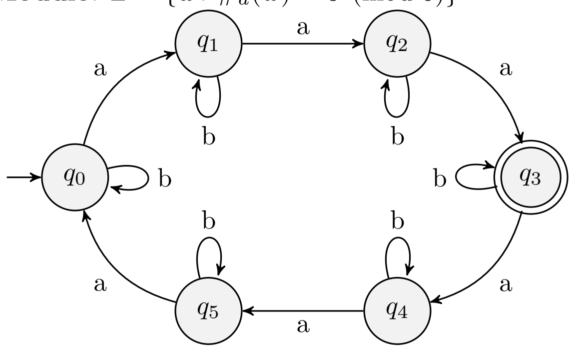
CMSC452 Elementary Theory of Computation

Michael Li

Contents

1 DFA $(Q, \Sigma, \delta, s, F)$

Modulo: $L = \{w \colon \#_a(w) \equiv 3 \pmod 5\}$



Intersection: $L = \{w \colon \#_a(w) \equiv 3 \pmod 5 \wedge \#_b(w) \equiv 2 \pmod 3\}$
$Q = \{0, \ldots, 4\} \times \{0, \ldots, 2\}$
$\Sigma = \Sigma$
$\delta((q_1, q_2), \sigma) = (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma))$
$s = (0, 0)$
$F = F_1 \times F_2$

Expanded Notation: Number written in base 10 with mod 7
$10^0 \pmod 7 = 1$
$10^1 \pmod 7 = 3$
$10^2 \pmod 7 = 2$
$10^3 \pmod 7 = 6$
$10^4 \pmod 7 = 4$
$10^5 \pmod 7 = 5$
$10^6 \pmod 7 = 1$

$Q = \{0, \ldots, 6\} \times \{0, \ldots, 5\}$ Keep track of weighted sum $\pmod 7$ and track digit placement $\pmod 6$
$\Sigma = \{0, \ldots, 9\}$
$\delta(a, 0), i) = (a + 1 * i \pmod 7, 1)$
$\delta(a, 1), i) = (a + 3 * i \pmod 7, 2)$
$\delta(a, 2), i) = (a + 2 * i \pmod 7, 3)$
$\delta(a, 3), i) = (a + 6 * i \pmod 7, 4)$
$\delta(a, 4), i) = (a + 4 * i \pmod 7, 5)$
$\delta(a, 5), i) = (a + 5 * i \pmod 7, 0)$
$s = (0, 0)$

Minimum Number of DFA States Proof: a^n requires n states
By pigeonhole principle, if a DFA requires $n-1$ states, 2 of the states, q_i, q_j $(i \neq j)$, must be the same. Then

$$a^i a^{n-i} = a^n \neq a^j a^{n-i}$$

Thus contradiction is reached and DFA requires at least n states.

DFA Complementation: $L(Q, \Sigma, \delta, s, Q - F)$

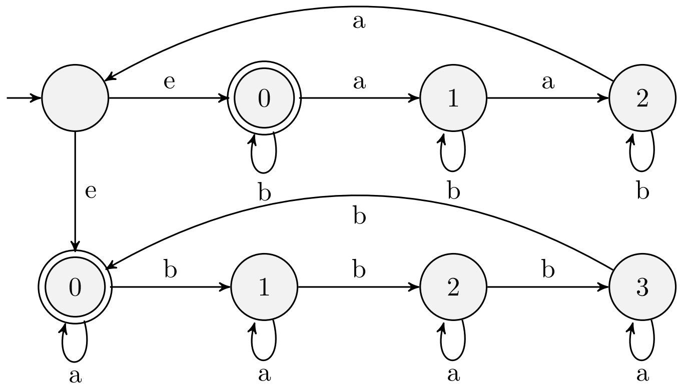
DFA Capabilities

- DFA can track $_a(w) \pmod{17}$
- DFA cannot track $_a(w)$
- If DFA M exists and $L(M) = L$, then L is regular

2 NFA $(Q, \Sigma, \Delta, s, F)$

Important: $\Delta\colon Q \times (\Sigma \cup \{e\}) \to 2^Q$ is a set of possible resultant states

Union: $L = \{w\colon \#_a \equiv 0 \pmod 3 \vee w\colon \#_b \equiv 0 \pmod 4\}$



Not Equivalent Modulo: $a^n\colon n \not\equiv 0 \pmod{15}$
Equivalent of $L = \{w\colon \#_a \not\equiv 0 \pmod 3 \vee w\colon \#_a \not\equiv 0 \pmod 5\}$

Equivalent Modulo: $a^n\colon n \equiv 0 \pmod{15}$
This requires 15 states. Proof using pigeon hole principle.

Converting NFA to DFA: result DFA will have $\leq 2^n$ states

1. Remove e-transitions and create a new transition function Δ_1

$$\Delta_1(q, \sigma) = \bigcup_{0 \leq i, j \leq n} \Delta(q, e^i \sigma e^j)$$

2. Define DFA that recognizes NFA $M(Q, \Sigma, \Delta_1, s, F)$
   DFA $(2^Q, \Sigma, \delta, \{s\}, F')$ will keep track of the **set of states** the NFA could be in
   $\delta\colon 2^Q \times \Sigma = 2^Q$
   $\delta(A, \sigma) = \bigcup_{q \in A} \Delta_1(q, \sigma)$
   $F' = \{A\colon A \cap F \neq \emptyset\}$

Complement:
Complement with NFA is difficult because we can't flip normal/final states. Instead we have to

1. Convert n-NFA to 2^n-DFA
2. Take complement of DFA $\implies$ 2^n-DFA
3. So we have a 2^n NFA

NFA Capabilities:

- If NFA M exists and $L(M) = L$, then L is regular

3 Regex

1. Base Case: contains e and $\sigma \in \Sigma$
2. If α and β are regular, then $\alpha \cup \beta$ and $\alpha\beta$ are regular
3. α is regular then α^* is regular

$L(\alpha)$ is the set of strings generate from a regex α

Proof Regex $\subseteq$ NFA
Base Case: e and $\{\sigma\}$ have NFAs
IH: Assume for every regex β with $|\beta|$ ¡ n, $L(\beta)$ is recognized by an NFA
IS: Show α is a regex, with $|\alpha| = n$

- Case 1: $\alpha = \alpha_1 \cup \alpha_2$. Since $|\alpha_1|, |\alpha_2 < n$, we can apply IH and generate NFAs N_1 and N_2 such that $L(N_1) \cup L(N_2) = L(\alpha)$
- Case 2: similar for $\alpha = \alpha_1 \circ \alpha_2$
- Case 3: similar for $\alpha = \alpha_1^*$

Thus, regex $\subseteq$ NFA $\subseteq$ DFA

Proof DFA $\subseteq$ Regex
For the sake of the proof, we can extend $\delta\colon Q \times \Sigma^* \to Q$ to handle strings
so $\delta(q, w =$ state we end up at if we start at q and input w

Key idea is for every pair of states (i, j), we find the regex that represents the string that takes from state i to state j
$R(i, j, k) = \{w\colon \delta(i, w) = j$ using only states $\{1, \ldots, k\}\}$

Base case: $R(i, j, 0)$ for $1 \le i, j \le n$. Strings with no intermediary state so a single transition or $i = j$ and string is e

$$R(i,j,0) = \begin{cases} \{\sigma\colon \delta(i,\sigma) = j\} & i \neq j \\ \{\sigma\colon \delta(i,\sigma) = j\} \cup \{e\} & i = j \end{cases}$$

IH: Assume for $1 \le i, j \le n$, $R(i, j, k-1)$ is a regex
IS: Prove for all $1 \le i, j \le n$, $R(i, j, k)$ is a regex

$$R(i,j,k) = R(i,j,k-1) \cup R(i,k,k-1) \circ R(k,k,k-1)^* \circ R(k,j,k-1)$$

Capabilities of Regex

- Regex can't cleanly represent complement, although it is regular: idea is to convert n-NFA to 2^n-DFA, take the complement, then convert to a 2^{2^n} regex
- Regex can't cleanly represent intersection, although it is regular: idea is to convert to an NFA then convert back to regex

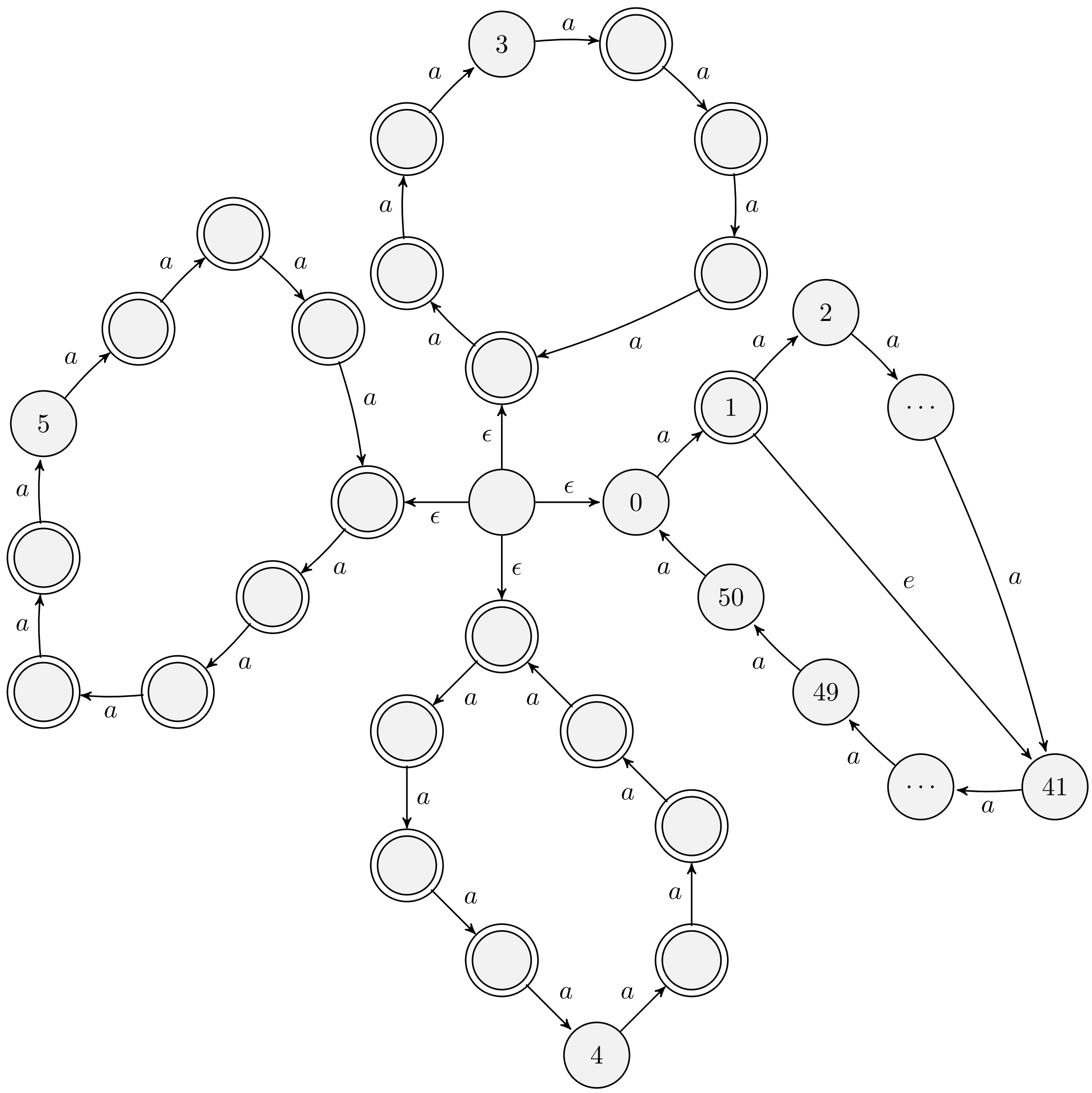
3.1 Trex

1. Base Case: contains e and $\sigma \in \Sigma$
2. If α and β are regular, then $\alpha \cup \beta$ and $\alpha\beta$ are regular
3. α is regular then α^* is regular
4. If α is a trex and $n \in \mathbb{N}$ then α^n takes $O(\lg n)$ space

4 Number of States

Small NFA: $L = \{a^i : i \neq 500\}$

- For $i \geq 501$, use
  Frobenius Theorem: for all $z \geq xy - x - y + 1$ there is $c, d \in \mathbb{N}$ such that $z = cx + cy$
  For all $z \geq 500 = 51 * 11 - 51 - 11 + 1$, there is a $c, d \in \mathbb{N}$ such that $z = cx = cy$
  Thus, $z + 1 \geq 501 = 51 * 11 - 51 - 11 + 2$ and we create an NFA for this
- For $i \leq 499$, use the following property of coprimes:
  Let $\{q_1, \ldots, q_k\}$ be a set of coprimes such that $\prod_{i=1}^{k} q_i \geq n$
  Then the set of i such that:
  $i \not\equiv n \pmod{q_1}$
  $\ldots$
  $i \not\equiv n \pmod{q_k}$
  Contains $\{1, \ldots, n-1\}$ but does not contain n
  This will take $O((\log n)^2 \log \log n)$ to represent using NFAs



Above, $\pmod 7$. Below $\pmod 8$. Left $\pmod 9$

Proof $\Sigma^* a \Sigma^n$ Requires 2^{n+1} DFA States

Number of States DFA/NFA/Regex:

Closure Property	DFA	NFA	Regex
$L_1 \cup L_2$	$n_1 n_2$	$n_1 + n_2 + 1$	$L_1 + L_2$
$L_1 \cap L_2$	$n_1 n_2$	$n_1 n_2$	X
$L_1 \circ L_2$	X	$n_1 + n_2$	$L_1 + L_2$
$\overline{L}$	n	X	X
L^*	X	$n + 1$	$L + 1$

5 Pumping Lemma

If L is regular then there exists n_0, n_1 such that for all $w \in L$ where $|w| \geq n_0$, there exists an x, y, z such that

- $w = xyz$ and $y \neq e$
- $|xy| \leq n_1$ (aka xy is short)
- for all $i \geq 0$, $xy^i z \in L$

To prove L is not regular need to find some i such that $xy^i z \notin L$

Example: $L = \{a^n b^n : n \in \mathbb{N}\}$ not regular
Let xy contain only a's so
$x = a^{m_1}$
$y = a^{m_2}$
$z = a^{n - m_1 - m_2} b^n$
Take $i = 2$ then

$$a^{m_1 + 2m_2 + n - m_1 - m_2} b^n = a^{n + m_1} b^n$$

which is clearly not in L

Example: $L = \{w \colon \#_a(w) \neq \#_b(w)\}$ not regular
Pumping Lemma doesn't work b/c there's not way of controlling the number of a output
However we know that $L_2 = \{w \colon \#_a(w) = \#_b(w)\}$ is not regular so we can take the complement of L_2, which will be not regular.

Example: $L = \{a^{n^2} : n \in \mathbb{N}\}$ not regular Let $x = a^{n_1}$, $y = a^{n_2}$, and $z = a^{n_3}$ so

$$a^{n_1} (a^{n_2})^i a^{n_3} \in L$$

So $\forall i \geq 0, n_1 + i n_2 + n_3$ is a square
$(n_1 + n_3) = x^2$
$(n_1 + n_3) + n_2 \geq (x+1)^2$
$\ldots$
$(n_1 + n_3) + n_i \geq x^2 + 2ix + i^2 \implies \frac{(n_1 + n_3)}{i} + n_2 \geq i$ so LHS decreases while RHS grows so this can't hold for all i

Example: $L = \{a^n b^m : n > m\}$ not regular
Revise pumping lemma to bound $|yz|$ then do pumping on the b's

Example: $L = \{a^{n_1} b^m c^{n_2}\}$ not regular
Let $w = a^n b^{n-1} c^n$ and $x = a^{n_1}, y = a^{n_2} z = a^{n - n_1 - n_2} b^{n-1} c^n$. Take $i = 0$. Then

$$xy^0 z = a^{n - n_1} b^{n-1} c^n$$

$\#_a$ on the left side is clearly $\leq$ than $\#_b$, which is $(n-1)$. Thus $xy^0 z \notin L$.

6 CFG (N, Σ, R, S)

Not CFL Examples

- $\{a^n b^n c^n\}$
- $\{a^{n^2}\}$
- If $L \subseteq a^*$ and L is not regular, then L is not context free
- $L_1 \cap L_2$
- $\overline{L}$

Proof Regex $\subseteq$ CFG
Base case $|\alpha| = 1$ then σ or e are both CFL's
IH: For regex β with $|\beta| < n$ there exists a CFG G such that $L(\beta) = L(G)$
IS: Take a regex α with $|\alpha| = n$
Case 1: $\alpha = \beta_1 \cup \beta_2$. By IH, β_1 and β_2 are CFL, and by closure under $\cup$, $L(\alpha)$ is a CFL
Case 2: Similar closure for $\alpha = \beta_1 \circ \beta_2$
Case 3: Similar closure for $\alpha = \beta^*$

Chomsky Normal Form

1. $A \to BC$ where $A, B, C \in N$
2. $A \to \sigma$ where $A \in N, \sigma \in \Sigma$
3. $S \to e$ where S is the start state

Difference between DFA, NFA, CFG Sizes: $L = \{a,b\}^* a \{a,b\}^n$
DFA requires $\Theta(2^n)$
NFA requires $n + \Theta(1)$
CFG requires $\Theta(\lg(n))$
CFG can be constructed by having $L = L_1 \circ L_2$ where

- $L_1 = \{a,b\}^* a$ which requires 5 rules:

  $A \to AS$

  $S \to BS$

  $S \to a$

  $A \to a$

  $B \to b$

- $L_2 = \{a,b\}^n$ which can be constructed in $\lg(n)$ rules assuming n is a power of 2:

  $S \to S_1 S_2$

  $S_1 \to S_2 S_2$

  $\ldots$

  $S_{\lg(n)} \to a$

  $S_{\lg(n)} \to b$