CSCE 355
1/12/2026

$\underbrace{\underbrace{\text{https://cse.sc.edu/\textasciitilde fenner}}_{\text{my homepage}}\text{/csce355}}_{\text{class homepage}}$ ①

3 topics:

1. Automata & regular languages
2. Context-free languages
3. Universal computation, incl. decidability/undecidability

Alphabets & strings

Def: An alphabet is any nonempty finite set.

If Σ is an alphabet, then we call elements of Σ symbols or letters.

Ex: $\Sigma = \{0\}$ unary alphabet

$\Sigma = \{0,1\}$ the binary alphabet

$\Sigma = \{0,\dots,n-1\}$ the ~~#~~ n-ary alphabet.

Def: Let Σ be an alphabet. A string over Σ is any finite sequence of symbols from Σ. (0 or more)

Ex: $\Sigma = \{a, b, c\}$ strings: a
acb
ccbaca
ε (empty string)

Def: Let x be a string over alphabet Σ.

$|x|$ denotes the # of symbols in x (the length of x)

Ex: $|a| = 1$ $\qquad$ $|\varepsilon| = 0$ (unique string of length 0)

$|acb| = 3$

$|ccbaca| = 6$

Def: For alph. Σ, Σ^* denotes the set of all strings over Σ.

Def: Let x and y be strings over Σ.

1. The concatenation of x (followed by) (and) y is ~~the string~~ denoted xy.

   If $x = a_1 \cdots a_n$, $y = b_1 \cdots b_m$ $\quad$ ($a_1, \ldots, b_1, \ldots \in \Sigma$)

   $xy = a_1 \cdots a_n b_1 \cdots b_m$

[Note: Identify Σ with the strings in Σ^* of length 1.

2. Concatenation is not commutative:

   $xy \neq yx$ for some x, y

   but it is associative: $(xy)z = x(yz)$

3. $|xy| = |x| + |y|$]

2. x is a <u>prefix</u> of y means that there exists $z \in \Sigma^*$, $y = xz$

3. x is a <u>suffix</u> of y if $\exists z \in \Sigma^*$, $y = zx$

4. x is <u>substring</u> of y if x is a prefix of some suffix of y, equiv.

$$\exists w, z \in \Sigma^*, y = wxz.$$

<u>String induction:</u>

<u>Idea</u>: use a ~~fab~~ fact about shorter strings to prove ~~a~~ the fact about longer strings.

<u>Basic Principle</u>: Let $x \in \Sigma^*$ be any string over Σ. Exactly one of the following holds:

a) $x = \varepsilon$

b) $x = ya$ ~~where~~ for unique strings y & a with $|a| = 1$ $(a \in \Sigma)$ and $|y| = |x| - 1$.

Call a the <u>last symbol</u> of x ~~ang~~ and y the <u>principal prefix</u> of x

Two uses of string induction:

1. Let ~~P(x) be some~~ $P(x)$ be some assertion about string x.

   Establishing $P(\varepsilon)$

   and $P(y) \to P(ya)$ $\forall a \in \Sigma$

   suffices to conclude that $P(x)$ holds for all $x \in \Sigma^*$.

2. Can uniquely define a function $f: \Sigma^* \to ?$ if you can define

   $f(\varepsilon)$ outright

   and if $\forall a \in \Sigma$ and string $y \in \Sigma^*$ can define $f(ya)$ in terms of y, a, and $f(y)$, then you have defined f uniquely.

Ex: $\Sigma = \{a, b, c\}$. ~~For~~ For $x \in \Sigma^*$ want

$f(x) = $ # of a's that occur in x.

$$\boxed{\begin{aligned} & f(\varepsilon) := 0 \\ \forall y \in \Sigma^* \quad & f(ya) := f(y) + 1 \\ & f(yb) := f(y) \\ & f(yc) := f(y) \end{aligned}}$$

$f(x)$ = longest suffix of x consisting of just a's

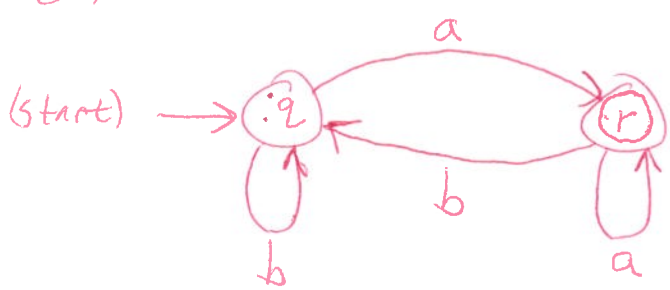
$$\boxed{\begin{aligned} & f(\varepsilon) := 0 \\ \forall y: \quad & f(ya) := f(y)+1 \\ & f(yb) := 0 \\ & f(yc) := 0 \end{aligned}}$$

Deterministic Finite Automata (DFAs)

(pl. of automaton)

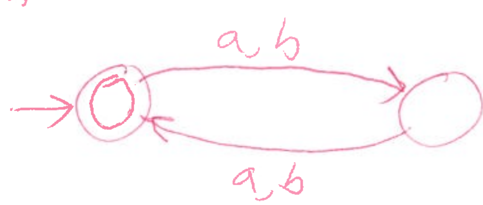
$\Sigma = \{a, b\}$

"x ends with a"

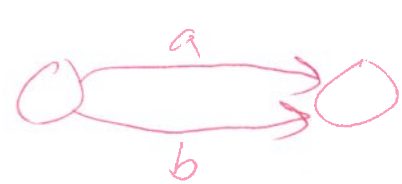


Input: x = abbab   end in state r iff x ends in a.

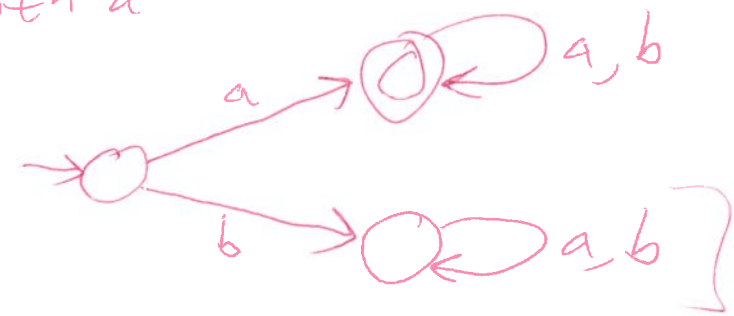
"x has even length"



is shorthand for

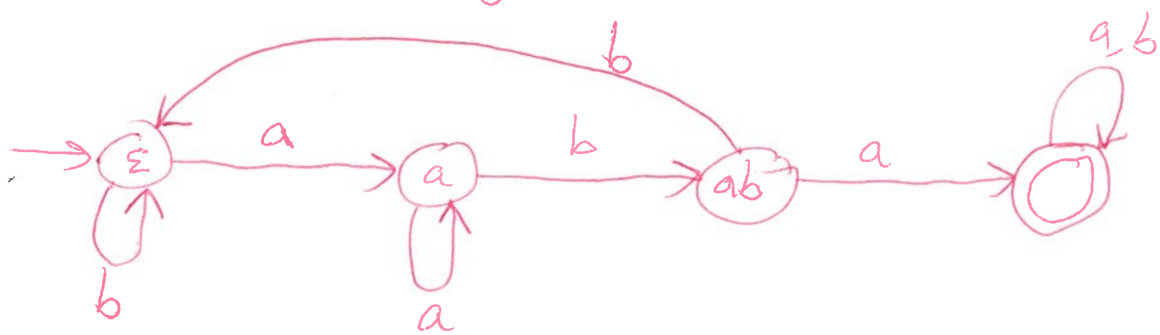


"x starts with a"



dead state

"x has aba as a substring"



Def: Σ alphabet. A language over Σ is any set of strings over Σ.

Def: Let A be a DFA & let x be string. A accepts x if A ends in an accepting state upon reading all of x. Otherwise A rejects x.

The language of A (denoted $L(A)$) is the set of strings accepted by A.
(The language recognized by A.)