

Problem 1: Representations of Regular Languages (40 pts.)**Background****Task**

We have seen that a regular language (i.e., a language that is accepted by a finite state machine) can also be described by a regular expression or by a formula in $\text{MSO}[<]$.

Recall that u is a factor of v if $\exists x, y \in \Sigma^* (v = xuy)$ and $u = u_1u_2 \dots u_n$ is a subword of v if $\exists x_0, \dots, x_n \in \Sigma^* (v = x_0u_1x_1u_2x_3 \dots u_nx_n)$.

Let K be the language of all words over alphabet $\Sigma = \{a, b, c\}$ containing exactly two factors ab . Likewise, let L be the language of all words over alphabet $\Sigma = \{a, b, c\}$ containing exactly two subwords ab .

- Construct the minimal automaton for K and L .
- Give a regular expression for K and L .
- Give a formula in $\text{MSO}[<]$ for K and L .

Comment

For the regular expressions and formulae try to find short, elegant answers. Make sure to explain what you are doing.

Problem 2: Primitivity and Minimality (30 pts.)**Background**

Let w be a non-empty word. Then z is the *root* of w if $w \in z^*$ but there is no shorter word with this property. A word is *primitive* if it is its own root. For example, ab is the root of $abababab$. The primitive words of length 3 over $\{a, b\}$ are $aab, aba, abb, baa, bab, bba$.

Given a binary word $u = u_0u_1 \dots u_{n-1}$ define the corresponding subset of $S(u) \subseteq \{0, 1, \dots, n-1\}$ by $i \in S(u) \iff u_i = 1$ (in other words, think of u as a bitvector for membership). Define a DFA $M(u)$ over the one-letter alphabet $\{a\}$ as follows:

$$M(u) = \langle \{0, 1, \dots, n-1\}, \{a\}, \delta; 0, S(u) \rangle$$

where $n = |u|$ and $\delta(p, a) = p + 1 \pmod n$.

Task

- A. Show that $M(u)$ is minimal if, and only if, u is primitive by reasoning about the behavior of the states in $M(u)$.
- B. Reprove part (B) by “running” a minimization algorithm on $M(u)$.
- C. **Extra Credit:** Figure out how to characterize arbitrary minimal DFAs over a one-letter alphabet.

Comment

Since minimization algorithms compute behaviors a lawyer might argue that part (C) is also an answer to part (B). Don't even think about it.

Problem 3: Acceptance for Büchi Automata (30 pts.)

Background

We have seen that acceptance of a Büchi automaton is undecidable if we consider sufficiently complicated input words (e.g., words specified by a computable function). However, for sufficiently simple words acceptance is still decidable.

Let $\mathcal{A} = \langle Q, \Sigma, \tau; I, F \rangle$ be a Büchi automaton.

Task

- A. Show how to decide whether \mathcal{A} accepts a “constant” input word $U = a^\omega$ where $a \in \Sigma$.
- B. Generalize to periodic inputs $U = u^\omega$ where $u \in \Sigma^*$.
- C. Generalize to ultimately periodic inputs $U = vu^\omega$ where $u, v \in \Sigma^*$.
- D. What is the running time of your three algorithms?

Comment

One might try to push things a bit further and consider inputs that are not periodic but have some reasonably simple description. For example, define

$$U(k) = \begin{cases} 1 & \text{if } k \text{ is prime,} \\ 0 & \text{otherwise.} \end{cases}$$

So

$$U = 0011010100010100010100010000010100000100010100010000010000010 \dots$$

Do you feel it is decidable whether some arbitrary Büchi automaton accepts U ?