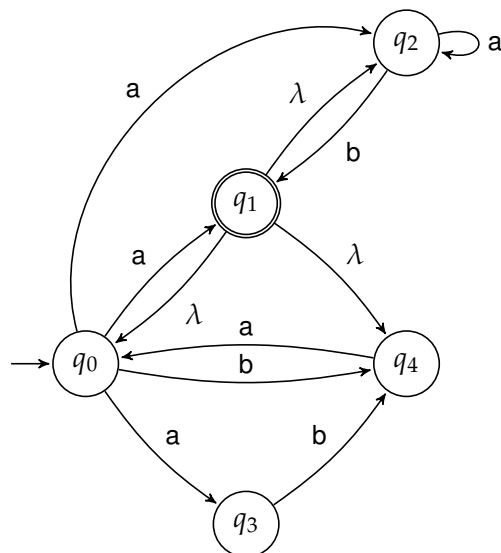


The following abbreviations are used in this paper.

| | |
|-------------|--|
| DFA | <i>deterministic finite-state automaton</i> |
| NDFA | <i>non-deterministic finite-state automaton</i> |
| FSA | <i>finite-state automaton</i> (either DFA or NDFA) |
| NPDA | <i>non-deterministic pushdown stack automaton</i> |
| CFG | <i>context-free grammar</i> |
| ID | <i>instantaneous description</i> |
| TM | <i>Turing machine</i> |

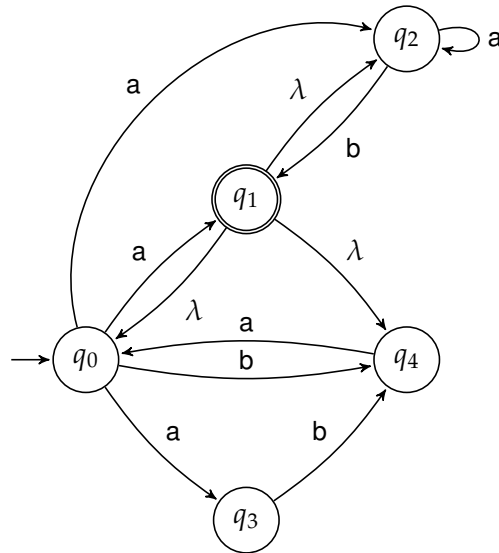
1 (5 marks)

State the five components of the NDFA given below:



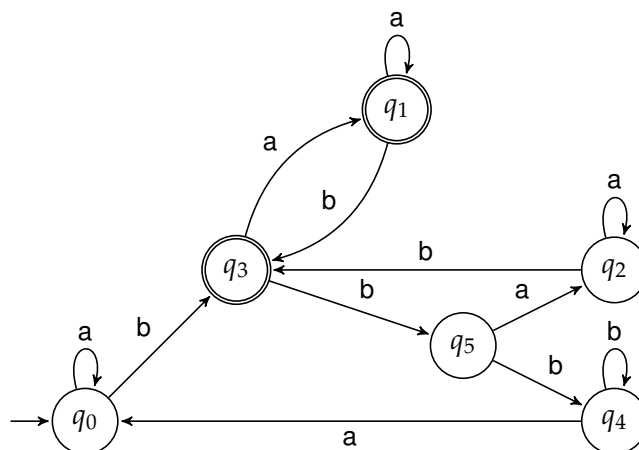
2 (10 marks)

Using a systematic procedure, convert the following NFA into an equivalent DFA, and draw the resulting automaton. Show clearly all the steps involved.



3 (10 marks)

By constructing a tree, identify the distinguishable states for the DFA below. Hence draw a DFA which is equivalent to this one, but which has a minimal number of states. Note that this DFA is complete; you should ensure that your answer is also a complete DFA.



4 (5 marks)

Consider the DFA in Question 3 and convert it into a regular expression using a systematic procedure. *[Hint: It may be easier to calculate the regular expression for the minimal DFA.]*

5 (5 marks)

The company you are working for has a number of systems that exchange information in binary format. The company wishes to add a primitive error-detection scheme where prior to communicating a binary string, a system must attach either a 0 or a 1 to the end of the original string according to the following conditions:

- The value attached is 0 if the number of 0s in the original string is greater than the number of 1s.
- The value attached is 1 otherwise; that is if the number of 1s is greater than or equal to the number of 0s.

Your manager has asked you to implement a DFA that checks whether a binary string received by a system contains an error or not.

Your task here is to either produce such a DFA or prove to your boss that a DFA cannot be created to solve the task.

[Hint 1: If it is easier, you may first construct an NFA and then convert it into a DFA.]

[Hint 2: You may use the pumping lemma to prove that the language is not regular, and therefore no DFA can be created.]

6 (5 marks)

Now, the company has decided to modify the error-detection schema. Prior to communicating a binary string, a system now must attach either a 0 or a 1 to the end of the original string according to the following conditions:

- The value attached is 0 if the number of 1s in the original string is even;
- The value attached is 1 otherwise; that is if the number of 1s in the original string is odd.

Your manager has asked you to implement a DFA that checks whether a binary string received by a system contains an error or not.

Your task here is to either produce such a DFA or prove to your boss that a DFA cannot be created to solve the task.

[Hint 1: If it is easier, you may first construct an NFA and then convert it into a DFA.]

[Hint 2: You may use the pumping lemma to prove that the language is not regular, and therefore no DFA can be created.]

7 (5 marks)

Give a context-free grammar for the following language:

$$L = \{a^i b^j c^k : i > k, 0 \leq j < 3, k \geq 0\}$$

8 (5 marks)

Consider the following context-free grammar:

$$S \rightarrow 0S0 \mid 1S1 \mid 0 \mid 1 \mid \lambda$$

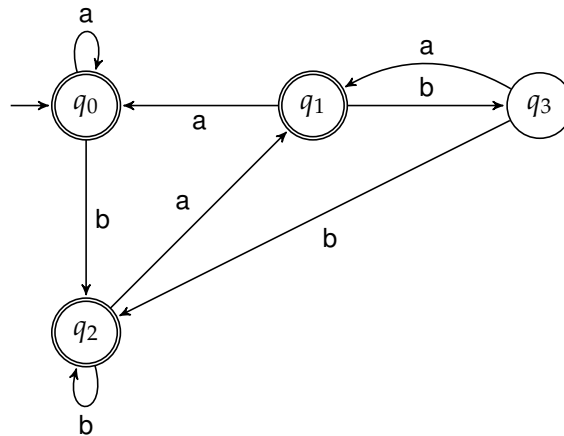
This grammar generates the language

$$L = \{w \in \{0,1\}^* : w = w^R\} \text{ where } w^R \text{ is the reverse of string } w.$$

Using the grammar, draw an NPDA which accepts the same language L , by empty stack, explaining how you have derived all of the states and transitions of the NPDA.

9 (5 marks)

Consider the following DFA:



Construct an equivalent CFG, using the structure of the DFA, and explain the process followed for each component of the grammar.

10 (5 marks)

Consider the following context-free grammar:

$$S \rightarrow aSb \mid bSa \mid SS \mid \lambda$$

(a) [2 marks] Give a derivation for the string *aabb*.

(b) [3 marks] Show that the grammar is ambiguous.

11 (5 marks)

Consider the following NPDA (accepting by final state)

$$M = \langle \{q_0, q_1, q_2\}, \{0, 1\}, \{0, 1, z\}, \delta, q_0, z, \{q_2\} \rangle$$

with the transition function δ defined by

$$\begin{aligned}\delta(q_0, 0, z) &= \{(q_1, 0), (q_2, \lambda)\} \\ \delta(q_1, 1, 0) &= \{(q_1, 1)\} \\ \delta(q_1, 1, 1) &= \{(q_1, 1)\} \\ \delta(q_1, 0, 1) &= \{(q_2, \lambda)\}\end{aligned}$$

- (a) [3 marks] Use a sequence of IDs to determine whether or not the string 0110 is accepted (by final state) by this NPDA
- (b) [2 marks] What is the language accepted by the NPDA?

12 (10 marks)

- (a) [5 marks] Draw an NPDA that accepts, by final state, the following language:

$$L = \{a^{i+j}b^ic^j : i \geq 0, j \geq 0\}$$

- (b) [5 marks] Give a CFG for the language L from part (a), and draw a parse tree for the string a^4bc^3 .

13 (5 marks)

Suppose that you are using the CYK algorithm to determine whether a string $abcdef$ can be generated by a particular context-free grammar (written in Chomsky normal form). Part-way through this process, you have produced the following table:

| | | | | | | |
|--|----------|----------|-------------|--------|--------|-----|
| | X_{16} | | | | | |
| | X_{15} | X_{26} | | | | |
| | A | X_{25} | X_{36} | | | |
| | A, B | B, C | B | S, A | | |
| | A, C | B | \emptyset | A | B | |
| | A | C | B | A | S, A | E |
| | a | b | c | d | e | f |

- (a) [1 mark] For the next entry X_{25} , which substring of $abcdef$ is being considered?
- (b) [3 marks] For that next entry X_{25} , which pairs of entries lower in the table are considered?

- (c) [1 mark] If the grammar does not generate $abcdef$, what can you say about the completed table?

14 (5 marks)

Consider the following context-sensitive grammar:

$$\begin{aligned} S &\rightarrow aSBC \mid aBC \\ CB &\rightarrow BC \\ aB &\rightarrow ab \\ bB &\rightarrow bb \\ bC &\rightarrow bc \\ cC &\rightarrow cc \end{aligned}$$

- (a) [2 marks] What is the language generated by this grammar?
- (b) [3 marks] Give a derivation to show how the string $aabbcc$ can be derived in the grammar.

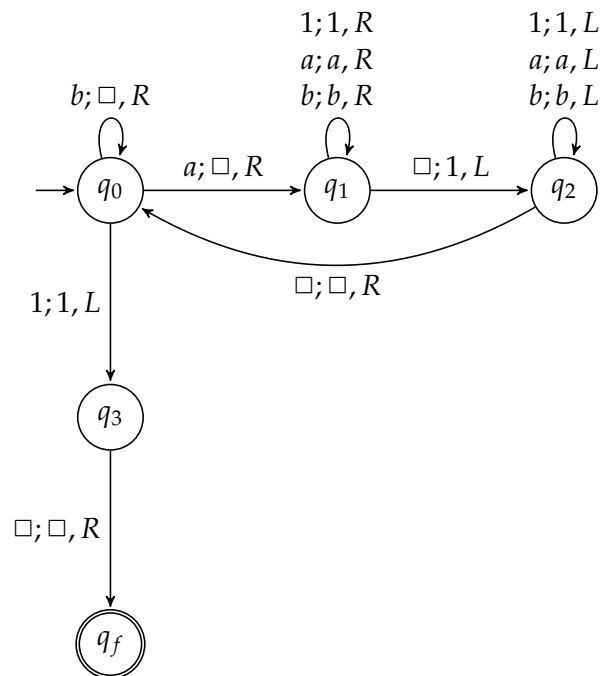
15 (10 marks)

Use the Pumping Lemma for context-free languages to show that the following language is not context-free:

$$L = \{0^j : j \geq 0\}$$

16 (5 marks)

Consider the following Turing machine:



- (a) [3 marks] Give a sequence of configurations / IDs to show how this Turing machine below processes an input of $abba$.
- (b) [2 marks] How is the contents of the tape at the end of processing a string on the tape related to the original contents of the string? You may assume that the string is not empty, and is made up from as and bs .