15-354: CDM	**K. Sutner**
Assignment 4	Due: **Sep. 27, 2024, 24:00**.

1. Word Binomials (40)

Background

By a subsequence or subword of a word $v = v_1 v_2 \ldots v_m$ we mean any word $u = v_{i_1} v_{i_2} \ldots v_{i_r}$ where $1 \leq i_1 < i_2 < \ldots i_r \leq m$ is a strictly increasing sequence of indices. In other words, we can erase a few letters in v to get u. 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

Recall the Kronecker delta defined by $\delta_{a,b} = 1$ iff $a = b$, 0 otherwise. Let $a, b \in \Sigma$ 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

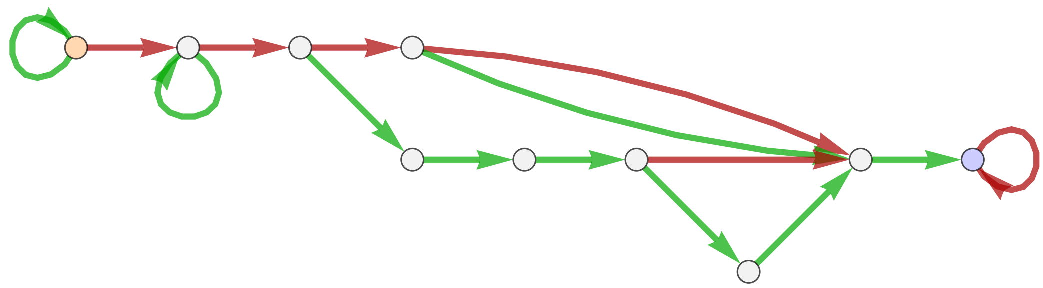
$$L(u,r) = \{\, v \in \Sigma^\star \mid C(v,u) = r \,\}$$

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.

subword aab 6-count



2. Semilinear Counting (30)

Background

It is often stated that "finite state machines cannot count." To a point, that 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

$$t + p\mathbb{N} = \{\, t + i\,p \mid i \geq 0 \,\}$$

where $t, p \in \mathbb{N}$; for $p = 0$ this is just the singleton $\{t\}$ (think of transient and period). Let $L_C = \{\, 0^\ell \mid \ell \in C \,\} \subseteq 0^\star$, the numbers in C written in unary.

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

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

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.

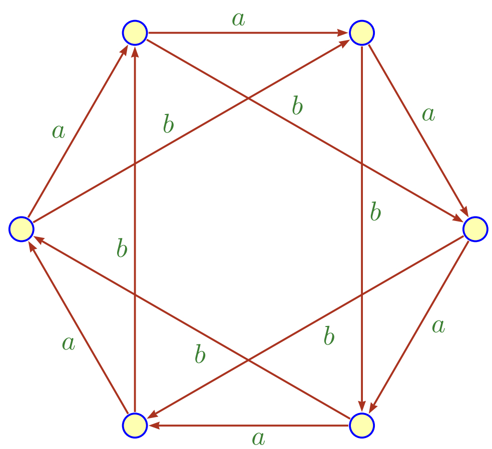
C. Show that $L(U, C)$ is regular.

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 $C = \mathsf{evens}$.

3. Blowup (30)

Background

Write $\mathcal{A}_n$ for the (boring) automaton on n states whose diagram is the circulant with n nodes and strides 1 and 2. The edges with stride 1 are labeled a and the edges with stride 2 are labeled b. For example, the following picture shows $\mathcal{A}_6$. We assume $I = F = Q$.



Let $\mathcal{B}_n$ be the (interesting) automaton obtained from $\mathcal{A}_n$ by switching one of the b labels to an a label; write K_n for the acceptance language of $\mathcal{B}_n$.

Task

A. Show that determinization of $\mathcal{B}_n$ produces an accessible automaton $\mathcal{B}'_n$ of 2^n states.

B. Argue that $\mathcal{B}'_n$ is already reduced and conclude that K_n has state complexity 2^n.

Comment

The language K_n has no particular significance (as far as I know). Thinking about pebble automata might help with the argument.

Extra credit: If you switch an a to a b, there is still full blow-up for odd n, but for even n the power automaton has only size $2^n - 1$.