Theory of Computation
CSCI 341, Fall 2016

Recitation 8
2016-11-08

From Sipser's textbook

- Exercise 5.3 page 239
- Problem 5.12 page 239

  Solution:

  $$L = \{< M > \mid M \text{ is a TM, such that for all } w \text{ M ever writes a blank over a non-blank}\}$$

  For any Turing machine M it is possible to construct a Turing machine M' that works on a new blank symbol $\sqcup'$ instead of the usual $\sqcup$ symbol. To do that you can change the transitions of the form $(a \to \sqcup, m)$ to $(a \to \sqcup', m)$ for any $a \in Sigma$, and also add transitions from every state to itself $(\sqcup \to \sqcup', S)$ to make sure that if M creates a $\sqcup$, M' will immediately change it to $\sqcup'$.

  Suppose L is decided by a Turing Machine R. Let's define a Turing Machine S that decides A_{TM}.

```
S = " on input <M,w>:
      1) construct <M'> that works another blank symbol.
      2) construct M1 defined as:
         "on input x:
          1) run M' on w
          2) if M' ever accept
             then perform the specific transition (* -> blank, R)
             else loop"
      3) Run R on M1
      4) if R accepts then accepts
      5) if R rejects then rejects"
```

  Assuming R decides L, then S decides A_{TM}. Since A_{TM} is undecidable we conclude that L is undecidable.

- Problem 5.13 page 239 Solution:

  $$L = \{< M > \mid M \text{ is a TM, such that } \exists q \in Q, \forall w, \text{M never enters state} q \text{ when run on } w\}$$

  For any Turing machine M it is possible construct a Turing machine M' with the same behaviour of as M with respect to reaching the state q_{accept} and such that M', by design, reaches all the states except

q_{accept}. We can do that by creating extra transitions that traverse all the states one after another and go back to $qstart$ and then start executing M normally.

Suppose L is decided by a Turing Machine R. Let's define a Turing Machine S that decides A_{TM}.

```
S = " on input <M,w>:
      1) construct <M'> equivalent to <M> that traverse
         all the states for sure except qaccept.
      2) construct M1 the same as M' except that there is a transition
         from qaccept to a new state qaccept' and qaccept' is the new
         accept state of M1.
      3) Run R on M1
      4) if R accepts then accepts
      5) if R rejects then rejects"
```

Assuming R decides L, then S decides A_{TM}. Since A_{TM} is undecidable we conclude that L is undecidable.

- Problem 5.14 page 240 Solution:

$L = \{< M, w > \mid M$ is a TM, such that on input w, M attempts to move left on the left-most tape cell$\}$

For any Turing machine M it is possible to construct a Turing machine M' equivalent to M, such that M' is never going to attempt to move left on the left-most tape cell. To do that, we add a specific symbol # to Σ that corresponds to the leftmost character on the tape and add transitions $(\# \rightarrow \#, Right)$ to each state in M'. M on any input w behaves as M' on $\#w$. By construction M' never attempts to move left on the left-most character #.

Suppose L is decided by a Turing Machine R. Let's define a Turing Machine S that decides A_{TM}.

```
S = " on input <M,w>:
      1) construct <M'> that does not ever try to move left on the leftmost tape cell.
      2) construct M1 defined as:
         "on input x:
          1) run M' on #w
          2) if M' ever accept then keep moving left"
      3) Run R on M1
      4) if R accepts then accepts
      5) if R rejects then rejects"
```

Assuming R decides L, then S decides A_{TM}. Since A_{TM} is undecidable we conclude that L is undecidable.

- Problem 5.17 page 240
- Problem 5.19 page 240
- Problem 5.28 page 241