NATIONAL UNIVERSITY OF SINGAPORE

CS 5236 – Advanced Automata Theory

(Semester 1: AY 2020/2021)

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 in the space provided or handwritten one page each question.
4. This is an **OPEN BOOK** assessment.
5. During the exam, you are not permitted to communicate with other people over the internet or to post anything in forums or social networks.
6. Every question is worth SIX (6) marks. The maximum possible marks are 60.
7. Either handwrite and scan or type into pdf-file. The uploaded file should be student-number followed by module code and "Finalexam". For example A01234567Z-CS5236-Finalexam.pdf.

STUDENT NO: _____

Question 1 [6 marks] **CS 5236 – Solutions**

Construct a **context-free grammar** for the language L of all words $w \in \{0,1\}^*$ such that, if the length of the word is n, the following holds: (a) $n \geq 4$, (b) w contains a subword of length $n-2$ which is a palindrome, (c) n is odd. Furthermore, the grammar should not have any rule of the form $A \to \varepsilon$, so each left side is one single non-terminal and each right side is a string of symbols of length at least one.

For example, 00111 is a member of L and the palindrome subword is 111; 011110 is not in L as the length is even; 010 is not in L as 010 is too short; 0101010 is a member of L and possible palindrome subwords of length $n-2$ are 01010 and 10101.

Furthermore, after making the grammar, supply sample derivations for the above words 00111 and 010101010 in L.

Solution. The non-terminals are S, T, U, the terminals are $0, 1$ and the start symbol is S. The rules are the following: $S \to TTU \,|\, TUT \,|\, UTT$, $T \to 0 \,|\, 1$, $U \to 0U0 \,|\, 1U1 \,|\, 0T0 \,|\, 1T1$.

The derivation of 00111 is $S \Rightarrow TTU \Rightarrow TT1T1 \Rightarrow 0T1T1 \Rightarrow 001T1 \Rightarrow 00111$.

The derivation of 0101010 is $S \Rightarrow TUT \Rightarrow 0UT \Rightarrow 0U0 \Rightarrow 01U10 \Rightarrow 010T010 \Rightarrow 0101010$.

Question 2 [6 marks] **CS 5236 – Solutions**

Consider a grammar for a language $L \subseteq \{0,1,2\}^*$ with start symbol S and the following rules:

$$S \to SXYZ \mid XYZ, XY \to YX, XZ \to ZX, YZ \to ZY, YX \to XY, ZX \to XZ, ZY \to YZ, X \to 0, Y \to 1, Z \to 2.$$

Determine the complexity of the language L in terms of the Chomsky hierarchy:

☐ (a) context-senstive but not context-free,

☐ (b) context-free but not regular,

☐ (c) regular.

If the answer is (a) or (b), use pumping lemmas to show that it does not go better; if the answer is (b) or (c), provide the corresponding grammars to show that the language goes onto this level.

Solution. The grammar generates the same amount of X, Y, Z and allows to reorder them arbitrarily; afterwards X becomes 0, Y becomes 1 and Z becomes 2. Thus the language is that of all words with the same amount of 0, 1 and 2, where the empty word is not in the language. The grammar is clearly context-sensitive. Furthermore it is well-known that this language is not context-free. The required proof can be done with the pumping lemma for context-free languages. Let k be the pumping constant and consider the word $0^k1^k2^k$. This word can now, by assumption on the lemma, be split into $vwxyz$ so that $|wxy| \leq k$ and $xy \neq \varepsilon$ and $\forall h\,[vw^hxy^hz \in L]$. Now wxy can only contain at most two of the three symbols $0, 1, 2$. Thus the word $vwwxyyz$ has one or two of the symbols $0, 1, 2$ in their quantity strictly increased, as $wy \neq \varepsilon$, but the quantity of the third symbol did not increase, so $vwwxyyz \notin L$. Thus the language L does not satisfy the context-free pumping lemma and does therefore not have a context-free grammar.

Question 3 [6 marks] **CS 5236 – Solutions**

Let $\Sigma = \{0, 1, 2, 3\}$ and construct a dfa which accepts whenever at least three digits occur in the input and which rejects whenever at most one digit occurs in the input. It does not matter what happens if exactly two digits occur in the input and it does not matter how often the digits occur (whether once or several times). Use at most four states and prove that it cannot be done with two states.

Solution. First that it does not go in two states. Note that the start state cannot be accepting, as the empty word has to be rejected. Furthermore, any state reacheable from the start state in one step has to be rejecting, as otherwise a word consisting of a single digit could be accepted. As there must be at least one accepting state, there must be at least three states.

The following four-state dfa solves the given task; it has the states start, odd, even and end. The transitions are as in this table.

State	Type	On 0 and 2	On 1 and 3
start	reject	even	odd
even	reject	even	end
odd	reject	end	odd
end	accept	end	end

Words which contain no digits are rejects, which contain only one type of digits have either all digits odd or all digits even and the automaton will be in one of the states odd and even after processing the full word. On words which contain at least three types of digits, the dfa will see both even and odd digits and therefore first go to one of odd, even and later go on to end. So these words will be accepted. In summary the dfa accepts a word iff it contains both, odd and even digits.

Consider the following survival game: The game is played on the field $\{0, 1, 2, \ldots, r-2, r-1\}$ and the players move alternately, but do differet types of moves which are modulo r: Anke can add 1 or 2 or 3 to the current position, Boris can multiply with one of $2, 3, 5, 7$. Determine for $r = 4, 6, 7, 9, 17, 30$ which player has a winning strategy and give a short outline of this winning strategy. Whenever the game reaches 0 then Boris wins; whenever the game runs forever without ever reaching 0 then Anke wins. The start position is 2 in all games and Anke is first to move.

Solution. For $r = 4$, Anke has a winning strategy, she moves in each move to an odd number. Boris can then either multiply with $3, 5, 7$ and stay on an odd number or multiply with 2 which returns to the start position. Note that from every number one can reach an odd number by adding one of $1, 2, 3$.

For $r = 6$, Boris has a winning strategy. After Anke's move, if the game is on $2, 4$ he multiplies with 3 and reaches 0 (modulo 6); if the game is on 3 Boris multiplies with 2 and again reaches 0; if the game is in 5 Boris multiplies with 5 and reaches 1; if the game is in 1 Boris multiplies with 7 ad stays on 1. Now the game is either in 0 (Boris has already won) or 1. From 1, Anke can only move to $2, 3, 4$ which all have a winning counter move of Boris. Hence Boris wins.

For $r = 7$, Boris has a trivial win by multiplying with 7 what gives 0, independently of the current situation. The same would apply for $r = 2, 3, 5$ which are not part of the question.

For $r = 9$, Anke has a winning strategy is very similar to that of 4. Anke always moves to a number which is not a multiple of 3 and then either Boris multiplies with 3 ending up in 3 or 6 or multiplies with one of the other primes which preserves that the number is not a multiple of 3. Thus Anke can avoid the game going to 0 all the time and wins.

For $r = 17$, Anke has a winning strategy. Boris cannot win by own moves, as all moves go from nonzero positions to nonzero positions. As Anke can from the current position p always to move to one of $p+1, p+2, p+3$ (modulo r), at least two of these positions are nonzero and so she can avoid that the game becomes 0.

For $r = 30$, Anke wins. 30 is the product of three distinct prime factors $(2, 3, 5)$. When before Anke's move the game is in nonzero position p, then she can select among $p + 1, p + 2$ (modulo 30) that position which has at most one of the three primes $2, 3, 5$ dividing it; this position always exist as none of these primes divides two neighbouring numbers. Thus Anke moves always to a position where at most one of the primes $2, 3, 5$ divides this position and so Boris cannot make the number to 0 as that requires that all three primes divide the number and the primefactor selected by Boris can increase the number of prime factors among $2, 3, 5$ only by one.

Question 5 [6 marks] **CS 5236 – Solutions**

Construct a deterministic Büchi automaton recognising the ω-language of all words $\alpha \in \{0,1,2\}^\omega$ with infinitely many subwords of the form 012 and no subword of the form 02. The Büchi automaton should be complete (never get stuck) and have at most six states.

Solution. The Büchi automaton has the states s, t, u, v, w and the following transition table:

State	Type	at 0	at 1	at 2
s	start, rej	u	s	s
t	rej	t	t	t
u	rej	u	v	t
v	rej	u	s	w
w	acc	u	s	s

Question 6 [6 marks] **CS 5236 – Solutions**

Assume that $(Q, \Sigma, \delta, s, F)$ is a nondeterministic Büchi automaton for some ω-language $L \subseteq \Sigma^\omega$ and that for every $q \in Q$ and $a \in \Sigma$ there is an $r \in Q$ with $(q, a, r) \in \delta$. Now consider the following Büchi game on the graph $Q \cup (Q \times \Sigma)$ where Q is considered to be disjoint from $Q \times \Sigma$. Furthermore let $E = \{(q, (q, a)) : q \in Q \wedge a \in \Sigma\} \cup \{((q, a), r) : (q, a, r) \in \delta\}$. The accepting states of the game are nodes in F. Now Anke wins a play iff the play goes infinitely often through a state in F. Furthemore, in contrast to the usual setting, Boris starts to move from the start state s, not Anke, so that Boris always moves from nodes q to (q, a) and Anke from nodes (q, a) to nodes r.

Assume that Σ is one of $\{0\}$, $\{0, 1\}$, $\{0, 1, 2\}$. Decide in dependence on the alphabet Σ and the ω-languages $L \subseteq \Sigma^\omega$, which of the following options applies:

(a) Anke has a winning strategy for the game based on any given Büchi automaton for L;

(b) Boris has a winning strategy for the game based on any given Büchi automaton for L;

(c) It depends on the actual Büchi automaton for L whether Anke or whether Boris has a winning strategy.

Solution. Note that Boris always moves from a state q of the Büchi automaton to (q, a) where a is the next symbol of the word to be processed and that Anke always moves from (q, a) to a state r with $(q, a, r) \in \delta$; by assumption such a state r always exisits.

If L is a proper subset of Σ^ω then there is an ω-word α which cannot be accepted and Boris has a winning strategy by feeding this word, that is, by going to (q, α_k) for the k-th symbol α_k in α when it is Boris' k-th time to move. Thus the game cannot go through an accepting state infinitely often, as that play of the game can then be translated into an accepting run of the original automaton on α which does not exist.

If $L = \{0\}^\omega$ and $\Sigma = \{0\}$ then Boris' moves have no effect (as he always has to choose to go from some q to $(q, 0)$) and Anke can move such that the play defines an accepting run on the ω-word.

If $\Sigma = \{0, 1\}$ or $\Sigma = \{0, 1, 2\}$ and $L = \Sigma^\omega$ then it depends on the automaton whether Anke or whether Boris has a winning strategy; the two cases are similar, so assume that $\Sigma = \{0, 1, 2\}$.

If the Büchi automaton consists only of one state (which has to be accepting) then Anke trivially wins, as the game goes through this state infinitely often.

Now assume that Büchi automaton consists of four states s, x, y, z with the following transition table:

State	Type	On 0	On 1	On 2
s	start, reject	s, x	s, y	s, y
x	reject	z	x	x
y	reject	y	z	z
z	accept	z	z	z

If 0 occurs twice then the Büchi automaton can go on the first occurrence from s to x and on the second occurrence from x to z and stay in the corresponding states in all other cases; if symbols from $1, 2$ occur twice then the Büchi automaton can go on the first occurrence from s to y and on the second occurrence from y to z. The automaton waits in s until the opportunity comes to follow a symbol which appears again. Thus Σ^ω is the ω-language recognised by this Büchi automaton is Σ^ω.

However, Boris wins the corresponding game. If the game is in s, he goes to $(s, 0)$ and Anke can choose between returning to s and going to x. If the game is in x, Boris goes to $(x, 1)$ so that Anke needs to return to x. Thus the game will always go through s or x and never through y or z. So no accepting state is visited ever and Boris wins the Büchi game.

The difference between nondeterministic acceptance by the automaton and winning of the game is due to the following: The automaton decides its nondeterministic moves based on the full ω-word, thus this ω-word has to be fixed in advance. The game however has that the player Boris who puts the digits of the word can react to the state in which Anke is, thus the game alternates between Anke to move and Boris to fix the next symbol. This is exploited to make the automaton remain eventually forever in either s or x.

Question 7 [6 marks] **CS 5236 – Solutions**

Determine a minimal dfa and the syntactic monoid of the language $L = \{0,1,2\}^* \cdot \{00,11,22\} \cdot \{0,1,2\}^*$ with the alphabet being $\{0,1,2\}$.

Solution. The dfa has five states: s, q_0, q_1, q_2, t. From s and q_b on input a with $a \neq b$, the automaton goes to q_a. From q_a, t on input a, the automaton goes to t. The start state is s and the state t is accepting.

The syntactic monoid has eleven functions. $f_\varepsilon, f_0, f_1, f_2, f_{00}, f_{01}, f_{02}, f_{10}, f_{12}, f_{20}, f_{21}$. f_ε is the identity function. f_a with $a \in \{0,1,2\}$ maps s, q_b with $b \neq a$ to q_a and q_a, t to t. If $a \neq b$ and there is no double character in awb then $f_{awb} = f_{ab}$ and maps a, t to t and all other states to q_b. If awa does not contain a double character then f_{awa} is equal to f_a and maps a, t to t and all other states to q_a. If awb contains a double character then $f_{awb} = f_{00}$ and maps all states to t; in particular, $f_{00} = f_{11} = f_{22}$.

Question 8 [6 marks] **CS 5236 – Solutions**

Assume that an automatic structure $(A, <)$ is given where A is the set of natural numbers and $<$ is the usual order on the natural numbers. Let S be the set of sums $0+1+\ldots+n$ with $n \in \mathbb{N}$, so $S = \{0, 1, 3, 6, 10, 15, \ldots\}$. Furthermore, $<$ is the usual order on $\mathbb{N}$. Now state the following:

(a) Is the successor-function f from x to $x+1$ automatic in $(A, <)$?

☐ Always Yes, ☐ Always No, ☐ Depends on the representation $(A, <)$.

(b) Is the set S automatic in $(A, <)$?

☐ Always Yes, ☐ Always No, ☐ Depends on the representation $(A, <)$.

Prove the answer for both questions; if it is the third choice, provide two automatic representations, one where f respectively S is automatic and one where f respectively S is not automatic.

Solution. The successor-function is always automatic, as one can first-order define it from the order:

$$f(x) = y \Leftrightarrow x < y \wedge \forall z\,[x < z \Rightarrow y = z \vee y < z]$$

Thus by the Theorem of Khoussainov and Nerode, the successor-function f is automatic.

For the set S, it depends on the automatic structure A. One can select $A = \{conv(x, y) : x, y \in \mathbb{N} \wedge y \leq x\}$. Now one orders this set A lexicographically: $(x, y) < (x', y')$ iff either $x = x'$ and $y < y'$ or $x < x'$. The set S is now the set of all pairs $conv(x, 0)$ and one sees that it is the sum set: Between $conv(x, 0)$ and $conv(x+1, 0)$ are exactly x non-elements $conv(x, 1), conv(x, 2), \ldots, conv(x, x)$.

One can also select $A = \{0\}^*$ and equip this with the length-comparing order. All regular subsets of A are eventually periodic and that would mean, they satisfy $\exists c, c' \forall x \geq c'\,[S(x + c) = S(x)]$, a formula not satisfied by the sum-set S. Hence in this automatic model, the set S is not regular.

A Mealy machine is a finite automaton where each transition from one state to another contains an (input,output)-pair to be processed; the final output is the concatenation of all the output words written while reading the corresponding input words. For all runs on the same input which read the input completely and end up in an accepting state, the same output must be produced.

Which of the following partial functions can be computed by a Mealy machine on a binary alphabet $\{0,1\}$ with parameters $n, m > 0$:

(a) $0^n1^m \mapsto 0^{2n+3}1^{2m+3}$;

(b) $0^n1^m \mapsto (01)^{n+m}$;

(c) $0^n1^m \mapsto 0^n1^n$.

If the function can be computed by a Mealy machine then **provide this Mealy machine** else **give reasons why it does not exist**. Note that on inputs not of the form 0^+1^+, the Mealy machine should not have any accepting run; that is, it should not happen that the machine reads the full input and ends up in an accepting state.

Solution. For (a), there are three states s, t, u. The Mealy automaton goes with $(0, 00000)$ from s to t, on t it goes with $(0, 00)$ to itself and with $(1, 11111)$ to u. On u it goes with $(1, 11)$ to itself. s is the start state and u is the only accepting state.

For (b), there are again three states s, t, u with s being the start state and u being the only accepting state. Only the transitions change. From s one can go on $(0, 01)$ to t and from t with $(0, 01)$ to t and $(1, 01)$ to u and on u with $(1, 01)$ to u itself. There are no other transitions.

For (c), there is no automaton. The reason is that the output language is context-free and not regular while a transducer maps every regular language to a regular language. Thus a transducer cannnot exist in this case.

Question 10 [6 marks] **CS 5236 – Solutions**

Let $H = \{0,1,2\}^+$ and let $|z|$ denote the length of the string z. Define the automatic family $\{L_x : x \in H\}$ in the three cases (1), (2), (3) by the corresponding formula:

(1) $L_x = \{y \in H : |y| \neq |x| - 1\}$;

(2) $L_x = \{y \in H : y \neq x\}$;

(3) $L_x = \{y \in H : y \geq_{ll} x\}$.

Consider learning from positive data (text). Determine for each of (1), (2), (3) whether the family (a) has an automatic learner with hypothesis-sized memory, (b) has a learner but no automatic learner with hypothesis-sized memory, (c) is not learnable at all. Give reasons for the decisions, each option occurs exactly once.

Solution. For case (1), option (c) applies. The sets L_0, L_1, L_2 are all equal to H while all sets L_x with $|x| > 1$ are proper subsets of H. Furthermore, if F_0 would be a tell-tale set for L_0, then there is an x much longer than all strings in F_0 and thus $F_0 \subseteq L_x \subset L_0$, a contradiction to the choice of the set F_0. Hence Angluin's tell-tale condition is not satisfied and therefore the class is not learnable from positive data.

For case (2), option (b) applies. Every finite subset F of H is contained in all L_x with $x \notin F$, whenever the learner converges on learning some L_x to a hypothesis, all the data seen so far is in many possible hypotheses and the learner needs to archive this data in order to be able to remember it when the need for a mind change comes up. As these are infinitely many data items of which more and more have to be memorised, the learner cannot obey any mind change bound depending on its current hypothesis, as such a bound allows only a finite amount of bits stored in the memory. On the other hand, the simple learner conjecturing L_x for the length-lexicographic least x not yet observed in the text is a learner, though not an automatic one. For that reason, in case (b) there is a learner, but none which obeys a hypothesis-sized memory bound.

For case (3), option (a) applies. There is an automatic learner, as the learner just has to track the length-lexicographically least string observed so far. This learner has an hypothesis-sized mind change bound, as the current hypothesis L_x needs the memory x as of the least element seen so far to be archived; as long as the learner has not yet seen any element, it conjectures some default set and has the pause symbol as memory. For automaticity, one extends the order such that every string is below the pause symbol, initialises the memory with the pause symbol and always takes the minimum of the current datum and the memory content when updating the memory. The hypothesis can be computed from the memory with an automatic function and also the update function is also automatic, as the order is.

END OF PAPER