CSCE 355
1/31/2022

Today: DFA minimization

Given a DFA $A = \langle Q, \Sigma, \delta, q_0, F \rangle$

1) Remove any state of A that is not reachable from the start state.

Result: $\cancel{\forall w \in \Sigma^*}$ $\forall q \in Q \; \exists w \in \Sigma^*,$

$$\hat{\delta}(q_0, w) = q$$

[q is "reachable"]

we will call such a DFA sane.

2) Merge states that are indistinguishable from each other.

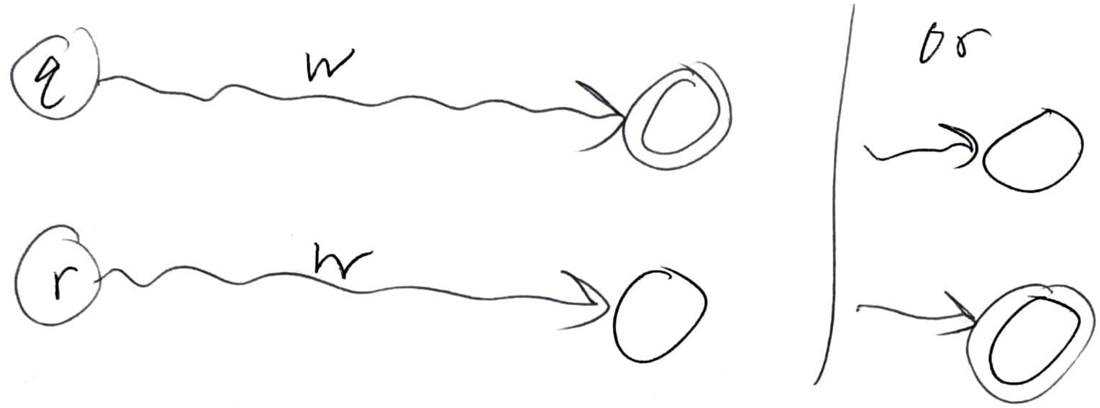
Def: Let $q, r \in Q$ be states of A.

A string w dististinguishes q from r if one of $\hat{\delta}(q, w)$ and $\hat{\delta}(r, w)$ is is accepting and one is rejecting.

q and r are distinguishable if there exists a string that distinguishes them.

② (Otherwise indistinguishable)

w dist. q from r



Program: find all pairs of dist. states.
Any pairs left over are indist.

~~Theorem:~~

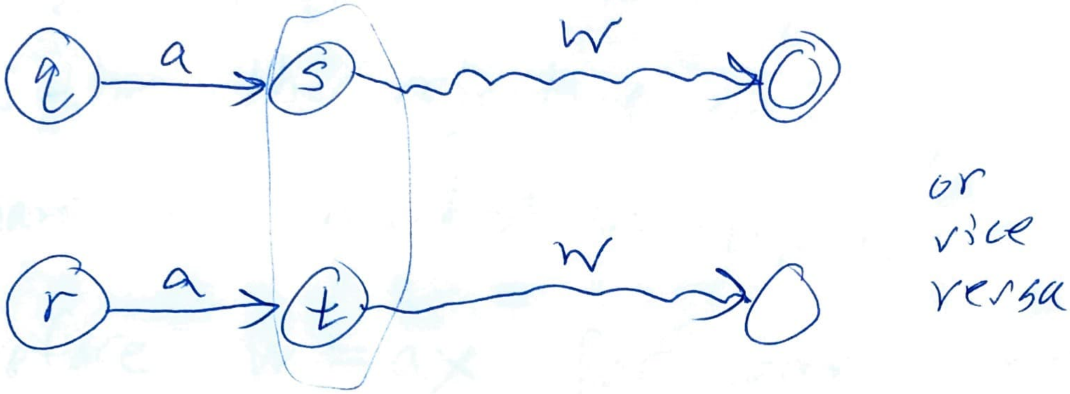
Dist Pairs (A): // A is sane

1) $\forall q, r \in Q$, if one is accepting & the other rejecting, then mark this pair $\{q, r\}$ as distinguishable. // dist. by ε

2) While there exists a pair $\{q, r\}$ of states unmarked and $\exists a \in \Sigma$,
→ $\{\delta(q,a), \delta(r,a)\}$ is marked,
then mark $\{q, r\}$

end-while

(3)



$\{q, r\}$ are dist. by the string aw

~~// Each iteration that~~

// If a pair $\{q, r\}$ is ever marked
// then q & r are dist. (Soundness)
// Proof by induction on the number of
// iterations of the while-loop

<u>Theorem</u>: Any pair of states left unmarked by the algo above are indistinguishable.

<u>Proof</u>: Suppose not, i.e, there exist a pair of states that are dist. but unmarked.

Choose a string $w \in \Sigma^*$ of minimum length such that there exist a pair of states q and r dist by w but $\{q, r\}$ is unmarked.

④ $-w \neq \varepsilon$. Otherwise one of q, r is accepting & the other rejecting, so $\{q, r\}$ would have been marked in Step 1.

Therefore $w = ax$ for some $a \in \Sigma$ and some $x \in \Sigma^*$ and $|x| = |w| - 1 < |w|$ (x is shorter than w).

$ax = w$ dist. q + r

So x distinguishes $\delta(q, a)$ from $\delta(r, a)$

By choice of ~~...~~ w (min length)

The algo marks the pair $\{\delta(q,a), \delta(r,a)\}$ at some point. But then at some later point the algo marks $\{q, r\}$ ↯

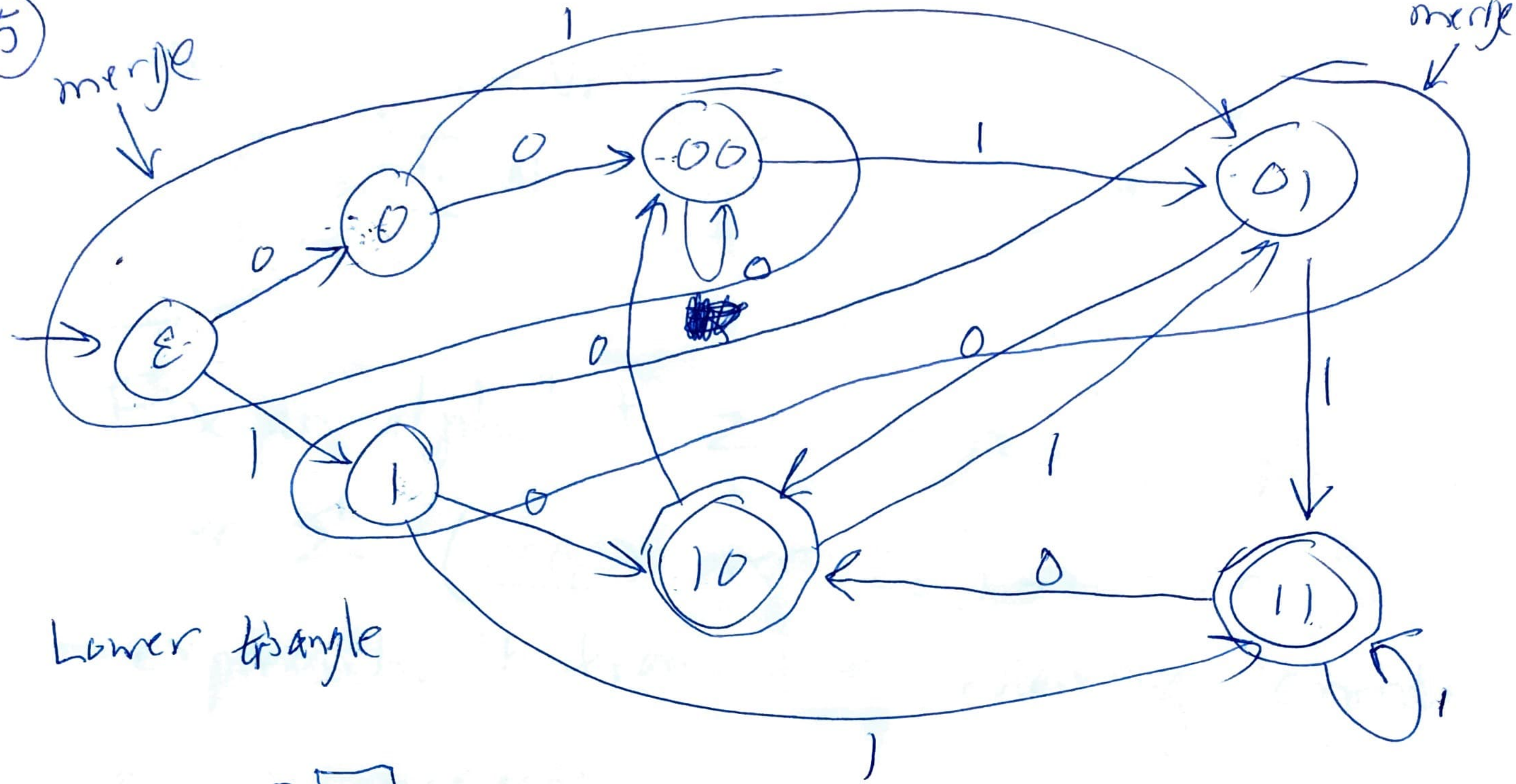
∴ Every dist. pair gets marked by the algo. ☑

("Completeness" or "Adequacy")

Ex: Recall the DFA recognizing

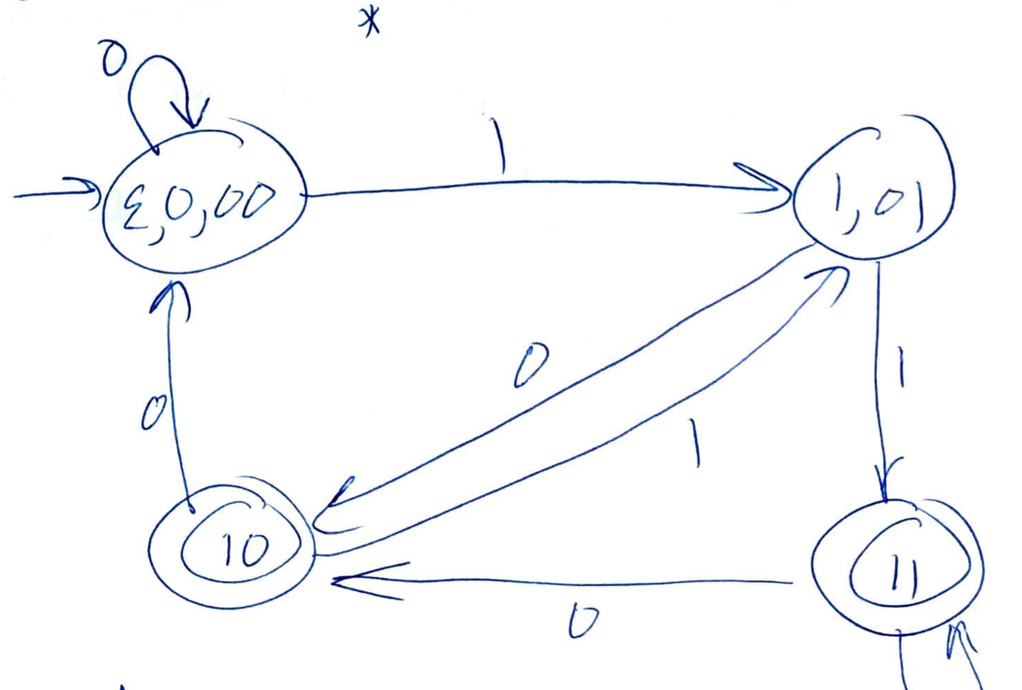
$L = \{ w \in \{0,1\}^* :$ the 2nd last digit of w is $1 \}$

5



Lower triangle

0						
1	X	X				
00			X			
01	X	X		X		
* 10	X	X	X	X	X	
* 11	X	X	X	X	X	X
	ε	0	1	00	01	10 *



Myhill-Nerode Theorem

6

$$\begin{tikzcd}
\varepsilon\text{-NFA} \arrow[r] \arrow[d, bend left] & \text{NFA} \arrow[l, bend left, "\text{trivial}"] \arrow[r] & \text{DFA} \arrow[l, bend left, "\text{trivial}"] \arrow[r] & \text{min DFA} \arrow[l, bend left, "\text{trivial}"] \\
\text{regular expression} \arrow[u, bend left] & & &
\end{tikzcd}$$

Fix an alphabet Σ. A regular expression over Σ (regex over Σ) is an expression built from the following elements:

Atomic regexes:

$\emptyset$

a (for all $a \in \Sigma$)

Nonatomic regexes: Let s & t be regexes. The following are also regexes:

$s + t$ (union of s with t)

st (concatenation of s with t)

s^* ($*$-closure, Kleene closure)

Syntax: Can use parentheses for grouping if necessary. But operator precedence may allow you to drop parens:

$+$ — lowest precedence (associative & commutative)

$$(s+t)+u = s+(t+u) = s+t+u$$

$$s+t = t+s$$

~~st~~ concat — next higher (assoc.)

$*$ — highest precedence

A regex denotes a language over Σ:

atomic
- $\emptyset$ denotes the empty lang.
- a " " $\{a\}$

$s+t$ " union of langs of s and of t

st " concat " " " " " "

s^* union of 0 or more concats of s with itself.