15-354: CDM **K. Sutner**

Assignment 4 Due: **Sep. 29, 2023, 24:00**.

1. Primitive Words (30)

Background

Let w be a non-empty word. A word z is the root of w if $w \in z^\star$ 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.

Task

A. Show that any two non-empty words u, v that commute, $uv = vu$, have the same root.

B. Show that any two non-empty words u and v commute iff they have equal powers: $u^k = v^\ell$ for positive k and l.

C. Show that a non-empty word w is primitive iff $w^2 = xwy$ implies that $x = \varepsilon$ or $y = \varepsilon$.

D. Count the number of primitive words of length n over a k letter alphabet.

E. Show that the language of primitive words over an alphabet of size at least 2 is not regular.

F. As part of some algorithm it is necessary to store a few thousand binary words of length 20. Prof. Dr. Wurzelbrunft suggests to speed things up by storing the roots of words, rather than the words directly. What professional advice can you give Wurzelbrunft?

Comment

Möbius inversion might be helpful for part (D).

It is conjectured that primitive words are not even context-free, but that is currently an open problem. Extra credit.

2. Word Binomials (40)

Background

By a subsequence of a word $v = v_1 v_2 \dots v_m$ we mean any word $u = v_{i_1} v_{i_2} \dots v_{i_r}$ where $1 \leq i_1 < i_2 < \dots i_r \leq m$ is a strictly increasing sequence of indices. Thus *bbc* and *cab* are subsequences of *ababacaba* but *cbb* is not.

Note that a specific word can occur multiple times as a subsequence of another. For example, *aab* appears 7 times in *ababacaba*. We write

$$\binom{v}{u} = C(v,u) = \text{ number of occurrences of } u \text{ as a subsequence of } v.$$

The notation is justified since "word binomials" generalize ordinary binomial coefficients: $\binom{n}{k} = \binom{a^n}{a^k}$. Note that instances of u as a subsequence of v in general overlap, e.g., $C(a^3, a^2) = 3$.

Task

Let $\delta_{a,b} = 1$ iff $a = b$, 0 otherwise, be the Kronecker delta; $a, b \in \Sigma$ and $u, v, u_i, v_i \in \Sigma^\star$.

A. Show that

$$\binom{va}{ub} = \binom{v}{ub} + \delta_{a,b}\binom{v}{u}$$

B. Show that

$$\binom{v_1 v_2}{u} = \sum_{u=u_1 u_2} \binom{v_1}{u_1}\binom{v_2}{u_2}$$

C. Give an efficient algorithm to compute word binomials.

D. Give a simple description (in terms of union, concatenation and Kleene star) of the language

$$L = \{\, v \in \{a,b\}^\star \mid C(v, ab) = 3 \,\}$$

E. Construct the minimal DFA for L by diagram chasing (aka doodling).

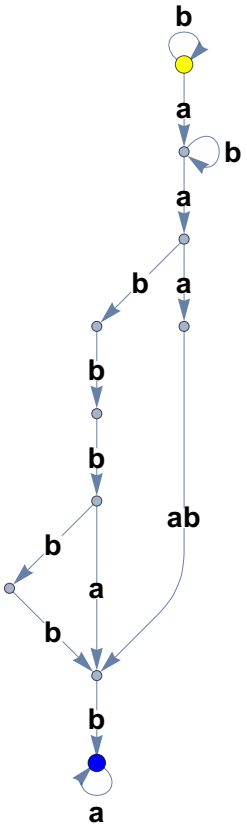
F. Generalize: given a word u and an integer r construct a DFA that accepts

$$L(u,r) = \{\, v \in \Sigma^\star \mid C(v,u) = r \,\}$$

Is your machine always minimal?

Comment

For what it's worth, here is a picture of the smallest possible DFA checking for 6 subwords *aab*. Make sure you understand how this machine works. Your construction will probably produce a much larger machine–but one that is also much easier to describe than this minimal one.


b
a
b
a
b
a
b
b
b
b
ab
b
a
b
b
a

3. Semilinear Counting (30)

Background

It is often stated that "finite state machines cannot count." To a point, this is correct, but there are special cases when a finite state machine is perfectly capable of counting. Here are some fairly involved examples of counting in zero space.

Recall that a set $C \subseteq \mathbb{N}$ is semilinear if it is a finite union of sets of the form $t + p\,\mathbb{N}$, where $t, p \in \mathbb{N}$ (for $p = 0$ this is just $\{t\}$) (transient and period). Let $L_C = \{\, 0^\ell \mid \ell \in C \,\} \subseteq 0^\star$.

Let $U \subseteq \Sigma^+$ be a regular language. A U-factorization of $x \in \Sigma^+$ is a sequence $u_1, \ldots, u_\ell$ of words in U such that $x = u_1 \ldots u_\ell$, $\ell \geq 1$. Write $\mathsf{fac}(x, U)$ for the set of all U-factorizations of x and define

$$L(U, C) = \{\, x \mid |\mathsf{fac}(x, U)| \in C \,\}$$

Thus, $L(U, C)$ collects all words that have exactly ℓ many U-factorizations where $\ell \in C$.

Task

A. Construct the minimal automaton for L_C.

B. Conclude that the semilinear sets form a Boolean algebra.

C. Show that $L(U, C)$ is regular.

Comment For (A), make sure your automaton is really minimal. For the last part, you probably want to use a pebbling argument and closure properties. Try $C = \{3\}$ first, then $C = \mathsf{evens}$.