Homework 9: Decidability

CSCI 161

Due 9:00pm, Saturday, November 21, 2020

Ben Dreyer

Overview. This week's homework primarily concerns the decidability of languages. You will give (partial) classifications for six languages. The following gives additional per-problem guidance:

- The bonus question is an ungraded practice question. I'll present a solution in office hours next week upon request as an example of a decider algorithm involving creative and mechanical problem solving. Although the language is analogous, the solution does not directly influence the solution to Question 6.
- As with most decidability arguments, showing decidability of each of the languages in Questions 1-4 will involve giving an algorithm and arguing that it correctly accepts or rejects any possible input in finite time. Your algorithms can use procedures discussed in class – and in several cases this will probably be helpful! – but you should describe at a high level whatever algorithm you are calling on. You can probably rely on the correctness of your algorithms to speak for themselves, but a sentence or two describing why they work as designed may help with the clarity of your answer.
- Turing recognizability can be established by giving an algorithm that is guaranteed to accept all inputs in the language in finite time, and co-Turing-recognizability (Question 5) can be established by giving an algorithm that is guaranteed to reject all inputs not in the language in finite time.
- Undecidability is generally established by using an assumed decider for the language to construct a contradiction. The hint for Question 6 suggests what contradiction you may wish to construct.

As usual, edit this tex document and submit it with your compiled pdf for 5% extra credit.

Question 0. List all collaboration on this assignment.

Solution Cole and the Textbook

Bonus Q. Show that $REV_{\text{DFA}} = \{\langle D \rangle \mid D \text{ is a DFA and } L(D) = \{w^{\mathcal{R}} \mid w \in L(D)\}\}$ is decidable.

Question 1. Consider the problem of determining whether a DFA and a regular expression are equivalent. Express this problem as a language EQ_{DR}, and show that it is decidable.

Solution In order to show the language $EQ_{\text{DR}}$is decidable we can construct a turning machine T that decides it as follows:

1. Run TM T on input x:

   If x is not the encoding of a DFA and a regular expression, reject

   Else, refer to the inputs as DFA D and Regex R

   Convert Regex R into an equivalent DFA DR

   Construct a TM H that takes an input of the encoding of two DFA's

   (a) Run H on input $< D, DR >$:

   Construct a DFA M whose language is the symmetric difference of the languages of D and DR

   Construct a TM E that takes an input of the encoding of one DFA

i. Run E on input $< M >$
   Mark states reachable from the start state
   If one of these states is an accept state: reject
   else: accept

ii. Forward answer from E to H

(b) Forward answer from H to T

This solution works because we know that a Regex can be converted into a DFA based on theorems from Unit 1. I then simply defined the TM that recognizes the language EQ_{DFA} that we learned in classn in order to check if the converted Regex and DFA are equivalent, which we know will always reject or accept.

Question 2. Let $SUBSET_{\text{REX}} = \{\langle R, S\rangle \mid R, S \text{ are regular expressions and } L(R) \subseteq L(S)\}$. Show that $SUBSET_{\text{REX}}$ is decidable.

Solution To show that $SUBSET_{\text{REX}}$ is a decidable language we can construct a TM T that decides it as follows:

1. Run TM T on input x:

   if x is not the encoding of two regular expressions: reject

   else refer to input as Regex R1 and R2

   Convert R1 and R2 into respectively equivalent DFA's D1 and D2

   Construct a DFA D2C whose language is the compliment of D2

   Construct a DFA DI whose language is the intersection of D2C and D1

   Construct a TM H that takes the input of an encoding of a DFA

   (a) Run H on input $< DI >$:
      Mark states reachable from the start state
      If one of these marked states is an accept state: reject DI
      else: accept DI

   (b) If H accepted DI , accept x

   (c) Else reject x

This solution works because like in the previous question, we can convert a Regex into an equivalent DFA. Likewise in order to find out if a Language of a DFA is the subset of another's language, we can take the compliment of the larger DFA's language and intersect that with the smaller. If there are any elements in this language, we know that the smaller DFA is not a subset because if has values outside of the language of the Larger DFA. The TM H in my code is the TM that recognizes E_{DFA} from class.

Question 3. Any regular language intersected with a context-free language is context-free. Use this fact to show that $IMB_{\text{DFA}} = \{\langle D\rangle \mid D \text{ is a DFA and } w \in L(D) \text{ for some } w \text{ with more 1's than 0's}\}$ is decidable.

Solution To show that IMB_{DFA} is decidable we can construct a Turing Machine T that decides it as follows:

1. Run TM T on input x:

   if x is not the encoding of a DFA: reject

   else refer to the input as DFA D

   Construct a DFA R whose language is some string with more 1's than 0's

   Construct a TM H that takes an input of the encoding of two DFAs

   (a) Run TM H on input $< R, D >$

      Construct a DFA DC whose language is the compliment of D

      Construct a DFA DI whose language is the intersection of R and DC

      Construct a TM E that takes the input of the encoding of a DFA

      i. Run E on input $< DI >$

         Mark states reachable from the start state

         If one of these marked states is an accept state: reject DI

         else: accept DI

      ii. If E accepted DI: accept $< R, D >$ else: reject $< R, D >$

   (b) If H accepted $< R, D >$: accept x

   (c) else: reject x

This solution works by passing the input DFA and then constructing a new DFA whose language is some string with more 1's than 0's. We then check if this new DFA is a subset of the input DFA using the logic from question 2.

Question 4. Let $A\epsilon_{\text{CFG}} = \{\langle G \rangle \mid G \text{ is a CFG and } \epsilon \in L(G)\}$. Show that $A\epsilon_{\text{CFG}}$ is decidable.

Solution To prove that $A\epsilon_{\text{CFG}}$ is a decidable language we can construct a Turing Machine T that decides it as follows:

1. Run T on input x:

   if x is not the encoding of a CFG: reject

   else refer to input as CFG G

   Convert G into an equivalent Chomsky Normal Form CFG G'

   If the starting rule of G' includes a transition to epsilon, accept x

   else reject x

For this question, we need to check if the CFG generates simply the epsilon character. We need to have our CFG in CNF because a non CNF CFG could generate epsilon in multiple steps not obvious through the start variable. After converting to CNF, simply check if our start variable includes a rule that is $S \rightarrow \epsilon$.

Question 5. Recall that $EQ_{\text{CFG}} = \{\langle G, H \rangle \mid G, H \text{ are CFGs and } L(G) = L(H)\}$. Show that EQ_{CFG} is co-Turing-recognizable.

Solution To show that EQ_{CFG} is co-Turing-recognizable, we can show that the compliment of the language, $\overline{EQ_{CFG}}$, is Turing-recognizable by building a Turing Machine D that recognizes it.

1. Run Turing Machine D on input x

   if x is not the encoding of two CFGs, accept

   else refer to the inputs as CFG G1 and G2

   Convert both inputs into equivalent CFGs in Chomsky Normal Form, G1' and G2'

   Generate all possible terminal strings from the alphabet of G1', generate all possible terminal strings from the alphabet of G2'

   Union these two sets of strings into S

   For each item in S

   Run CYK algorthim on CFG G1' with s_i and CFG G2' with s_i

   if only one of G1' and G2' can generate s_i, accept x

Question 6. Show that $REV_{\text{TM}} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \{w^{\mathcal{R}} \mid w \in L(M)\}\}$ is undecidable.

Hint: Show $A_{\text{TM}} \rightarrow REV_{\text{TM}}$.

Solution To show that REV_{TM} is undecidable, we can reduce A_{TM} to it and prove with contradiction that if we can decide A_{TM} using REV_{TM}, that REV_{TM} is undeidable since we know from Thm 4.11 that A_{TM} is undecidable.

Assume the the Turing Machine H decides REV_{TM}, then construct a TM D as follows:

1. A = on input $< M, w >$ M is a Turing Machine and w is a string

   Construct a Turing Machine M2 as follows

   M2 = on input $< x >$ where x is a string

   if $x \in L(00 * 11*)$, accept

   else, run w on M

   if M accepts w, accept

   if M rejects w, reject

   Run H on input $< M2 >$

   If H accepts, accept

   If H rejects, reject

We can see that we are using REV_{TM} to decide A_{TM}, therefore we know that REV_{TM} is undecidable. We define our M2 as we do because if the input string x is a string that is in the language of 00*11*, we know its reverse will not be in that language.