$\sim$ Assignment 3 Computational Models Spring 2022 $\sim$

Saar Barak

Exercise 1

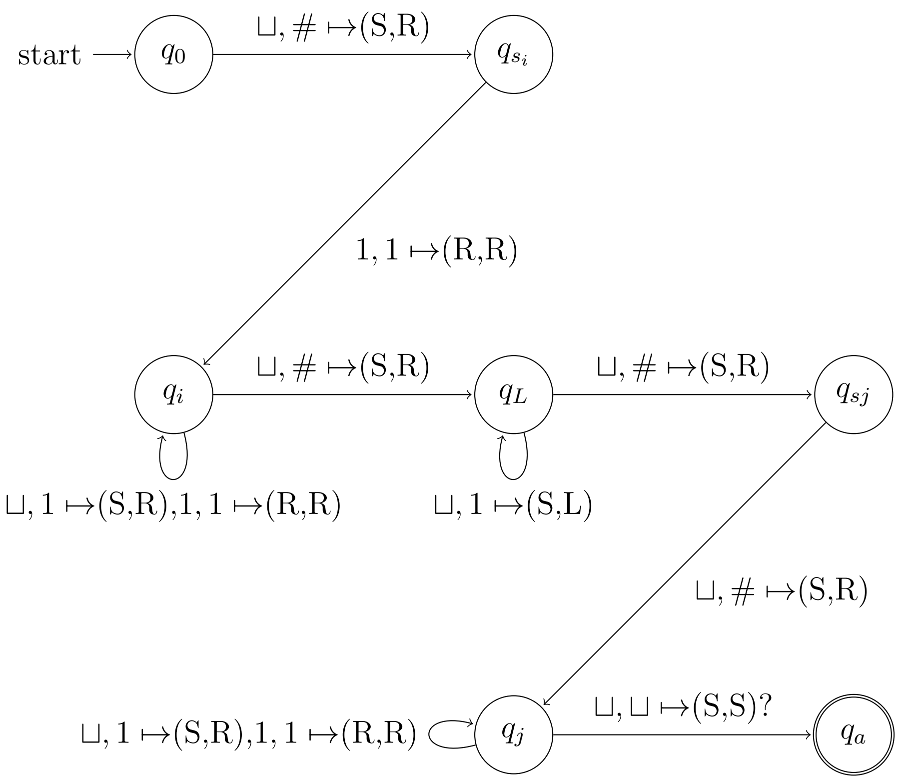
Considering the following language

$$\mathcal{L} = \{\#1^n\#x_1 \dots x^n : \exists i \neq j \text{ s.t } x_i \neq x_j\}$$

Lets describe an 2-tape NTM $\mathcal{M}$ that allow the head to stay at the same place, that recive input $\#1^n\#(n > 1)$ and implements the first part of decide $\mathcal{L}$

$$Q = \{q_0, q_a, q_r\}, \Sigma = \{1, \#\}, \Gamma = \{1, \#, \sqcup\}$$

and define δ :



The idea behind the way I generated δ is first to verify we start "legal" sequence i.e $i \neq j$ and $i, j > 0$ and split non-deterministic for any i, j the NTM run

Exercise 2

(a)

For DFA $A = (Q, \Sigma, \delta, q_0, F)$. lets describe a 2-tape TM M that accept $\mathcal{L}(A)$

$$Q = \{q_m, q_a, q_r\}, \Sigma = \Sigma, \Gamma = \Sigma \cup \{\{q \in Q\}, \sqcup\}, q_0 = q_m$$

$$\delta(q, (\sigma_1, \sigma_2)) = \begin{cases} (q_m, (\sigma_1, \delta(q_0, \sigma_1), (R, L)) & \text{if } \sigma_1 \in \Sigma \wedge \sigma_2 = \sqcup \\ (q_m, (\sigma_1, \delta(\sigma_1, \sigma_2)), (R, L)) & \text{if } \sigma_1 \in \Sigma \wedge \sigma_2 = q \in Q/F \\ (q_a, (\sigma_1, \sigma_2), (R, L)) & \text{if } \sigma_1 = \sqcup \wedge \sigma_2 = q \in F \\ (q_r, (\sigma_1, \sigma_2), (R, L)) & \text{if } \sigma_1 = \sqcup \wedge \sigma_2 = q \notin F \end{cases}$$

The idea is to save the last visted state on of A.

(b)

For TM $M = (Q, \Sigma, \delta, q_0, F)$. lets describe a 2-tape TM M_2 that accept $\mathcal{L}(M)$

$$Q = \{q_{m2}, q_a, q_r\}, \Sigma = \Sigma, \Gamma = \Sigma \cup \{q \in Q\}, q_0 = q_{m2}$$

Lets define $\hat{q}, \hat{\sigma}, \hat{D}$ to be the triplet such that $\delta(q, \sigma) \longmapsto (\hat{q}, \hat{\sigma}, \hat{D})$

$$\delta(q, (\sigma_1, \sigma_2)) = \begin{cases} (q_m, (\hat{\sigma}, \hat{q}), (\hat{D}, L)) & \text{if } \sigma_1 \in \Sigma \wedge \sigma_2 = \sqcup \\ (q_m, (\hat{\sigma}, \hat{q}), (\hat{D}, L)) & \text{if } \sigma_1 \in \Sigma \wedge \sigma_2 = q \in Q \\ (q_a, (\sigma_1, q_a), (R, L)) & \text{if } \sigma_2 = q_a \wedge \forall \sigma_1 \\ (q_r, (\sigma_1, q_r), (R, L)) & \text{if } \sigma_2 = q_r \wedge \forall \sigma_1 \end{cases}$$

The idea is kind of same as above but now we track the direction of the head of M

Exercise 3

Proposition 1 *Disprove $\mathcal{RE}$ is not closed under complement.*

For $L, \overline{L}$, if both are semi-decidable $\Rightarrow$ both are decidable, (can just flip the nation reject accept) $\Rightarrow$ if we assume $\mathcal{RE}$ closed under complement, $\forall L \in \mathcal{RE}, \overline{L} \in \mathcal{RE} \Rightarrow$ we get that $\mathcal{RE} = \mathcal{R} \Rightarrow \clubsuit$

Proposition 2 *co$\mathcal{RE}$ is closed under intersection.*

By detention $L \in co\mathcal{RE}$ if $\overline{L} \in \mathcal{RE}$. using the fact that $\mathcal{RE}$ Closures under union we can apply De-Morgan law for some $\overline{L_1}, \overline{L_2} \in co\mathcal{RE}$ which lead us to :

$$L_1, L_2 \in \mathcal{RE} \Rightarrow L_1 \cup L_2 \in \mathcal{RE} \Rightarrow \overline{L_1 \cup L_2} \in co\mathcal{RE} \Rightarrow \overline{L_1} \cap \overline{L_2} \in co\mathcal{RE}$$

Proposition 3 *$\mathcal{R}$ is closed under Kleene star.*

w.l.o.g $\mathcal{L} \in \mathcal{R}$ such that exist TM T that decide $\mathcal{L}$.now we can build an TM M^* that accept L^* for input x .

- if $x = \epsilon$ Accept.
- partition $|x| = n$ to any combination possible ways lets say $x_{power} := (\{x_1\}, \{x_2\} \dots \{x_{2^{n-1}}\})$
- run L for any $\forall w \in \{x_i\}$, if for some x_i L accept **all** $w \in \{x_i\}$ Accept.
- else Reject.

Proposition 4 *$\mathcal{RE}$ is closed under Prefix, but $\mathcal{R}$ is not closed under Prefix,*

i. Since $\mathcal{L} \in \mathcal{RE}$, there exists an enumerator $f_{\mathcal{L}}$. Its will be sufficient to describe f_{prefix} to proof that $\mathcal{L} \in \mathcal{RE} \Rightarrow$ Prefix($\mathcal{L}$) $\in \mathcal{RE}$ Now lets built new f_{prefix}.

$$f_{prefix} = w := \{\sigma_1 \dots \sigma_k\} \forall k : 0 < k < |w| \text{ for any } f_{\mathcal{L}} = w$$

the idea is to use $f_{\mathcal{L}}$ output and apply Prefix on any world to create f_{prefix}

ii. Lets assume $\mathcal{R}$ closed under Prefix. and lets look at my favourite TM M_F for $L_f \in \mathcal{RE}/\mathcal{R}$. now lets construct new $\hat{L}$ consisting of strings from the encode of M_F and # i.e M_F# now I claim that $\hat{L}$ Accept only when its see # ,otherwise its Reject. hence exist some deterministic TM that Reject/Accept. $\hat{L} \in \mathcal{R}$, but Prefix($\hat{L}$) $= L_f \notin \mathcal{R} \Rightarrow \clubsuit$

Exercise 4

Define Size($O(n)$):

$$\text{Size}(O(n)) = \{\mathcal{L} : \exists \mathcal{C} := \{C_n\}_{n\in N} \text{ s.t } \mathcal{L}(\mathcal{C}) = \mathcal{L} \wedge |C_n| \in O(n) \forall n \in N\}$$

i. Lets look at the unary language $\mathcal{L} \subseteq \{1^n : n \in N\}$, its immediate that $\mathcal{L} \in \text{Size}(O(n))$ since its needs to have a single "hardwired" bit for indicating 1^n. now lets look at the language $\mathcal{L}_{\mathcal{U}}$

$$\mathcal{L}_{\mathcal{U}} := \{1^n | \text{ The Turing machine encode to n halts on } \epsilon.\}$$

Hence its immediate from Rice's Theorem, or the fact that $H_{TM\epsilon} \leq_m \mathcal{L}_{\mathcal{U}}$ that $\mathcal{L}_{\mathcal{U}} \notin \mathcal{R}$ and $\mathcal{L}_{\mathcal{U}} \in \text{Size}(O(n))$

ii. We can look at some recursively define $\mathcal{L} \in \mathcal{R}$ for example:

$$\mathcal{L} = \{1^{2^n} : n \geq 0\}$$

Since we will need circuit for each possible input length, we may need exponential size circuit (in terms of n) to accept words in the language. and the following holds that $\mathcal{L} \notin \text{Size}(O(n))$

Exercise 5

(A) **Prove:** If $\mathcal{L}_1 \leq_m \mathcal{L}_2$ and $\mathcal{L}_2 \leq_m \mathcal{L}_3$, then $\mathcal{L}_1 \leq_m \mathcal{L}_3$.
Define $f_{1\to 2}, f_{2\to 3}$ to be a computable mapping redaction functions. now lets $w \in \mathcal{L}_1$, and by dentition we get :

$$w \in \mathcal{L}_1 \Leftrightarrow f_{1\to 2}(w) \in \mathcal{L}_2 \Leftrightarrow f_{2\to 3}(f_{1\to 2}(w)) \in \mathcal{L}_3$$

(B) **Disprove:** If $\mathcal{L}_1 \leq_m \mathcal{L}_2$ and $\mathcal{L}_2 \leq_m \mathcal{L}_1$, then $\mathcal{L}_1 = \mathcal{L}_2$.
for H_{TM}, A_{TM} we know that $H_{TM} \leq_m A_{TM}$ and $A_{TM} \leq_m H_{TM}$ but $A_{TM} \neq H_{TM} \Rightarrow \clubsuit$

(C) **Disprove:** If $\mathcal{L}_1 \subseteq \mathcal{L}_2$ then $\mathcal{L}_1 \leq_m \mathcal{L}_2$.
Lets look at $\mathcal{L} = \Sigma^*$,$\mathcal{L} \in \mathcal{R}$ since any other $\mathcal{L}' \subseteq \mathcal{L}$. if we assume that $\forall \mathcal{L}' \leq_m \mathcal{L}$ its will lead to $\forall \mathcal{L}' \in \mathcal{RE} \Rightarrow \clubsuit$

(D) **Disprove:** For every $\mathcal{L}_1, \mathcal{L}_2$, then $\mathcal{L}_1 \leq_m \mathcal{L}_2$ or $\mathcal{L}_2 \leq_m \mathcal{L}_1$
Lets look at $A_{TM}, \overline{A_{TM}}$. If we assume $\overline{A_{TM}} \leq_m A_{TM}$ Since $\overline{A_{TM}} \notin \mathcal{RE}$ its will lead to $A_{TM} \in \mathcal{R} \Rightarrow \clubsuit$,
and if $A_{TM} \leq_m \overline{A_{TM}}$ Since $A_{TM} \notin$ co-$\mathcal{RE} \Rightarrow \overline{A_{TM}} \in \mathcal{R} \Rightarrow \clubsuit$

(E) **Prove:** If $\mathcal{L}$ is regular, then $\mathcal{L} \leq_m HALT$
Lets A be an DFA s.t $L(A) = \mathcal{L}$ when $\mathcal{L}$ is regular. based on 2(a) we define some TM M that accept $L(A)$ and Let M_{loop} be TM that loop forever for any input. hence we can define computable f :

$$f(w) = \begin{cases} \langle M, w \rangle & \text{if } w \in M \\ \langle M_{loop}, w \rangle & \text{if } w \notin M \end{cases}$$

Which imply $\mathcal{L} \leq_m HALT$

Exercise 6

$$\textbf{(a)}\ \mathcal{L} = \{\langle M \rangle : M \text{ is a TM and } |L(M)| > 10\} \in \mathcal{RE}/\mathcal{R}$$

Lets describe algorithm A_{10} that accept $\mathcal{L}$

Algorithm 1 A_{10} on input $\langle M \rangle$

Require: $\langle M \rangle$ is a valid encode of TM
  else **Reject**
  $k \leftarrow 1$
  while $k < \infty$ **do**
    counter $\leftarrow 0$
    for $i \leftarrow 1$ **to** k **do**
      simulate M on w_i for k steps $\triangleright$ w_i generated from Σ of M
      if M accept **then**
        counter + 1
      end if
    end for
    if counter > 10 **then Accept**
    end if
  end while

Its implement that we cover any optional input on M and while discovered more then 10 worlds in $\mathcal{L}$ we accept. hence $\mathcal{L} \in \mathcal{RE}$
Now lets proof $\mathcal{L} \notin \mathcal{R}$ using reduction from HALT$\leq_m \mathcal{L}$. now lets define $\mathcal{M}$

Algorithm 2 $\mathcal{M}$ on input $\langle M, \hat{w} \rangle$.

Require: $\langle M, \hat{w} \rangle$ is a valid encode of TM and word
  else **Reject**
  Ignore $\hat{w}$
  Write M and w on the tape
  run M on w **Accept** if M halt
  else **Reject**

I claim that exists maping function from H_{TM} to $\mathcal{L}$ s.t:

- If $\langle M, \hat{w} \rangle \in H_{TM} \Rightarrow M(\hat{w})$ halt $\Rightarrow \mathcal{M}$ accept any input $\Rightarrow |L(\mathcal{M})| > 10 \Rightarrow \mathcal{M} \in \mathcal{L}$
- If $\langle M, \hat{w} \rangle \notin H_{TM} \Rightarrow M(\hat{w})$ not halt $\Rightarrow \mathcal{M}$ dont accept any input $\Rightarrow |L(\mathcal{M})| < 10 \Rightarrow \mathcal{M} \notin \mathcal{L}$

$$f(\langle M, w \rangle) = \begin{cases} \langle \mathcal{M} \rangle & \text{if } \langle M, w \rangle \text{ is valid input of TM and word} \\ \langle \mathcal{M}_{favourite} \rangle & \text{otherwise} \end{cases}$$

Where $\mathcal{M}_{favourite}$ is my favourite TM that hold $|L(\mathcal{M}_{favourite})| = 10$

(B) $\mathcal{L} = \{M : \langle M \rangle$ is a TM that accepts 1 but does not accept 0$\} \in \overline{\mathcal{RE} \cup \text{co-}\mathcal{RE}}$

Lets look at

$$\mathcal{L}_{1\wedge\neg 2} = \{\langle M_1, w_1, M_2, w_2\rangle | w_1 \in L(M_1) \wedge w_2 \notin L(M_2)\}$$

since we know that $\mathcal{L}_{1\wedge\neg 2} \in \overline{\mathcal{RE} \cup \text{co-}\mathcal{RE}}$ we can can apply mapping reduction. Now lets construct a TM $\mathcal{M}_{1\wedge\neg 2}$ that work as follow:

Algorithm 3 $\mathcal{M}_{1\wedge\neg 2}$ on input w.

If $w = 0$ **Then** simulate M_2 on w_2
If $w = 1$ **Then** simulate M_1 on w_1
If $w \neq 0 \wedge w \neq 1$ **Accept**

I claim that exists maping function f from $\mathcal{L}_{1\wedge\neg 2}$ to $\mathcal{L}$ s.t:

$$f(\langle M_1, w_1, M_2, w_2\rangle) = \begin{cases} \langle \mathcal{M}_{1\wedge\neg 2}\rangle & \text{if } \langle M_1, w_1, M_2, w_2\rangle \text{ is valid input of 2 TMs and 2 words} \\ \langle \mathcal{M}_0 \rangle & \text{otherwise} \end{cases}$$

where $\mathcal{M}_0$ is some TM that accept 0

- If $\langle M_1, w_1, M_2, w_2\rangle \in \mathcal{L}_{1\wedge\neg 2} \Rightarrow w_1 \in L(M_1) \wedge w_2 \notin L(M_2)$
  $\Rightarrow 1 \in L(\mathcal{M}_{1\wedge\neg 2}) \wedge 0 \notin L(\mathcal{M}_{1\wedge\neg 2}) \Rightarrow \langle \mathcal{M}_{1\wedge\neg 2}\rangle \in \mathcal{L}$
- If$\langle M_1, w_1, M_2, w_2\rangle \notin \mathcal{L}_{1\wedge\neg 2} \Rightarrow w_1 \in L(M_1) \downarrow w_2 \notin L(M_2)$
  $\Rightarrow 1 \in L(\mathcal{M}_{1\wedge\neg 2}) \downarrow 0 \notin L(\mathcal{M}_{1\wedge\neg 2}) \Rightarrow \langle \mathcal{M}_{1\wedge\neg 2}\rangle \notin \mathcal{L}$

(c) $\mathcal{L} = \{\langle M_1, M_2\rangle : M_1, M_2$ are TMs and $\mathcal{L}(M_1) \cap \mathcal{L}(M_2) = \emptyset\} \in \text{co-}\mathcal{RE}/\mathcal{R}$

Lets describe an algorithm $\mathcal{A}$ that reject for $\mathcal{L}$

Algorithm 4 $\mathcal{A}$ on input $\langle M_1, M_2\rangle$

$k \leftarrow 1$
while $k < \infty$ **do**
  for $i \leftarrow 1$ **to** k **do**
    simulate M_1 and M_2 on w_i for k steps
    If both accept then **Reject**
  end for
end while

Hence its hold that if both M_1, M_2 accept same word then $\mathcal{L}(M_1) \cap \mathcal{L}(M_2) \neq \emptyset$, $\mathcal{L} \in \text{co-}\mathcal{RE}$. Now lets proof $\mathcal{L} \notin \mathcal{R}$ using reduction from $E_{TM} \leq_m \mathcal{L}$. and its will be sufficient to see that $\overline{E_{TM}} \leq_m \overline{\mathcal{L}}$ now lets define ND-TM $\mathcal{M}_E$

Algorithm 5 $\mathcal{M}_E$ on input $\langle \hat{M} \rangle$.

Guess w from Σ^*
Simulate non-deterministic M on w

And let define f s.t:

$$f(\langle M \rangle) = \begin{cases} \langle \mathcal{M}_E, \mathcal{M}_{ACC} \rangle & \text{if } \langle M \rangle \text{ is valid input} \\ \langle \mathcal{M}_E, \mathcal{M}_{\emptyset} \rangle & \text{otherwise} \end{cases}$$

Where $\mathcal{M}_{ACC}$ is TM that accept any input, and $\mathcal{M}_{\emptyset}$ is TM that don't accept any word.

- If $\langle M \rangle \in \overline{E_{TM}} \Rightarrow \mathcal{L}(M) \neq \emptyset \Rightarrow \mathcal{L}(\mathcal{M}_E) \cap \mathcal{L}(\mathcal{M}_{ACC}) \neq \emptyset \Rightarrow \langle \mathcal{M}_E, \mathcal{M}_{ACC} \rangle \in \overline{\mathcal{L}}$
- If $\langle M \rangle \notin \overline{E_{TM}} \Rightarrow \mathcal{L}(M) = \emptyset \Rightarrow \mathcal{L}(\mathcal{M}_E) \cap \mathcal{L}(\mathcal{M}_{ACC}) = \emptyset \Rightarrow \langle \mathcal{M}_E, \mathcal{M}_{ACC} \rangle \notin \overline{\mathcal{L}}$.

(d) $\mathcal{L} = \{\langle M_1, M_2 \rangle : M_1, M_2 \text{ are TMs and } \mathcal{L}(M_1) \subseteq \mathcal{L}(M_2)\} \in \overline{\mathcal{RE} \cup \text{co-}\mathcal{RE}}$

Lets look at

$$EQ = \{\langle M_1, M_2 \rangle : \text{are TMs and } \mathcal{L}(M_1) = \mathcal{L}(M_1)\}$$

Since we know that $EQ \in \overline{\mathcal{RE} \cup \text{co-}\mathcal{RE}}$ we can can apply mapping reduction $EQ \leq_m \mathcal{L}$. Now lets construct a TM $\mathcal{M}_{EQ}$ that work as follow:

Algorithm 6 $\mathcal{M}_{EQ}$ on input $\langle M_1, M_2 \rangle$.

Run $M_{\mathcal{L}}\langle M_1, M_2 \rangle$ $\triangleright$ we assume that exist some TM $M_{\mathcal{L}}$ that halt
if $M_{\mathcal{L}}$ reject **then Reject**
end if
if $M_{\mathcal{L}}$ accept **then** simulate $M_{\mathcal{L}}\langle M_2, M_1 \rangle$
  Answer like $M_{\mathcal{L}}\langle M_2, M_1 \rangle$
end if

We can see that $\mathcal{M}_{EQ}$ decides EQ, since $\mathcal{M}_{EQ}$ accept $\Leftrightarrow \mathcal{L}(M_1) \subseteq \mathcal{L}(M_2) \wedge \mathcal{L}(M_2) \subseteq \mathcal{L}(M_1)$. from the way we construct $\mathcal{M}_{EQ}$ its guaranteed that will act like any $\mathcal{M}_{\mathcal{L}}$ we plug in (halt/loop). Hence if we assume $\mathcal{M}_{EQ}$ halt and accept $\Rightarrow EQ \in \mathcal{R}$. on the other hand if we assume $\mathcal{M}_{EQ}$ halt and reject $\Rightarrow EQ \in \text{co-}\mathcal{RE}$
both lead to contradiction ♣