

**CS 341-H01, Fall 2026, Face-to-Face Section**  
**Solutions for Exam 1**

1. Multiple choice.

1.1. Answer: (b).

- Because  $A$  has a regular expression,  $A$  is regular by Kleene's Theorem (Theorem 1.54), so  $A^*$  is regular since the class of regular languages is closed under Kleene star (Theorem 1.49). Also,  $B$  is regular because it is finite (slide 1-95), so  $\overline{B}$  is regular because the class of regular languages is closed under complementation (HW 2, problem 3). The class of regular languages also closed under intersection (HW 2, problem 5), so  $A^* \cap \overline{B}$  is regular, making (b) correct.
- Suppose that  $A$  has regular expression  $(aa)^*a$ , so  $A$  is the set of strings of  $a$ 's of odd length. Because  $A$  has a regular expression, it is regular by Kleene's Theorem (Theorem 1.54). Note that  $a \in A$  and  $aaa \in A$ , but their concatenation  $aaaa \notin A$ , so  $A$  is not closed under concatenation, showing that (a) is incorrect, so (d) is also incorrect. (While the *class* of regular languages is closed under concatenation by Theorem 1.47, this example shows that a *particular* regular language may not be closed under concatenation.) Also, the same language  $A$  is infinite, showing that (c) is incorrect.

1.2. Answer: (c).

- The languages  $L_1 = \{a^n b^n c^n \mid n \geq 0\}$  and  $L_2 = \{b^n a^n c^n \mid n \geq 0\}$  are non-context-free languages (slide 2-96), with  $L_1 \cap L_2 = \{\varepsilon\}$ , which is regular because it is finite (slide 1-95), so (d) is incorrect. The intersection is also context-free by Corollary 2.32, making (a) incorrect.
- If  $L_1 = L_2 = \{a^n b^n c^n \mid n \geq 0\}$ , then  $L_1 \cap L_2 = L_1$ , which is non-regular and non-context-free, so (b) is incorrect.
- The previous two examples show that (c) is correct.

1.3. Answer: (d).

- By slide 2-99,  $A$  is non-context-free, so Theorem 2.20 shows that  $A$  cannot have a PDA, making (b) incorrect.
- Because  $A$  is non-context-free,  $A$  is also not regular (Corollary 2.32), so (a) is incorrect. We can see that the regular expression  $(0 \cup 1)^*(0 \cup 1)^*$  in (a) is wrong because it generates the string  $01 \notin A$ .
- The strings  $0101$  and  $11$  are in  $A$ , but their concatenation  $010111 \notin A$ , so  $A$  is not closed under concatenation, making (c) incorrect.
- For any strings  $w_1$  and  $w_2$ , we have that  $(w_1 w_2)^R = w_2^R w_1^R$ . Thus, for any string  $ww \in A$ , its reverse is  $(ww)^R = w^R w^R \in A$ , so  $A$  is closed under reversal, making (d) correct.

1.4. Answer: (b).

- Since the class of regular languages is closed under intersection (HW 2, problem 5), option (a) is incorrect.
- The class of languages recognized by NFAs is the same as the class of regular languages by Corollary 1.40. Thus, this class of languages is closed under complementation (HW 2, problem 3), so option (b) is correct.
- The language  $\{0^n 1^n \mid n \geq 0\}$  is nonregular (slide 1-105), so it is not recognized by any NFA (Corollary 1.40), making option (c) incorrect. Also, this language is context-free (slide 2-5), making (d) incorrect.

1.5. Answer: (b).

- For each  $i \geq 0$ , the string  $1^i \in A$ , so  $A$  is infinite, making (a) incorrect.
- The language  $A$  has regular expression  $1^*0^*$ , so Kleene's Theorem (Theorem 1.54) implies that  $A$  is regular, and  $A$  is also infinite. Thus, (b) is correct, and (c) is incorrect. By Corollary 2.32, if a language is non-context-free, then it must also be non-regular, so (d) is incorrect.

1.6. Answer: (e).

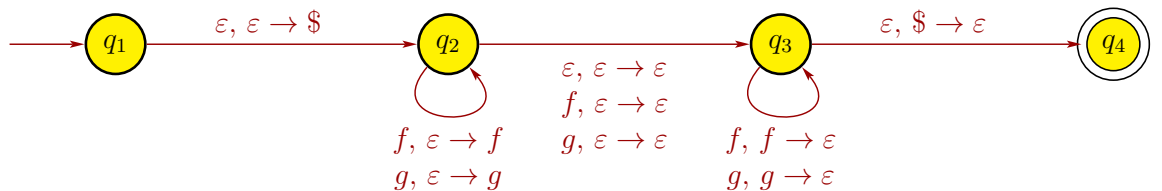
- The class of regular languages is closed under union, concatenation, Kleene star, intersection, and complementation. Because  $A$ ,  $B$ , and  $C$  are regular, each of  $\overline{B}(A^* \cup C)$ ,  $(B \cap C^*)\overline{A}$ ,  $A^*(\overline{B} \cup C)$ , and  $(B \cap \overline{C})A^*$  is regular, so they are also context-free by Corollary 2.32. Thus, (e) is correct.

1.7. Answer: (e).

- Suppose that  $B = \Sigma^*$  for  $\Sigma = \{a, b, c\}$ . Then  $B$  is regular because it has a regular expression (Kleene's Theorem), so it is also context-free (Corollary 2.32).
- Now let  $A = \{a^n b^n c^n \mid n \geq 0\}$ , which is non-context-free (slide 2-96), with  $A \subseteq B$ , so (c) is incorrect. Moreover,  $A$  being non-context-free implies  $A$  is also nonregular by Corollary 2.32, making (a) also incorrect.
- For the same  $B$ , let  $A = B$ , so  $A$  is context-free with  $A \subseteq B$ , so (d) is incorrect. Also because  $A$  is regular, we have that (b) is incorrect.

1.8. Answer: (b).

- For  $\Sigma = \{f, g\}$ , the PDA



recognizes the language  $L_1 = \{w \in \Sigma^* \mid w = w^R\}$ , which is a slight variation of HW 6, problem 1(c).

- The PDA accepts string  $\varepsilon$ , so  $\varepsilon \in A$ , but  $\varepsilon \notin L_2$ , so  $A \neq L_2$ .
- The language  $L_3$  is not context-free, which can be proven by slightly modify-

ing the proof on slide 2-99, so  $L_3$  cannot have a PDA (Theorem 2.20). Thus,  $A \neq L_3$ .

1.9. Answer: (e).

- The regular expression  $b^*a^*b^*$  generates the string  $bbabbb \notin A$ , so (a) is incorrect.
- The regular expression  $(ba)^*b^*$  generates the string  $bababb \notin A$ , so (b) is incorrect.
- From the CFG with rules  $S \rightarrow bSa \mid bS \mid \varepsilon$ , we get the derivation  $S \Rightarrow bSa \Rightarrow bbSaa \Rightarrow bbbSaa \Rightarrow bbbaa \notin A$ , so (c) is incorrect.
- The language  $A$  is context-free, so (d) is incorrect. A CFG for  $A$  is  $G = (V, \Sigma, R, S)$ , with  $V = \{S, X, Y\}$ ,  $\Sigma = \{a, b\}$ , starting variable  $S$ , and the set  $R$  of rules are

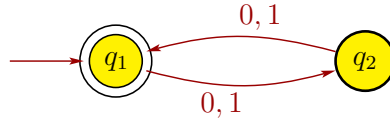
$$\begin{aligned} S &\rightarrow XY \\ X &\rightarrow bXa \mid \varepsilon \\ Y &\rightarrow bY \mid \varepsilon \end{aligned}$$

1.10. Answer: (d).

- The class of context-free languages is closed under union (Homework 5, problem 3a), so  $B \cup C$  is context-free. Also, the class of context-free languages is closed under Kleene-star (Homework 5, problem 3c), so  $A^*$  is context-free. Finally, the class of context-free languages is closed under concatenation (Homework 5, problem 3b), so  $A^*(B \cup C)$  is context-free, making (d) correct.
- We know that the class of context-free languages is *not* closed under complementation (Homework 6, problem 2b), so there exists some context-free language  $D$  whose complement  $\overline{D}$  is not context-free. Also, let  $B = C = \{\varepsilon\}$ , which is finite, so  $B$  and  $C$  are regular (slide 1-95), making them also context-free (Corollary 2.32). Also,  $B^* = \{\varepsilon\}$ , so  $B^* \cup C = \{\varepsilon\}$ . Let  $A = D$ , so  $\overline{A}(B^* \cup C) = \overline{A} = \overline{D}$  is non-context free, making (a) incorrect.
- Let  $A$  and  $C$  be any context-free languages, and let  $B = \emptyset$ , which is finite, so  $B$  is regular (slide 1-95), and also context-free (Corollary 2.32). But then  $B \cap C^* = \emptyset$ , so  $\overline{A}(B \cap C^*) = \emptyset$ , which is regular so also context-free, making (b) incorrect.
- For  $\Sigma = \{a, b\}$ , let  $A$  be the language of all strings over  $\Sigma$  that don't begin with  $a$ . Now  $A$  has regular expression  $\varepsilon \cup b(a \cup b)^*$ , so Kleene's Theorem (Theorem 1.54) implies that  $A$  is a regular language, making  $A$  also context-free (Corollary 2.32). Also,  $\overline{A}$  is the set of all strings over  $\Sigma$  that begin with  $a$ ; e.g.,  $a \in \overline{A}$ . Also, let  $B = \{b\}$  and  $C = \{b\}$ , each of which are finite so also regular (slide 1-95) and context-free (Corollary 2.32). Also,  $B \cup C = \{b\} = C \cup B$ . Then, we have that  $ab \in \overline{A}(B \cup C)$ , but  $ab \notin (C \cup B)\overline{A}$ , so  $\overline{A}(B \cup C) \neq (C \cup B)\overline{A}$ , making (c) incorrect.

1.11. Answer: (f).

- For  $\Sigma = \{0, 1\}$ , let  $A$  be the language of all strings over  $\Sigma$  that have an odd number of 1's or even length. Note that  $A = A_1 \cup A_2$ , where  $A_1$  is the language of strings in  $\Sigma^*$  with an odd number of 1's, and  $A_2$  is the language of strings in  $\Sigma^*$  of even length. Thus, if we have a regular express  $r_1$  for  $A_1$  and a regular expression  $r_2$  for  $A_2$ , then a regular expression for  $A = A_1 \cup A_2$  is  $R = r_1 \cup r_2$ .
- The regular expression  $R_2 = ((0 \cup 1)(0 \cup 1))^* \cup (0^*10^* \cup 0^*1)(0^*10^*1)^*$  cannot generate the string  $w = 11100 \in A$ , so  $A \neq L(R_2)$ .
- We now show that  $R_1 = (00 \cup 01 \cup 10 \cup 11)^* \cup 0^*1(0 \cup 10^*1)^*$  satisfies  $A = L(R_1)$ . We can obtain regular expressions  $r_1$  and  $r_2$  for  $A_1$  and  $A_2$ , respectively, by converting DFAs for the languages into regular expressions. A DFA  $M_2$  for  $A_2$  is

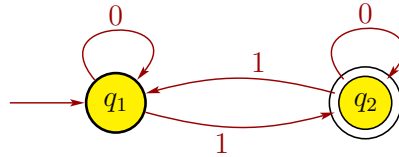


While we can use the algorithm in part of the proof of Kleene's theorem (Lemma 1.60) to convert the DFA  $M_2$  into a regular expression  $r_2$ , the DFA is simple enough to be able to analyze it directly to obtain  $r_2$ . Specifically, note that every string accepted by  $M_2$  has to be processed as follows:

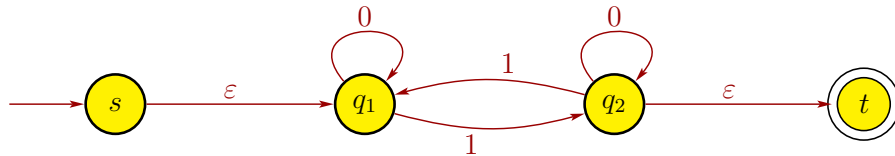
- start in  $q_1$ ,
- loop from  $q_1$  back to  $q_1$  zero or more times.

Looping from  $q_1$  back to  $q_1$  corresponds to  $(0 \cup 1)(0 \cup 1) = (00 \cup 01 \cup 10 \cup 11)$ , so looping zero or more times yields  $((0 \cup 1)(0 \cup 1))^*$  or  $(00 \cup 01 \cup 10 \cup 11)^*$ . Thus, we get  $r_2 = ((0 \cup 1)(0 \cup 1))^*$  and  $r'_2 = (00 \cup 01 \cup 10 \cup 11)^*$  as regular expressions for  $A_2$ .

A DFA  $M_1$  recognizing  $A_1$  is



To obtain a regular expression corresponding  $M_1$ , we apply the algorithm in Kleene's theorem (Lemma 1.60) to convert  $M_1$  into a regular expression  $r_1$  as follows. First, convert  $M_1$  into an equivalent GNFA:



We will first eliminate state  $q_1$ , so we define  $I_1 = \{s, q_2\}$  as the set of states

(except for  $q_1$ ) with edges directly into  $q_1$ , and  $O_1 = \{q_2\}$  as the set of states (except for  $q_1$ ) with edges directly from  $q_1$ . Eliminating  $q_1$  by taking into account all paths going directly from a state in  $I_1$  to  $q_1$ , looping in  $q_1$  zero or more times, and then directly going to a state in  $O_1$  results in

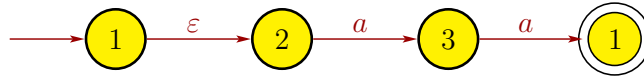


Next removing state  $q_2$  results in the regular expression  $r_1 = 0^*1(0 \cup 10^*1)^*$  for  $A_2$ . Thus, a regular expression for  $A = A_1 \cup A_2 = A_2 \cup A_1$  is  $R = r'_2 \cup r_1 = (00 \cup 01 \cup 10 \cup 11)^* \cup 0^*1(0 \cup 10^*1)^*$ , which is  $R_1$ .

- For  $R_3 = (0^*10^*10^*)^*0^*10^* \cup ((0 \cup 1)(0 \cup 1))^*$ , we can show that  $A = L(R_3)$  as follows. First, write  $A = A_1 \cup A_2$ , with  $A_1$  and  $A_2$  as defined above. We can again use regular expression  $r_2 = ((0 \cup 1)(0 \cup 1))^*$  for  $A_2$ . For  $A_1$ , note that  $r_3 = (0^*10^*10^*)^*$  defines the language of strings in  $\Sigma^*$  with an even number of 1s. Also,  $r_4 = 0^*10^*$  defines the language of strings in  $\Sigma^*$  with exactly a single 1. Concatenating these two gives a regular expression  $r'_1 = r_3r_4 = (0^*10^*10^*)^*0^*10^*$  for the language of strings in  $\Sigma^*$  with an odd number of 1s. Thus, a regular expression for  $A$  is  $r'_1 \cup r_2 = (0^*10^*10^*)^*0^*10^* \cup ((0 \cup 1)(0 \cup 1))^*$ , which is  $R_3$ .

1.12. Answer: (b).

- $N$  accepts  $aa$  as follows:



so (a) is incorrect.

- $N$  accepts all strings generated by the regular expression  $(baa)^*$ , as well as other strings, so  $|L(N)| = \infty$ , making (c) incorrect.

1.13. Answer: (b).

- HW 4, problem 5c, shows that (a) is incorrect, and that (b) is correct.
- Slightly modifying the proof on slide 1-105 shows that the language  $L_1 = \{a^n b^n \mid n \geq 1\}$  is non-regular. Adding  $\varepsilon$  to  $L_1$  leads to the language  $L_2 = \{a^n b^n \mid n \geq 0\}$ , which is context-free (with CFG having rules  $S \rightarrow aSb \mid \varepsilon$ ), so (c) is incorrect.
- Slightly modifying the proof on slide 2-96 shows that the language  $L_1 = \{a^n b^n c^n \mid n \geq 1\}$  is non-context-free, so it is also non-regular by Corollary 2.32. Adding  $\varepsilon$  to  $L_1$  leads to the language  $L_2 = \{a^n b^n c^n \mid n \geq 0\}$ , which is non-context-free by the proof on slide 2-96, so (d) is incorrect.

1.14. Answer: (a).

- For the alphabet  $\Sigma = \{a, b\}$ , first consider the language  $\Sigma^*$ , which satisfies  $A \subseteq \Sigma^*$ . For each  $k \geq 0$ , there are  $|\Sigma|^k$  strings of length  $k$  in  $\Sigma^*$ . Since  $A \subseteq \Sigma^*$ , there are at most  $|\Sigma|^k$  strings of length  $k$  in  $A$ . Each string  $w \in A$  has length  $|w| \leq 100$ , so  $|A| \leq \sum_{k=0}^{100} |\Sigma|^k < \infty$ , implying that  $A$  is regular (slide 1-95). Thus,  $A$  has a regular expression by Kleene's Theorem (Theorem 1.54), making (a) correct.
- Because  $A$  is regular, it must have a regular expression, so (b) is incorrect. Also, (d) is incorrect by Corollary 1.40. In addition, every regular language is also context-free (Corollary 2.32), so  $A$  must have a PDA (Theorem 2.20), making (c) incorrect.

1.15. Answer: (b).

- $\varepsilon \notin \emptyset$ , so (a) is incorrect.
- $\emptyset$  is the empty set, and  $\varepsilon$  is the empty string, so they aren't equal, making (c) incorrect.
- $\emptyset^* = \{\varepsilon\}$  making (b) correct, and (d) incorrect.

2. Short answers.

- (a) [**10 points**] Suppose that language  $A_1$  has CFG  $G_1 = (V_1, \Sigma, R_1, S_1)$  and language  $A_2$  has CFG  $G_2 = (V_2, \Sigma, R_2, S_2)$ . Give a CFG  $G_3$  for  $A_3 = A_2 \circ A_1 \cup A_1^*$  in terms of  $G_1$  and  $G_2$ . You do not have to prove the correctness of your CFG  $G_3$ , but do not give just an example.

**Answer:**  $G_3 = (V_3, \Sigma, R_3, S_3)$ , where

- $V_3 = V_1 \cup V_2 \cup \{S_3, S_4\}$ , where we assume that  $V_1 \cap V_2 = \emptyset$ , and  $S_3 \notin V_1 \cup V_2$ , and  $S_4 \notin V_1 \cup V_2$ .
- $\Sigma$  is the same as in  $G_1$  and  $G_2$ ,
- rules  $R_3 = R_1 \cup R_2 \cup \{S_3 \rightarrow S_2 S_1 \mid S_4, S_4 \rightarrow S_4 S_1 \mid \varepsilon\}$ ,
- $S_3$  is the start variable.

An **incorrect** answer has  $G'_3 = (V'_3, \Sigma, R'_3, S'_3)$ , where

- $V'_3 = V_1 \cup V_2 \cup \{S_3\}$ , where we assume that  $V_1 \cap V_2 = \emptyset$  and  $S_3 \notin V_1 \cup V_2$ .
- $\Sigma$  is the same as in  $G_1$  and  $G_2$ ,
- rules  $R'_3 = R_1 \cup R_2 \cup \{S_3 \rightarrow S_2 S_1 \mid S_3 S_1 \mid \varepsilon\}$ ,
- $S_3$  is the start variable.
- The problem with CFG  $G'_3$  is that a derivation starting with  $S_3 \Rightarrow S_3 S_1 \Rightarrow S_2 S_1 S_1 \xRightarrow{*} \dots$  can derive strings not in  $A_3$ .

- (b) [**15 points**] Define a language  $L$  as follows. First define  $\Omega = \{a, b, c, \dots, z\}$  as the set of lower-case Roman letters. Also, define  $\Lambda = \{.\}$  and  $\Phi = \{@\}$ , and let  $\Sigma = \Omega \cup \Lambda \cup \Phi$ ; i.e.,  $\Sigma$  is the set consisting of all the lower-case Roman letters, the dot (or period), and the @ symbol. (The symbols  $\Omega$ ,  $\Lambda$ , and  $\Phi$  are the capital Greek letters Omega, Lambda, and Phi, respectively.) Define the following sets of strings:

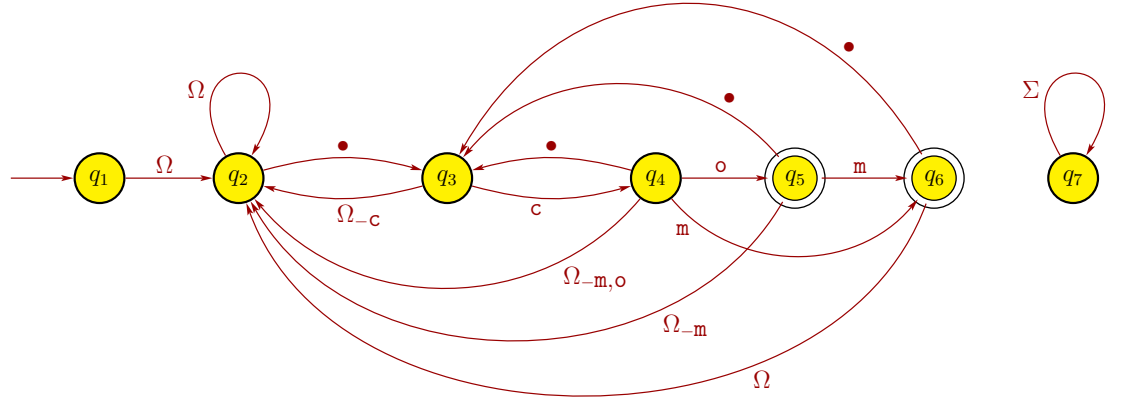
- $S_1 = \Omega\Omega^*$
- $S_2 = \Lambda\Omega\Omega^*$
- $S_3 = \{.co\}$
- $S_4 = \{.com\} \cup \{.cm\}$

Give a DFA for

$$L = S_1 S_2^* (S_3 \cup S_4)$$

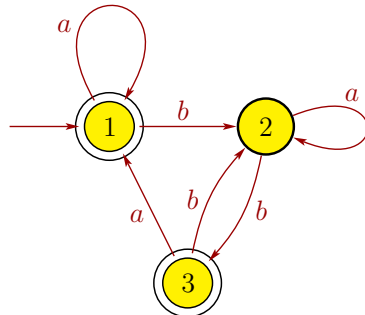
Give only the DFA drawing (state diagram); do not specify the 5-tuple. To simplify your DFA drawing, you should omit any edges going from any state  $q$  to a “trap state” (i.e., a non-accepting state from which the DFA can never leave). All other edges must be included in your drawing. Clearly identify which state is the trap state in the drawing of your DFA, and your drawing should include a note stating that all edges not specified go to a trap state. You will lose points if your DFA has more states than necessary or is overly complicated.

**Answer:** Below is a DFA for  $L$ .

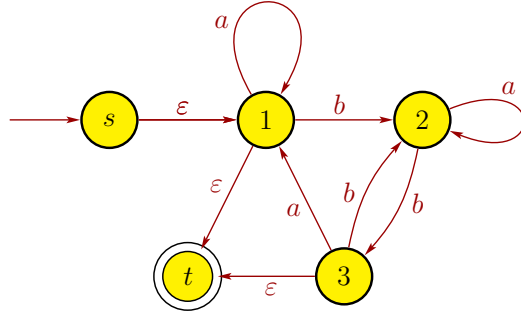


All edges not specified go to state  $q_7$ , which is a trap state. For example, there is an edge from  $q_1$  to  $q_7$  labeled with a dot. Also, we use the notation that  $\Omega_{-\ell} = \Omega - \{\ell\}$  for any symbol  $\ell \in \Omega$ ; i.e.,  $\Omega_{-\ell}$  is the set of all lower-case Roman letters except for  $\ell$ . For example,  $\Omega_{-c} = \{a, b, d, e, \dots, z\}$ .

3. Let  $M$  be the following DFA with  $\Sigma = \{a, b\}$ . Give a regular expression for  $L(M)$ .



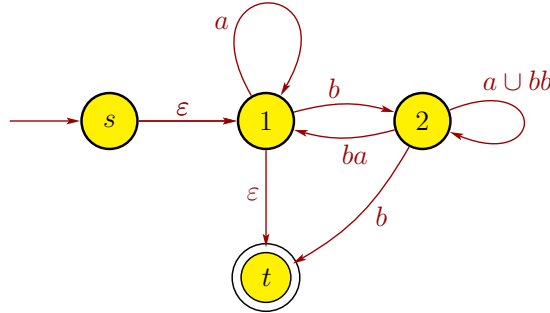
**Answer:** First convert DFA  $M$  into an equivalent GNFA  $G$ .



Next, we eliminate the states of  $G$  (except for  $s$  and  $t$ ) one at a time. The order in which the states are eliminated does not matter. However, eliminating states in a different order from what is done below may result in a different (but also correct) regular expression. We arbitrarily choose to first eliminate state 3. To do this, we define the set  $I_3 = \{2\}$  of states with edges directly into state 3, and set  $O_3 = \{1, 2, t\}$  of states with edges directly from state 3. We then need to account for paths that start in a state in  $I_3$ , go immediately to state 3, and then go immediately to a state in  $O_3$ . Thus, we need to account for the paths

- $2 \rightarrow 3 \rightarrow 1$ , which will create an arc from 2 to 1 labelled with  $ba$ ;
- $2 \rightarrow 3 \rightarrow 2$ , which will create an arc from 2 to 2 labelled with  $bb$ ; and
- $2 \rightarrow 3 \rightarrow t$ , which will create an arc from 2 to  $t$  labelled with  $b\epsilon = b$ .

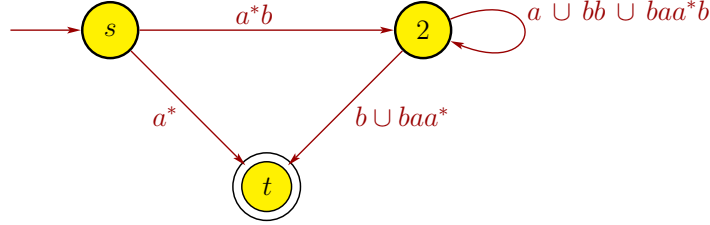
We combine the previous arc from 2 to 2 labelled  $a$  with the new one labelled  $bb$  to get the new label  $a \cup bb$ .



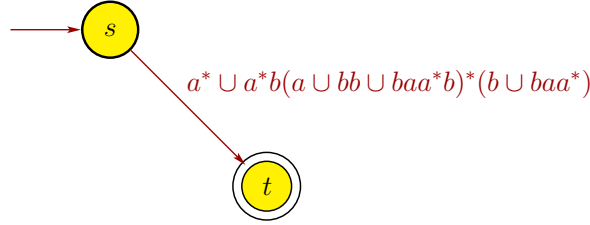
We next choose to eliminate state 1. To do this, we define  $I_1 = \{s, 2\}$  as the set of states with edges directly into state 1 (except for state 1), and  $O_1 = \{2, t\}$  as the set of states with edges directly from state 1 (except for state 1). Thus, need to account for the following paths from a state in  $I_1$ , going immediately to state 1, looping immediately back to 1, and then immediately going to a state in  $O_1$ :

- $s \rightarrow 1 \rightarrow 2$ , which will create an arc from  $s$  to 2 labelled with  $\epsilon a^* b = a^* b$ .
- $s \rightarrow 1 \rightarrow t$ , which will create an arc from  $s$  to  $t$  labelled with  $\epsilon a^* \epsilon = a^*$ .

- $2 \rightarrow 1 \rightarrow 2$ , which will create an arc from 2 to 2 labelled with  $baa^*b$ . We combine this with the existing 2 to 2 arc to get the new label  $a \cup bb \cup baa^*b$ .
- $2 \rightarrow 1 \rightarrow t$ , which will create an arc from 2 to  $t$  labelled with  $baa^*\varepsilon = baa^*$ . We combine this arc with the existing arc from 2 to  $t$  to get the new label  $b \cup baa^*$ .



Finally, we eliminate state 2 by adding an arc from  $s$  to  $t$  labelled  $a^*b(a \cup bb \cup baa^*b)^*(b \cup baa^*)$ . We then combine this with the existing  $s$  to  $t$  arc to get the new label  $a^* \cup a^*b(a \cup bb \cup baa^*b)^*(b \cup baa^*)$ .



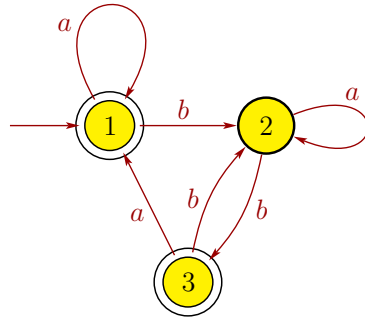
So a regular expression for the language  $L(M)$  recognized by the DFA  $M$  is

$$a^* \cup a^*b(a \cup bb \cup baa^*b)^*(b \cup baa^*)$$

Writing this as

$$\underbrace{a^*}_{\text{stay in 1}} \cup \underbrace{(a^*b)}_{\text{1 to 2}} \underbrace{(a \cup bb \cup baa^*b)^*}_{(2 \text{ to } 2)^*} \underbrace{(b \cup baa^*)}_{\text{end in 3 or 1}}$$

should make it clear how the regular expression accounts for every path that starts in 1 and ends in either 3 or 1, which are the accepting states of the given DFA.



4. For  $\Sigma = \{d, h, u\}$  (note the alphabet), let

$$L = \{h^i u^j d^k \mid i, j, k \geq 0 \text{ and } i = j + k\}.$$

**Answer:** This is similar to HW 5, problem 1(e) and HW 6, problem 4(e).

- (a) One CFG  $G_1 = (V, \Sigma, R, S)$  for language  $L$  has a set of variables  $V = \{S, T\}$ , where  $S$  is the start variable; set of terminals  $\Sigma = \{d, h, u\}$ ; and rules

$$\begin{aligned} S &\rightarrow hSd \mid T \\ T &\rightarrow hTu \mid \varepsilon \end{aligned}$$

There are infinitely many other correct CFGs for  $L$ . Another correct CFG  $G_2 = (V, \Sigma, R, S)$  for  $L$  has variables  $V = \{S, T, X\}$ ,  $\Sigma = \{d, h, u\}$ , and rules

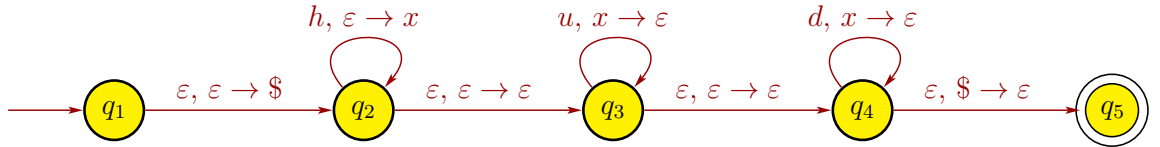
$$\begin{aligned} S &\rightarrow hSd \mid T \mid X \\ T &\rightarrow hTu \mid X \\ X &\rightarrow \varepsilon \end{aligned}$$

An **incorrect** CFG has rules

$$S \rightarrow hSd \mid hSu \mid \varepsilon$$

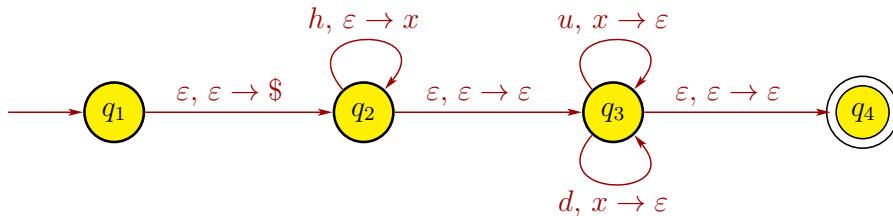
The derivation  $S \Rightarrow hSu \Rightarrow hhSdu \Rightarrow hhdu$  shows that the rules generate the string  $hhdu$ , which is not in  $L$  because the  $d$  and  $u$  are in the wrong order.

- (b) There are infinitely many correct PDAs for  $L$ . Here is one:



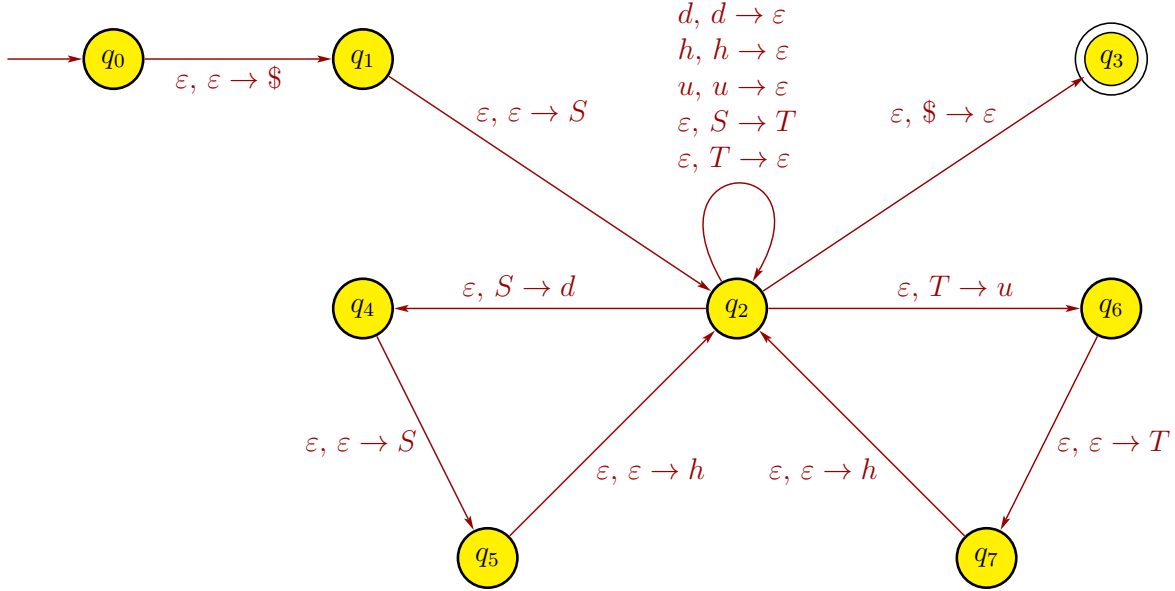
where an edge label “ $x, y \rightarrow z$ ” means read  $x$ , pop  $y$ , and push  $z$ . For every  $h$  read in the first part of the string, the PDA pushes an  $x$  onto the stack. Then for each  $u$  read, it pops an  $x$  off the stack. Then for each  $d$  read, it pops an  $x$  off the stack.

An **incorrect** PDA is



which incorrectly accepts  $hhdu \notin L$ .

We can also design a correct PDA for  $L$  by applying the algorithm from Lemma 2.21 to convert the second correct CFG  $G_1$  in part (a) into a PDA.



Note that

- The path  $q_2 \rightarrow q_4 \rightarrow q_5 \rightarrow q_2$  corresponds to the rule  $S \rightarrow hSd$ , where the symbols on the right side of the rule are pushed in reverse order.
- The path  $q_2 \rightarrow q_6 \rightarrow q_7 \rightarrow q_2$  corresponds to the rule  $T \rightarrow hTu$ , where the symbols on the right side of the rule are pushed in reverse order.

5. For  $\Sigma = \{d, h, u\}$  (note the alphabet), let

$$A = \{ h^i u^j d^k \mid i, j, k \geq 0 \text{ and } i = j + k \}.$$

Is  $A$  a regular or nonregular language? If  $A$  is regular, give a regular expression **and** DFA (only state diagram) for  $A$ . If  $A$  is not regular, prove that it is a nonregular language.

**Answer:** The language  $A$  is **nonregular**. Note that  $A$  is from problem 3d from HW 4 but with a different  $\Sigma$ . For a generic symbol  $c \in \Sigma$  and string  $w \in \Sigma^*$ , let  $n_c(w)$  be the number of  $c$ 's in string  $w$ .

To prove that  $A$  is nonregular, suppose that  $A$  is a regular language. Let  $p$  be the “pumping length” of the pumping lemma (Theorem 1.70). Consider the string  $s = h^p u^p$ . Note that  $s \in A$ , and  $|s| = 2p \geq p$ , so the pumping lemma will hold. Thus, we can split the string  $s$  into 3 parts  $s = xyz$  satisfying the properties

- (a)  $xy^i z \in A$  for each  $i \geq 0$ ,
- (b)  $|y| > 0$ ,

(c)  $|xy| \leq p$ .

Since the first  $p$  symbols of  $s$  are all  $h$ 's, the third property implies that  $x$  and  $y$  consist only of  $h$ 's. So  $z$  will be the rest of the  $h$ 's, followed by  $u^p$ . The second property states that  $|y| > 0$ , so  $y$  has at least one  $h$ . More precisely, we can then say that

$$\begin{aligned}x &= h^j \text{ for some } j \geq 0, \\y &= h^k \text{ for some } k \geq 1, \\z &= h^m u^p \text{ for some } m \geq 0.\end{aligned}$$

Since  $h^p u^p = s = xyz = h^j h^k h^m u^p = h^{j+k+m} u^p$ , we must have that  $j+k+m = p$ . The first property implies that  $xy^2z \in A$ , where

$$\begin{aligned}xy^2z &= h^j h^k h^k h^m u^p \\&= h^{p+k} u^p\end{aligned}$$

since  $j+k+m = p$ . But  $xy^2z \notin A$  since  $k \geq 1$ , so we get a contradiction. Therefore,  $A$  is a nonregular language.