HW 3: Extra problems

Instructors: Har-Peled

CS/ECE 374A: Intro. Algorithms & Models of Computation, Fall 2024 Version: **1.11**

The following problems are not for submission or grading. No solutions for them will be provided but you can discuss them on Piazza (however, some of them already contain a solution).

1 (100 PTS.) NFAs (Fall 2020).

For each of the following languages over $\Sigma = \{3, 7, 4\}$, draw an NFA that accepts them. Your NFA should have a small number of states (at most say 14 states). Provide a brief explanation for your solution.

1.A. (20 PTS.) $\Sigma^* 3 \Sigma^* 7 \Sigma^* 4 \Sigma^*$

1.B. (20 PTS.) All strings in Σ^* that contain the substrings 374 and 473.

1.C. (20 PTS.) All strings in Σ^* that do not contain 374 as a substring.

1.D. (20 PTS.) All strings in Σ^* that contain the substring 374 and an odd number of 7s.

1.E. (20 PTS.) All strings in Σ^* such that every maximal substring of consecutive 7s is even in size.

2 (100 PTS.) DFAs to NFAs (Fall 2020)

Given a DFA $M = (\Sigma, Q, \delta, s, A)$ that accepts L, construct an NFA $N = (\Sigma, Q', \delta', s', A')$ that accepts the following languages. You can assume $\Sigma = \{0, 1\}$ in **2.A.** and **2.C.**. Provide a brief explanation for your solution.

2.A. (30 PTS.) $\mathrm{DelOnes}(L) := \left\{ 0^{\#_0(w)} \;\middle|\; w \in L \right\}$; i.e., removes all 1s from the strings of L.

2.B. (30 PTS.) $\mathit{ThereAndBack}(L) := \left\{ xy \;\middle|\; x \in L \text{ and } y^R \in L \right\}$

2.C. (40 PTS.) $\mathit{XOR}(L) := \{ z \mid z = \mathtt{XOR}(x, y) \text{ for some } x \in L, y \in L, \text{ such that } |x| = |y| = |z| \}$, where $\mathtt{XOR}(x, y)$ computes the element-wise $\mathtt{XOR}$ of x and y (so for each index i, $z_i = x_i \ \mathtt{XOR}\ y_i$).

2.D. (**Not for submission**) Consider, if you must, the language

$$\mathit{Middle}(L) := \{ y \in L \mid xyz \in L \text{ for some } x, z \text{ such that } |x| = |y| = |z| \}.$$

Prove that this language is regular.

3 (100 PTS.) Fooling Sets (Fall 2020)

Prove that the following languages are not regular by providing a fooling set. You need to provide an infinite set and also prove that it is a valid fooling set for the given language.

3.A. (20 PTS.) $L = \left\{ w w^R w \;\middle|\; w \in \{0, 1\}^* \right\}$.

3.B. (20 PTS.) $L = \left\{ 0^i 1 0^j \;\middle|\; i \text{ is divisible by } j \right\}$.

3.C. (20 PTS.) $L = \left\{ a^i b^j \;\middle|\; i, j \in \mathbb{N}, \text{ and } j = \log_2 i \right\}$.

3.D. (20 PTS.) $L = \left\{0^i 0^j \;\middle|\; i, j \in \mathbb{N}, \text{ and } j = \sqrt{i}\right\}$.

3.E. (20 PTS.) $L = \left\{wcd^{\#_a(w)} \mid w \in \{a, b\}^*\right\}$.

4 (100 PTS.) Draw me a goat.

For each of the following languages, draw (or describe formally) an NFA that accepts them. Your automata should have a small number of states. Provide a short explanation of your solution, if needed.

4.A. All strings over $\{a, b, c\}^*$ in which every nonempty maximal substring of consecutive as is of even length.

4.B. $\Sigma^* a \Sigma^* b \Sigma^* c \Sigma^*$.

4.C. $(a(a+b)^*a + b(b+c)^*b + c(c+a)^*c)^*$.

4.D. $\Big(\big((aa+aab)^*(bab+bb)^* + c\big)b\Big)^* + bb$.

4.E. All strings in 1^* of length that is divisible by at lease one of the following numbers $2, 3, 5, 7$. For full credit your automata should have less than (say) 20 states.

4.F. All strings in a^* of length that is NOT divisible by any of the following numbers $2, 3, 5, 7$.

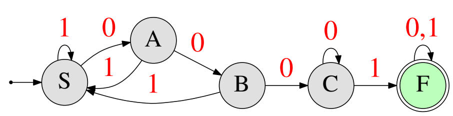
5 (100 PTS.) Blip blop.

For two binary strings $x, y \in \{0, 1\}^*$, of the same length, their ***Hamming distance*** $d_H(x, y)$ is the number of bits in which they differ. For example $d_H(1111, 1111) = 0$, $d_H(0001, 1111) = 3$, and $d_H(1111001, 1111011) = 1$. As a negative example, observe that $d_H(11, 1011)$ is not defined.

Let $L \subseteq \{0, 1\}^*$ be a regular language.

5.A. Consider the language $L_{\leq 1} = \{x \in \{0, 1\}^* \mid \exists y \in L \text{ s.t. } d_H(x, y) \leq 1\}$. Describe in words what the language $L_{\leq 1}$ is.

5.B. Consider the following DFA M.



What is its language $L = L(M)$?

5.C. By modifying the given DFA give above, describe an NFA that that accepts the language $L_{\leq 1}$. Explain your construction.

5.D. More generally, demonstrate that if a language $L \subseteq \{0, 1\}^*$ is regular, then $L_{\leq 1}$ is a regular language (for simplicity, you can assume $\varepsilon \notin L$). Specifically, consider a DFA for L, and describe in detail how to modify it to an NFA for $L_{\leq 1}$. (The description of the NFA does not have to be formal here.) Explain why the constructed NFA accept the desired language.

5.E. Prove, that for any constant k, the language $L_{\leq k}$ is regular. Your proof has to be formal and provide all necessary details. (I.e., you need to provide an explicit formal description of the resulting NFA for the new language, and prove that the NFA accepts the language $L_{\leq k}$).

6 Suppose $N_1 = (Q_1, \Sigma, \delta_1, s_1, A_1)$ and $N_2 = (Q_2, \Sigma, \delta_2, s_2, A_2)$ are NFAs. Formally describe a DFA that accepts the language $L(N_1) \setminus L(N_2)$. This combines subset construction and product construction to give you practice with formalism.

7 Suppose $M = (Q, \Sigma, \delta, s, A)$ is a DFA. For states $p, q \in Q$ (p can be same as q) argue that $L_{p,q} = \{w \mid \delta^*(p, w) = q\}$ is regular. Recall that $\text{PREFIX}(L) = \{w \mid wx \in L, x \in \Sigma^*\}$ is the set of all prefixes of strings in L. Express $\text{PREFIX}(L(M))$ as $\cup_{q \in Z} L_{s,q}$ for a suitable set of states $Z \subseteq Q$. Why does this prove that $\text{PREFIX}(L(M))$ is regular whenever L is regular?

8 For a language L let $\text{MID}(L) = \{w \mid xwy \in L, x, y \in \Sigma^*\}$. Prove that $\text{MID}(L)$ is regular if L is regular.

9

1. Draw an NFA that accepts the language $\{w \mid \text{there is exactly one block of 0s of even length}\}$. (A "block of 0s" is a maximal substring of 0s.)
2. (a) Draw an NFA for the regular expression $(010)^* + (01)^* + 0^*$.

   (b) Now using the powerset construction (also called the subset construction), design a DFA for the same language. Label the states of your DFA with names that are sets of states of your NFA.

10 This problem is to illustrate proofs of (the many) closure properties of regular languages.

1. For a language L let $\text{FUNKY}(L) = \{w \mid w \in L \text{ but no proper prefix of } w \text{ is in } L\}$. Prove that if L is regular then $\text{FUNKY}(L)$ is also regular using the following technique. Let $M = (Q, \Sigma, \delta, s, A)$ be a DFA accepting L. Describe a NFA N in terms of M that accepts $\text{FUNKY}(L)$. Explain the construction of your NFA.
2. In Lab 3 we saw that $\mathit{insert1}(L)$ is regular whenever L is regular. Here we consider a different proof technique. Let r be a regular expression. We would like to show that there is another regular expression r' such that $L(r') = \mathit{insert1}(L(r))$.

   (a) For each of the base cases of regular expressions $\emptyset, \epsilon$ and $\{a\}, a \in \Sigma$ describe a regular expression for $\mathit{insert1}(L(r))$.

   (b) Suppose r_1 and r_2 are regular expressions, and r_1' and r_2' are regular expressions for the languages $\mathit{insert1}(L(r_1))$ and $\mathit{insert1}(L(r_2))$ respectively. Describe a regular expression for the language $\mathit{insert1}(L(r_1 + r_2))$ using r_1, r_2, r_1', r_2'.

   (c) Same as the previous part but now consider $L(r_1 r_2)$.

   (d) Same as the previous part but now consider $L((r_1)^*)$.

11 Recall that for any language L, $\overline{L} = \Sigma^* - L$ is the complement of L. In particular, for any NFA N, $\overline{L(N)}$ is the complement of $L(N)$.

Let $N = (Q, \Sigma, \delta, s, A)$ be an NFA, and define the NFA $N_{\text{comp}} = (Q, \Sigma, \delta, s, Q \setminus A)$. In other words we simply complemented the accepting states of N to obtain N_{comp}. Note that if M is DFA then M_{comp} accepts $\Sigma^* - L(M)$. However things are trickier with NFAs.

1. Describe a concrete example of a machine N to show that $L(N_{\text{comp}}) \neq \overline{L(N)}$. You need to explain for your machine N what $\overline{L(N)}$ and $L(N_{\text{comp}})$ are.

2. Define an NFA that accepts $\overline{L(N)} - L(N_{\text{comp}})$, and explain how it works.
3. Define an NFA that accepts $L(N_{\text{comp}}) - \overline{L(N)}$, and explain how it works.

Hint: For all three parts it is useful to classify strings in Σ^* based on whether N takes them to accepting and non-accepting states from s.

12 Let L be an arbitrary regular language. Prove that the language $\mathit{half}(L) := \{w \mid ww \in L\}$ is also regular.

> **Solution:** Let $M = (\Sigma, Q, s, A, \delta)$ be an arbitrary DFA that accepts L. We define a new NFA $M' = (\Sigma, Q', s', A', \delta')$ with ε-transitions that accepts $\mathit{half}(L)$, as follows:
>
> $$\begin{gathered} Q' = (Q \times Q \times Q) \cup \{s'\} \\ s' \text{ is an explicit state in } Q' \\ A' = \{(h, h, q) \mid h \in Q \text{ and } q \in A\} \end{gathered}$$
>
> $$\begin{aligned} \delta'(s', \varepsilon) &= \{(s, h, h) \mid h \in Q\} \\ \delta'((p, h, q), a) &= \big\{\big(\delta(p, a), h, \delta(q, a)\big)\big\} \end{aligned}$$
>
> M' reads its input string w and simulates M reading the input string ww. Specifically, M' simultaneously simulates two copies of M, one reading the left half of ww starting at the usual start state s, and the other reading the right half of ww starting at some intermediate state h.
>
> - The new start state s' non-deterministically guesses the "halfway" state $h = \delta^*(s, w)$ without reading any input; this is the only non-determinism in M'.
> - State (p, h, q) means the following:
>   - The left copy of M (which started at state s) is now in state p.
>   - The initial guess for the halfway state is h.
>   - The right copy of M (which started at state h) is now in state q.
> - M' accepts if and only if the left copy of M ends at state h (so the initial non-deterministic guess $h = \delta^*(s, w)$ was correct) and the right copy of M ends in an accepting state.

Rubric: 5 points =

- \+ 1 for a formal, complete, and unambiguous description of a DFA or NFA
  - – No points for the rest of the problem if this is missing.
- \+ 3 for a correct NFA
  - – -1 for a single mistake in the description (for example a typo)
- \+ 1 for a *brief* English justification. We explicitly do *not* want a formal proof of correctness, but we do want one or two sentences explaining how the NFA works.

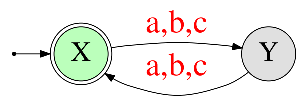
13 (100 PTS.) Codes.

Let Σ be finite alphabet. A ***code*** is a mapping $f : \Sigma \to \{0, 1\}^+$. For example, if $\Sigma = \{a, b, c\}$, a code f might be $f(a) = 00010$, $f(b) = 000$, and $f(c) = 1$. (To simplify things, we assume that $f(a) \neq \varepsilon$, for any character $a \in \Sigma$.)

For a string $w_1 w_2 \cdots w_m \in \Sigma^*$, we define $f(w) = f(w_1)f(w_2)\cdots f(w_m)$. In the above code, we have

$$f(abcba) = 00010 \bullet 000 \bullet 1 \bullet 000 \bullet 00010. = 00010000100000010.$$

13.A. (10 PTS.) Let L be the language of the following DFA M. What is L?



13.B. (20 PTS.) Working directly on the DFA M from (A) construct an NFA for the language $f(L)$. Here $f(L) = \{f(w) \mid w \in L\}$ is the *code language*. Where f is code from the above example.

13.C. (30 PTS.) Let $L \subseteq \Sigma^*$ be am arbitrary regular language. Prove that the encoded language $f(L) = \{f(w) \mid w \in L\}$ is regular.

Specifically, given a DFA $M = (Q, \Sigma, \delta, s, A)$ for L, describe how to build an NFA M' for $f(L)$. Give an upper bound on the number of states of M'.

(I.e., You need to prove the correctness of your construction – that the language of the constructed NFA is indeed the desired language $f(L)$.)

(Rubric: Half the credit is for a correct construction, and the other half is for a correct proof of correctness.)

13.D. (40 PTS.) Let $L \subseteq \{0, 1\}^*$ be a regular language. Consider the decoded language $L_f = \{w \in \Sigma^* \mid f(w) \in L\}$.

Prove that L_f is a regular language. As above, given a DFA M for L, describe how to construct an NFA for L_f.

(Rubric: Half the credit is for a correct construction, and the other half is for a correct proof of correctness.)

14 (100 PTS.) Draw me a giraffe.

For each of the following languages in **14.A.–14.C.**, draw an NFA that accepts them. Your automata should have a small number of states. Provide a short explanation of your solution, if needed.

14.A. (25 PTS.) All strings in $\{0, 1, 2\}^*$ such that at least one of the symbols 0, 1, or 2 occurs at most 4 times. (Example: 1200201220210 is in the language, since 1 occurs 3 times.)

14.B. (25 PTS.) $\big((01)^*(10)^* + 00\big)^* \cdot (1 + 00 + \varepsilon) \cdot (11)^*$.

14.C. (25 PTS.) All strings in $\{0, 1\}^*$ such that the last symbol is the same as the third last symbol. (Example: 1100101 is in the language, since the last and the third last symbol are 1.)

14.D. (25 PTS.) Use the power-set construction (also called subset construction) to convert your NFA from **14.C.** to a DFA. You may omit unreachable states.

15 (100 PTS.) Fun with parity.

Given $L \subseteq \{0, 1\}^*$, define $\mathit{even}_0(L)$ to be the set of all strings in $\{0, 1\}^*$ that can be obtained by taking a string in L and inserting an even number of 0's (anywhere in the string). Similarly, define $\mathit{odd}_0(L)$ to be the set of all strings x in $\{0, 1\}^*$ that can be obtained by taking a string in L and inserting an odd number of 0's.

(Example: if $01101 \in L$, then $01010000100 \in \mathit{even}_0(L)$.)

(Another example: if L is 1^*, then $\mathit{even}_0(L)$ can be described by the regular expression $(1^*01^*0)^*1^*$.)

The purpose of this question is to show that if $L \subseteq \{0, 1\}^*$ is regular, then $\mathit{even}_0(L)$ and $\mathit{odd}_0(L)$ are regular.

15.A. (30 PTS.) For each of the base cases of regular expressions $\emptyset$, ε, 0, and 1, give regular expressions for $\mathit{even}_0(L(r))$ and $\mathit{odd}_0(L(r))$.

15.B. (60 PTS.) Given regular expressions for $e_j = \mathit{even}_0(L(r_j))$ and $o_j = \mathit{odd}_0(L(r_j))$, for $j \in \{1, 2\}$, give regular expressions for

(i) $\mathit{even}_0(L(r_1 + r_2))$
(ii) $\mathit{odd}_0(L(r_1 + r_2))$
(iii) $\mathit{even}_0(L(r_1 r_2))$
(iv) $\mathit{odd}_0(L(r_1 r_2))$
(v) $\mathit{even}_0(L(r_1^*))$
(vi) $\mathit{odd}_0(L(r_1^*))$

Give brief justification of correctness for each of the above.

15.C. (10 PTS.) Using the above, describe (shortly) a recursive algorithm that given a regular expression r, outputs a regular expression for $\mathit{even}_0(L(r))$ (similarly describe the algorithm for computing $\mathit{odd}_0(L(r))$).

16 (100 PTS.) "+1".

Let binary(i) denote the binary representation of a positive integer i. (Note that the string binary(i) must start with a 1.)

Given a language $L \subseteq \{0, 1\}^*$, define $\mathrm{INC}(L) = \{\mathrm{binary}(i + 1) \mid \mathrm{binary}(i) \in L\}$. For the time being assume that L does not contain any string of 1^*.

(Example: for $L = \{100, 101011, 1101\}$, we have $\mathrm{INC}(L) = \{101, 101100, 1110\}$.)

16.A. (30 PTS.) Given a DFA $M = (Q, \Sigma, \delta, s, A)$ for L, describe **informally** (in a few sentences) how to construct an NFA M_w for $\mathrm{INC}(L)$.

16.B. (30 PTS.) Given a DFA $M = (Q, \Sigma, \delta, s, A)$ for L, describe **formally** how to construct an NFA M' for $\mathrm{INC}(L)$.

16.C. (30 PTS.) Prove formally the correctness of your construction from (**16.B.**). That is, prove that $\mathrm{INC}(L) = L(M')$.

16.D. (10 PTS.) Describe formally how to modify the construction of M' from above, to handle that general case (without the above assumption) that L might also contain strings of the form 1^*. You do not need to provide a proof of correctness of the new automata.

17 (100 PTS.) String flip [Fall 24].

Let $\Sigma = \{0, 1\}$, and let $L \subseteq \Sigma^*$ be a regular language. For a string $s = s_1 \dots s_n \in \Sigma^*$, let $s^R = s_n s_{n-1} \dots s_1$ be the *reverse* of s. Consider the following language

$$L_8 = \left\{ xy^R z \;\middle|\; x, y, z \in \Sigma^*, |y| \leq 8, \text{ and } \quad xyz \in L \right\}.$$

Thus, if $0101000011110101 \in L$, then $0101\underline{11110000}0101 \in L_8$ as is $01010\underline{100}1110101 \in L_8$. Prove that L_8 is a regular language.

To this end, you are given a DFA M for L – provide an NFA N for L_8. Describe formally how you construct N from M, and argue why your construction is correct. A formal proof that your construction works is not required.

Hints: (A) Your NFA should use its ability to guess things, and remember constant amount of information (how?). (B) To build up to the solution consider special cases, and solve them first, such as: (i) $x = \varepsilon$, (ii) $z = \varepsilon$, and (iii) $|y| = 2$.

18 (100 PTS.) Highly irregular [Fall 24].

For each of the following languages prove that they are not regular using fooling sets. Here $\Sigma = \{0, 1\}$.

18.A. (30 PTS.) For a string $w = w_1 w_2 \dots w_k$, let $\mathrm{odd}(w) = w_1 w_3 w_5 \cdots$ be the string formed by the odd characters of w. Consider the language $L_A = \{w \in (0 + 1)^* \mid \mathrm{odd}(w) \text{ is a palindrome}\}$.

18.B. (30 PTS.) $L_B = \{w \in \Sigma^* \mid 10^n10^n1 \text{ is a substring of } w, \text{ where } n \text{ is an integer}\}$.

18.C. (40 PTS.) $L_C = \left\{ 0^i 1^j \;\middle|\; i + j = k^2, \text{ where } k \text{ is an integer} \right\}$.