Homework 2: DFAs and NFAs

Will Griffin

January 28, 2026

1. Divisibility Tests

Define, for all $k > 0$,

$$D_k = \{w \in \{0, \ldots, 9\}^* \mid w \text{ is the decimal representation of } k\}$$

where ε is considered to represent the number 0. For example, the strings ε, `0`, `1234`, and `01234` all belong to D_2, but `99` and `099` do not.

($*$) For any $k > 0$, the language D_k is regular to the DFA $M = (Q, \Sigma, \delta, q_0, q_0)$ where

- $Q = \{q_0, \ldots, q_{k-1}\}$
- $\Sigma = \{0, \ldots, 9\}$
- $\delta(q_n, w \in \Sigma^*) = q_{(n \times 10 + w \pmod k)}$

For any $k > 0$, any whole number divided by k has a remainder between 0 and $k-1$. Therefore, the number of possible remainders for dividing a number by k is equal to k, and thus the number of states for the DFA representing dividing a number by k is equal to k. Q is the set of all states of this DFA, where each state is numbered by the corresponding remainder ($Q = \{q_0, \ldots, q_{k-1}\}$).

By having each state correspond to a remainder, the DFA then stores the remainder of the number in the string $w \in \Sigma^*$ as each digit is processed.

Mathematically, the transition function δ operates as such on string $w \in \Sigma^*$:

(a) Base Case (w_1):

The start state is q_0, so $r_{prev} = 0$. Therefore, $\delta(q_0, w_1) = q_{(0 \times 10 + w_1 \pmod k)} = q_{(w_1 \bmod k)}$

(b) Recursive Case (w_i, $0 < i \leq |w|$):

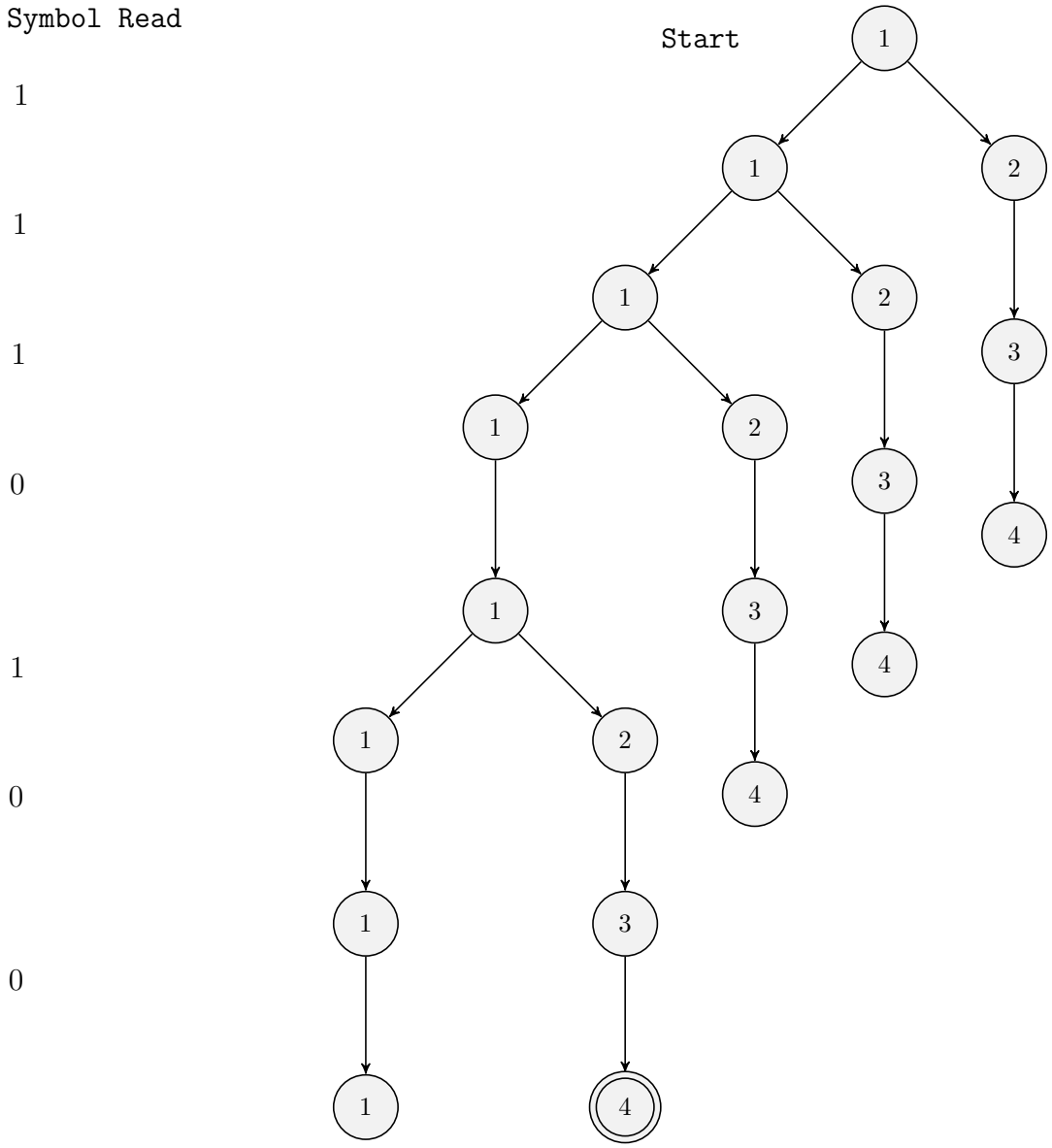
The remainder of a number can be found through calculating the remainder of its digits one by one. If N is the string of already read digits, and d is the next digit, $N_{new} = N \times 10 + d$. Taking the modulus with respect to k, we obtain the remainder of N_{new} calculated from the remainder of N:

$$\begin{aligned} N_{new} \bmod k &= (N \times 10 + d) \bmod k \\ r_{new} &= (((N \bmod k)(10 \bmod k) \bmod k) + (d \bmod k)) \bmod k^1 \\ r_{new} &= ((R \times (10 \bmod k) \bmod k) + (d \bmod k)) \bmod k \\ r_{new} &= (R \times 10 + d) \bmod k \end{aligned}$$

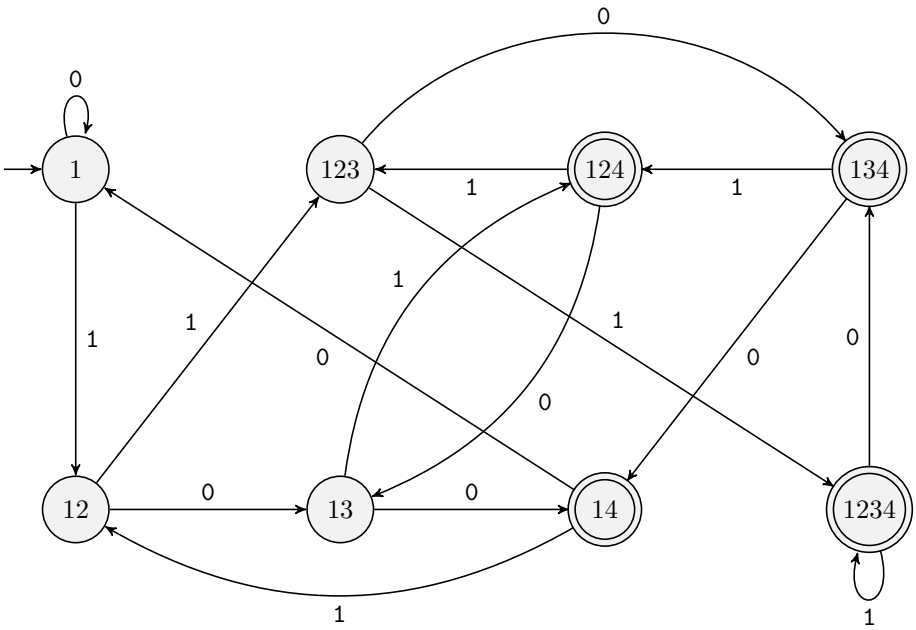
We can apply this to the transition function δ, such that for a DFA at current state q_j, $j \in [0, k-1]$, processing symbol $w_i \in w$, $\delta(q_j, w_i) = q_{((j \times 10 + w_i) \bmod k)}$.

2. Nondeterminism

(a) The computation of `1110100` on N_2.



(b) The DFA M of the NFA N_2



(c) The order `1110100` goes through M is:

- 1
- 12
- 123
- 1234
- 134
- 124
- 134
- 14

(d) The states in the DFA of Figure 1.31 have subscripts containing the last three digits read. It starts out assuming the last three digits read were `0`s, essentially a cleared input stream. As `1`s are read, they are input in the last digit and shifted left as more digits are read. It is also easier to obviously see success states; each state with notation q_{1xx} will be a success state as it has a `1` in the third to last position.

3. Procrustean Closure Properties

(a) Prove that for any regular language L, STRETCH(L) is also regular.

If L is a regular language, there must be some DFA M that recognizes it. Let that DFA $M = (Q, \Sigma, \delta, s, F)$.

To prove STRETCH(L) is also regular, let another DFA $M' = (Q', \Sigma, \delta', s', F')$ exist that recognizses STRETCH(L).

Since every symbol in L is doubled when STRETCH(L) is applied, we need to add a waiting state for every character in Σ. We also need to add a dead state so that when in any waiting

state, if the succeeding character is different, the DFA moves to a dead state with no motion to success. Therefore, $Q' = Q \cup \{q_{\texttt{wait},a} | q \in Q, a \in \Sigma\} \cup \{\texttt{DEAD}\}$.

The transition function δ' will have three operations for any $q \in Q; a, b \in \Sigma, a \neq b$:

- $\delta'(q, a) = q_{\texttt{wait},a}$
- $\delta'(q_{\texttt{wait},a}, a) = \delta(q, a)$
- $\delta'(q_{\texttt{wait},a}, b) = \texttt{DEAD}$

Since both M and M' are at the very beginning of their computation before reading characters, therefore $s' = s$.

Since L is regular to M and STRETCH(L) is regular to M', and δ' after passing a `wait` state is the result of the original transition function δ, clearly $F' = F$.

Since we have constructed a DFA $M' = (Q', \Sigma, \delta', s', F')$ that accepts language STRETCH(L) from a DFA M that accepts language L, for any regular language L, STRETCH(L) is also regular.

(b) Prove that for any regular language L, CHOP(L) is also regular.

If L is a regular language, there must be some DFA M that recognizes. Let that DFA $M = (Q, \Sigma, \delta, s, F)$.

To prove CHOP(L) is a regular language, let a DFA $M' = (Q', \Sigma, \delta', s', F')$ exist that recognizes CHOP(L).

Since the CHOP(L) operation removes the first symbol in the string, let the start state's transitions be defined as $\delta'(s', w_2) = \delta(\delta(s, w_1), w_2)$ where $w \in L, w = w_1 w_2 \dots w_{n-1} w_n$.

Since the CHOP(L) operation also removes the last symbol in the string, let the state of accept states F' be defined as $F' = \{q \mid \delta(q, a) = q_f, a \in \Sigma, q_f \in F\} \cup \{s' \mid \varepsilon \in \text{CHOP}(L)\}$.

Let $Q' = (Q \setminus (\{s\} \cup \{q \mid \delta(s, w_1) = q, w \in L, w = w_1 w_2 \dots w_n\} \cup F)) \cup \{s'\} \cup F'$.

Since we have constructed a DFA M' that accepts language CHOP(L), for any regular language L, CHOP(L) is also regular.

4. References

(1) "Modulo/Properties." *Wikipedia*, Wikimedia Foundation, 21 Jan. 2026, `https://en.wikipedia.org/wiki/Modulo#Properties_(identities)`.