Faculty of Information Technology

Monash University

FIT2014 Theory of Computation FINAL EXAM

2nd Semester 2017

Working Space

Question 1 **(4 marks)**

Suppose we have propositions A, K and L, with the following meanings.

A: alphaGo is the equal-best Go player in the world.
K: Kē Jié is the equal-best Go player in the world.
L: Lee Se-dol is the equal-best Go player in the world.

Use A, K and L to write a proposition that is True if and only if **exactly two** of alphaGo, Kē Jié and Lee Se-dol are the equal-best Go players in the world.

For full marks, your proposition should be in Conjunctive Normal Form.

Question 2 **(3 marks)**

Suppose you have predicates `computableByGoedel`, `computableByChurch`, and `computableByTuring` with the following meanings, where variable $F : \{\mathtt{a}, \mathtt{b}\}^* \to \mathbb{N} \cup \{0\}$ represents an arbitrary function from finite strings over $\{\mathtt{a}, \mathtt{b}\}$ to nonnegative integers:

`computableByGoedel`(F):	the function F is a *recursive function*, according to Kurt Gödel's definition.
`computableByChurch`(F):	the function F has a *lambda expression*, in the sense of Alonzo Church's Lambda Calculus.
`computableByTuring`(F):	the function F is *computable*.

In the space below, write a statement in predicate logic with the meaning:

> Every function that satisfies any one of the three definitions of computability — recursive functions (Gödel), lambda calculus (Church), or Turing computability — also satisfies each of the others.

To do this, you may only use: the above three predicates; quantifiers; logical connectives. (In particular, you may not use set theory symbols such as $\subseteq$, $\subset$, $\cap$, $\cup$, etc, and in fact they would not help.)

Official use only
7

Question 3 **(6 marks)**

Let E_n be the following Boolean expression in variables $x_1, x_2, \dots, x_n$:

$$((\cdots((((\neg x_1 \vee x_2) \wedge \neg x_2) \vee x_3) \wedge \neg x_3) \cdots) \vee x_n) \wedge \neg x_n$$

For example, $E_1 = \neg x_1$, and $E_2 = (\neg x_1 \vee x_2) \wedge \neg x_2$. In general, $E_n = (E_{n-1} \vee x_n) \wedge \neg x_n$.

Prove by induction on n that, for all $n \geq 1$, the expression E_n is satisfiable.

Official use only
6

Question 4 **(4 marks)**

Write down all strings of length ≤ 6 that match the following regular expression:

$$\varepsilon \cup (\texttt{ab})^*\texttt{b}(\texttt{aaa} \cup \texttt{bb})^*$$

Official use only
4

Working Space

Question 5 **(4 marks)**

Let R be any regular expression.

(a) Give, in terms of R, a regular expression for the language of all strings that can be divided into two substrings, each of which matches R.

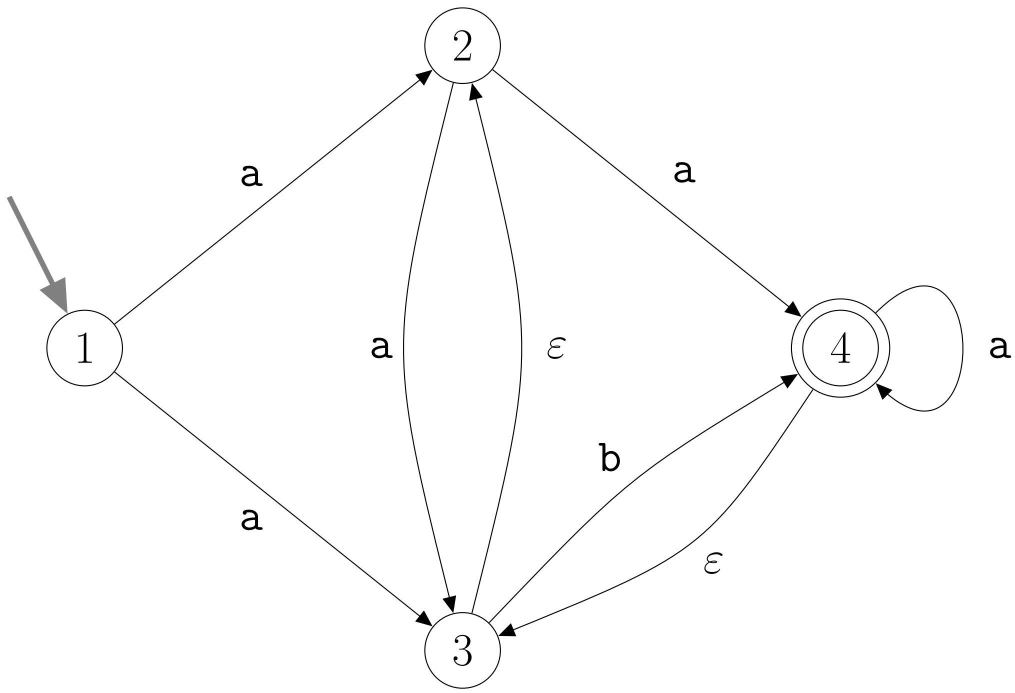
(b) Prove that the language EVEN(R) of all strings that can be divided into *any even number* of substrings, each of which matches R, is regular.

> For example, if R is $\texttt{a} \cup \texttt{bb}$, then the string $w = \texttt{aabba}$ can be divided into four substrings $\texttt{a},\texttt{a},\texttt{bb},\texttt{a}$ which each match R. So this w belongs to EVEN(R). The empty string is also in EVEN(R), noting that zero is an even number. But $\texttt{aabb}$ and $\texttt{bbb}$ do not belong to EVEN(R).

Official use only
4

Question 6 **(8 marks)**

(a) Convert the following Nondeterministic Finite Automaton (NFA) into an equivalent Deterministic Finite Automaton (FA).



Your FA must be presented by filling in some rows in the following table. You may not need all the rows available.

state	a	b

(b) Give a regular expression for the language accepted by the above NFA.

(You should not need to apply a general automaton-to-regexp conversion algorithm. Just think about what the automaton does. The equivalent FA should help.)

Question 7 **(5 marks)**

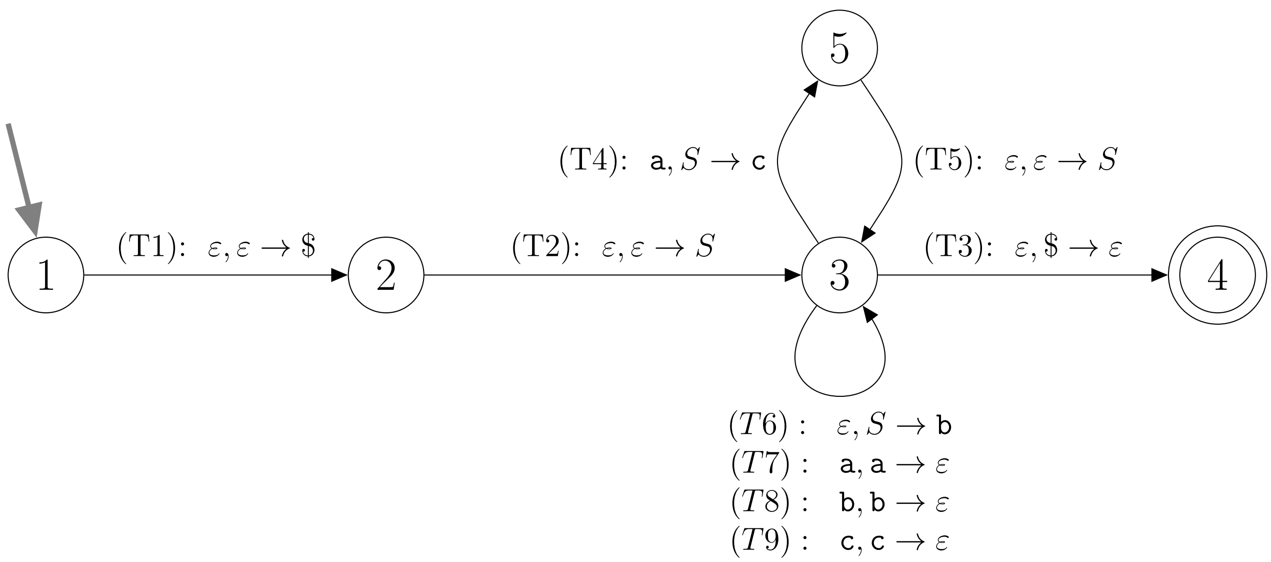
Give an algorithm which takes, as input, a Finite Automaton represented as a table, and finds another Finite Automaton that accepts the same language as the first one and has the minimum number of states among all FAs that accept that language.

A pseudocode description is fine.
Do not try to write your algorithm as a Turing machine.

Official use only
13

Question 8 **(12 marks)**

Consider the following Pushdown Automaton (PDA), with input alphabet {a,b,c} and extra stack symbols S and \$.



(a) Show that the single-letter string b is accepted by this PDA, by giving the sequence of transitions that leads to acceptance of b.

Use the names of the transitions, i.e., $T1, T2, \ldots$, etc.

(b) Prove the following statement by induction on n:

For all $n \geq 0$:
If the PDA is in State 3, the top symbol on the stack is S, and the remaining input begins with $\mathtt{a}^n\mathtt{b}\mathtt{c}^n$, **then** after reading $\mathtt{a}^n\mathtt{b}\mathtt{c}^n$ the PDA is again in State 3 and the stack is the same except that the S on the top has been removed.

(c) Hence prove that, for all $n \geq 0$, the string $\texttt{a}^n\texttt{b}\texttt{c}^n$ is accepted by this PDA.

Official use only
12

Question 9 **(13 marks)**

Let GOAL be the language generated by the following Context-Free Grammar:

$$S \rightarrow \mathtt{goo}X\mathtt{al} \qquad (1)$$

$$X \rightarrow \mathtt{oo}X\mathtt{a} \qquad (2)$$

$$X \rightarrow \varepsilon \qquad (3)$$

GOAL consists of all strings of the form $\mathtt{g(oo)}^{2n}\mathtt{a}^n\mathtt{l}$, where $n \geq 1$. The first few strings in this language, in order of increasing length, are:

`gooal`, `gooooaal`, `gooooooaaal`, ...

(a) Give a derivation for the string `gooooaal`.

Each step in your derivation must be labelled, on its right, by the number of the rule used.

(b) Give a parse tree for the same string, `gooooaal`.

(c) Use the Pumping Lemma for Regular Languages to prove that GOAL is not regular.

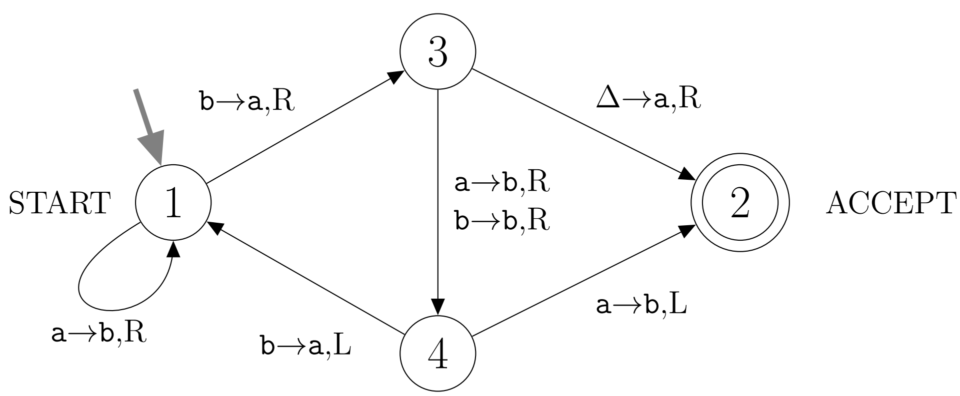
Official use only
13

Question 10 **(2 marks)**

The Cocke-Younger-Kasami (CYK) algorithm is less efficient than most commonly used parsing algorithms. What is one significant advantage it has over those algorithms?

Question 11 **(6 marks)**

Consider the following Turing machine.



Trace the execution of this Turing machine, writing your answer in the spaces provided on the next page.

The lines show the configuration of the Turing machine at the start of each step. For each line, fill in the state and the contents of the tape. On the tape, you should indicate the currently-scanned character by underlining it, and you should show the first blank character as Δ (but there is no need to show subsequent blank characters).

You should not need all the lines provided.

To get you started, the first line has been filled in already.

At start of step 1:	State:	1	Tape:	<u>b</u>	b	b	a	Δ	
At start of step 2:	State:		Tape:						
At start of step 3:	State:		Tape:						
At start of step 4:	State:		Tape:						
At start of step 5:	State:		Tape:						
At start of step 6:	State:		Tape:						
At start of step 7:	State:		Tape:						
At start of step 8:	State:		Tape:						
At start of step 9:	State:		Tape:						
At start of step 10:	State:		Tape:						
At start of step 11:	State:		Tape:						
At start of step 12:	State:		Tape:						

Official use only
8

Question 12 **(5 marks)**

(a) Name three variations on Turing machines that give the same class of computable functions.

(b) Give one way of modifying the definition of Turing machines so that they can still recognise all regular languages but can no longer recognise all decidable languages.

For full marks, your modification should be as simple as possible. It should involve altering just one part of the definition of Turing machines. Replacement of an entire machine by something else is not acceptable.

No proof is required for this question.

Official use only
5

Question 13 **(4 marks)**

For each of the following decision problems, indicate whether or not it is decidable.

You may assume that, when Turing machines are encoded as strings, this is done using the Code-Word Language (CWL).

Decision Problem	your answer (tick **one** box in each row)	
Input: two Turing machines M_1 and M_2. Question: Does M_1 eventually halt, when given M_2 as input?	☐ Decidable	☐ Undecidable
Input: two Turing machines M_1 and M_2. Question: Does M_1 have the same number of states as M_2?	☐ Decidable	☐ Undecidable
Input: two Turing machines M_1 and M_2. Question: Is M_1 equivalent to M_2 (i.e., do M_1 and M_2 have the same sets of accepted strings and the same sets of rejected strings)?	☐ Decidable	☐ Undecidable
Input: two Turing machines M_1 and M_2. Question: If each machine is given itself as input, does M_1 finish before M_2?	☐ Decidable	☐ Undecidable

Official use only
4

Question 14 **(12 marks)**

The Venn diagram on the next page shows several classes of languages. For each language (a)–(j) in the list below, indicate which classes it belongs to, and which it doesn't belong to, by placing its corresponding letter in the correct region of the diagram.

If a language does not belong to any of these classes, then place its letter above the top of the diagram.

You may assume that, when Turing machines are encoded as strings, this is done using the Code-Word Language (CWL), with input alphabet {`a`,`b`} and tape alphabet {`a`,`b`,`#`,Δ}.

(a) The set of all binary strings.

(b) The set of all strings in which every `a` occurs before every `b`.

(c) The set of all strings in which `a` occurs more times than `b`.

(d) The set of all strings in which every `a` occurs before every `b`and ALSO `a` occurs more times than `b`.

(e) The set of adjacency matrices of 2-colourable graphs.

(f) The set of adjacency matrices of 4-colourable graphs.

(g) The set of all Boolean expressions in Conjunctive Normal Form (CNF), whether satisfiable or unsatisfiable. (Assume the variables are $x_1, x_2, \ldots, x_n$, and variable x_i is represented by its index i as a binary positive integer.)

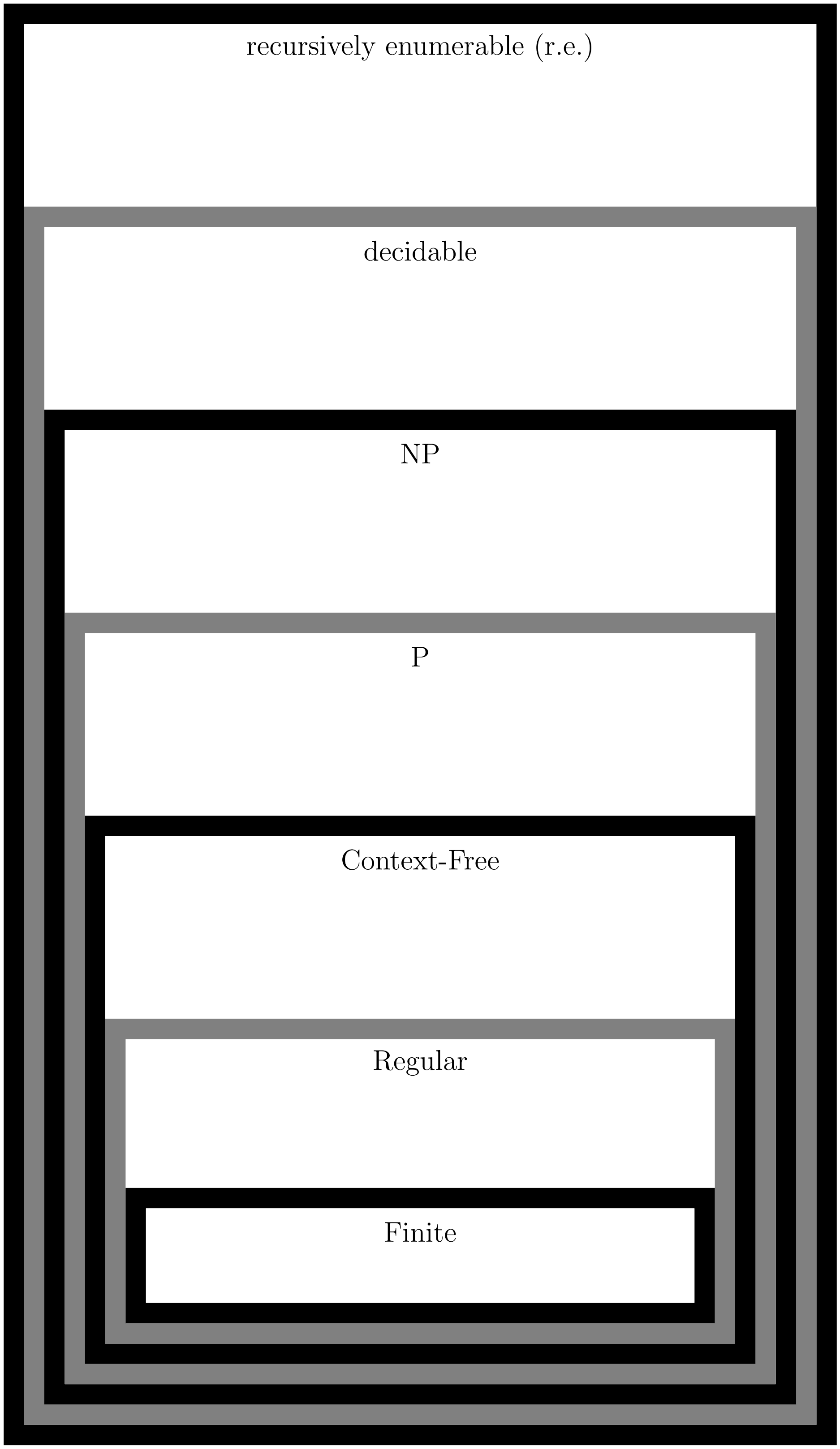
(h) The set of all encodings of Turing machines that accept regular languages.

(i) The set of all encodings of Turing machines that accept non-context-free languages.

(j) The set of all encodings of Turing machines that loop forever for some input.

(k) The set of all arithmetic expressions involving positive integers, in decimal notation, and addition, subtraction, multiplication and division, but with no parentheses.

(l) The set of all arithmetic expressions involving positive integers, in decimal notation, and addition, subtraction and parentheses, but no multiplication or division.



Official use only
12

Working Space

Question 15 **(8 marks)**

Let L and K be languages. Suppose that K is finite.

Definition: The **symmetric difference** $L \Delta K$ consists of all strings that belong to L or K, *but not both.* In other words, $L \Delta K = (L \cup K) \setminus (L \cap K)$.

(a) Prove that there exists a mapping reduction from L to $L \Delta K$.

(b) Prove that L is undecidable if and only if $L \Delta K$ is undecidable.

Official use only
8

Question 16 **(5 marks)**

An enumerator for a language is normally allowed to output a given string in the language more than once. A **direct enumerator** is an enumerator which never outputs a string more than once.

Prove that if a language has an enumerator then it has a direct enumerator.

Official use only
5

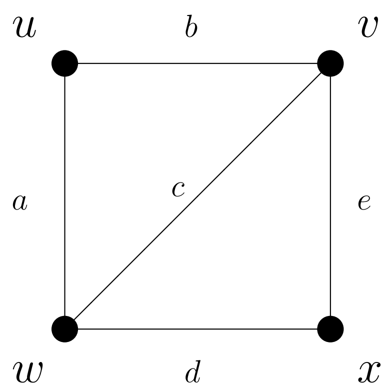
Question 17 **(11 marks)**

A **vertex cover** in a graph G is a set X of vertices that meets every edge of G. So, every edge is incident with at least one vertex in X.

The VERTEX COVER decision problem is as follows.

VERTEX COVER
Input: Graph G.
Question: Does G have a vertex cover?

For example, in the following graph, the vertex set $\{u, w, x\}$ is a vertex cover, and so is $\{v, w\}$. But $\{u, x\}$ is not a vertex cover, since it does not meet every vertex. (Specifically, it misses the diagonal edge c.)



Let W be the above graph.

(a) Construct a Boolean expression E_W in Conjunctive Normal Form such that the satisfying truth assignments for E_W correspond to vertex covers in the above graph W.

(b) Give a polynomial-time reduction from VERTEX COVER to SATISFIABILITY.

Official use only
11

Question 18 **(8 marks)**

Prove that the problem LONG PATH is NP-complete, using reduction from HAMILTONIAN PATH. You may assume that HAMILTONIAN PATH is NP-complete.

Definitions:

HAMILTONIAN PATH
Input: Graph G.
Question: Does G have a path that contains every vertex?

LONG PATH
Input: Graph G, with an even number of vertices.
Question: Does G contain a path of length $\geq n/2$?

In each of these definitions, n is the number of vertices in the graph G.

Official use only
8

Working Space

Working Space

END OF EXAMINATION