DELFT UNIVERSITY OF TECHNOLOGY
Faculty of Electrical Engineering, Mathematics and Computer Science

Resit TI2316
Automata, Languages & Computability

August 13, 2019, 9:00–12:00

- Total number of pages (without this cover page): 10.
- This exam consists of 10 open questions, the weight of each subquestion is indicated on the exam.
- Consulting handouts, readers, notes, books or other sources during this exam is prohibited. The use of electronic devices such as calculators, mobile phones etc is also prohibited.
- A single exam cannot cover all topics, so do not draw conclusions based on this exam about topics that are never tested.
- Formulate your answers in correct English and write legibly (use scrap paper first). Do not give irrelevant information, this could lead to a deduction of points.
- Before handing in your answers, ensure that your name and student number is on every page and indicate the number of pages handed in on (at least) the first page.
- **Note:** for some exercises a maximum is stated for the number of lines an answer can consist of! Exceeding this number may lead to deduction of points.

Question:	1	2	3	4	5	6	7	8	9	10	Total
Points:	5	4	8	6	6	6	4	4	4	9	56

Learning goals coverage, based on the topics of the study assignments:

Goal	M16	E16	M17	E17	R17	MT18	ET18	RT 18
Strings & Operations								
Inductive proof over strings								
Languages & Operations								
(Formal) def. of DFA	2b	1a	1a, 1b		1a, 5c	1a		1a
Accepting words & Recognising languages (DFA)				1a	1b			
(Formal) def. of δ^*	1b		1a, 1b	1b		1b		
Regular languages & operations						1c		1b
Extension input alphabet of a DFA				2b				
Closure under union, complement, intersection	2b	2c		2a	1c	2a,2b		2
(Formal) def. of NFA	1a		1c	3c	5c			3a
Equivalence of NFA & DFA		2c	3a,3c		2b			
Accepting words & Recognising languages (NFA)			2a		2a	3 a		3b
Closure under concat and star	1d	1c	2b		2c	3 b		
(Formal) def. of regexp		2a		3a,3c	2d, 5c	3 b		
(Formal) def. of GNFA			1c			4		
Equivalence of NFA & regexp	1 c	2b	3b,3c	3d				3c
Accepting words & Recognising languages (GNFA)				3b				
Nonregular languages	2a		4	4		5		
Pumping lemma for reg. languages	2a		4	4		7		4
Basic concepts of grammar	3a		5a					
(Formal) def. of CFGs	3c	1b,3a,3b	5b		3a,3c	5 a		5a
Derivations & Ambiguity	3b				3b	5 b		5a
Closure of context-free lang	3d	3 c	5c					
Converting a CFG into CNF		3d		5a				5b
(Formal) def. of PDAs	4a, 4b, 4c				4a			
Equivalence of CFGs & PDAs				5b	4b	6		
(Formal) def of (multitape) DTMs		4a,4b,4c					1a,1c	6a
(Formal) def of (multitape) NTMs				8a,8b	5a,5b, 5c		2a,2b,2c	6b
Deciders		4d,5b,5c		8c			3a	7
Equivalence of TMs				6b			1b	6c
Comp. power of NTMs and DTMs					5d			
Differences between NTMs and DTMs								
König's Lemma							2	
Enumerators				9a	6a			8a
Recognisers		5b		9a			3b	
Hilbert's Entscheidungsproblem								
Churing-Turing Thesis								
Encoding TMs/Problems								
Decidable languages				6a	6b			
Countable vs Uncountable					8a			
Hilbert Hotel								
(Un)Countability of $\mathbb{Q}$ and $\mathbb{R}$					8b			
Halting problem				6c,6d	9a			8b
Acceptance problem		5a(?)		9a			4a	
Universal TMs								
co-Turing-recognizability				9b			4b	
(Formal) def. of a reduction				7a,7b			5a,5b	
Direct reductions							7	9
Computable functions				8a	9b			
Mapping/Many-to-one reducability		5b,6		10a,10b	9c			10
Rice's theorem					10		6	
Reduction via computation histories								

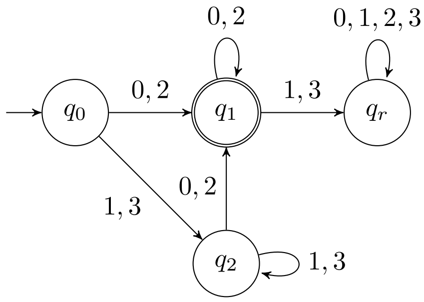
1. (a) (3 points) Consider a modification of the DFA model, called SFA. The SFA model is identical to the DFA with the extra requirement that $|F| = 1$. Is the SFA model equally expressive as the DFA model? If so, explain how we can construct an SFA from a DFA. If not, give an example of a DFA that has no SFA equivalent and argue why (no full proof is needed).

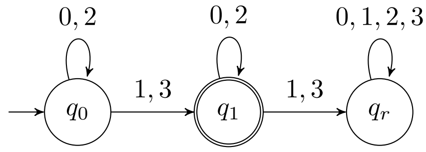
   Solution: Take a machine D that returns true only if the remainder when dividing by 3 is at most 1. This has two accept states q_0 (the start state) and q_1 (for a remainder of 1), and only q_2 rejects. It needs all three of these states however, as we need to keep track of a 3 option counter. Thus we cannot create an SFA for this as we cannot collapse q_0 and q_1 together.

   (b) (2 points) Take two DFAs D_1 and D_2. Now take the language $L = (L(D_1) \cup L(D_2))^*$. Is L regular? Explain your answer in at most 5 lines.

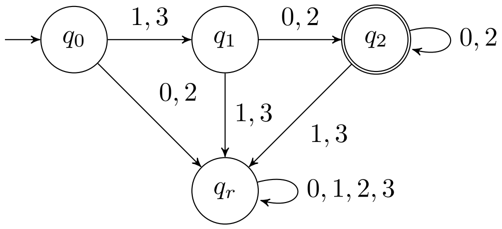
   Solution: Yes, regularity is closed under both $\cup$ and $*$.

2. (4 points) Consider the two DFAs D_1 and D_2 below. Create a new DFA D_3 with at most 5 states, so that language: $L(D_3) = L(D_1) \cap L(D_2)$. Explain how you constructed your answer in at most 5 lines.

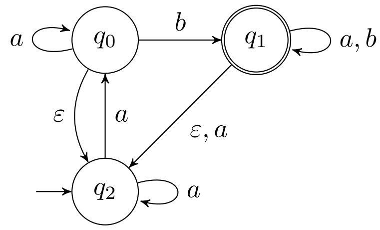




Solution: $L(D_1)$ is described by the regular expression: $(1 \cup 3)^*(0 \cup 2)(0 \cup 2)^*$. $L(D_2)$ is described by the regular expression: $(0 \cup 2)^*(1 \cup 3)(0 \cup 2)^*$. Thus $L(D_3)$ (the intersection), must have at least one 1 or 3(this must be at the start) and then followed by at least one 0 or 2, followed by zero or more iterations of 0 or 2. Thus the regular expression: $(1 \cup 3)(0 \cup 2)(0 \cup 2)^*$. In a DFA:



3. Consider the NFA $N = (\{q_0, q_1, q_2\}, \{a, b, c\}, \delta, q_2, \{q_1\})$, whose transition graph is depicted below.



(a) (2 points) Give a formal description of δ in the form of a table.

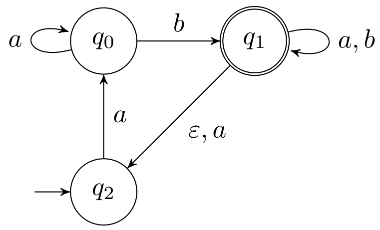
Solution:

	a	b	c	ε
q_0	$\{q_0\}$	$\{q_1\}$	$\emptyset$	$\{q_2\}$
q_1	$\{q_1, q_2\}$	$\{q_1\}$	$\emptyset$	$\{q_2\}$
q_2	$\{q_0, q_2\}$	$\emptyset$	$\emptyset$	$\emptyset$

(b) (1 point) Give a word $w \in L(N)$ such that $|w| = 5$ and $w^R \notin L(N)$.

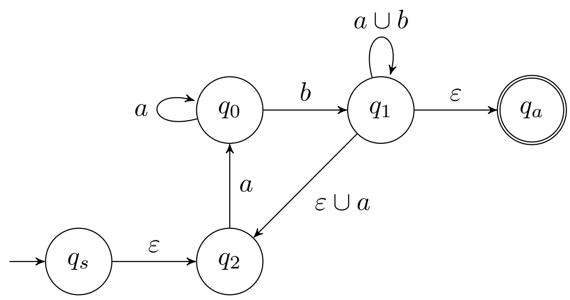
Solution: For example $aaaab$.

(c) (5 points) Consider now the NFA N', whose transition graph is depicted below.

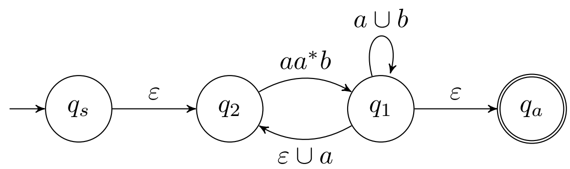


Construct a regular expression R such that $L(N') = L(R)$ using the method from Sipser. Show all intermediate steps.

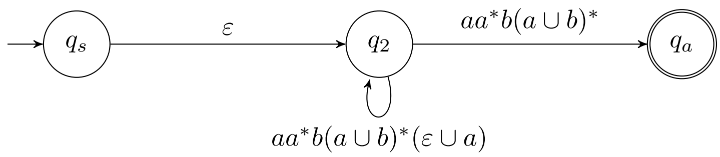
Solution:
Add start state and accept state:



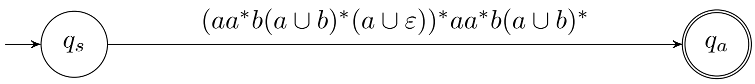
Remove q_0:



Remove q_1:



Remove q_2:



So take $R = (aa^*b(a \cup b)^*(a \cup \varepsilon))^*aa^*b(a \cup b)^*$

4. (6 points) Consider the language of correctly closed angular brackets that uses the alphabet $\Sigma = \{\langle, \rangle\}$.

$$L = \{w \mid w \text{ is a well-formed sequence of angular brackets}\}$$

For example $\langle\langle\rangle\rangle$ and $\langle\langle\rangle\langle\rangle\rangle$ are both elements of L, but $\langle\langle\rangle$ is not. Is L regular? If so, prove it by constructing a regular expression R such that $L = L(R)$. If not, prove it using the pumping lemma.

Solution:

Proof. Proof by contradiction:

- Suppose L is regular.
- This means there must exist some pumping length $p > 0$ for L such that all words w longer than p can be split up into three parts x, y and z, with $|y| > 0$ and $|xy| \leq p$.
- For this division of w, and any $i \geq 0$, $xy^iz \in L$.
- Let's take the word $w = \langle^p\rangle^p$ which is in L.

- This word is longer than p, so the above holds for this word.
- Given the requirements, we know that there is only one possible split
  - $x = \langle^{\alpha}, y = \langle^{\beta}$ and $z = \langle^{p-\alpha-\beta}\rangle^{p}$, with $0 \leq \alpha < p$, $0 < \beta \leq p$ and $\alpha + \beta \leq p$.
    * Now, taking $i = 0$, we get $\langle^{\alpha} b^{p-\alpha-\beta}\rangle^{p} = \langle^{p-\beta}\rangle^{p}$, which is clearly not in L since $\beta > 0$.
- We have obtained a contradiction, so L must not be regular.

$\square$

5. Consider the following CFG $G = (\{A, B, C\}, \{m, i, a\}, R, B)$, with R described as:

$$\begin{aligned} &A \to m \mid B \\ &B \to m \mid AiC \mid \varepsilon \\ &C \to iBa \end{aligned}$$

(a) (1 point) Give a word $w \in L(G)$, such that $3 \leq |w| \leq 5$ and such that w contains the letter m exactly twice.

Solution: For example: $B \Rightarrow AiC \Rightarrow miC \Rightarrow miiBa \Rightarrow miima$

(b) (5 points) Convert G to Chomsky normal form using the method from Sipser. Show all intermediate steps.

Solution: Add a new start state S:

$$\begin{aligned} &S \to B \\ &A \to m \mid B \\ &B \to m \mid AiC \mid \varepsilon \\ &C \to iBa \end{aligned}$$

Remove ε-rules (first from B):

$$\begin{aligned} &S \to B \mid \varepsilon \\ &A \to m \mid B \mid \varepsilon \\ &B \to m \mid AiC \\ &C \to iBa \mid ia \end{aligned}$$

Now from A:

$$\begin{aligned} &S \to B \mid \varepsilon \\ &A \to m \mid B \\ &B \to m \mid AiC \mid iC \\ &C \to iBa \mid ia \end{aligned}$$

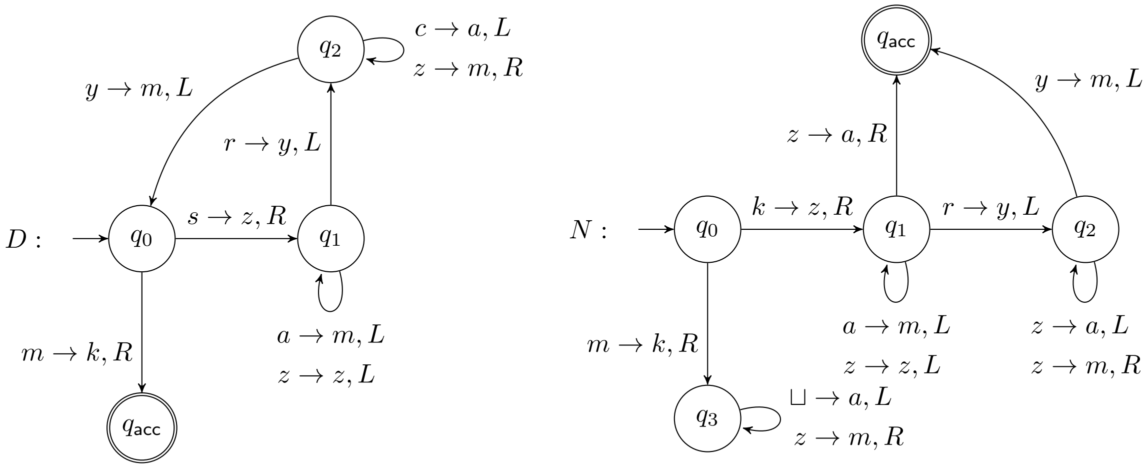
Remove unit rules:

$$\begin{aligned} &S \to m \mid AiC \mid iC \mid \varepsilon \\ &A \to m \mid AiC \mid iC \\ &B \to m \mid AiC \mid iC \\ &C \to iBa \mid ia \end{aligned}$$

Fix remaining rules

$$\begin{aligned} S &\to m \mid AU_{\mathsf{ic}} \mid U_{\mathsf{i}}C \mid \varepsilon \\ A &\to m \mid AU_{\mathsf{ic}} \mid U_{\mathsf{i}}C \\ B &\to m \mid AU_{\mathsf{ic}} \mid U_{\mathsf{i}}C \\ C &\to U_{\mathsf{ib}}U_{\mathsf{a}} \mid U_{\mathsf{i}}U_{\mathsf{a}} \\ U_{\mathsf{ic}} &\to U_{\mathsf{i}}C \\ U_{\mathsf{ib}} &\to U_{\mathsf{i}}B \\ U_{\mathsf{i}} &\to i \end{aligned}$$

6. Consider the deterministic TM D on the left (you should assume all missing transitions lead to a reject state) and the non-deterministic TM N on the right. For both machines $\Sigma = \{k, r, a, n, s\}$ and q_{acc} is the accepting state. You should also assume the tape is bound on the left side for both machines.



(a) (2 points) What is $L(D)$? Explain your answer in at most 5 lines.

Solution: All words that start with sr are accepted.. The word cannot start with m as $m \notin \Sigma$, so it must start with s, from here the only path that gets us to q_{acc} is for the word sr. Thus $L(D) = \{srx \mid x \in \Sigma^*\}$.

(b) (2 points) Give a word of length 3 that is in the language $L(N)$ but not in the language $L(D)$. Give a series of configurations that shows $w \in L(N)$.

Solution: For example: kan (though ka followed by any symbol from Σ will work):

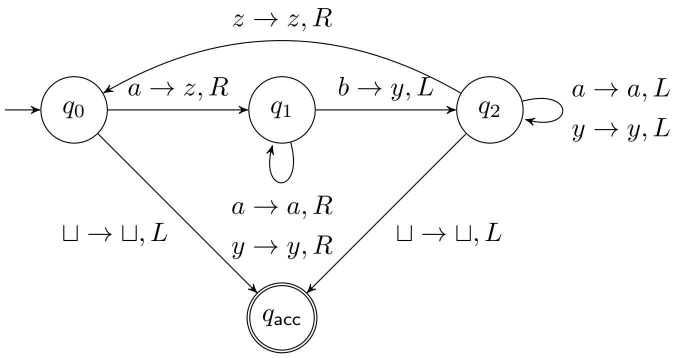
- $q_0 kan\sqcup$
- $zq_1 an\sqcup$
- $q_1 zmn\sqcup$
- $aq_{\text{acc}} mn\sqcup$

(c) (2 points) Add at most 2 transition (labels) to N to create N', so that $L(D) \subseteq L(N')$. Only give the added labels, and the states between which they should be added, for your answer (not a full diagram).

> **Solution:** Add the transition label $s \to s, R$ and $r \to r, R$ between q_0 and q_1, and q_1 and q_{acc} respectively.

7. (4 points) Construct an NTM of at most 5 states that decides the language: $L = \{w \mid w = a^n b^n \text{ such that } n \geq 0\}$. A transition diagram suffices as an answer.

Solution:



8. For each of the following claims, either explain why they are true, or give a counterexample if they are false. Start your answer with either the word "True" or "False" indicating which of the two options applies.

(a) (2 points) Enumerators only exist for finite languages.

> **Solution:** False. Consider the enumerator for the infinite set $\mathbb{N}$: 1. Write 0 on the tape. 2. Add 1 to the previous number and write the result on the tape. 3. Go to 2.

(b) (2 points) If a problem is undecidable, it is also *not* co-Turing-recognisable.

> **Solution:** False, consider for example EQ_{CFG} which is undecidable, but is co-Turing-recognisable. You can enumerate all words and if you find one that is not recognised by both grammars, it is not in EQ_{CFG}.

9. (4 points) Consider the following *faulty* proof to show that $HALT_{\mathsf{TM}}$ is undecidable.

   Proof. Assume for the sake of contradiction that TM R decides $HALT_{\mathsf{TM}}$. We construct a TM S to decide A_{TM}.
   $S =$ "On input $\langle M, w\rangle$, an encoding of a TM M and a string w:

   1. Run TM R on input $\langle M, w\rangle$.
   2. Simulate M on w until it halts.
   3. If M has accepted, *accept*; if M has rejected, *reject*."

   Clearly, if R decides $HALT_{\mathsf{TM}}$, then S decides A_{TM}. Because A_{TM} is undecidable, $HALT_{\mathsf{TM}}$ must also be undecidable. □

   This tentative proof contains a flaw, which causes it to be invalid. In what step(s) does this flaw occur and what is the flaw (for 2 points), and what should this/these step(s) in the proof look like instead (for 2 points)?

   Solution: Proof can be found on page 217 of the book.

10. Suppose we have the following language

$$L = \{\langle M, D\rangle \mid \text{When we run } M \text{ on } hi\text{, it outputs } D \text{ on the tape, which is a DFA that accepts the word } hello.\}$$

   We intend to prove that this language is undecidable. To that end we use a mapping reduction from $HALT_{TM}$ to L. The reduction f is computed by the following TM F:

   $F =$"On input $\langle M, w\rangle$:

   1. Construct the following TM M':
      $M' =$"On input x:
      1. Clear the input from the tape.
      2. Run M on w.
      3. See question a.
      4. See question a.
   2. Write $\langle M', D\rangle$ to the output tape, where D is a DFA with 1 state which is accepting and transitions to itself for all input letters."

   (a) (4 points) What should be done in steps 3 and 4 of M'?

   Solution:

   3. Create a DFA D with 1 state which is accepting and transitions to itself with all input letters.

   4. Write D to the tape and accept."

(b) (5 points) Provide a proof showing f satisfies the requirements of a mapping reduction.

Solution:

Proof. We need to show that f is computable and reliable. F computes f, so f is computable. f is also reliable, because:

- $\langle M, w\rangle \in HALT_{\mathsf{TM}} \Rightarrow f(\langle M, w\rangle) \in L$ holds, since if M halts on w, M' leaves D on the tape on input *hi* (it always does this regardless of input) and this is indeed a DFA that accepts the word $hello$. So, $\langle M', D\rangle \in L$.
- $\langle M, w\rangle \notin HALT_{\mathsf{TM}} \Rightarrow f(\langle M, w\rangle) \notin L$ holds, since if M does not halt on w, M' will never write the correct output to the tape.

$\square$

End of the exam