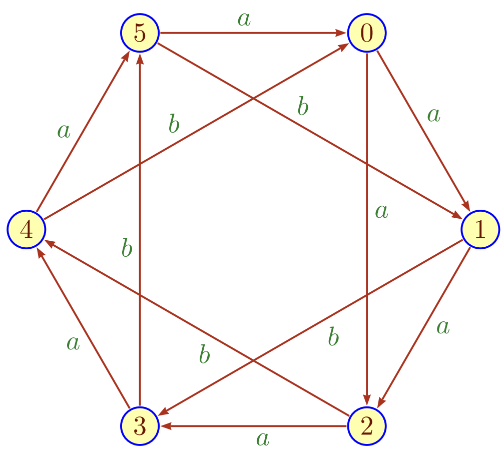
15-354: CDM **K. Sutner**

Assignment 5 Due: **Oct. 6, 2023, 24:00**.

1. Blow-Up (40)

Background

Let n be a positive integer and write $\mathcal{A}$ for the semi-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, except for one. For example, the following picture shows $\mathcal{A}$ for $n = 6$.



Note the a-transition from 0 to 2, if this were a b-transition, the automaton would be completely boring (recall that in a semi-automaton $I = F = Q$, so the language would simply be $\Sigma^\star$). Write K for the acceptance language of $\mathcal{A}$. Call a set $P \subseteq Q$ persistent if $\bigcap_{p \in P} [\![p]\!]_{\mathcal{A}} - [\![Q - P]\!]_{\mathcal{A}} \neq \emptyset$. A string in the non-empty set is a witness for P.

Task

A. Find the length-lex minimal witness for $P = \emptyset$.

B. Repeat for $P = Q$, but with the extra condition that the witness must contain a letter b.

C. Show that all $P \subseteq Q$ are persistent.

D. Show that determinization of $\mathcal{A}$ produces an accessible DFA $\mathcal{B}$ with 2^n states.

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

Comment

In particular for part (C), arguing in terms of pebbles is probably the best approach.

Solution: Blow-Up

Part A: Witness $\emptyset$

We need a string w such that $\delta(Q, w) = \emptyset$. Hence, at least n letters b are necessary. For odd n, b^n actually works. For even n, we need $b^{n/2}ab^{n/2}$ because the stride-2 edges form 2 disjoint cycles of length $n/2$ that must both be eliminated.

Part B: Witness Q

$a^n b$: We need a string in the behavior of all states. Clearly, any string in $a^\star$ works, but, since we need a b, consider a witness $w = a^i bu$. One can check that for $i < 6$, $\delta(p, a^i b) = \emptyset$ for some state $p \neq 0$.

Part C: Persistence

Informally, P is persistent if there is a word in the behavior of all $p \in P$, but not in the behavior of any $q \notin P$.

To this end, place a black pebble on each state in P, and a red pebble on all the other states. If a black pebble sits on state 0, we fire an a, but we underestimate the result: assume the pebble does not split and just moves to 1. The advantage of this underestimate is that it avoids collisions. So, letter a induces a cyclic rotation on black pebbles. Otherwise, everything moves according to the standard pebbling rules.

As long as there is a red pebble somewhere, let i minimal such that $\delta(P, a^i)$ has a red pebble on 0, then fire a letter b. This reduces the number of red pebbles, but does not affect the number of black ones. By induction, there is a witness for P.

Part D: Reachability

Write δ^{op} for the transition relation of the reversal of the machine.

We claim that for all $P \subseteq Q$: P is persistent iff P is reachable in $\mathcal{B}^{\mathsf{op}}$ from Q.

To see why, note that P is persistent iff $w = x^{\mathsf{op}}$ is a witness where $\delta^{\mathsf{op}}(Q, x) = P$.

Here comes a trick: $\mathcal{A}$ and $\mathcal{A}^{\mathsf{op}}$ are isomorphic, so by part (C) every subset of Q is of the form $\delta(Q, x)$, and we are done.

Part E: Minimality

Let $P_1 \neq P_2 \subseteq Q$, say, $p \in P_1 - P_2$. Since $\{p\}$ is persistent and $P_2 \subseteq Q - \{p\}$ we are done.

2. Window Languages (30)

Background

In this problem we only consider languages in Σ^+, so the empty word causes no technical problems. A language L is a window language if membership in L can be tested by sliding a window of size 2 across the word and observing the 2-factors of the word.

Here is a formalization of this idea. Define $\mathsf{fact}_2(x) = \{ ab \in \Sigma^2 \mid x = uabv, u, v \in \Sigma^\star \}$ to be the set of all 2-factors of x. Define an equivalence relation $\approx$ on Σ^+ as follows:

$$x \approx y \iff x_1 = y_1 \wedge \mathsf{fact}_2(x) = \mathsf{fact}_2(y) \wedge x_{-1} = y_{-1}$$

Then L is a window language if it saturates $\approx$: $L = \bigcup_{x \in L}[x]$. Write Σ^{++} for all words of length at least 2. Given $F \subseteq \Sigma^2$ let $L_F = \{ x \in \Sigma^{++} \mid \mathsf{fact}_2(x) \subseteq F \}$.

Task

A. Find a fast algorithm to check whether L_F is finite. What is the running time of your algorithm?

B. Find a simple, infinite language $L \subseteq \Sigma^{++}$ that is not of the form L_F.

C. Show that $\approx$ is a congruence: $x \approx y$ and $u \approx v$ implies $xu \approx yv$.

D. Show that every window language is regular.

E. Show that every regular language is a homomorphic image of a window language.

Comment

For the last part you need to produce a regular language $R \subseteq \Gamma^\star$ for some alphabet Γ and a homomorphism $\Phi : \Gamma^\star \to \Sigma^\star$ such that $\Phi(R) = L$. Γ will depend strongly on a DFA for the given regular language.

This is perhaps a little counterintuitive: the window seems to be too narrow for arbitrary regular languages with long-distance constraints (say, every a is followed by a b after at most 123 letters).

Solution: Window Languages

Part A: Finiteness

Consider the subgraph G of the de Bruijn graph over Σ of order 2 whose nodes are given by F. G can be constructed in $O(k^2)$ steps where k is the cardinality of Σ. Then L_F is finite iff G has only trivial strongly connected components. The latter property can be checked in linear time.

Part B: Non-Window

Let $\Sigma = \{a, b\}$ and define $L \subseteq \Sigma^{++}$ to be the collection of all words that do not contain a block aaa. Assume that for some $F \subseteq \Sigma^2$ we have $L_F = L$. Note that $\mathsf{fact}_2(L) \subseteq F$, so that $F = \Sigma^2$. But then $L_F = \Sigma^{++}$, contradiction.

Part C: Congruence

By definition, $\approx$ is the kernel relation of a function

$$f : \Sigma^+ \to \Sigma \times \mathfrak{P}(\Sigma^2) \times \Sigma$$

and thus an equivalence relation. To see that it is also a congruence, note that

$$\mathsf{fact}_2(xu) = \mathsf{fact}_2(x) \cup \mathsf{fact}_2(u) \cup \{x_{-1}u_1\}$$

Part D: Regular

Given x, we can construct a partial DFA that accepts the $\approx$-equivalence class of x on state set $\{\perp, x_1\} \cup \mathsf{fact}_2(x)$ with the obvious de Bruijn type transitions. Final states are all $ab \in \mathsf{fact}_2(x)$ such that $b = x_{-1}$. If $x = x_1$ we omit the 2-factors and make x_1 final. Done by closure under union.

Alternatively, let

$$F = \Sigma^2 - \mathsf{fact}_2(x)$$

be the set of forbidden 2-factors. Then the class of x is

$$(x_1\Sigma^\star \cap \Sigma^\star x_{-1}) - \Sigma^\star F \Sigma^\star$$

Done by closure.

Part E: Image

Suppose $\mathcal{A}$ is a DFA for L over Σ. Define

$$\Gamma = \{\, (p, a, q) \in Q \times \Sigma \times Q \mid \delta(p, a) = q \,\}$$

and define R over Γ by only allowing only words x such that

- $x_1 = (q_0, a, p)$ and $x_{-1} = (p, a, q)$ where $q \in F$.
- $(p, a, q)(p', b, q')$ is a 2-factor if $q = p'$.

Clearly, R is a window language. Furthermore, a word in R codes an accepting trace of $\mathcal{A}$ in a straightforward way. Define the homomorphism by $\Phi((p, a, q)) = a$.

3. The Un-Equal Language (30)

Background

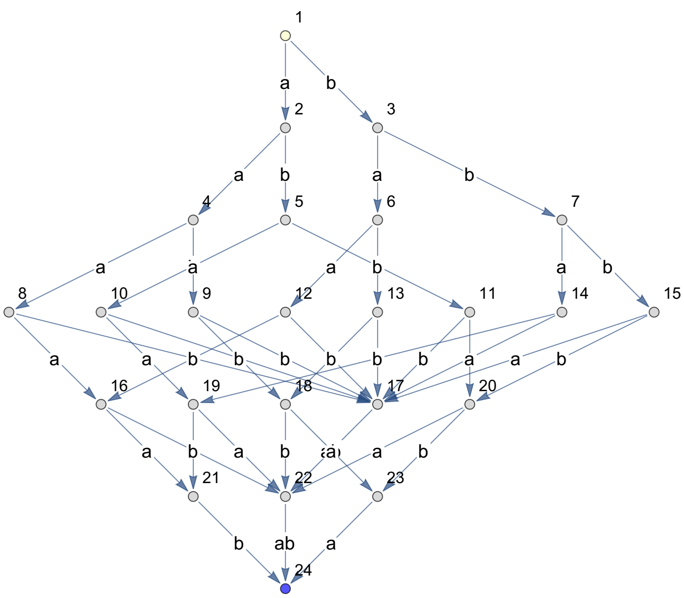
Consider the language of all strings of length $2k$ that are not of the form uu:

$$L_k = \{\, uv \in \{a, b\}^\star \mid |u| = |v| = k, u \neq v \,\}.$$

These languages are finite, hence trivially regular. The following table shows the state complexity of L_k up to $k = 6$.

k	1	2	3	4	5	6
sc	5	12	25	50	99	196

The minimal DFA for L_3 looks like so (the layout algorithm is not too great):



This is the partial DFA without sink, the top state is initial and the bottom state final.

Task

A. What happens when you run Moore's algorithm on this DFA? How many rounds are there?

B. Determine all quotients for L_2.

C. Generalize. In particular explain the diagram for L_3.

D. Determine the state complexity of L_k.

E. Determine the state complexity of $K_k = \{\, uu \mid u \in \{a, b\}^k \,\}$.

Comment For part (A), and in the interest of maintaining TA sanity, use the state numbering from above (sink is state 25).

From the diagram and the table it is not hard to conjecture a reasonable closed form for the state complexity. For a proof one can exploit the description of the minimal DFA in terms of quotients.

Solution: The Un-Equal Language

Part A: Moore

Initially there are two classes: 1 and 24. Step by step we get classes

$$
\begin{array}{l}
1, 24 \\
1, 21, 22, 23, 24 \\
1, 16, 17, 18, 19, 20, 21, 22, 23, 24 \\
1, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24 \\
1, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24 \\
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24 \\
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25
\end{array}
$$

In other words, in each round the next higher level in the graph becomes separated. In the last round, the sink moves into its own class.

Part B: Quotients L_2

The quotients of L_2 organized by length of the dividing

$$
\begin{array}{ll}
0 & L_2 \\
1 & (aab, aba, abb, baa, bba, bbb), (aaa, aab, abb, baa, bab, bba) \\
2 & (ab, ba, bb), (aa, ba, bb), (aa, ab, bb), (aa, ab, ba) \\
3 & (a), (b), (a, b) \\
4 & (\epsilon) \\
5 & \emptyset
\end{array}
$$

Part C: Quotients

Ignoring the sink, states are organized in layers, at layer ℓ we have $x^{-1}L_k = P \subseteq \{a, b\}^{2k-\ell}$ where $|x| = \ell$. In particular for $\ell = k$ we have $u^{-1}L_k = \{a, b\}^k - \{u\}$ and the structure of the quotient automaton down to level k is a complete binary tree; the number of states in this part is $2^{k+1} - 1$.

The remainder of the machine has two kinds of states: those where a witness for inequality of u and v has already been found, and those where we are still waiting for such a witness.

Fix some $u \in \{a, b\}^k$. The first type of state is of the form $\delta(q_0, ux)$ where x is not a prefix of u and has behavior $\{a, b\}^{k-|x|}$, where $|x| \leq k$. The second type is of the form $\delta(q_0, ux)$ where x is a prefix of u and has behavior $\{a, b\}^{k-|x|} - \{y\}$ where $u = xy$. But then there are k states of the first type, and $2^k - 1$ states of the second type (these form another tree which is upside-down compared to the first, and has as root the sink of the DFA).

Part D: State Complexity

It follows from the analysis in part (B) that the state complexity of L_k is $3 \cdot 2^k + k - 2$.

Part E: Repetitions

The state complexity of K_k is $3 \cdot 2^k - 1$.