CS 373 Homework 9 Anchu A. Lee
December 6, 2017

1. Let Some$_{DFA}$ ={ $\langle A \rangle$ | A is a DFA and $L(A)$ is not empty and $L(A)$ is not equal to Σ^* }. Show that Some$_{DFA}$ is decidable.

   Construct a Turing machine M to decide the Some$_{DFA}$ problem.

   On input d that is a DFA:

   1. Run $\langle d \rangle$ on a Turing machine T that decides E$_{DFA}$, if T rejects then continue, if T accepts, then reject.
   2. Construct a DFA d^C that is the compliment of d.
   3. Run $\langle$d$^C\rangle$ on a turing machine T, if T rejects then accept d, if T accepts then reject d.

   If d is not a DFA, then M rejects. If $L(d)$ is empty, then M rejects. If the compliment of d is empty, meaning $L(d) = \Sigma^*$, then M rejects. Otherwise M accepts. All conditions are handled in M so $M =$ Some$_{DFA}$.

2. Let Alot$_{RE}$ ={ $\langle A \rangle$ | A is a regular expression and $L(A)$ is infinite }. Show that Alot$_{RE}$ is decidable.

   Construct a Turing machine M to decide the Alot$_{RE}$ problem.

   On input r that is a RE:

   1. Construct a DFA A that is equivalent to r.
   2. For s that is the number of states in A, construct DFA B that accepts all strings over the alphabet in A that are at least length s.
   3. Construct a DFA C so that $L(C) = L(A) \cap L(B)$.
   4. Run $\langle C \rangle$ on a turing machine T that decides E$_{DFA}$.
   5. If T accepts then reject r, if T rejects then accept r.

   In order for a DFA to accept an infinite language, it must contain a loop. If a DFA contains a string with a length greater than the number of states, then it contains a loop and therefore accepts an infinite language. The Turing machine M checks if a RE accepts an infinite language by intersecting the language accepted by r to the language of strings with length greater than the number of states of the DFA for r. M accepts if the intersection is non-empty and rejects otherwise. Therefore $M =$ Alot$_{RE}$.

3. Let Complimentary$_{RE,DFA}$ ={ $\langle A, B \rangle$ | A is a regular expression and B is a DFA such that $L(A) \cup L(B) = \Sigma^*$ and $L(A) \cap L(B)$ =∅}. Show that Complimentary$_{RE,DFA}$ is decidable.

   Construct a Turing machine M to decide the Complimentary$_{RE,DFA}$ problem.

   On input r that is a RE and d that is a DFA:

   1. Create a DFA A that is equivalent to r.
   2. Construct a DFA B so that $L(B) = L(A) \cup L(d)$.
   3. Cosntruct a DFA B^C that is the compliment of B.
   4. Run $\langle B^C \rangle$ on a Turing machine T that decides E$_{DFA}$.
   5. If T accepts, then continue, if T rejects then reject r, d.
   6. Construct a DFA C so that $L(C) = L(A) \cap L(d)$.
   7. Run $\langle C \rangle$ on turing machine T.
   8. If T accepts, then accept r, d. Otherwise reject.

   M rejects if $L(A) \cup L(B) \neq \Sigma^*$. M rejects if $L(A) \cap L(B) \neq \emptyset$. M only accepts if both conditions are met, therefore $M =$ Complimentary$_{RE,DFA}$.

4. Let ALL$_{DFA}$ ={ $\langle A \rangle$ | A is a DFA and $L(A) = \Sigma^*$}. Show that ALL$_{DFA}$ is decidable.

   Construct a Turing machine M to decide the ALL$_{DFA}$ problem.

   On input d that is a DFA:

   1. Create a DFA d^C that is the compliment of d.
   2. Run $\langle d^C \rangle$ on a Turing macine T that decides E$_{DFA}$.
   3. If T rejects, then reject d. If T accepts, then accept d.

   M only accepts if the language of the compliment of the input DFA is an empty language. For a DFA to equal Σ^*, it must accept all possible strings over the alphabet and therefore the compliment of such a DFA must be empty. This shows that $M =$ ALL$_{DFA}$.

5. Let $N_{\varepsilon CFG} = \{ \langle G \rangle \mid G$ is a CFG and G does not generate the empty string $\}$. Show that $N_{\varepsilon CFG}$ is decidable.

   Construct a Turing machine M to decide the $N_{\varepsilon CFG}$ problem.

   On input g that is a CFG:

   1. Construct A that is a equivalent CFG to g except in Chomsky normal form.
   2. If A contains the rule $S \to \epsilon$ reject, otherwise accept.

   The CFG A can only generate the empty string if there exists a rule $S \to \epsilon$, so if A does not include the rule then $\epsilon \notin L(A)$. So $M = N_{\varepsilon CFG}$.

6. Let X be the set $\{1, 2, 3, 4, 5\}$ and Y be the set $\{6, 7, 8, 9, 10\}$. We describe the functions $f : X \to Y$ and $g : X \to Y$ in the following tables. Answer each part and give a reason for each negative answer.

n	$f(n)$
1	6
2	7
3	6
4	7
5	6

n	$g(n)$
1	10
2	9
3	8
4	7
5	6

   (a) Is f onto?

   No, there exists elements in Y that are not mapped to X.

   (b) Is f a correspondence?

   No, f is not one-to-one as $f(1) = f(3)$ and is not onto.

   (c) Is g onto?

   Yes

   (d) Is g a correspondence?

   Yes

7. Let $U = \{ \langle A, B, C \rangle \mid A, B, C$ are DFA's and $|L(A)| = |L(B)| + |L(C)| \}$. Show that U is decidable.

   Construct a Turing machine M to decide U.

   On input a, b, c which are DFAs:

   1. Run each DFA on ALL_{DFA}, if any accept then reject. Otherwise continue.
   2. Let l be the number of states in b added with the number of states in c.
   3. Construct a DFA A which accepts all strings over Σ of length l.
   4. Construct a Time Machine to go to the future and figure out how to solve this problem.

8. Let $A = \{ \langle R \rangle \mid R$ is a regular expression describing a language containing at least one string w that has 111 as a substring (i.e., $w = x111y$ for some x and y)$\}$. Show that A is decidable.

   Define the language $B = \{w \in \Sigma^* \mid w$ has 111 as a substring$\}$

   Construct a Turing machine M to decide A.

   On input r which is a RE:

   1. Construct a DFA C that is equivalent to r.
   2. Construct a DFA D that accepts the language $L(D) = B \cup L(C)$.
   3. Run D on a Turing machine T that decides E_{DFA}.
   4. If T accepts, reject. If T rejects, accept r.

   M accepts only if the input RE shares a word with the language that contains all w that contain 111 as a substring. M rejects otherwise. Thus M decides A.

9. Let $E_{PDA} = \{ \langle P \rangle \mid P$ is a pushdown automata and $L(P)$ is empty $\}$. Show E_{PDA} is decidable.

   Construct a Turing machine M to decide E_{PDA}.

   On input p which is a PDA:

   1. Construct a CFG A that is equivalent to p.
   2. Run A on a Turing machine T that decides E_{CFG}

3. If T accepts then accept p, otherwise reject.

M accepts only if the CFG equivalent to p, which has the same language as p, is the empty language. We] know checking if a CFG is empty is decidable so M must be able to decide E_{PDA} as well.

10. A ***useless state*** in a pushdown automaton is never entered on any input string. Consider the problem of determining whether a pushdown automaton has any useless states. Formulate this problem as a language and show that it is decidable.

    asdf