Assignment 2

CMPUT 474

Pranav Wadhwa
1629510

March 18, 2022

1. A **Turning mahine with doubly infinite tape** is similar to an ordinary Turing machine, but its tape is infinite to the left as well as to the right. The tape is initially filled with blanks except for the portion that contains the input. Computations is defined as usual except that the head never encounters an end to the tape as it moves leftward. Show that this type of Turing Machine recognizes the class of Turing-recognizable languages.

Solution: Lets first examine what happen in a regular turing machine when its head is at the left-end of the tape. If the head of the machine is at the left-end and it tries to do an $\delta(q, \sigma) = (q', \sigma', L)$ but the head stays at the same place.

We want to show that Turing Machine with doubly infinite tape recognizes the class of Turing-recognizable languages. We can do this by making the new TM simulate the left bounded machine by adding a new symbol to mark the left end of the regular TM's tape and prevent the head from moving any further left off the that mark.

To simulate the doubly infinite tape by an regular TM we can use 2 tapes. As we already know that we can easily simulate multiple tapes on a single tape, we can use this knowledge to simulate the doubly infinite tape. The first tape would have the input symbol and the right bound and the second tape would have the left bound in reverse order. At the start of the computation the second tape would be black. This way we can get a regular TM to simulate a doubly infinite TM.

2. Show that the collection of decidable languages is closed under the operation of

   1. union
   2. concatenation
   3. star
   4. complementation
   5. intersection

Solution: Let there be 2 decidable languages L_1 and L_2 and their respective turing machines that accept then M_a and M_2. As both M_1 and M_2 are decidable both machine will halt.

1. union

   We construct a turing machine M that recognizes $L_1 \cup L_2$ such that i Read input string w

ii Run M_1 on string w.
If M_w return reject goto next step. Else return accept

iii Run M_2 on string w.
If M_2 return reject then return reject. Else return accept

2. concatenation

We will use the **non-deterministic** TM to prove this. The concatenation of L_1 and L_2 is defined as $L_1 \circ L_2 = \{xy | x \in L_1 \text{ and } y \in L_2\}$. Consider the non-deterministic TM M which would take an input w and partition it into a form $w = xy$ where $x \in L_1$ and $y \in L_2$.

i Read the input w and partition it into 2 string $w = xy$

ii Simulate M_1 on x and simulate M_2 on y

iii return accept if both M_1 and M_2 accept it, else return reject.

3. star

4. complementation

We have a TM M which simulates L_1. If L_1 rejects input w then M return accepts and it L_1 accepts input w then M returns rejects.

5. intersection

Intersection is similar to union. Let M be a TM which simulates $L_1 \cap L_2$ where we run an input w on both machines and M return accept only when both L_1 and L_2 return accept else it return reject.

3. Show that the collection of Turing-recognizable languages is closed under the operation of

1. union
2. concatenation
3. star
4. intersection
5. homomorphism

(Note that you can fine the definition of homomorphism on Page 93, Problem 1.66)

Solution: Let there be 2 Turing-recognizable languages L_1 and L_2 and their respcitive TM M_1 and M_2.

1. union

We construct TM U that recognizes $L_1 \cup L_2$ such that:

1. Simulate M_1 and M_2 on string w simultaneously.
2. Do a step of M_1 first then do a step of M_2
3. If any of the machine return accept then return accept.

2. concatenation This is similar to concatenation of decidable languages. Let a non-deterministic TM simulate M_1 and M_2 on a string w which can be partitoned into $w = xy$ where $x \in L_1$ and $y \in L_2$. We simulate M_1 on x and M_2 on y simulataneouly. If both accept then return accept.
3. star
4. intersection

   This operation is similar to the union of decidable languages. Let a TM I simulate input w on M_1 then on M_2. If both return accept then return accept else return reject.
5. homomorphism

   Def: A *homomorphism* is a function $f : \Sigma \to \Gamma^*$ from one alphabet to strings over another alphabet.

4. Prove the following language is decidable

$$L = \{\langle M\rangle : M \text{ is a DFA that accepts some string of the form } ww^R \text{ for } w \in \{0,1\}^*\}$$

Solution:

The string of the form ww^R for $w \in \{0,1\}^*$ is not a regular language but it is a context-free language. Let $A = \{x|$ x is of form $w \in \{0,1\}^*\}$.

It is already known that context-free langugaes are decidable. We need to show that M accept string of form A is decidable. We can contruct a decider TM D_M which decides on L.
D_M on input $< M >$ where M is a DFA:

1. Construct $B = A \cap L(M)$. B is a CFL as A is a CFL. (This is done through another TM which converts the DFA into a RE and does the intersection with A)
2. If B is empty return reject else return accept.

Therefore, L is decidable.

5. Show that the following langauges are undecidable

   a) Set of descriptions of Turing machines $\langle M\rangle$ such that $\emptyset \in L(M)$

   b) Set of decriptions of pairs of Turing machines $\langle M, M'\rangle$ such that $L(M) \cap L(M') = \emptyset$. Also show that this language is recognizable.

Solution:

a) Let $L = \{\langle M\rangle | M \in TM \text{ and } \emptyset \in L(M)\}$ and we show that L is undecidable.

Assume the contrary that L is decidable and TM M_L can decide on L. We show that M_L can be used to decide A_{TM}. Then we construct TM D such that:

$D(\langle M\rangle)$

1. call $M_L(\langle M\rangle)$
2. if M_L accepts $\langle M\rangle$
   Simulate M on string x (must halt)
   if M accepts, return accept, else return reject
3. If M_L rejects $\langle M\rangle$ return reject.

This contradicts the undecidability of A_{TM} so L is undecidable.

b) We have 2 TM M and M' which simulate the languages $L(M)$ and $L(M')$. We know that decidable languages are closed under intersection then $L(M) \cap L(M')$ is also a decidable language which must have an equivalent TM say I. Let $L = \{\langle M, M'\rangle | L(M) \cap L(M') = \emptyset\} = \{\langle I\rangle | I \in TM \text{ and } L(I) = \emptyset\}$

We know that E_{TM} is undecidable and L is also of the same form. Therefore, this language L is also undecidable.

6. Let A and B be two disjoint languages. Say that language C separates A and B if $A \subseteq C$ and $B \subset \bar{C}$. Show that any two disjoint co-recognizable languages are separable by some decidable language.

7. Say that a variable A in CFL G is usable if it appears in some derivation of some strings $w \in G$. Given a CFG G and a variable A, consider the problem of testing whether A is usable. Formulae this problem as a language and show that it is decidable.

Solution: Let there be a CFG G which has a variable A. We define that a variable is usable if it appears in some derivation of some strings. Or, more precisely, A appears in the the rule of some other variable such as X such that $X \to X_1AX_2$ is a rule in CFG G. Here X_1 and X_2 can be ϵ or derive other languages $L(X_1)$ or $L(X_2)$.

Consider the language $U = \{\langle G, A\rangle | G \text{ is a CFG }, A \text{ is usable for G }\}$. We show that U is decidable. Let D be a decider TM which decides on U and it works such that

On Input of $\langle G, A\rangle$:

1. Let S be a starting variable. It uses breath first search to look for any derivatation from S to A. If there are no derivation to A it returns reject else move to next step.

2. We check using E_{CFG} that $L(A)$ is not empty. If E_{CFG} returns reject then then D returns accept else D returns reject.

Therefore, U is decidable

8. Let L be a CFL. Is $\{1\}^* \subseteq L$ a decidable problem? Is $\{1\}^* = L$ a decidable problem?

Solution: We know that $\{1\}^*$ is a regular language and the intersection of a regular language and a CFL is a CFL.

We will use this knowledge to construct a decider TM D such that:

On input of a CFL L

1. take the intersection: $L \cap \{1\}^* = A$
2. Check that if $A = \emptyset$
3. If A is empty then return false else return true.

So $\{1\}^* \subseteq L$ is a decidable problem.

Now to check if $\{1\}^* = L$ is decidable problem. We can't take the intersection like the previous problem as EQ_{CFG} is undecidable. Instead,we will look at how the CFG generates it strings. The idea is to look at all the rules in the CFG in the chomsky normal form and check if the terminal variables are all $1s$ or not.

So let a TM D be a decider such that:

On input of a CFL L:

1. Convert the CFL into chomsky normal form.
2. Look at all the terminal variables of the CFG.
3. If there is a production rule (other than the start variable) such that it has a terminal symbol other than 1, return reject.
4. if all production rules with a terminal symbols are 1, then return accept.

So, $\{1\}^* = L$ is also a decidable problem.

9. Consider the problem of determining whether a turing machine M on an input w ever attempts to move its head left at any point during its computation on w. Formulate this problem as a language and show that it is decidable.

Solution:

let $L = \{\langle M, w\rangle | M \text{ moves its head left on input } w\}$. We show that L is decidable using a TM LM.

We construct TM LM such that it simulates M on input w. We will have a few cases during the execution on w such as:

a) If the head ever moves left then return accept.

b) If the head never moves left and accepts the input.

c) if the head stays in the same position during the execution after reading through the input and going over $|Q|$ (number of states) steps then return reject.

d) if the head moves right and have gone over the input and $|Q|$ (number of states) steps we can conclude that M is in a cycle and thereby never moving left.

We can formulate this into an high level description of the machine LM such that if M moves left withing $w + |Q|$ steps then return accept else return reject as after that steps the machine enters into a cycle.

Therefore, the language L is decidable.

10. Consider the problem of determining whether a Turing machine M on an input w ever attempts to move its head left when its head is in the left-most tape cell. Formulate this problem as a language and show that it is undecidable.

Solution:

Let $L = \{\langle M, w\rangle |$head attempts to move its head left when head is in the left-move tape cell $\}$. We show that L is undecidable using reduction of $HALT_{TM}$.

Let LM be the TM that run M on input w but puts a # symbol on the left end of the tape and shifts the input after # symbol.

Let LM' be the decider TM and it run on input $\langle M, w\rangle$ as follows:

1. Run TM LM to simulate M on input w as described above.
2. If LM reaches the # symbol during its execution, then it moves it head one step to the right but remains in the same state, then it return accept, else return reject.
3. if LM accept
   Simulate M on w (must halt) if M return accept, return accept else return accept
4. If LM rejects M and w
   then return reject

This contradicts the undecidability of $HALT_{TM}$ so L is also undecidable.