Introduction to the Theory of Computation 2020 — Final exam

Solutions (may be updated)

Problem 1 (15 pts). Let $\Sigma = \{0\}$. Consider

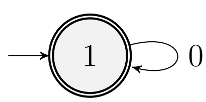
$$EQ_{\text{DFA}} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}.$$

Recall that to prove EQ_{DFA} is decidable, we construct DFA C such that

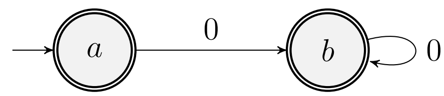
$$L(C) = \left(L(A) \cap \overline{L(B)}\right) \cup \left(\overline{L(A)} \cap L(B)\right),$$

and then run T on C where T decides $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$. Now consider

$\boxed{A}$



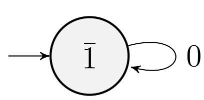
$\boxed{B}$



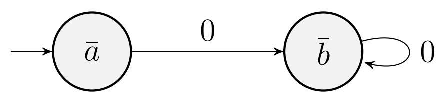
Construct a DFA C with the above A and B, and see if T accepts or rejects $\langle C\rangle$ by running the algorithm in Theorem 4.4 of the textbook. For $\cap$, use a procedure similar to the one for $\cup$ in Theorem 1.25 of the textbook.

Solution.

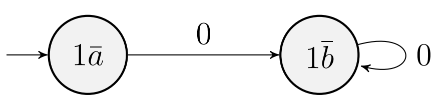
$\boxed{\text{DFA recognizing } \overline{L(A)}}$



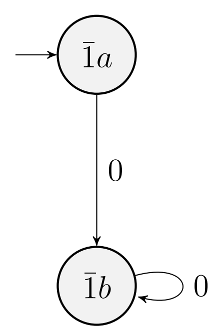
$\boxed{\text{DFA recognizing } \overline{L(B)}}$



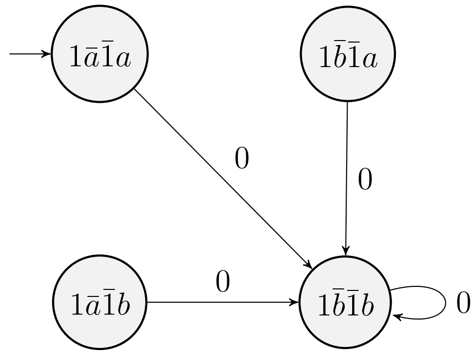
$\boxed{\text{DFA recognizing } L(A) \cap \overline{L(B)}}$



$\boxed{\text{DFA recognizing } \overline{L(A)} \cap L(B)}$

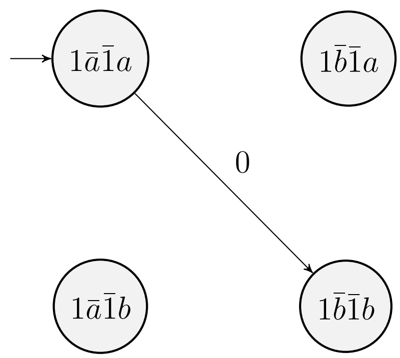


$$\boxed{\text{DFA } C \text{ recognizing } \Big(L(A) \cap \overline{L(B)}\Big) \cup \Big(\overline{L(A)} \cap L(B)\Big)}$$



Now we test T on C as follows: mark $1\bar{a}\bar{1}a$; mark $1\bar{b}\bar{1}b$; no new states can be marked. Because no accept state is marked, accept.

Common mistake: For some unknown reason, some get the following diagram

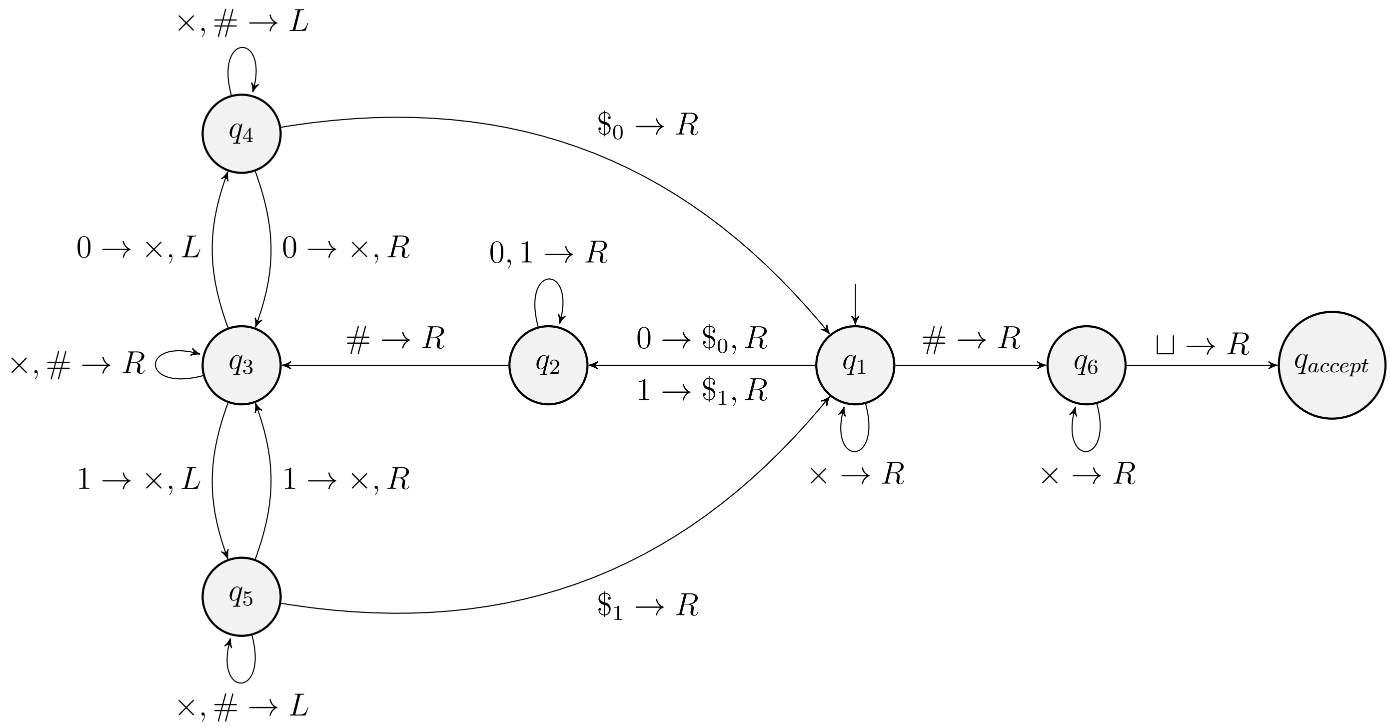


But remember that we have a DFA!

Problem 2 (30 pts). Let $\Sigma = \{0, 1, \#\}$. In the previous exam we consider the language

$$D = \{w\#w^R \mid w \in \{0,1\}^*\}.$$

(a) (20 pts) In Problem 5(b) of the previous exam, we design the following TM that decides D.

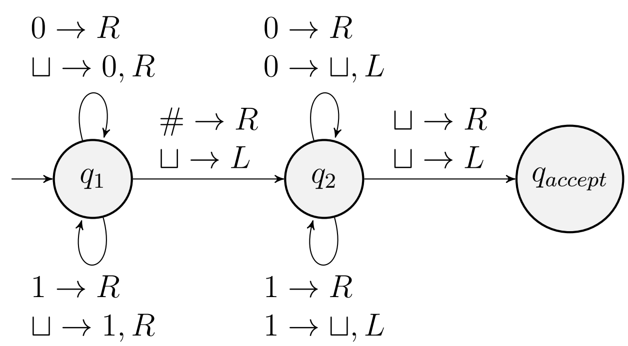


Assume the input is $w\#w^R$ with

$$|w| = n.$$

Count the number of steps needed as a function of n. You must explain the calculation instead of just giving a function of n. Note that we want the exact number instead of just an estimation. Verify your solution by simulating on 01#10.

(b) (10 pts) Redo (a) by considering the following two-tape TM designed in Problem 5(c) of the previous exam.



Verify your solution by simulating on 01#10.

Solution.

(a) If $n = 0$, then it takes 2 steps to go from q_1 to q_6 and then to q_{accept}. Otherwise $n \geq 1$, and the steps are counted as follows.

- 1 step from q_1 to q_2
- $n - 1$ steps when looping on q_2

- 1 step from q_2 to q_3
- consider one iteration of the loop from

  $$\cdots \# q_3 \underbrace{\times \cdots \times}_{k} 0 \cdots$$

  to

  $$\cdots \# q_3 \underbrace{\times \cdots \times}_{k+1} 0 \cdots$$

  (Note that we can assume the next digit is 0 by symmetry between 0 and 1.) We need

  $$\begin{array}{ll} k \text{ steps to} & \cdots \# \underbrace{\times \cdots \times}_{k} q_3 0 \cdots \\ 1 \text{ step to} & \cdots \# \underbrace{\times \cdots \times}_{k-1} q_4 \times \times \cdots \\ k \text{ steps to} & \cdots 0 \underbrace{\times \cdots \times}_{k} q_4 \# \underbrace{\times \cdots \times}_{k} \times \cdots \\ k+1 \text{ steps to} & \cdots q_4 0 \underbrace{\times \cdots \times}_{k} \# \underbrace{\times \cdots \times}_{k} \times \cdots \\ 1 \text{ step to} & \cdots \times q_3 \underbrace{\times \cdots \times}_{k} \# \underbrace{\times \cdots \times}_{k} \times \cdots \\ k+1 \text{ steps to} & \cdots \times \underbrace{\times \cdots \times}_{k} \# q_3 \underbrace{\times \cdots \times}_{k} \times \cdots \end{array}$$

  Therefore, before the last iteration for handling $\$_0$ or $\$_1$, the number of steps is

  $$\sum_{k=0}^{n-2} (4k+4).$$

  Now consider the last iteration from

  $$\$_0 \underbrace{\times \cdots \times}_{n-1} \# q_3 \underbrace{\times \cdots \times}_{n-1} 0$$

  to

  $$q_4 \$_0 \underbrace{\times \cdots \times}_{n-1} \# \underbrace{\times \cdots \times}_{n}$$

  Similar to the above discussion, we need

  $$(n-1) + 1 + (n-1) + n = 3n - 1$$

  steps.
- 1 step from q_4/q_5 back to q_1
- $n-1$ steps when looping on q_1
- 1 step from q_1 to q_6
- n steps when looping on q_6
- 1 step from q_6 to q_{accept}

The number of steps is thus

$$n+1+\sum_{k=0}^{n-2}(4k+4)+3n-1+2n+2=2n^2+4n+2=2(n+1)^2.$$

Simulation on 01#10 takes $2(2+1)^2=18$ steps:

$$\begin{array}{lllll}
 & q_1 01\#10 & \Rightarrow \$_0 q_2 1\#10 & \Rightarrow \$_0 1 q_2 \#10 & \Rightarrow \$_0 1\# q_3 10 \\
\Rightarrow & \$_0 1 q_5 \# \times 0 & \Rightarrow \$_0 q_5 1\# \times 0 & \Rightarrow \$_0 \times q_3 \# \times 0 & \Rightarrow \$_0 \times \# q_3 \times 0 \\
\Rightarrow & \$_0 \times \# \times q_3 0 & \Rightarrow \$_0 \times \# q_4 \times \times & \Rightarrow \$_0 \times q_4 \# \times \times & \Rightarrow \$_0 q_4 \times \# \times \times \\
\Rightarrow & q_4 \$_0 \times \# \times \times & \Rightarrow \$_0 q_1 \times \# \times \times & \Rightarrow \$_0 \times q_1 \# \times \times & \Rightarrow \$_0 \times \# q_6 \times \times \\
\Rightarrow & \$_0 \times \# \times q_6 \times & \Rightarrow \$_0 \times \# \times \times q_6 \sqcup & \Rightarrow \$_0 \times \# \times \times \sqcup q_{accept} \sqcup &
\end{array}$$

(b) The steps are counted as follows:

- n steps when looping on q_1
- 1 step from q_1 to q_2
- n steps when looping on q_2
- 1 step from q_2 to q_{accept}

The number of steps is thus $2n+2=2(n+1)$.

Simulation on 01#10 takes $2(2+1)=6$ steps:

$$\begin{array}{llll}
 & q_1 \begin{smallmatrix} 0 & 1 & \# & 1 & 0 \\ \sqcup & \sqcup & \sqcup & \sqcup & \sqcup \end{smallmatrix} & \Rightarrow \begin{smallmatrix} 0 \\ 0 \end{smallmatrix} q_1 \begin{smallmatrix} 1 & \# & 1 & 0 \\ \sqcup & \sqcup & \sqcup & \sqcup \end{smallmatrix} & \Rightarrow \begin{smallmatrix} 0 & 1 \\ 0 & 1 \end{smallmatrix} q_1 \begin{smallmatrix} \# & 1 & 0 \\ \sqcup & \sqcup & \sqcup \end{smallmatrix} \\
\Rightarrow & \begin{smallmatrix} 0 & & 1 & \# & q_2 & 1 & 0 \\ 0 & q_2 & 1 & \sqcup & & \sqcup & \sqcup \end{smallmatrix} & \Rightarrow \begin{smallmatrix} & 0 & 1 & \# & 1 & q_2 & 0 \\ q_2 & 0 & \sqcup & \sqcup & \sqcup & & \sqcup \end{smallmatrix} & \Rightarrow \begin{smallmatrix} & 0 & 1 & \# & 1 & 0 & q_2 & \sqcup \\ q_2 & \sqcup & \sqcup & \sqcup & \sqcup & \sqcup & & \sqcup \end{smallmatrix} \\
\Rightarrow & \begin{smallmatrix} & 0 & 1 & \# & 1 & 0 & \sqcup & q_{accept} & \sqcup \\ q_{accept} & \sqcup & \sqcup & \sqcup & \sqcup & \sqcup & \sqcup & & \sqcup \end{smallmatrix} & &
\end{array}$$

Problem 3 (20 pts). Consider positive functions $f, g : \mathbb{N} \to \mathbb{R}^+$. In our slides we respectively define

$$f(n) = O(g(n))$$

and

$$f(n) = o(g(n))$$

by

$$\begin{aligned} &\exists c > 0, \exists n_0, \forall n \geq n_0, f(n) \leq cg(n) \\ &\forall c > 0, \exists n_0, \forall n \geq n_0, f(n) \leq cg(n). \end{aligned} \tag{1}$$

We also define

$$f(n) = 2^{O(g(n))}$$

if

$$\exists c > 0, \exists n_0, \forall n \geq n_0, f(n) \leq 2^{cg(n)}$$

(a) (10 pts) We would like to define $f(n) = 2^{o(g(n))}$ by following the setting in (1):

$$\forall c > 0, \exists n_0, \forall n \geq n_0, f(n) \leq 2^{cg(n)} \tag{2}$$

We know that from (1), $f(n) = o(g(n))$ is the same as

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0.$$

Can you write the definition in (2) similarly by using limit?

(b) (10 pts) Give an example of $f(n)$ and $g(n)$ such that $f(n) = 2^{O(g(n))}$ but $f(n) \neq 2^{o(g(n))}$.

Solution.

(a) Noting that $f(n) \leq 2^{cg(n)} \iff \log_2 f(n) \leq cg(n)$, we can rewrite as

$$\lim_{n\to\infty} \frac{\log_2 f(n)}{g(n)} = 0.$$

Common mistake:

$$\lim_{n\to\infty} \frac{f(n)}{2^{g(n)}} = 0.$$

Example: $f(n) = 2^n, g(n) = 2n$. Then

$$\lim_{n\to\infty} \frac{\log_2 f(n)}{g(n)} = \lim_{n\to\infty} \frac{n}{2n} = \frac{1}{2} \neq 0,$$
$$\lim_{n\to\infty} \frac{f(n)}{2^{g(n)}} = \lim_{n\to\infty} \frac{2^n}{2^{2n}} = 0.$$

(b) $f(n) = 2^n, g(n) = n$. By taking $c = n_0 = 1$, we see that $f(n) = 2^{O(g(n))}$. To prove that $f(n) \neq 2^{o(g(n))}$, it is sufficient to calculate $\lim_{n\to\infty} \frac{\log_2 f(n)}{g(n)} = 1 \neq 0$. If you would like to use the definition in (2), you need to prove the opposite statement:

$$\exists c > 0, \forall n_0, \exists n \geq n_0, f(n) > 2^{cg(n)}.$$

Common mistake: Many failed to correctly prove this statement.

Problem 4 (10 pts). Consider $f(n) = e^{-n}, g(n) = \sin n + 2$. Check whether $f(n) = o(g(n))$ or not **by the definition in** (1). Thus to prove it you must find n_0 for every c, and to disprove it you must find one c such that no n_0 satisfies the condition.

Solution. Yes. Given any $c > 0$, by letting $n_0 = \max(1, \lceil -\ln c \rceil)$, we have

$$e^{n_0} \geq e^{-\ln c} = \frac{1}{c},$$

where the inequality is from the monotonicity of exp and the fact $x \leq \lceil x \rceil$ for any real x. Therefore

$$e^{-n} \leq e^{-n_0} \leq c \leq c(\sin n + 2),\ \forall n \geq n_0,$$

where the last inequality follows from $\sin n \geq -1$.

Problem 5 (25 pts). Let $\Sigma = \{0, 1\}$. Recall that the concatenation of two languages is defined by

$$A \circ B = \{xy \mid x \in A, y \in B\}.$$

(a) (5 pts) Prove that the class of context-free languages is closed under concatenation. In other words, show that if A, B are CFLs, then $A \circ B$ is also a CFL.

(b) (10 pts) Let

$$L_{conc} = \{\langle G_1, G_2, w\rangle \mid G_1, G_2 \text{ are CFGs, } w \in \Sigma^*, \text{ and } w \in L(G_1) \circ L(G_2)\}$$

Prove that L_{conc} is decidable by using the result of (a) and the property that A_{CFG} is decidable. Basically what you need to do is to construct a TM that decides L_{conc}.

(c) (10 pts) Now suppose we do not know (a). Can you prove L_{conc} is decidable in another way without using the result of (a)? You may still use the property that A_{CFG} is decidable. That is, here you need a different TM from that in (b). *Hint: Split w.*

Solution.

(a) Suppose G_1, G_2 are CFGs that generate A, B, respectively. Let $G_1 = (V_1, \Sigma, R_1, S_1), G_2 = (V_2, \Sigma, R_2, S_2)$. We assume $V_1 \cap V_2 = \emptyset$; otherwise rename the variables and modify R_1, R_2, S_1, S_2 accordingly. Then $G = (V, \Sigma, R, S)$ generates $A \circ B$, where $V = V_1 \cup V_2 \cup \{S\}$, and $R = R_1 \cup R_2 \cup \{S \to S_1 S_2\}$. To prove that $L(G) = A \circ B$, on one hand, for any $w = w_1 w_2 \in A \circ B$, $S_1 \overset{*}{\Rightarrow} w_1$ and $S_2 \overset{*}{\Rightarrow} w_2$ imply that $S \overset{*}{\Rightarrow} w_1 w_2$. Thus $A \circ B \subseteq L(G)$. On the other hand, for any $w \in L(G)$, the only rule from $S \to S_1 S_2$ means that $S \to S_1 S_2 \overset{*}{\Rightarrow} w$. Thus w can be split into $w_1 w_2$ with $S_1 \overset{*}{\Rightarrow} w_1$ and $S_2 \overset{*}{\Rightarrow} w_2$. Then $w \in A \circ B$, and thus $L(G) \subseteq A \circ B$. Finally we have $L(G) = A \circ B$.

(b) The TM described by the following procedure decides L_{conc}.

- On input $\langle G_1, G_2, w \rangle$, check the format is correct.
- Construct CFG G that generates $L(G_1) \circ L(G_2)$ by the procedure in (a).
- Run the decider S of A_{CFG} on input $\langle G, w \rangle$.
- If S accepts, accept; otherwise, reject.

(c) The TM described by the following procedure decides L_{conc}.

- On input $\langle G_1, G_2, w \rangle$, check the format is correct.
- Let $n = |w|$. For $i = 0, 1, \ldots, n$,
  - Split $w = xy$, where $|x| = i, |y| = n - i$.
  - Run the decider S of A_{CFG} on inputs $\langle G_1, x \rangle$ and $\langle G_2, y \rangle$.
  - If S accept both inputs, accept.
- Reject.