Theories of Computation

Solutions

Main Summer Examinations 2023

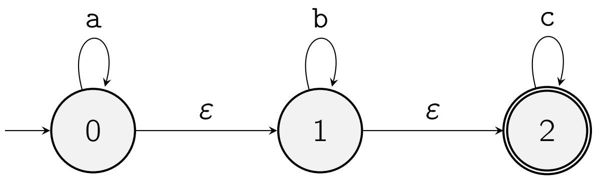
Note

Answer ALL questions. Each question will be marked out of 20. The paper will be marked out of 60, which will be rescaled to a mark out of 100.

Question 1

Take $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ to be the alphabet for this question.

(a) Explain how to remove ε-transitions from an εNFA and illustrate the procedure on the following automaton.



[6 marks]

(b) Consider the following context-free grammar $\mathcal{G}$

$$\begin{array}{rrcl} \Rightarrow & S & ::= & B\mathtt{c} \mid S\mathtt{c} \\ & B & ::= & \mathtt{a}B\mathtt{b} \mid \varepsilon \end{array}$$

where the $\Rightarrow$ symbol indicates the Start variable.

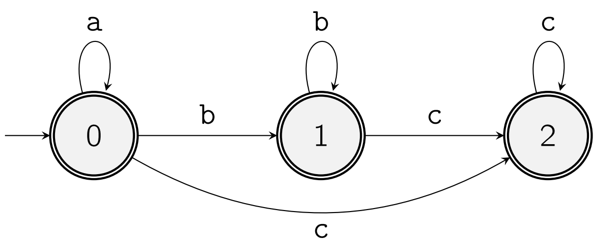
(i) Is the word abcc generated by this grammar? If yes, give a derivation tree for it; if no, explain why not. **[4 marks]**

(ii) What is the language $L(\mathcal{G})$ generated by the grammar $\mathcal{G}$? **[4 marks]**

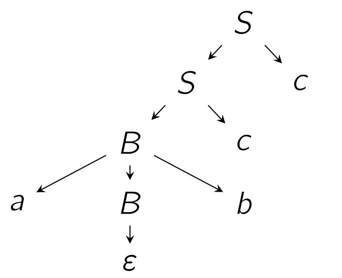
(c) Is the language $\{\mathtt{a}^n\mathtt{b}^n\mathtt{c}^n \mid n \geq 0\}$ regular? Explain your answer. **[6 marks]**

Model answer / LOs / Creativity:

(a) The first step is to make state 0 and 1 accepting since they have a ε-transition to 2 which is accepting. The second step is to add c-transitions from 0 and 1 to 2 and a b-transition from 0 to 1, while removing the ε-transitions.



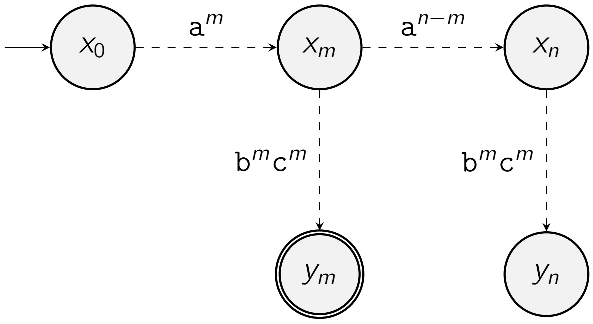
(b) (i) Yes.



(ii) $L(\mathcal{G}) = \{\mathtt{a}^n\mathtt{b}^n\mathtt{c}^m \mid n \geq 0, m \geq 1\}$

(c) L is not regular.

Suppose that we can give a deterministic finite automaton (DFA) D that recognizes L. For any $n \in \mathbb{N}$, let x_n denote the state of D reached after reading $\mathtt{a}^n$. A given state x_n does not accept the word $\mathtt{b}^m\mathtt{c}^m$ for any m such that $1 \leq m < n$, while $\mathtt{b}^m\mathtt{c}^m$ would be accepted by the corresponding state x_m.



Hence any x_n is inequivalent to all x_p for $p < n$. This means that the DFA D would need to consist of infinitely many different states, which contradicts its assumed finiteness.

Learning outcomes:

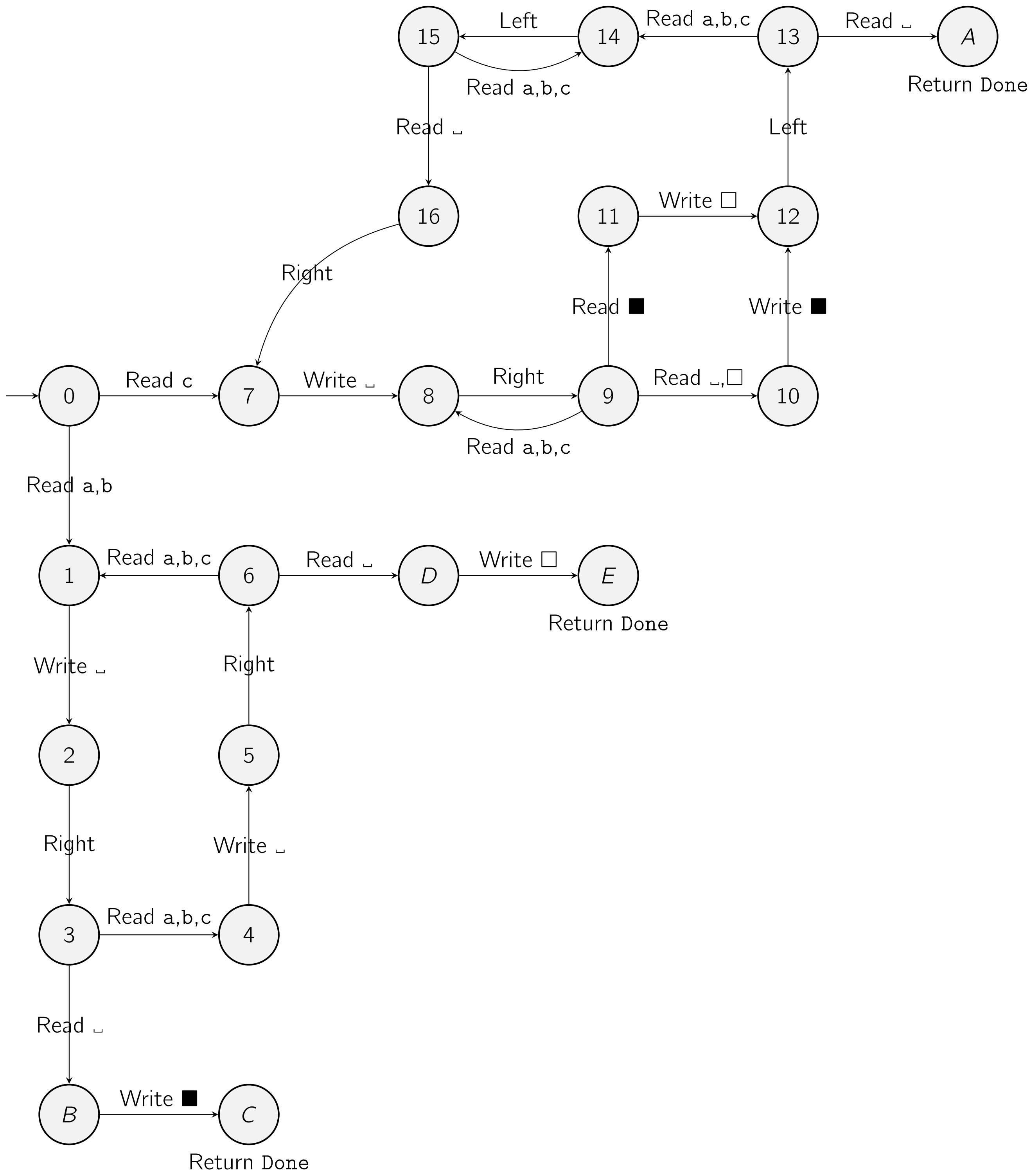
LO2 Explain and apply concepts from automaton theory, formal language theory, computability theory and complexity theory

LO3 Describe and use the connection between finite automata and regular language

(a) and (b)(i) are easy. (b)(ii) is creative. (c) is challenging.

Question 2

Consider the following Turing machine $\mathcal{M}$ on alphabet $\Omega = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \square, \blacksquare, \llcorner\!\lrcorner\}$ with return value set $V = \{\mathtt{Done}\}$.



The tape initially contains a non-empty block of `a`'s, `b`'s and `c`'s on an otherwise blank tape, with the head on the leftmost non-blank character.

(a) Give the complete run of machine $\mathcal{M}$ above on the word `abc`, with the head positioned on `a`.

At each step, indicate the tape contents, the position of the head, the current state and the instruction (including the result if it is a Read). **[6 marks]**

Hint: No more than 14 steps are required.

(b) The processing time on an input of length $n \geq 1$ is given by the function $f(n)$ when the first character of the input is `a` or `b`, and by the function $g(n) = 2n^2 + 9n - 1$ when the first character of the input is a `c`.

(i) Three students have tried to compute $f(n)$

- Akash found $f(n) = 3n + 3$;
- Belinda found $f(n) = 2n + 5$;
- Chao found $f(n) = 3n^2 + 1$.

Which student performed the correct computation? Explain your answer. **[3 marks]**

(ii) Assume that `a`, `b` and `c` are equally probable and all characters in the input are independent. What is the average processing time of machine $\mathcal{M}$? Prove that this average time is in $O(n^2)$. **[6 marks]**

(c) We would like to convert this Fancy Turing machine on alphabet $\Omega = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \square, \blacksquare, \text{␣}\}$ into a Simple one that only uses alphabet $\Sigma = \{a, b, c, \text{␣}\}$ via the following encoding

Character on Fancy tape	**Represented on Simple tape**
□	aa
■	ab
a	ba
b	bb
c	bc
␣	␣␣

The head of the Simple machine is situated on the leftmost of the two characters corresponding to the Fancy machine's head position. For example, the configuration

␣ a $\dot{\mathtt{b}}$ ■ ␣ would be represented as ␣ ␣ b a $\dot{\mathtt{b}}$ b a b ␣ ␣

Give a Turing machine that could be used as a macro on the Simple machine to simulate the Write □ instruction from the Fancy machine. **[5 marks]**

Model answer / LOs / Creativity:

(a)

␣ •a b c ␣	0	Read a
␣ •a b c ␣	1	Write ␣
␣ •␣ b c ␣	2	Right
␣ ␣ •b c ␣	3	Read b
␣ ␣ •b c ␣	4	Write ␣
␣ ␣ •␣ c ␣	5	Right
␣ ␣ ␣ •c ␣	6	Read c
␣ ␣ ␣ •c ␣	1	Write ␣
␣ ␣ ␣ •␣ ␣	2	Right
␣ ␣ ␣ ␣ •␣	3	Read ␣
␣ ␣ ␣ ␣ •␣	B	Write ■
␣ ␣ ␣ ␣ •■	C	Return Done

(b) (i) $f(n) = 3n + 3$.

When the first letter of the input is a or b, the machine transitions from state 0 to 1. For each letter of the input, there are three operations being performed: Read, Write ␣ and Right; this takes $3n$ steps.. Finally, after the last character is read, 3 more instructions are performed: Read ␣, Write □ (or ■) and Return Done.

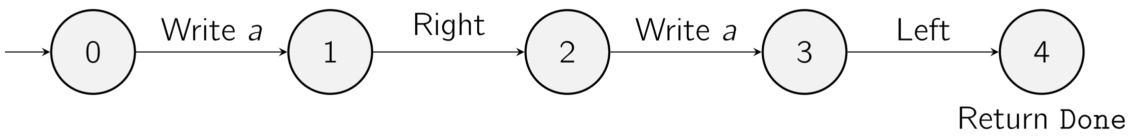
(ii) The average complexity is

$\frac{2}{3}f(n) + \frac{1}{3}g(n) = \frac{1}{3}(6n + 6 + 2n^2 + 9n - 1) = \frac{1}{3}(2n^2 + 15n + 5)$

$\frac{\frac{1}{3}(2n^2+15n+5)}{n^2} = \frac{2}{3} + \frac{5}{n} + \frac{5}{3n^2} \leq \frac{2}{3} + 5 + \frac{5}{3} \leq 8$ for all $n \geq 1$

so by taking $M = 1$ and $C = 8$ we have that $T(n) \in O(n^2)$.

(c)



Learning outcomes:

LO1 Explain and apply mathematical models of computations

LO2 Explain and apply concepts from automaton theory, formal language theory, computability theory and complexity theory

Question 3

(a) State Rice's theorem. **[4 marks]**

(b) A team of TAs for the Object Oriented Programming module have to mark the sorting programs of a class of 500 students. They want to write automated marking software that gives full marks to any correct answer and less than full marks to any incorrect one.

(i) Can the TAs do this in Java? Explain your answer. **[5 marks]**

(ii) Can they do this in a different language? Explain your answer. **[5 marks]**

(c) Recall that Primitive Java is a language with the type `nat` and the following basic facilities.

- `nat i = 0`
- `i++`
- `i--`, which does nothing if `i==0`
- `if (i == 0)` $\{M\}$ `else` $\{N\}$
- `repeat i times` $\{M\}$

The following facilities are derivable from the above basic facilities and may be used in answering this question.

- `nat j = i`
- `j = 0`
- `j = i`
- `if (i <= j)` $\{M\}$ `else` $\{N\}$
- `if (i < j)` $\{M\}$ `else` $\{N\}$
- `if (i == j)` $\{M\}$ `else` $\{N\}$
- `i = j + k`
- `i = j * k`
- `i = max(j-k,0)`

Note: In all of the instructions shown above, the variables used are arbitrary and can be replaced with any others.

To show that the exponentiation function is primitive recursive, give a Primitive Java encoding of:

$$\mathtt{i} = \mathtt{j}^{\mathtt{k}}$$

[6 marks]

Model answer / LOs / Creativity:

(a) Any semantic property of code that holds in some case and fails to hold in some case is undecidable.

(b) (i) To be a sorting program is a semantic property because it only depends on the input-output behaviour. We know moreover that it is possible to write a program that sorts a list and one that simply returns the input list without sorting it (for example). Therefore, by Rice's Theorem, the property of being a sorting program is undecidable.

(ii) By Church's thesis, this implies that there is no algorithm in any language that takes a program and returns True if it is a sorting program and False otherwise.

(c)

```
i = 0;
i++;
repeat k times {
  i = i*j;
}
```

Learning outcomes:

LO2 Explain and apply concepts from automaton theory, formal language theory, computability theory and complexity theory

LO4 Explain non-computability and undecidability issues

(a) is bookwork. (b) is relatively challenging. (c) is medium.