Henry Osei COSC 417
3/19/2020

Assignment 8

Instructions.

1. Due April 23.
2. This is a team assignment. Work in teams of 3-4 students. Submit one assignment per team, with the names of all students making the team.
3. You will submit on Blackboard one single pdf file with the solutions to all exercises. For this you'll take the .tex file for this assignment and modify it. In the box above replace Ann, Bob, Charlie with your names. Write down your anwers for each question after Answer:.

   For editing the above document with Latex, see the template posted on the course website.

   `http://orion.towson.edu/~mzimand/adatastruct/assignment-template.tex` and

   `http://orion.towson.edu/~mzimand/adatastruct/assignment-template.pdf`

   To append in the latex file a .jpg file (for a photo; for example, in case you draw a picture by hand and take a photo of it with your phone camera), use

   ```
   \includegraphics[angle=270,origin=c,width=\linewidth]{file.jpg}
   ```

   The parameter angle=270 is for rotating the photo, and you may have to change 270 to whatever angle works for your photo.

Exercise 1. Show that the language

$$L = \{< M_1 >| M_1 \text{ is a Turing machine that accepts } 0\}.$$

is Turing-recognizable. (You need to give an informal description of a Turing machine V that accepts $< M_1 >$ if and only if M_1 is a TM that accepts 0. Your description should start with "V on input $< M_1 >$ ...." and describe what V does. For a similar example see the description of a machine U that recognizes A_{TM} on page 202 in the textbook).

Answer:.

V = "on input" <M_1> where M_1 is a Turing Machine that accepts 0.

1. Simulate M_1 on input 0
2. If M_1 accepts, then accept. If M_1 shall halt then reject; reject

Turning Machine V recognizes A_{TM}, This is a universal TM that is prepared for arbitrary inputs should such be entered. If such is entered the TM will halt meaning rejecting the input was entered.

Exercise 2. (a) Give an informal description of a computable function f that on input a Turing machine M and an input string w, outputs a Turing machine M_1 (in other words $f(hM, wi) = hM_1i$) with the property

- if M accepts w, then M_1 accepts 0, and
- if M does not accept w, then M_1 does not accept 0.

Answer:.

M' = "On input string w"

1. Run M and M_1 alternately on w,
2. one step at a time. If either accepts, accept. If both halt and reject, reject

- reject is not recursively enumerable , while Accept is recursively enumerable because the universal Turing machine accepts/recognizes/solves Accept. Let us consider the complement of reject. M' recognizes M union M_1, first consider $w \in M \cup M_1$. Then w is in M or in M_1. If $w \in M$, then M_1 accepts w, so M' will eventually accept w.

(Recall that to describe M_1 you need to consider an arbitrary input string x, and say how M_1 operates on x. Thus your description of M_1 should start with: "M_1 on input x: .... ", and next you explain in English what M_1 does Of course, M_1 has to simulate M on input w and do certain things depending on the outcome of the simulation.)

(b) Interpret part (a) as a reduction from a certain language X and explain what it implies about the language

$$L = \{< M_1 >| M_1 \text{ is a Turing machine that accepts } 0\}.$$

- To create a reduction we need another way to solve this algorithm. Meaning what other input could be used on this Turning Machine or if modified it should still accept 0. X is an arbitrary value that is to be entered into the TM but it would not be able to work because of the conditions on the TM to halt when reject is instanced. This would cause the algorithm to be decidable, and Turning recognizable. Decidable because it handles all other strings that are inputted.

Answer:. (You need to say what the problem X mentioned above is, and whether you can conclude whether L is decidable/undecidable/Turing-recognizable/ not-Turing recognizable)

Exercise 3. Let

$$A = \{< M_1 >| M_1 \text{ is a Turing machine that does not accept } 0\}.$$

Explain what is wrong in the following alleged reduction $A_{TM} \leq_m A$.

Transform $< M, w >$ into the following Turing machine M_1:

M_1 on input x:

Simulate M on w and

if M accepts w, then M_1 enters the state q_{reject}.

if M does not accept w, then M_1 enters the state q_{accept}.

Answer: (The error can be that either the transformation $< M, w > 7\rightarrow < M_1 >$ is not computable, or the transformation does not map yes-instances of A_{TM} into yes-instances of A, or the transformation does not map no-instances of A_{TM} into no-instances of A.)

- We use a proof by contradiction. Suppose ATm is decided by some TM H, so H accepts <M,W> if TM M accepts W , and H rejects <M,W > if TM M doesn’t accept W .

<M,W > → [H] , accept if <M,W > ∈ ATM. and reject if <M,W > not subset of ATM.

another TM D using H as a subroutine.

So D takes as input any encoded TM <M>, then feeds <M,<M>> as input into H, and finally outputs the opposite of what H outputs. Because D is a TM, we can feed <D> as input into D.

that D accepts <D> if D doesn’t accept <D>, which is impossible. Thus, ATM must be undecidable.

Suppose there exists a TM H that decides ATM. TM H takes input <M,W >, where M is a TM and W is a string . If TM M accepts string W , then <M, W > ∈ ATM and H accepts input <M, W >. If TM M does not accept string W , then <M,W > not subset of ATM and H rejects input <M,W >. Consider the language L = { <M> | M is a TM that does not accept <M> }. Now construct a TM D for L using TM H as a subroutine:

D = “On input <M>, where M is a TM:

1. Run H on input <M,<M>>.

2. If H accepts, reject. If H rejects, accept.”

If we run TM D on input <D>, then D accepts <D> if and only if D doesn’t accept <D>. Because this is impossible, TM H must not exist, so ATM is undecidable.

ATM is Turing-recognizable in other side because,

The universal TM U recognizes ATM, where U is defined as follows:

U = “On input <M, w>, where M is a TM and w is a string:

1. Run M on W .

2. If M accepts W , accept; if M rejects W , reject.”

After that we can say, U only recognizes ATM and does not decide ATM Because when we run M on W , there is the possibility that M neither accepts nor rejects W but rather loops on W .