

15-354: Computational Discrete Math

K. Sutner

Solutions Assignment 4

Solution: Primitive Words

Part A: Commutativity

By induction on $|uv|$.

For $|uv| = 2$ the claim is obvious, so assume $|uv| > 2$. Also assume without loss of generality (check that this is really true!) $|v| < |u|$. But then $uv = vu$ implies $u = vu_0$ where $|u_0| < |u|$. Note that $u_0v = vu_0$, so by induction we may assume that for some w we have $u_0, v \in w^*$. Our claim follows.

Part B: Primitivity

If w fails to be primitive it is easy to violate the condition on the right hand side: let $w = u^r$ where $r \geq 2$. Then $ww = u w u^{r-1}$, done.

So suppose w is primitive but $ww = xwy$ where $x, y \neq \varepsilon$. Then $w = w_-w_+ = w_+w_-$ where $0 < |w_-| = |x| < |w|$. By part (A) w is not primitive.

Part C: Root Count

Clearly any word w of length $n > 0$ is either primitive or can be written as $w = u^r$ for some $r \geq 2$. Of course, r is uniquely determined. But then

$$k^n = \sum_{d|n} \pi(d)$$

By Möbius inversion we get

$$\pi(n) = \sum_{d|n} \mu(d) k^{n/d}.$$

where μ is the Möbius function. Good enough to compute $\pi(n)$ for any reasonable value of n .

Part D: Wurzelbrunft

As is usually (though not always!) the case, Wurzelbrunft is clueless: the fraction of imprimitive binary words of length 20 is about 0.999012, so only a handful of the “thousands” of words would be any shorter.

Solution: Divisibility in Reverse Binary

Part A: Divisibility by 3

The value of a string in reverse binary is

$$\nu(x_0x_1 \dots x_k) = \sum_{i \leq k} x_i 2^i$$

so that $\nu(xa) = \nu(x) + a2^{|x|}$. Modulo 3 the powers of 2 simply alternate between 1 and 2, so we can use a state set

$Q = \{0, 1, 2\} \times \{1, 2\}$ and a transition function

$$\delta((p, q), a) = (p + aq \bmod 3, 2q \bmod 3).$$

Initial state is $(0, 1)$ and final states are $(0, 1)$ and $(0, 2)$.

Part B: Generalization

To deal with modulus m and reverse base B we use a state set $Q = (m) \times T \subseteq (m) \times \mathbb{Z}_m$. For the second component, note that we do not need all of \mathbb{Z}_m in general, but only T , the multiplicative submonoid of \mathbb{Z}_m generated by B . The transition function is now $\delta((p, q), a) = (p + aq \bmod m, Bq \bmod m)$. The initial state is again $(0, 1)$ and the final states are of the form $(0, q)$.

Part C: Divisibility by 5, reverse binary

The machine from above has size 20 since in this case $T = \{1, 2, 4, 3\}$. But the minimal automaton has size 5. By state merging we can determine the behavioral equivalence classes, they are all of size 4 and of the form $\{(a_1, 1), (a_2, 2), (a_3, 3), (a_4, 4)\}$ where the a_i are either all 0 or a permutation of T .

$(0, 1)$	$(0, 2)$	$(0, 3)$	$(0, 4)$
$(1, 1)$	$(2, 2)$	$(3, 3)$	$(4, 4)$
$(1, 2)$	$(2, 4)$	$(3, 1)$	$(4, 3)$
$(1, 3)$	$(2, 1)$	$(3, 4)$	$(4, 2)$
$(1, 4)$	$(2, 3)$	$(3, 2)$	$(4, 1)$

Why? The key observation is that $\nu(x) = 0 \pmod{5}$ iff $\nu(0x) = 0 \pmod{5}$. Hence instead of checking x for acceptance we can also check $0x$, $00x$ and $000x$, corresponding to the states in the first class (which is the same as the set of final states). But the transition operation is reversible in the second component, so the image of these four states is always another group of 4 states. From the observation, these 4 states are either all final (i.e., we have gone back to the first class) or all non-final. Since every state is reachable, behavioral equivalence must partition the state set of the 20-state machine into block of size 4, so there must be exactly 5 of them.

Solution: Balance and Majority

Part A: Balance

Suppose for the sake of a contradiction that M is a DFA accepting L_{bal} . Then M accepts all inputs $0^i 1^i$, $i \geq 0$. Since the state set of M is finite there must be a loop in the diagram: more precisely, there exist $r < s$ such that $\delta(q_0, 0^r) = \delta(q_0, 0^s)$. But then

$$\delta(q_0, 0^r 1^s) = \delta(\delta(q_0, 0^r), 1^s) = \delta(\delta(q_0, 0^s), 1^s) = \delta(q_0, 0^s 1^s) \in F$$

contradiction. Unbounded counting is beyond a finite state machine.

Part B: Majority

Again suppose that we have a DFA M for L_{maj} . Clearly we can then build another DFA M' for

$$L'_{\text{maj}} = \{x \in \mathbf{2}^* \mid |x|_0 \geq |x|_1\}.$$

This amounts simply to interchanging the 0 and 1 transitions.

But then by the closure properties of regular languages we can also build a DFA for

$$L_{\text{maj}} \cap L'_{\text{maj}} = L_{\text{bal}}$$

contradicting part A.

The last argument exploits closure properties of regular languages and reduces the problem to a similar one that we already understand. Needless to say, one can also directly adapt the argument from the balance case.

Yet another way to tackle this problem is to use quotients: for both languages one can show that the number of quotients is infinite, so the languages cannot be regular. Exercise.