CS 341, Fall 2020
Solutions for Midterm 2, Hybrid

1. (a) True, by Corollary 3.18.

   (b) False. Theorem 4.11 shows that A_{TM} is undecidable, so no TM can decide A_{TM}.

   (c) True. Languages A and B are equal if and only if $\overline{A} \cap B = \emptyset$ and $A \cap \overline{B} = \emptyset$; see slide 4.13. Equivalently, Languages A and B are unequal if and only if $\overline{A} \cap B \neq \emptyset$ or $A \cap \overline{B} \neq \emptyset$.

   (d) False. The set $\mathcal{N} = \{1, 2, 3, \ldots\}$ is infinite and countable.

   (e) False. Every context-free language is decidable by Theorem 4.9, and every decidable language is Turing-recognizable because the definition of Turing-recognizable is less restrictive than the definition of decidable (also see slide 4.55). Thus, every context-free language is Turing-recognizable. and every decidable language is Turing-recognizable

   (f) False, by slide 4-38.

   (g) True. For example, the language A_{TM} is recognized by the universal TM, which is deterministic, so is also recognized by a nondeterministic TM by Corollary 3.18. But by Theorem 4.11, A_{TM} is undecidable, so there does not exist any TM (deterministic or nondeterministic) that decides A_{TM}. , by Theorem 3.16.

   (h) True, by slide 3-27.

   (i) False. $\overline{A_{\text{TM}}}$ is not Turing-recognizable by Corollary 4.23.

   (j) True. For example, let A be the set of positive integers, which is countable, and let $B = \Re$, which is uncountable. Also, we have that $A \subset B$.

2. (a) Yes, because $x \neq y$ with $x, y \in D$ implies that $f(x) \neq f(y)$.

   (b) No, because nothing in D maps to $d \in R$.

   (c) No, because f is not onto.

   (d) A language L_1 that is Turing-recognizable is recognized by a Turing machine M_1 that may loop forever on a string $w \notin L_1$. A language L_2 that is Turing-decidable is recognized by a Turing machine M_2 that always halts.

   (e) An algorithm is a Turing machine that always halts.

3. $q_1baba\#aab$ $xq_3aba\#aab$ $xaq_3ba\#aab$ $xabq_3a\#aab$ $xabaq_3\#aab$ $xaba\#q_5aab$ $xaba\#aq_{\text{reject}}ab$

4. (From slides 4-39 and 4-40). Let $\mathcal{L}$ be the collection of languages over an alphabet Σ, and let $\mathcal{B}$ be the set of infinite binary strings, which we know is uncountable (by a diagonalization argument, on slide 4-39). We will show that there is a correspondence between $\mathcal{L}$ and $\mathcal{B}$, so they have the same size. Let $s_1, s_2, s_3, \ldots$

be an enumeration of the strings in Σ^*, e.g., the enumeration can list the strings in string order. Define mapping $\chi : \mathcal{L} \to \mathcal{B}$ such that for a language $A \in \mathcal{L}$, the nth bit of $\chi(A)$ is 1 if and only if the nth string $s_n \in A$. We now show χ is a correspondence.

- To show that χ is one-to-one, suppose that $A_1, A_2 \in \mathcal{L}$ with $A_1 \neq A_2$. Then there is some string s_i such that s_i is in one of the languages but not the other. Then $\chi(A_1)$ and $\chi(A_2)$ differ in the ith bit, so χ is one-to-one.
- To show that χ is onto, consider any infinite binary sequence $b = b_1 b_2 b_3 \ldots \in \mathcal{B}$. Consider the language A that includes all strings s_i for which $b_i = 1$ and does not include any string b_j for which $b_j = 0$. Then $\chi(A) = b$, so χ is onto.

Since χ is one-to-one and onto, it is a correspondence. Thus, $\mathcal{L}$ and $\mathcal{B}$ have the same size, so $\mathcal{L}$ is uncountable because $\mathcal{B}$ is uncountable.

5. (This is HW 7, problem 2b.) For any two Turing-recognizable languages L_1 and L_2, let M_1 and M_2, respectively, be TMs that recognize them. We construct a TM M' that recognizes the union $L_1 \cup L_2$:

   M' = "On input string w:
   1. Run M_1 and M_2 alternately on w, one step at a time.
   If either accepts, *accept*. If both halt and reject, *reject*.

   To see why M' recognizes $L_1 \cup L_2$, first consider $w \in L_1 \cup L_2$. Then w is in L_1 or in L_2 (or both). If $w \in L_1$, then M_1 accepts w, so M' will eventually accept w. Similarly, if $w \in L_2$, then M_2 accepts w, so M' will eventually accept w. On the other hand, if $w \notin L_1 \cup L_2$, then $w \notin L_1$ and $w \notin L_2$. Thus, neither M_1 nor M_2 accepts w, so M' will also not accept w. Hence, M' recognizes $L_1 \cup L_2$. Note that if neither M_1 nor M_2 accepts w and one of them does so by looping, then M' will loop, but this is fine because we only needed M' to *recognize* and not *decide* $L_1 \cup L_2$.

6. Define the language as

$$A = \{ \langle R \rangle \mid R \text{ is a regular expression with alphabet } \Sigma = \{0, 1\} \text{ that generates at least one string that ends in } 10 \}.$$

   Consider the language C having regular expression $(0 \cup 1)^*10$, so C is the language of strings over Σ that end in 10. By Kleene's Theorem (Theorem 1.54), the language C is regular, so it has a DFA M. The proof of Theorem 4.4 defines a Turing machine T that decides the language $E_{\text{DFA}} = \{ \langle B \rangle \mid B \text{ is a DFA with } L(B) = \emptyset \}$.

Then the following Turing machine S decides A:

S = "On input $\langle R \rangle$, where R is a regular expression (with alphabet $\Sigma = \{0, 1\}$):

1. Convert R into an equivalent DFA D using the algorithm in the proof of Kleene's Theorem.
2. Construct a DFA K for the language $L(D) \cap L(M)$ using the algorithm for building a DFA for the intersection of two regular languages (HW 2, problem 5).
3. Run TM T for E_{DFA} on input $\langle K \rangle$.
4. If T accepts, *reject*. If T rejects, *accept*."