NATIONAL UNIVERSITY OF SINGAPORE

CS 5236 – Advanced Automata Theory

(Semester 1: AY 2018/2019)

Time Allowed: 2 Hours

INSTRUCTIONS TO CANDIDATES

1. Please write your Student Number. Do not write your name.
2. This assessment paper consists of TEN (10) questions and comprises TWENTYONE (21) printed pages.
3. Students are required to answer **ALL** questions.
4. Students should answer the questions in the space provided.
5. This is a **CLOSED BOOK** assessment.
6. It is permitted to use calculators, provided that all memory and programs are erased prior to the assessment; no other material or devices are permitted.
7. Every question is worth SIX (6) marks. The maximum possible marks are 60.

STUDENT NO: ______________________________

This portion is for examiner's use only

Question	Marks	Remarks	Question	Marks	Remarks
Q01:			Q06:		
Q02:			Q07:		
Q03:			Q08:		
Q04:			Q09:		
Q05:			Q10:		
			Total:		

Construct a **context-sensitive grammar** for the language

$$L = \{10^n 10^{n-1} \ldots 10^3 10010112 : n \in \mathbb{N}\}$$

and give a **sample derivation** of 10010112.

Note that $L = \{x_0, x_1, x_2, \ldots\}$ with $x_0 = 112$, $x_1 = 10112$ and $x_{n+1} = 10^{n+1} x_n$ for all n.

When making the grammar, a rule $l \to r$ is allowed whenever $1 \leq |l| \leq |r|$ and l contains at least one nonterminal.

Solution. The non-terminals are S, T, U, the terminals are $0, 1, 2$ and the start symbol is S. The rules are the following: $S \to T12$, $T \to 1|TU$, $U0 \to 0U$, $U1 \to 01U$, $U2 \to 12$.

A sample derivation of 10010112 is $S \Rightarrow T12 \Rightarrow TU12 \Rightarrow T01U2 \Rightarrow T0112 \Rightarrow TU0112 \Rightarrow T0U112 \Rightarrow T001U12 \Rightarrow T00101U2 \Rightarrow T0010112 \Rightarrow 10010112$.

Consider the set $L = \{v3w : v, w \in \{0,1,2\}^*$: for some $k, m, n \in \mathbb{N}$, v, w have, both, k zeroes, m ones and n twos$\}$.

Which of the following statements is true for L?

☐ L is regular, ☐ L is linear and not regular,

☐ L is context-free and not linear,

☐ L is context-sensitive and not context-free,

☐ L is recursively enumerable and not context-sensitive.

Give a justification for your choice, by providing the corresponding grammars or automata to show that the language is in that level and by giving evidence that the language does not go onto a better level (if applicable).

Solution. The language L is context-sensitive. The grammar for the language is $(\{S, U, V, W\}, \{0, 1, 2, 3\}, S, P)$ where P contains the following rules:

$$S \to USU|VSV|WSW|3,$$
$$UV \to VU,\ UW \to WU,$$
$$VU \to UV,\ VW \to WV,$$
$$WU \to UW,\ WV \to VW,$$
$$U \to 0,\ V \to 1,\ W \to 2.$$

The grammar produces before and after the S the same amount of U, the same amount of V and the same amount of W which can be reordered in any way and then transformed to 0, 1 and 2, respectively. Furthermore, S will eventually become 3 and there are no rules which allow $U, V, W, 0, 1, 2$ to cross the $S/3$ in the centre.

Assume now that L would be context-free. Then the intersection H of L with the regular set $\{0\}^* \cdot \{1\}^* \cdot \{2\}^* \cdot \{3\} \cdot \{012\}^*$ is also context-free. However, H is the language of all $0^n1^n2^n3(012)^n$, as for $a = 0, 1, 2$ there have to be as many a before as after the 3 and the symbols $0, 1, 2$ occur after the 3 the same number of times. Now assume by contradiction that the language H satisfies the context-free pumping lemma. Let n be larger than the pumping constant. Now pumping up once changes in most two of the four blocks of n items the number of repetitions; this, however, creates a word outside H and the context-free pumping lemma is not satisfied by H. Therefore neither H nor L are context-free.

Let $\Sigma = \{1,2,3,4,5,6,7,8,9\}$ be the set of nonzero decimal digits and

$$L = \{u \in \Sigma^* : u \text{ contains at least two different digits}\}.$$

Construct a **dfa of 11 states** and an **nfa of 8 states** for L and **explain** the basic idea behind the constructions.

Solution. The dfa has states s, t_a, u for all $a \in \Sigma$. From s, it goes on a to t_a and it stays in t_a as long as it sees an a and goes on all other symbols to u. From u it goes on all symbols to itself. s is the start state and u the accepting state. As there are 9 symbols, the overall number of states is 11.

The basic idea is that the dfa archives the first symbol and, as soon as a different symbol comes, it changes into the accepting state. As the derivatives $L_\varepsilon, L_1, \ldots, L_9, L_{12}$ are all different, one cannot make a smaller dfa, so the dfa is minimal.

For the nfa, one considers the following sets $A_1, \ldots, A_6$: $\{1,2,3\}$, $\{4,5,6\}$, $\{7,8,9\}$, $\{1,4,7\}$, $\{2,5,8\}$, $\{3,6,9\}$. Now the nfa has the states $\{s, t_1, t_2, t_3, t_4, t_5, t_6, u\}$ and s is the start state and u the only accepting state. From s one can on a go to t_k iff $a \in A_k$. If $a \in A_k$ then one can go from t_k to itself else one can go from t_k to u. From u, one goes on all symbols to u.

The idea to reduce the states is that for each symbol a there are exactly two sets A_k, A_ℓ which contain a and they satisfy $A_k \cap A_\ell = \{a\}$. On a word from $\{a\}^+ \cdot \{b\} \cdot \Sigma^*$ with $a \neq b$, the nfa can in s on symbol a select a state t_k with $a \in A_k$ and $b \notin A_k$; it remains in t_k until b shows up and then transits to u. To see the above property, note that A_1, A_2, A_3 as well as A_4, A_5, A_6 are partitions of Σ into three sets each such the first goes by the order (first three, second three and third three consecutive numbers) and the second partition goes by the remainder modulo 3 which never coincides for two numbers within a group of three consecutive numbers.

Let $F = \{0, 1, 2, \ldots, 14\}$ be a type of soccer field. Anke can kick the ball from a position p either to $p - 3$ or to $p + 3$ (modulo 15); Boris can kick it from a position p to either $p + 5$ or $p - 5$ (modulo 15). At the beginning, the ball is on 0 and Anke moves first; the players move alternately. Anke scores a goal when the ball reaches 2 and Boris scores a goal when the ball reaches 4. The player who scores two goals wins; if the game runs an infinite time without any player scoring two goals, it is a draw. Decide whether Anke has a winning strategy or Boris has a winning strategy or both players have a draw strategy. Furthermore, can the strategy / these strategies be made memoryless?

Solution. Both players have a draw strategy. Anke controls by kicking the remainder of the position p modulo 5 and Boris controls by kicking the remainder of the position p modulo 3. Anke makes sure that the remainder is never 4 modulo 5 so that the ball never reaches 4 and therefore Boris never scores a goal; Boris makes sure that the remainder is never 2 modulo 3 so that the ball never reaches 2 and therefore Anke never scores a goal. If both play like this, the game will remain a draw forever. Note that $p - 3$ and $p + 3$ have different remainders modulo 5 and $p - 5$ and $p + 5$ have different remainders modulo 3, so both players can always select a field with the wrong remainder and when the oppositing player plays, that player cannot change the corresponding remainder. Note that the move only depends on the value p modulo 5 / modulo 3 for Anke and Boris, respectively, thus the above draw strategies are memoryless.

In a Büchi game, both players move alternately with player Anke starting at start node s. Player Anke wins an infinite play iff the play goes infinitely often through some nodes in the set W of accepting nodes. The players move alternately with Anke moving first.

Consider the following Büchi game between Anke and Boris: Let $V = \{a_0a_1a_2a_3a_4 : a_0, a_1, a_2, a_3, a_4 \in \{0, 1, 2, \ldots, 9\}\}$ be the set of five-digit decimal numbers with leading zeroes allowed. The node 00000 is the start node and every player moves from $a_0a_1a_2a_3a_4$ to $a_1a_2a_3a_4a_5$ for a new decimal digit a_5 selected by the player. Let $W = \{a_0a_1a_2a_3a_4$: the decimal value of $a_0a_1a_2a_3a_4$ is a multiple of 7$\}$.

Which player has a **winning strategy** for this Büchi game, Anke or Boris? Provide the winning strategy and make it memoryless whenever possible.

Solution. Anke has a winning strategy for this game. If the game is in the node $a_0a_1a_2a_3a_4$, she computes the remainder b of the division of $a_1a_2a_3a_40$ (when viewed as a natural number) by 7 and then she chooses $a_5 = 7 - b$ so that $a_1a_2a_3a_4a_5$ is a multiple of 7. Thus she moves to a position in W. It follows that the game goes through positions in W infinitely often and Anke wins the game with a memoryless winning-strategy.

Let $\Sigma = \{0, 1, 2, 3\}$ and consider the ω-language of all ω-words w which contain infinitely many substrings of the form ddd for some $d \in \Sigma$. Construct a deterministic Büchi automaton with at most 9 states which recognises this ω-language.

Solution. A possible automaton is the following, where s is the start state and the only accepting state and the other states are named $t_0, t_1, t_2, t_3, u_0, u_1, u_2, u_3$. The successors for symbols 0,1,2,3 are given in the following table.

State	On 0	On 1	On 2	On 3
s	t_0	t_1	t_2	t_3
t_0	u_0	t_1	t_2	t_3
u_0	s	t_1	t_2	t_3
t_1	t_0	u_1	t_2	t_3
u_1	t_0	s	t_2	t_3
t_2	t_0	t_1	u_2	t_3
u_2	t_0	t_1	s	t_3
t_3	t_0	t_1	t_2	u_3
u_3	t_0	t_1	t_2	s

One can see that, except for the initial value, the Büchi automaton goes only to s when it has seen a sequence ddd of digits. If a digit d differs from the previous one, the automaton goes to t_d. If it is in t_d on seeing d, it goes to u_d, if it is in u_d on seeing d, it goes to s, as it has seen the sequence ddd. Note that it is sufficient when the Büchi automaton goes into s only on every third of the sequences ddd, as it must do this only infinitely often.

Consider the following finite monoid: $G = \{0, 1, 2, 3, 4\}$, $x \circ y$ is the minimum of $x + y$ and 4. Is there a regular language L such that $(G, \circ)$ is the syntactic monoid of this language? ☐ Yes, ☐ No.

If so, then provide a sample language L with $(G, \circ)$ being the isomorphic to the syntactic monoid of this language; if not, then prove that such a language does not exist.

Recall that given a language L, one constructs the syntactic monoid as follows: One computes the minimal deterministic and complete finite automaton $(Q, \Sigma, s, \delta, F)$ and then considers all mappings f_u from states to states where $f_u(q) = \delta(q, u)$. For these, one defines $f_v = f_w$ iff $f_v(q) = f_w(q)$ for all $q \in Q$ and $f_v * f_w = f_{vw}$, so that $*$ is induced by the word concatenation. Now the question is whether there is a regular language L such that $(\{f_u : u \in \Sigma^*\}, *)$ is isomorphic to $(G, \circ)$.

Solution. One considers the following language: L contains all words of length 4 and more. The minimal automaton has states $0, 1, 2, 3, 4$ representing the length of the word read so far; only state 4 is accepting. Now $f_v \circ f_w = f_{vw}$ and $f_v(q) = \min\{q + |v|, 4\}$ for every $q \in \{0, 1, 2, 3, 4\}$. Thus $f_v = f_w$ whenever $|v| = |w|$ or $|v| \geq 4 \wedge |w| \geq 4$. Furthermore, $(f_v * f_w)(q) = \min\{q + |v| + |w|, 4\} = f_w(f_v(q)) = f_{vw}(q)$. Let a denote a fixed member of the alphabet. Now the functions in the syntactic monoid are given by the following table:

$f \mid q$	0	1	2	3	4
f_ε	0	1	2	3	4
f_a	1	2	3	4	4
f_{aa}	2	3	4	4	4
f_{aaa}	3	4	4	4	4
f_{aaaa}	4	4	4	4	4

One can see that $f_{a^i} * f_{a^j} = f_{a^k}$ for $k = \min\{i + j, 4\}$. So $(G, \circ)$ is isomorphic to the syntactic monoid $(\{f_{a^0}, f_{a^1}, f_{a^2}, f_{a^3}, f_{a^4}\}, *)$ of L.

Assume an automatic representation of $(\mathbb{N}^3, +)$ is given. Furthermore, let $\leq$ denote the coordinate-wise comparison, so $(a, b, c) \leq (d, e, f)$ iff $a \leq d$ and $b \leq e$ and $c \leq f$. Is $\leq$ automatic?

☐ Always Yes, ☐ Always No, ☐ Depends on the representation.

Prove the answer; if it is the third choice, provide two automatic representations, one where $\leq$ is automatic and one where $\leq$ is not automatic.

Solution. One can first-order define the ordering by

$$(a, b, c) \leq (d, e, f) \Leftrightarrow \exists (h, i, j)\,[(a, b, c) + (h, i, j) = (d, e, f)]$$

where the $+$ is, as defined, taken component-wise. Note that the structure already provides the $+$ in this form. This is just mapping the result for triples back to the corresponding property for $\mathbb{N}$ that $a \leq d \Leftrightarrow \exists h \in \mathbb{N}\,[a + h = d]$ and similarly for the other two coordinates. As the addition $+$ is automatic and the domain of the structure is regular, the relation $\leq$ is first-order defined from automatic parameters and, by a theorem of Khoussainov and Nerode, the so defined relation is automatic.

A Moore machine is a finite automaton where each state contains a word which is output whenever the automaton visits this state; the final output is the concatenation of all these output words, provided that the Moore machine is in an accepting state after processing the whole input. For a non-deterministic Moore machine with input v, all runs leading to an accepting state must provide the same output w.

Which of the following functions can be computed by a Moore machine on a binary alphabet $\{0, 1\}$:

(a) $a_1 a_2 \ldots a_n \mapsto a_1 a_1 a_2 a_2 \ldots a_n a_n$;

(b) $a_1 a_2 \ldots a_n \mapsto a_1 a_2 \ldots a_n a_1 a_2 \ldots a_n$?

If the function can be computed by a Moore machine then **provide this Moore machine** else **give reasons why it does not exist**. Note that if f is computed by a Moore machine and L is regular, so is the range $\{f(x) : x \in L\}$.

Solution. (a) The Moore machine outputs ε on the start state. Furthermore, for each digit $d \in \{0, 1\}$, it has a state called t_d and it outputs dd in this state. Upon reading a digit d it goes to the state t_d wherever it is. All states are accepting.

State	Output	Type	On 0	On 1
s	ε	accepting	t_0	t_1
t_0	00	accepting	t_0	t_1
t_1	11	accepting	t_0	t_1

(b) Moore machines map regular languages to regular languages. Thus they map $\{0\}^* \cdot \{1\}$ to a regular language. However, the second mapping maps this language to $\{0^n 1 0^n 1 : n \in \mathbb{N}\}$ which is not regular. Therefore the second function cannot be computed by a Moore machine.

Consider the following class of sets: $A_{c,3^d} = \{c + 3^d \cdot e : e \in \mathbb{N}\}$. Find a representation for the natural numbers where addition and order are fully automatic and where the set of all $A_{c,3^d}$ forms an automatic family. Provide also a definition of the index set E of this automatic family and prove that the relation $\{(e, x) : x \in A_e\}$ is automatic. Here note that $A_{0,3^0} = \mathbb{N}$.

Is the so obtained automatic family automatically learnable from positive data?

☐ Yes, ☐ No.

If so, provide an automatic learner for it, if not, explain why it is not learnable.

Solution. One represents the natural numbers by ternary numbers, that is the domain is $\{0,1,2\}^* \cdot \{1,2\} \cup \{0\}$. Here $a_0 a_1 \ldots a_n$ represents $\sum_{m=0}^{n} a_m \cdot 3^m$. Addition on these numbers is automatic and also the order of the natural numbers. Furthermore, for two ternary numbers x, y in this representation with $x < y$, one defines $A_{x,y} = \{z : x \leq z$ and for every u with $u \preceq y$ and $u \preceq y$ it also holds that $u \preceq z\}$. Note that the relation $\preceq$ of being a prefix is automatic and therefore also the family of all $A_{x,y}$ is an automatic family. The index set is $E = \{conv(x, y) : x < y$ and x, y are ternary representations for natural numbers as defined above$\}$; so, formally, $A_{x,y}$ should be written as $A_{conv(x,y)}$.

Here x, y can be chosen as the least members of the set to be learnt. Therefore one can make a learner which does the following: While it has not seen any data, its memory is $conv(\infty, \infty)$ where ∞ is a special symbol not in the alphabet and $x < \infty$ for all $x \in \mathbb{N}$. The learner's memory is initially $conv(\infty, \infty)$ and at each point, its memory $conv(x, y)$ satisfies either $x = y = \infty$ or $x < y = \infty$ or $x < y < \infty$; if it now sees a datum z, the learner does the following: If $z < x$ then the memory is updated to $conv(z, x)$ else if $x < z < y$ then the memory is updated to $conv(x, z)$ else the memory remains unchanged. If both x, y are different from ∞ then the learner conjectures $conv(x, y)$ else the learner outputs some default hypothesis like $conv(0, 1)$.

END OF PAPER