Theories of Computation

Solutions

May 2012

Theories of Computation

Exam length: 1.5 hours. Answer THREE questions out of the four. The marks available total 102, but will be capped at 100.

Question 1 : Regular Languages and Automata

(a) (i) Write out a regular expression that is matched by exactly those strings over an alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ in which after any `b` there can be no more occurrences of `a`. **[4 marks]**

(ii) Consider the two strings `ccaccbcc`, which has the given property, and ccbccacc, which does not. Explain why ccaccbcc matches your pattern, but `ccbccacc` does not. **[8 marks]**

(b) Design a deterministic finite automaton that accepts exactly those strings that match the regular expression $(\mathtt{ab}^* \mid (\mathtt{ab})^*)$. **[12 marks]**

(c) Consider the language $\mathcal{L}$ consisting of all those strings over the alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}\}$ that contain more `a`'s than `b`'s. Show that $\mathcal{L}$ is not regular. **[10 marks]**

Solutions

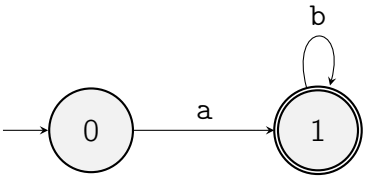
(a) (i) $(\mathtt{c} \mid \mathtt{a})^*(\varepsilon \mid \mathtt{b}(\mathtt{b} \mid \mathtt{c})^*)$

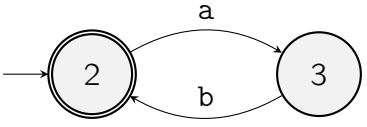
(ii) The string `ccaccbcc` consists of two parts

- `ccacc`, which matches $(\mathtt{c} \mid \mathtt{a})^*$
- `bcc` which matches $\mathtt{b}(\mathtt{b} \mid \mathtt{c})^*$ and therefore $\varepsilon \mid \mathtt{b}(\mathtt{b} \mid \mathtt{c})^*$

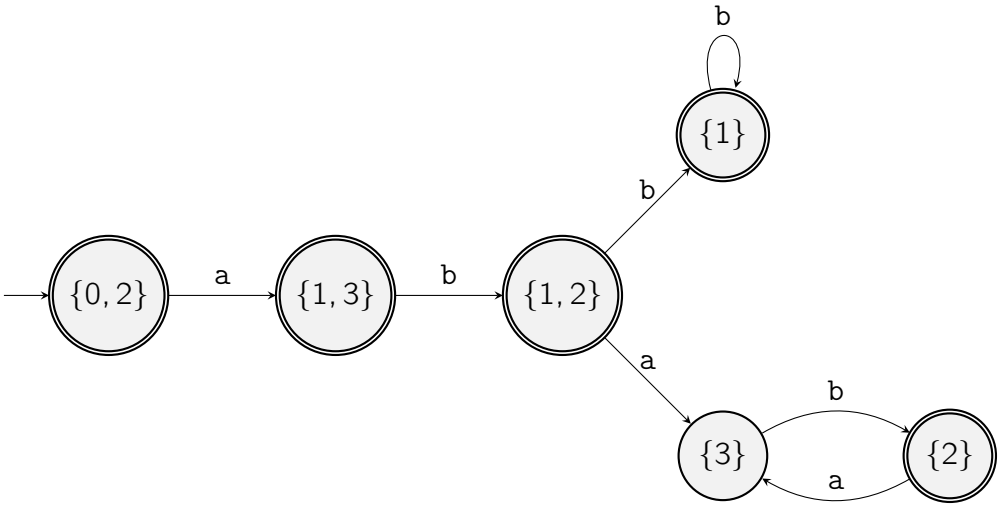
On the other hand `ccbccacc` does not divide up in any way that matches the pattern; if the second part is to be empty, the first part must be `ccbccacc`, which is impossible, and if the first part begins with `b`, the first part will be `cc` and the second part is `bccacc` but that doesn't match $\mathtt{b}(\mathtt{b} \mid \mathtt{c})^*$.

(b) One way to obtain the DFA is to start by design an NFA and then determinize it.





After determinization:



(c) If $\mathcal{L}$ is regular then there exists a DFA that recognizes it. Let x_n be the state we reach when we start at the initial state and read $\mathtt{a}^n$. We shall show these states are all distinct, which implies that there are infinitely many states, and result in a contradiction.

Suppose m and n are natural numbers with $m < n$. We want to show that $x_m \neq x_n$. If we start at x_m and read $\mathtt{b}^m$ we reach a rejecting state, because $\mathtt{a}^m\mathtt{b}^m \notin \mathcal{L}$, but if we start at x_n and read $\mathtt{b}^m$ we reach an accepting state, because $\mathtt{a}^n\mathtt{b}^m \in \mathcal{L}$. So x_m and x_n cannot be the same.

Question 2 : Context-free languages

In this question we consider a grammar given, over an alphabet $\Sigma = \{\mathtt{l}, \mathtt{r}, \mathtt{x}\}$, by

$$\begin{array}{rrcl} \Rightarrow & S & ::= & \varepsilon \mid TS \\ & T & ::= & \mathtt{x} \mid \mathtt{l}S\mathtt{r} \end{array}$$

(a) Of the two strings `xlrxrxlx` and `xlxlxrrx`, one is derivable (from the initial S) in the grammar and one is not. Which is which? Justify your answer by giving in each case either a derivation tree or a careful argument to show that every attempt to find a derivation fails. **[14 marks]**

(b) *Removed. Not on the syllabus this year.* **[6 marks]**

In the rest of the question we shall use the notion of **bracket count**. (Think of `l` and `r` as left and right brackets.) Imagine counting along from the left of a string, starting at 0 and adding $+1$ for each `l`, -1 for each `r` and 0 for each `x`. At each point of the string, the total so far is called the bracket count there. For example, in the string `xxlxlrrrx` the bracket counts are shown as subscripts as

$${}_0x_0x_0l_1x_1l_2r_1r_0r_{-1}x_{-1}$$

We say that a string is **well bracketed** if its bracket count finishes at 0 and never goes negative at any intermediate point.

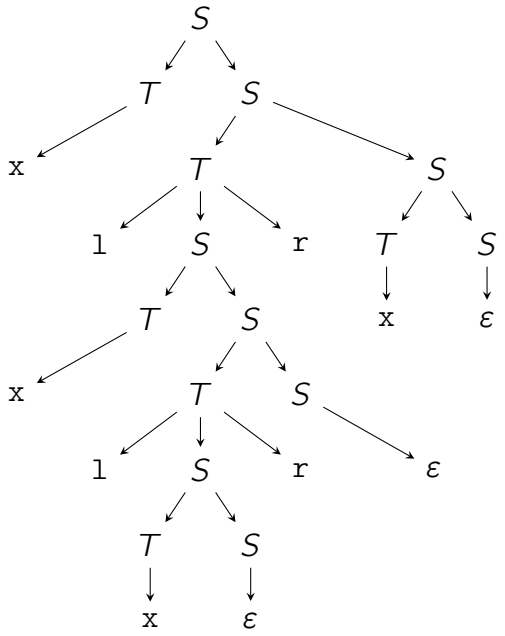
(c) Show by induction that every string derived from S is well bracketed. **[7 marks]**

(d) *Removed. Not on the syllabus this year.* **[7 marks]**

Solutions

(a) Suppose `xlrxrxlx` is an S-word. Since it is not empty, it has to divide into a T-word t and then an S-word s. Now t does not begin with `l` so must be `x`. Hence, s is `lrxrxlx`. Since it is not empty, it divides into a T-word t' then an S-word s'. Now t' is not `x` so it must begin with `l` and end with `r`; so it is either `lrxr` or `lr`. If $t' =$ `lrxr`; then since `lrxr` is not `x`, we must have that `rx` is an S-word and since it is not empty it must divide into a T-word then an S-word, but no prefix of `rx` is either `x` or begins with `l`, contradiction. So t' must be `lr` and s' is `xrxlx`, and the latter divides into a T-word, clearly `x`, followed by an S-word, which must be `rxlx`. This in turn, since it is not empty, must divide into a T-word followed by an S-word, but no prefix of `rxlx` is either `x` or begins with `l`, contradiction. Therefore `xlrxrxlx` is not an S-word.

Furthermore, we can give the following derivation tree for `xlxlxrrx`.



(b) *Removed. Not on the syllabus this year.*

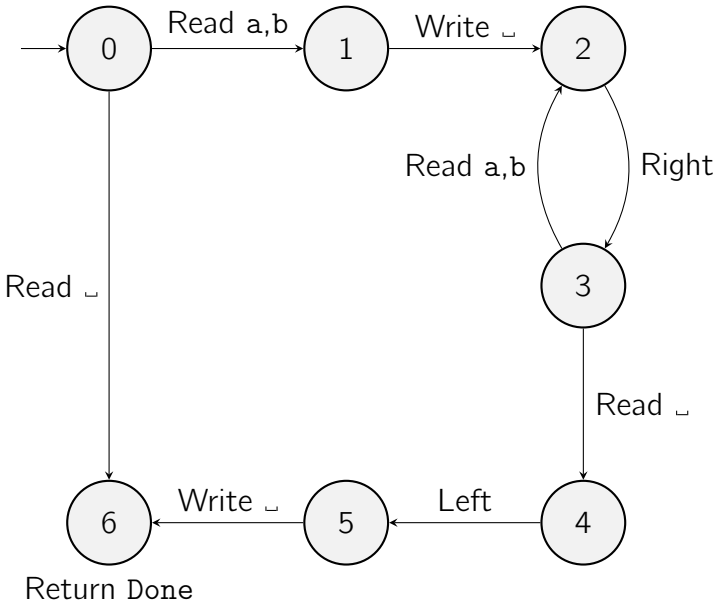
(c) We shall show by induction that every S-word and also every T-word is well-bracketed.

- ε is well-bracketed.
- If t is well-bracketed and s is well-bracketed then ts is well-bracketed, because the bracket count at the end of t is 0 so the bracket count in the s part is the same as in s alone.
- `x` is well-bracketed.
- If s is well-bracketed then `lsr` is well-bracketed, because the bracket count after `l` is 1, so the bracket count in the s part is 1 greater than it is in s alone, so it can never go < 1 and ends at 1. The final `r` then takes the bracket count to 0 as required.

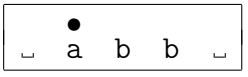
(d) *Removed. Not on the syllabus this year.*

Question 3 : Turing Machines and Complexity

(a) Consider the following deterministic Turing machine $\mathcal{M}$ on alphabet $\Omega = \{\mathtt{a}, \mathtt{b}, \sqcup\}$ with return value set $V = \{\mathtt{Done}\}$. The tape initially contains a non-empty block of a's and b's on an otherwise blank tape, with the head on the leftmost non-blank character. The transition function is given by the following diagram:



(i) *Removed. Not on the syllabus this year.* **[6 marks]**

(ii) Write out the configurations for the execution of this machine if it starts with input abb, i.e., the initial configuration is  **[6 marks]**

(b) What does it mean to say that a Turing machine executes in polynomial time?

Note. You may use the $\mathcal{O}$ notation without defining it. **[4 marks]**

(c) (i) Outline how a Turing machine $\mathcal{M}$ with two tapes, each with its own head, can be simulated by a Turing machine $\mathcal{N}$ with one tape. **[5 marks]**

(ii) Explain why if $\mathcal{M}$ executes in polynomial time then so does $\mathcal{N}$. **[5 marks]**

Note. Your outline in part (i) can be very brief, but it should include enough detail to support your explanation in part (ii).

(d) Which of the following operations are known to be executable in polynomial time on a Turing machine?

(i) Determining for two integers n and m whether m is a factor of n. **[2 marks]**

(ii) Determining for an integer n whether it has a factor other than 1 and n. **[2 marks]**

(iii) Finding the smallest factor (other than 1) of n. **[2 marks]**

How would your answers change if the "P = NP?" problem were solved, either way? Give reasons. **[2 marks]**

Solutions

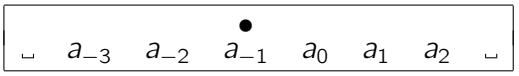
(a) (i) *Removed. Not on the syllabus this year.*

(ii)

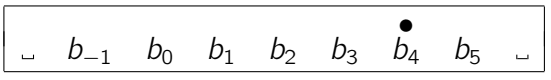
␣ **a** b b ␣	0	Read a
␣ **a** b b ␣	1	Write ␣
␣ ␣ b b ␣	2	Right
␣ ␣ **b** b ␣	3	Read b
␣ ␣ **b** b ␣	2	Right
␣ ␣ b **b** ␣	3	Read b
␣ ␣ b **b** ␣	2	Right
␣ ␣ b b **␣**	3	Read ␣
␣ ␣ b b **␣**	4	Left
␣ ␣ b **b** ␣	5	Write ␣
␣ ␣ b **␣** ␣	6	Return Done

(b) It means that there exists numbers N, C and k such that, if the length n of the input is at least N, then the machine terminates in time $\leq Cn^k$.

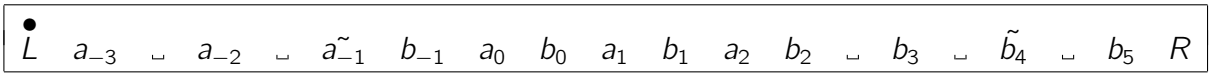
(c) (i) We intersperse the two tapes as follows: If the main tape is



and the secondary tape is



then the single tape simulating them is



in the same state. (We assume an extra tape character $\tilde{x}$ for each input character x, and extra tape characters L and R.)

- To work on the main tape we first move right (once and then two at a time) until we hit the˜-marked character representing the head position on the main tape.
  Then
  - to simulate a move right, we move two steps to the right, and mark with a˜, moving the R position if necessary.
  - likewise to simulate a move left
  - to simulate an output, output the appropriate˜-marked character. Finally we move left until we reach the L position.
- Likewise to work on the secondary tape.

(ii) Suppose $\mathcal{M}$ executes in a time polynomial in the length n of the input, so it is bounded by $\leq Cn^k$. Then the space used is $\leq n + Cxn^k$. Each step in the simulation takes at most this many steps, so the overall time taken by the simulation is

$$\leq Cn^k \times (n + Cn^k)$$

which is polynomial of order $2k$.

(d)
- Yes, this is polynomial in the length of n. Just do long division and see if there is a remainder.
- Yes, this is primality testing, which is polynomial in the length of n.
- Not known to be polynomial.

The third one is in NP since by the first one has a checking problem polynomial in n. Therefore it would be polynomial if $\mathsf{P} = \mathsf{NP}$. It is not known whether the third one is NP-complete, and it might be polynomial even if $\mathsf{P} \neq \mathsf{NP}$.

Question 4 : Computability and Models of Computation

(a) We consider Java programs that take their input from a file and print their output to `System.out` when the program terminates. What does it mean to say that a property of such programs is semantic? What does Rice's Theorem tell us about semantic properties? **[5 marks]**

(b) The following two properties of programs are both undecidable.

(i) The output of the program can differ, depending on the first character of the input.

(ii) The output of the program can differ, depending on the name of the input file.

Which one is a semantic property? Outline how you would prove undecidability of each property, using Rice's Theorem where appropriate. **[12 marks]**

(c) *Removed. Not on the syllabus this year.* **[5 marks]**

(d) *Removed. Not on the syllabus this year.* **[6 marks]**

(e) *Removed. Not on the syllabus this year.* **[6 marks]**

Solutions

(a) Being semantic means that the input-output behaviour (the infinite table showing next to every possible inputs, the corresponding output or nontermination) determines whether the property holds or not. Rice's theorem says that every such property, if it holds in some cases and fails to hold in some cases, is undecidable.

(b) The first one is a semantic property. It is satisfied by a program that just prints the first input character, and not satisfied by a program that just hangs. Therefore, by Rice's theorem it is undecidable.

As for the second, assuming the filename is supposed to be passed as an argument to the procedure, the input consists of both the filename and the contents of the file. So the property is semantic. It is satisfied by a program that just prints the first letter of the filename and not satisfied by a program that just hangs. Therefore, by Rice's theorem it is undecidable.

(c) *Removed. Not on the syllabus this year.*

(d) *Removed. Not on the syllabus this year.*

(e) *Removed. Not on the syllabus this year.*