ON DIVISION BY THREE

LEAH NEUKIRCHEN

I was thoroughly nerdsniped by a recent post on The Unix Heritage Society mailing list, where this anecdote was mentioned by Noel Chiappa:

> Steve Ward told another oral story which I'm pretty sure *is* true, though. They ask the candidate to design a state machine (or digital logic, I forget which) which can tell if a number is divisible by three (I think I have the details correct, but I'm not absolutely certain). So they describe one – and then point out that you can feed the number in from either end (most or least significant end first) – and proves that it will work either way! The committee was blown away.

Since this sounded like an interesting statement, I tried to find a proof myself and found two solutions, one using automata theory and one using ordinary math.

An automata-theoretic proof. On the first glimpse, we may be a bit surprised that divisibility by three can be detected by a state machine, *i.e.* the set of binary numbers divisible by three is a regular language. From school we know some divisibility rules for decimal numbers, certainly when a number is divisible by 2, 4, 5, 10, 20, 25, 50 etc., which just considers the last digits. These are clearly regular as they correspond to a regular expression like `/.*[05]/` for 5. Likewise, in binary, divisibility by powers of two can easily be checked by looking at trailing zeroes. But for more complicated divsibility rules such as the one for 3, 7, 11 we don't have a regular expression come to mind.

However, all divisibility relations can be checked by a finite state machine, and this becomes very clear when we consider the unary case, *i.e.* sequences of a single symbol (say `1`): $\Sigma = \texttt{1}^*$.

In order to check unary divisibility by n, we define an determinisic finite automaton with n states, make q_0 the initial and accepting state and define transitions $q_0 \to q_1 \to q_2 \to \cdots \to q_{(n-1)} \to q_0$. This is just counting in a finite ring.

A similar construction will yield a divisiblity automaton for a b-adic positional number system. Again, we need n states which have b-transitions each, for each digit symbol d that is an arrow from q_i to $q_{((b \cdot i + d) \mod n)}$.

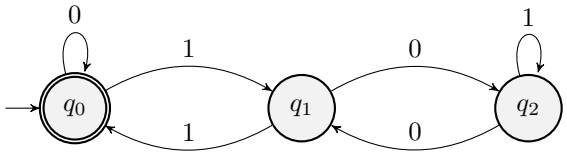
A priori its not clear that an unary regular language also implies that the corresponding language in, say, binary is also regular; there's possibly a more elementary proof for this, but if we wanted to use a big hammer, we can use Cobham's theorem, which states:

> Let S be a set of non-negative integers and let m and n be multiplicatively independent positive integers. Then S is recognizable by finite automata in both m-ary and n-ary notation if and only if it is ultimately periodic.

Since 1 is multiplicatively independent to every integer $n \geq 2$ and the language is trivially ultimately periodic this works in any base.

E-mail address: leah@vuxu.org.
Date: Feburary 2023.

Thus, we can define the requested automaton to check divisibility for three on binary numbers:



Now to the curious insight that this also works when the bit string is reversed: given a regular language L, the language of reversed words L^r is also regular. This can be seen by a construction on the DFA of L: define a NFA for L^r by exchanging start and accepting states and reversing all arrows (if there's more than one starting state now you need to prepend ε-transitions; this doesn't matter in our case). Then determinize the NFA again.

However, for above automaton this operation changes nothing, as it yields the same automaton! This makes it evident that divisibility by three is independent of endianess.

(I'm not sure how much convincing the committee mentioned above took if the automaton was presented in this way.)

The mathematical approach. Of course, we can also try to find a proof using high-school math, but this one doesn't generalize as nicely as above construction. In particular, proving things about "reversed digits" is often hard, and in fact, we only need a lot weaker assertion here.

Proof. An n-bit number k is written in base 2 as bits b_i, *i.e.* $k = \sum_{i=0}^{n} 2^i b_i$.

If we look at this number mod 3, it must be 0 when it's divisible by 3:

$$\sum_{i=0}^{n} 2^i b_i \equiv 0 \mod 3$$

Since $2 = -1 \mod 3$, we rewrite this as:

$$\sum_{i=0}^{n} (-1)^i b_i \equiv 0 \mod 3$$

Now we split this formula into two cases, for even and odd positions:

$$\sum_{i=0, i \text{ even}}^{n} 1 \cdot b_i + \sum_{i=0, i \text{ odd}}^{n} (-1) \cdot b_i \equiv 0 \mod 3$$

This can now be rewritten as

$$\sum_{i=0, i \text{ even}}^{n} b_i - \sum_{i=0, i \text{ odd}}^{n} b_i \equiv 0 \mod 3 \quad \text{or} \quad \sum_{i=0, i \text{ even}}^{n} b_i \equiv \sum_{i=0, i \text{ odd}}^{n} b_i \mod 3,$$

i.e. when the bit string has the same number of ones on even and odd positions (mod 3).

If we now reverse the indices of b, either even and odd bits stay the same, or they swap positions. But since both sides of the equation are the same, the direction of the bitstring doesn't matter in either case. □

Exercise. For which bases b and which divisors n does divisibility imply divisibility of the reversed number?