CSCE 355
1/27/2025

Prop: If $L_1, L_2 \subseteq \Sigma^*$ are regular lang's, then $L_1 \cap L_2$ is regular.

Proof use the product construction:

Given DFAs D_1 & D_2 such that

$$L(D_1) = L_1$$

$$L(D_2) = L_2$$

Build a DFA $D = D_1 \wedge D_2$ such that $L(D) = L_1 \cap L_2$. //

~~Def~~. REG_Σ is the class of all regular languages over alphabet Σ.

Cor: ~~REG~~ REG_Σ is closed under all Boolean set operations:

$$L_1 - L_2 = L_1 \cap \overline{L_2}$$

$$L_1 \cup L_2 = \overline{\overline{L_1} \cap \overline{L_2}}$$

$$L_1 \Delta L_2 = (L_1 - L_2) \cup (L_2 - L_1)$$ "symmetric difference"

$= (L_1 \cup L_2) - (L_1 \cap L_2)$

If L_1, L_2 are regular, then all of these are regular.

App of DFA: text search. Fix an alphabet Σ.
Fix some string $x \in \Sigma^*$ (the "search string")
Given $w \in \Sigma^*$ (the text document),
we want to decide if x occurs in w.

Def. $x, y \in \Sigma^*$.

x is a *prefix* of y if $\exists z \in \Sigma^*$, $y = xz$

x is a *suffix* of y if $\exists z \in \Sigma^*$, $y = zx$

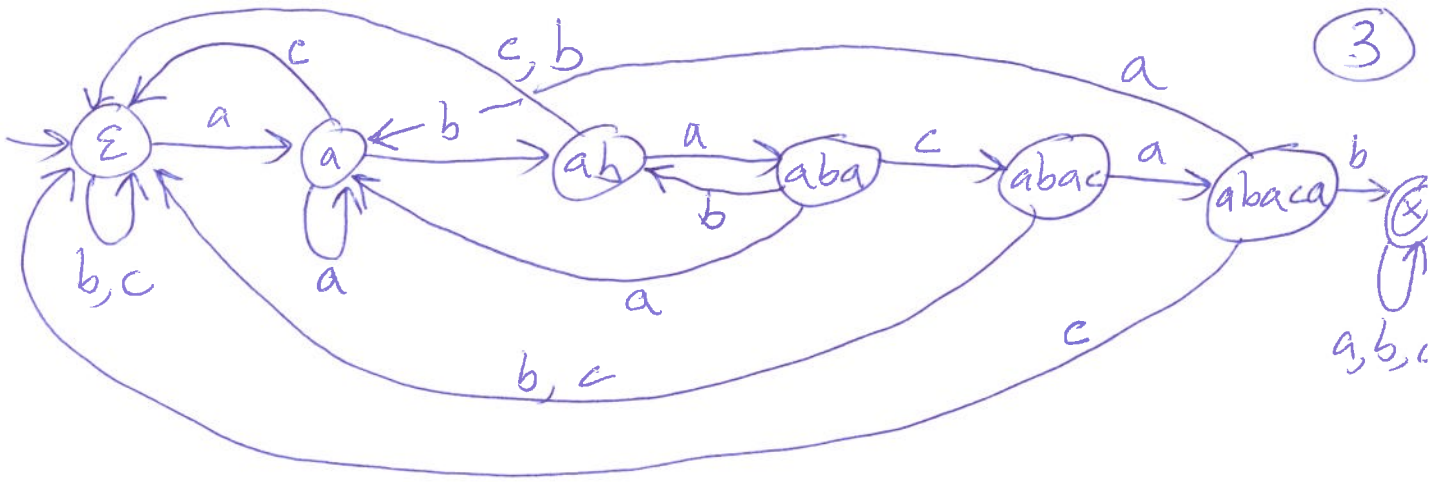
x is a *substring* of y if $\exists z, w \in \Sigma^*$, $y = wxz$

(equiv: x is a prefix of some suffix of y, or
" " " suffix " " prefix " ")

E: $\Sigma = \{a, b, c\}$, $x = abacab$

label each state with a ~~#~~ the corresponding prefix of x by reading from the start state.

Each label ~~repre~~ represents the longest prefix of x that is a suffix of what has been read so far.

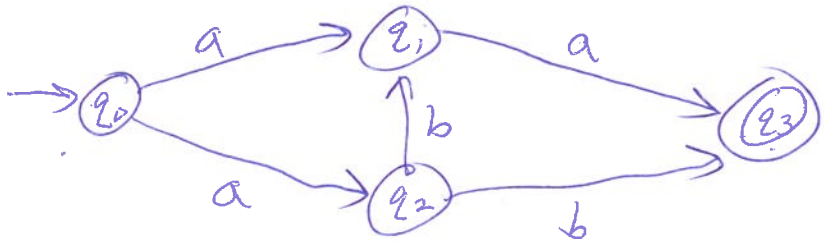


Knuth-Morris-Pratt algo builds a DFA given string x, to find x in w.

Nondeterminism

DFA: transition diagram requires exactly one edge out of any given state with any given label.

NFA (nondeterministic finite automaten):
no such restriction.

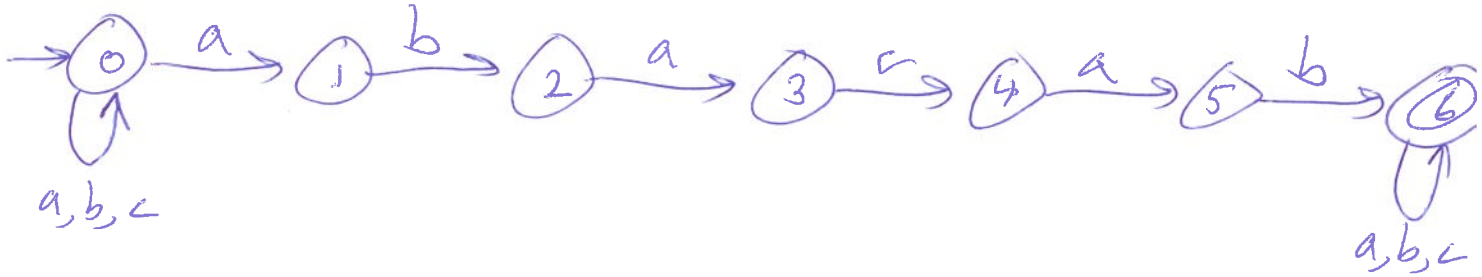


Accept iff there is some way to get from the start state to an accepting state by reading the entire input.

a — ✗
aa — ✓
aaa — ✗

✗ = reject
✓ = accept

NFA for text search: x = abacab



Accepts w iff w has x as a substring.

~~Simulate~~ Simulating an NFA efficiently:

~~Maintain~~

As the input chars are read, maintain a list (set) of all possible states the NFA could be in reading the input so far.

input	possible states
initially	0
a	0, 1
b	0, 2
a	0, 1, 3
b	0, 2
a	0, 1, 3
c	0, 4
a	0, 1, 5
b	0, 2, 6
b	0, 6

w = ababacabb

Accept if last list includes an accepting state

Ex: $\Sigma = \{0,1\}$

$L = \{w \in \Sigma^* :$ the 3rd to last symbol of x is 1$\}$

NFA for L:

Theorem: For any NFA N, there is a DFA D such that $L(D) = L(N)$.

(D and N are equivalent)

Idea: Given NFA N, the states of the equiv. DFA are subsets of the state set of N.