeterministic Turing machine can be converted to a deterministic machine, thus $A \circ B$ is Turing recognizable.

   Note: Instead of using nondeterminism, we can run all possible splits in parallel.

2. The formal language is as follows

$$L = \{\langle D, R\rangle | D \text{ is a DFA, } R \text{ is a regex, } L(D) = L(R)\}$$

   Let M_{EQ} be a machine that decides EQ_{DFA} - that is, it checks if two DFAs are equivalent. We can decide the language using the following machine M:

   1. M takes $\langle D, R\rangle$ as input
   2. Convert R to an equivalent DFA D_2
   3. Run M_{EQ} on $\langle D, D_2\rangle$ - that is, check if they're equivalent
   4. If M_{EQ} accepts $\langle D, D_2\rangle$, then M accepts $\langle D, R\rangle$
   5. If M_{EQ} rejects $\langle D, D_2\rangle$, then M rejects $\langle D, R\rangle$

3. Let M_{EQ} be a machine that decides EQ_{DFA} - that is, it checks if two DFAs are equivalent. We can decide ALL_{DFA} using the following machine M:

1. M takes $\langle D \rangle$ as input
2. Construct a DFA D_2 that recognizes Σ^* (which is a regular language).
3. Run M_{EQ} on $\langle D, D_2 \rangle$ - that is, check if they're equivalent
4. If M_{EQ} accepts $\langle D, D_2 \rangle$, then M accepts $\langle D \rangle$
5. If M_{EQ} rejects $\langle D, D_2 \rangle$, then M rejects $\langle D \rangle$

4. We will design a machine M to decide L as follows:

   1. M takes $\langle D \rangle$ as input
   2. Create a DFA D_2 that recognizes the set of strings with more than four 1's
   3. Create a DFA D_3 that recognizes $L(D) \cap L(D_2)$
   4. Check if $L(D_3) = \emptyset$. If so, accept $\langle D \rangle$. Otherwise, reject.

   This checks that $L(D)$ has nothing in common with the set of strings that contain more than four 1's.

5. As noted in the hint, every string is in either $\overline{A}$ or $\overline{B}$. Suppose $w \notin \overline{A}$ and $w \notin \overline{B}$. Then $w \in A \cap B$ - but this is impossible since A and B are disjoint.

   We also know that $\overline{A}$ and $\overline{B}$ are Turing-recognizable. Thus there exist machines $M_{\overline{A}}$ and $M_{\overline{B}}$ that recognize $\overline{A}$ and $\overline{B}$. If $w \in \overline{A}$ then $M_{\overline{A}}$ is guaranteed to halt and accept w; similarly, $w \in \overline{B}$ then $M_{\overline{B}}$ is guaranteed to halt and accept w.

   We will construct a machine M that separates A and B.

   1. M takes w as input
   2. Run $M_{\overline{A}}$ and $M_{\overline{B}}$ on w in parallel
   3. If $M_{\overline{B}}$ accepts w, then M accepts w
   4. If $M_{\overline{A}}$ accepts w, then M rejects w

   Let $C = L(M)$ - that is, C is the set of strings that are accepted by M. We claim that C separates A and B, and C is decidable.

   First we'll show that C separates A and B. Suppose $w \in A$. Then $w \in \overline{B}$, so $M_{\overline{B}}$ accepts w. Therefore M accepts w, so $w \in L(M) = C$. Thus, $A \subseteq C$.

   Now suppose $w \in B$. Then $w \in \overline{A}$, so $M_{\overline{A}}$ accepts w. Therefore M rejects w, so $w \notin L(M) = C$. This means $w \in \overline{C}$. Thus, $B \subseteq \overline{C}$.

   Finally, we show that C is decidable. To do this, we'll show that M (which recognizes C) always halts. As noted earlier, because $\overline{A}$ and $\overline{B}$ are disjoint, every string is in either $\overline{A}$ or $\overline{B}$. Thus, either $M_{\overline{A}}$ or $M_{\overline{B}}$ accepts w. Thus, M will always halt.