Automata & Formal Languages
Michaelmas Term 2022
Part II of the Mathematical Tripos
University of Cambridge
Prof. Dr. B. Löwe

Example Sheet #3

(31) Let $\Sigma = \{a, b\}$. Consider the register machine M given by the following program:

$$\begin{aligned} q_{\mathrm{S}} &\mapsto\ ?(0, a, q_0, q_1) \\ q_0 &\mapsto\ +(1, a, q_2) \\ q_1 &\mapsto\ +(2, b, q_2) \\ q_2 &\mapsto\ -(0, q_{\mathrm{H}}, q_{\mathrm{S}}) \\ q_{\mathrm{H}} &\mapsto\ ?(0, \varepsilon, q_{\mathrm{H}}, q_{\mathrm{H}}) \end{aligned}$$

Assume that the register content of the registers indexed 0, 1, and 2 is *abbaa*, *bbaa*, and *aaa*, respectively. Determine the first twelve snapshots of the computational sequence produced by this machine on the given input. Will the computation eventually halt? If so, what are the halting time and the register content at time of halting?

(32) Let $\Sigma = \{a\}$ and consider the register machine given by the following program. Determine the operation performed by this register machine.

$$\begin{array}{llll} q_{\mathrm{S}} \mapsto -(0, q_0, q_3) & q_0 \mapsto +(2, a, q_1) & q_1 \mapsto +(3, a, q_4) & q_2 \mapsto +(2, a, q_7) \\ q_3 \mapsto ?(1, \varepsilon, q_6, q_{\mathrm{H}}) & q_4 \mapsto +(4, a, q_0) & q_5 \mapsto +(4, a, q_3) & q_6 \mapsto -(2, q_5, q_7) \\ q_7 \mapsto -(3, q_2, q_8) & q_8 \mapsto ?(4, a, q_9, q_3) & q_9 \mapsto +(2, a, q_7) & q_{\mathrm{H}} \mapsto ?(0, \varepsilon, q_8, q_9) \end{array}$$

[*Remark.* It helps to visualise the register machine with a diagram in the style of the graphical representations of automata.]

(33) Consider the following operation: given a word $w = a_0 ... a_n$, duplicate the final two letters, i.e., produce $a_0 ... a_{n-2} a_{n-1} a_{n-1} a_n a_n$; if $w = \varepsilon$, don't do anything; if $w = a$, produce aa. Explicitly write down a register machine M_0 (giving all program lines in detail) that performs this operation.

Find a second machine M_1 that is different from M_0, strongly equivalent to M_0, and has the same upper register index. Find a third machine M_2 that is different from M_0, strongly equivalent to M_0, and has a different upper register index. Find a fourth machine M_3 that is not strongly equivalent to M_0, but performs the same operation.

(34) Show that the question "*Is the content of registers i and j the same?*" can be answered by a register machine and that the operation "*Replace all occurrences of the letter a in register i by the letter b*" can be performed by a register machine. [In this example, you may use all operations performed and questions answered by register machines from § 4.2; in particular, you do not have to present a complete description of the program of the register machine.]

(35) We consider the following notions of equivalence between register machines M and M':

(i) The machines M and M' are *operation equivalent* if they perform the same operations.

(ii) The machines M and M' are *function equivalent* if for all k, they define the same partial functions on $\mathbb{W}^k$, i.e., if $f_{M,k} = f_{M',k}$.

(iii) The machines M and M' are *weakly equivalent* if they define the same languages, i.e., $\mathrm{W}_M = \mathrm{W}_{M'}$ (where $\mathrm{W}_M := \mathrm{dom}(f_{M,1})$).

Show that strong equivalence implies operation equivalence, operation equivalence implies function equivalence, function equivalence implies weak equivalence, and that none of the implications can be reversed.

(36) Let $\Sigma = \{a, b, c\}$. Show without using the result from (44) that the language $\{a^n b^n c^n \,;\, n > 0\}$ is computable.

(37) Let $X \subseteq \mathbb{W}^k$. In the lectures, we said that a total function $f : \mathbb{W}^k \to \mathbb{W}$ with the property

$$f(\vec{w}) \neq \varepsilon \iff \vec{w} \in X$$

is a *characteristic function of* X and a partial function $f : \mathbb{W}^k \dashrightarrow \mathbb{W}$ with $\mathrm{dom}(f) = X$ a *pseudo-characteristic function of* X. Show that X is finite if and only if every characteristic function of X is computable if and only if every pseudocharacteristic function of X is computable.

(38) Show that the class of computable sets is closed under finite unions, but not under countable unions.

(39) Prove that the shortlex ordering is irreflexive, transitive, and trichotomous.

(40) Let $\Sigma = \{0, 1\}$ with the ordering $0 < 1$. On $\mathbb{W} = \Sigma^*$, define the standard *lexicographic order*: $w \leq_{\text{lex}} w'$ if either $w \subseteq w'$ or, if k is the least number such that $w(k) \neq w'(k)$, then $w(k) < w'(k)$. Show that $(\mathbb{W}, \leq_{\text{lex}})$ is not isomorphic to $(\mathbb{N}, \leq)$.

(41) Suppose that $f : \mathbb{W} \to \mathbb{W}$ is a total function that is *increasing* in the following sense: if $v < w$, then $f(v) < f(w)$ (where $<$ is the shortlex ordering). Prove that if f is computable, then the range of f is computable.

(42) Let $\Sigma = \{0, 1\}$ with the ordering $0 < 1$ and consider the Σ-words

$$w_0 := 01010, \qquad w_1 := 110101, \qquad w_2 := 00111,$$
$$w_3 := 100, \qquad w_4 := 111110, \text{ and} \qquad w_5 := 11111.$$

Compute the following words:

(i) $w_0 * w_1$;
(ii) $(w_2)_{(0)}$;
(iii) $(w_3)_{(1)}$;
(iv) $(w_4)_{(0)} * (w_4)_{(1)}$;
(v) $s(w_4)$;
(vi) $s(w_5)$;
(vii) u such that $\#(u) = \#(w_4) + \#(w_5)$;
(viii) u such that $\#(u) = \#(w_3) \cdot \#(w_5)$.

(43) Show that the following functions are primitive recursive:

(i) $f(w) = v$ if and only if $w = v = \varepsilon$ or $\#(w) = \#(v) + 1$;

(ii) $f(w, v) = u$ if and only if $\#(w) - \#(v) = \#(u)$ or $\Big(\#(v) > \#(w)$ and $u = \varepsilon\Big)$.

(44) Let $\Sigma := \{\mathbf{0}, \mathbf{1}, \mathbf{a}, \mathbf{A}, \mathbf{(}, \mathbf{)}, \mathbf{\rightarrow}\}$. Define a (natural) encoding of all grammars over the alphabet $\{0, 1\}$ by words in $\mathbb{W} = \Sigma^*$, i.e., a function code that assigns elements of $\mathbb{W}$ to each word, string, production rule, and grammar over $\{0, 1\}$. Re-visit the proof of Theorem 1.18 ("The word problem for noncontracting grammars is solvable") and transform it into a proof of the statement that

$$\{(\mathrm{code}(G), \mathrm{code}(w)) \,;\, G \text{ is a noncontracting grammar and } w \in \mathcal{L}(G)\}$$

is a computable set. Conclude that if G is a noncontracting grammar, $\mathcal{L}(G)$ is a computable set.

[*Suggestion.* E.g., you could define the encoding by letting **a0** stand for 0, **a1** stand for 1, **(A10001)** stand for the variable A_{17} (note that 10001 is 17 in binary), **((A10001)(A111) → a1a0(A10001))** stand for the production rule $A_{17}A_7 \to 10A_{17}$, etc.]