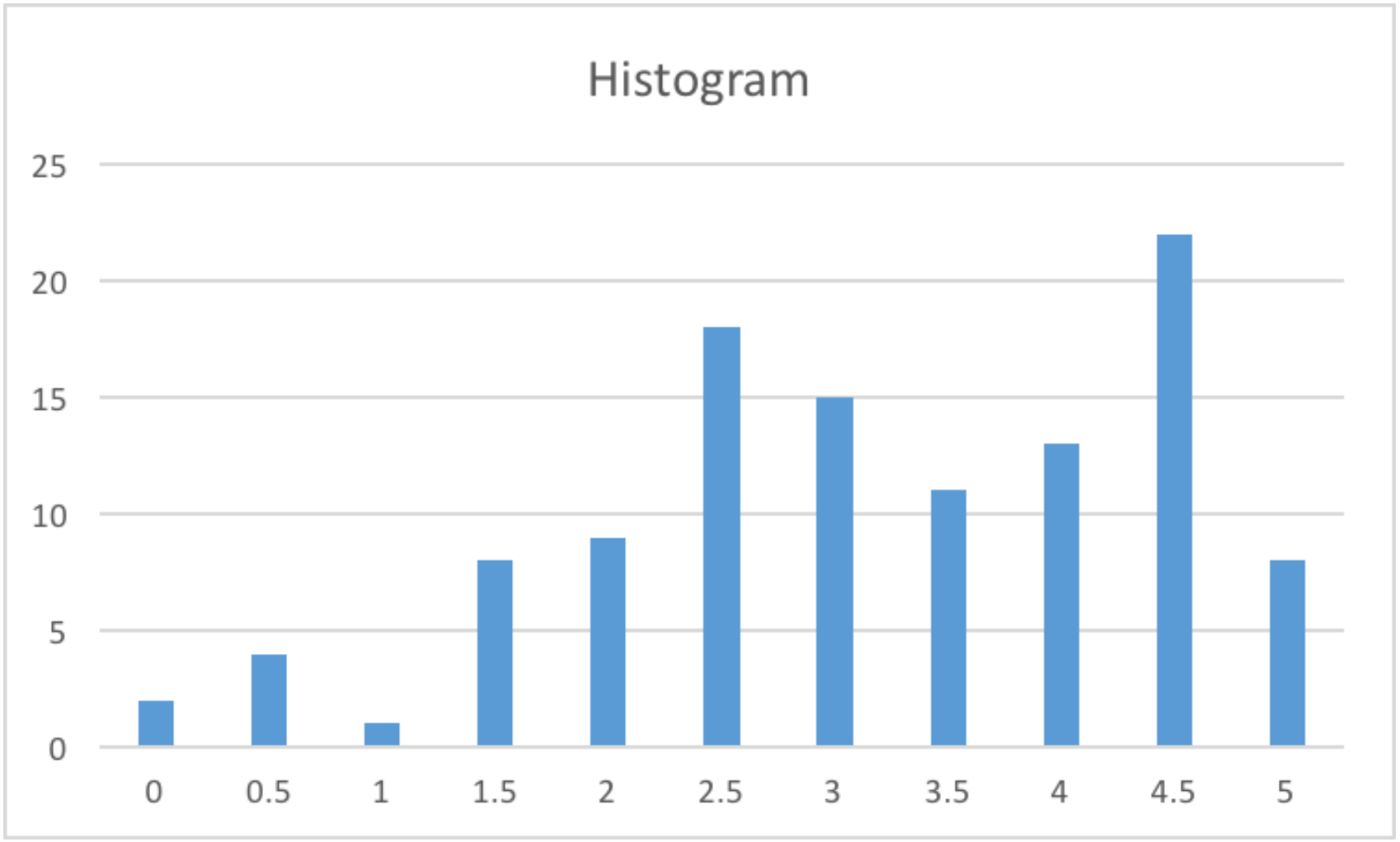
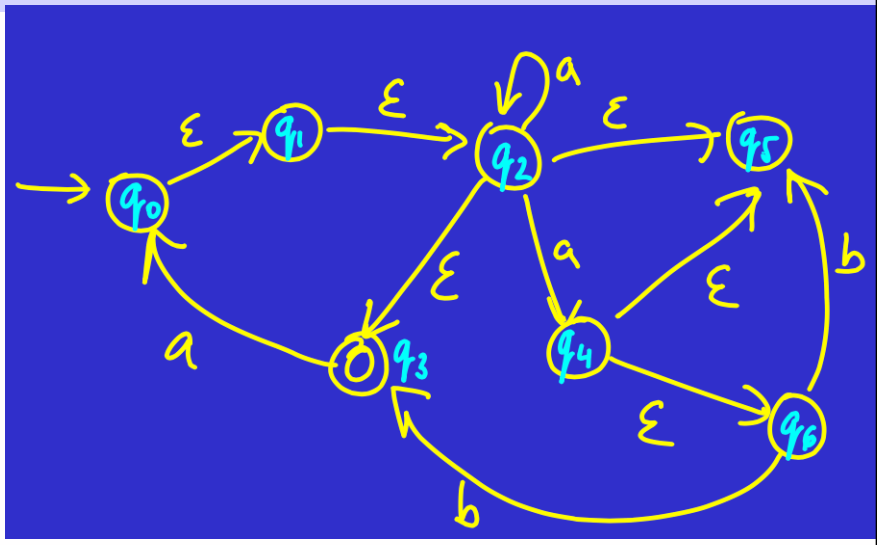
Quiz 3 distribution

- mean = 3.18, mode = 4.5, stddev = 1.25



Epsilon-Closure

- def. 0 (informal, Sipser):
  $E(q) = \{p \mid p$ is reachable from q by 0 or more ε edges$\}$
- def. 1 (default): $E(q)$ is the **smallest** set such that:
  - $q \in E(q)$
  - if $p \in E(q)$, then $\delta(p, \varepsilon) \subseteq E(q)$
- def. 2 (inductive): $E(q) = \cup_i E_i(q)$
  - $E_0(q) = \{q\}$
  - $E_{i+1}(q) = \cup_{p \in E_i(q)} \delta(p, \varepsilon)$
- from closure of state to closure of set:
  - $E(R) = \cup_{q \in R} E(q)$

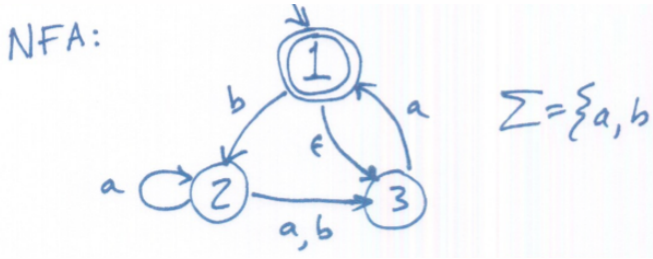


$E(q_0) = \{q_0, q_1, q_2, q_3, q_5\}$
$E(q_2) = \{q_2, q_3, q_5\}$

$E(\{q_2\}) = \{q_3, q_5\}$
$E(\{q_0\}) = \{q_0, q_1, q_2, q_3, q_5\}$
$E(\{q_1, q_4\}) = \{q_1, q_2, q_3, q_5, q_4, q_6\}$

Computation of NFA w/ epsilon

- string *w* is accepted iff. $\delta^*(q_0, w) \cap F \neq \varnothing$
- have to refine δ^* using epsilon-closure E(...)



without epsilon

$$\delta : Q \times \Sigma \mapsto P(Q)$$

$$\delta^* : Q \times \Sigma^* \mapsto P(Q)$$

$$\delta^*(q, \epsilon) = \{q\}$$

$$\delta^*(q, xa) = \bigcup_{p \in \delta^*(q,x)} \delta(p, a)$$

alternatively

$$\delta^*(q, ax) = \bigcup_{p \in \delta(q,a)} \delta^*(p, x)$$

with epsilon

$$\delta : Q \times (\Sigma \cup \{\epsilon\}) \mapsto P(Q)$$

$$\delta^* : Q \times \Sigma^* \mapsto P(Q)$$

$$\delta^*(q, \epsilon) = E(\{q\})$$

$$\delta^*(q, xa) = E\Big(\bigcup_{p \in \delta^*(q,x)} \delta(p, a) \Big)$$

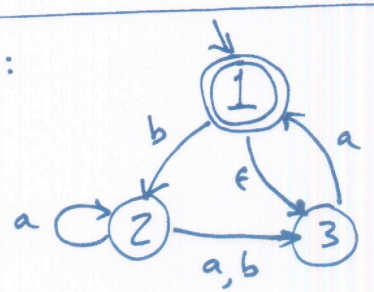
alternatively

$$\delta^*(q, ax) = \bigcup_{p \in E\left(\delta(q,a)\right)} \delta^*(p, x)$$

Example: NFA=>DFA w/ epsilon

EXAMPLE: CONVERT AN NFA TO A DFA.

NFA:



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

What states in the DFA?

$Q' = \{ \quad \} = P(Q)$

DFA:

$\emptyset$ 1 2 3 12 13 23 123

Start State?

$q_0' = E(\{1\}) = \{1, 3\}$

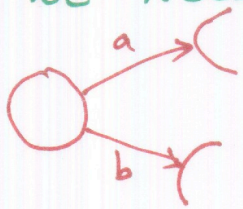
$E(1) = 13$

ACCEPT STATES? ANY THAT CONTAIN 1

$F' =$ 1 12 13 123

TRANSITION FUNCTION?

FOR EACH STATE, WE NEED TWO EDGES:



$\delta'(1, a) = \emptyset$ Since no edges labeled "a" leave 1

$\delta'(1, b) = 2$ From 1 we can get to 2, but no further with ε-edges

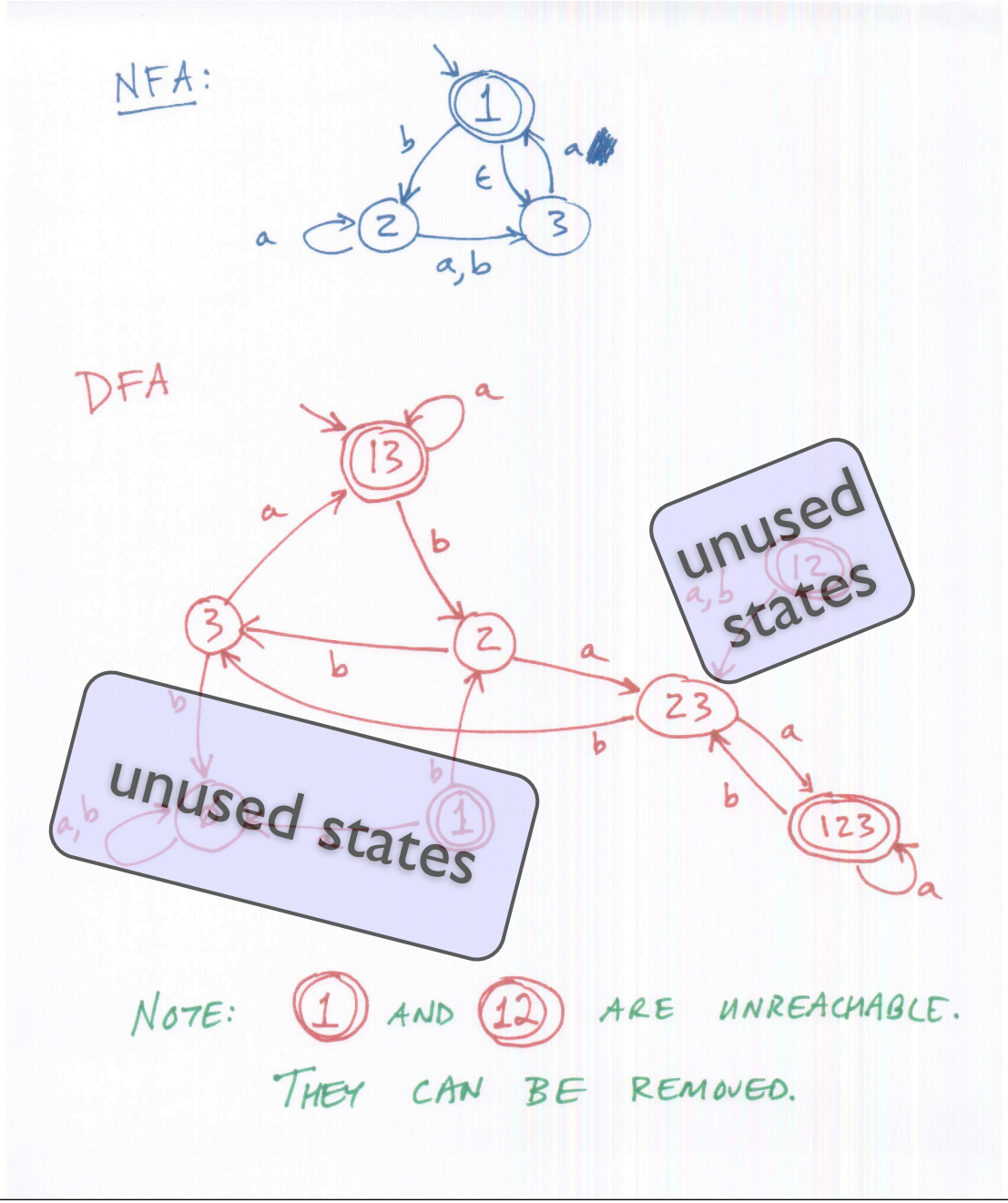
$\delta'(2, a) = 23$

$\delta'(2, b) = 3$

$\delta'(3, a) = 13$ Can get to 1 but can get to 3 by following ε-edge.

$\delta'(3, b) = \emptyset$

Example: NFA=>DFA w/ epsilon



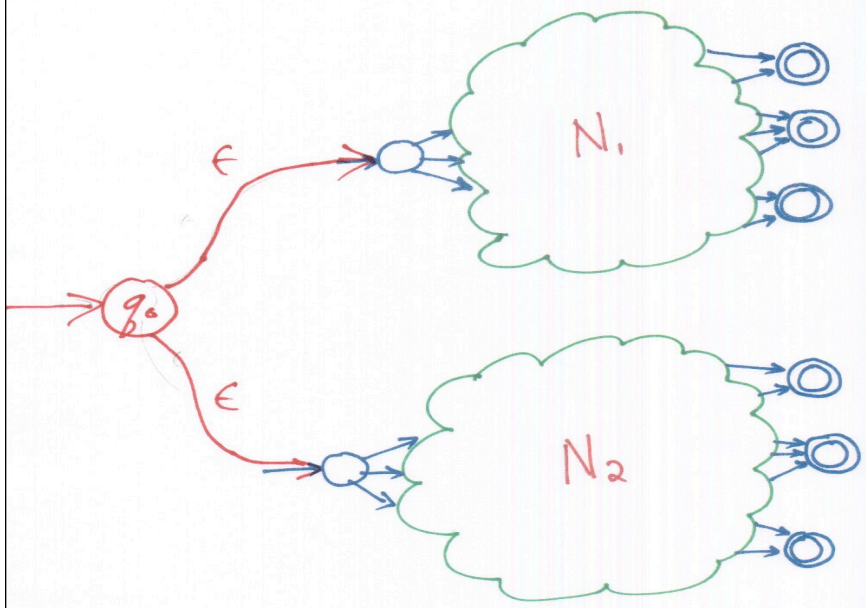
Closed under Union (with NFA)

THEOREM

THE CLASS OF REGULAR LANGUAGES IS "CLOSED" UNDER UNION.

"CLOSURE" OF A LANGUAGE.

PROOF



FORMALLY:

Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$

$N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$

CONSTRUCT

$N = (Q, \Sigma, \delta, q_0, F)$

$Q = Q_1 \cup Q_2 \cup \{q_0\}$

q_0 is new start state

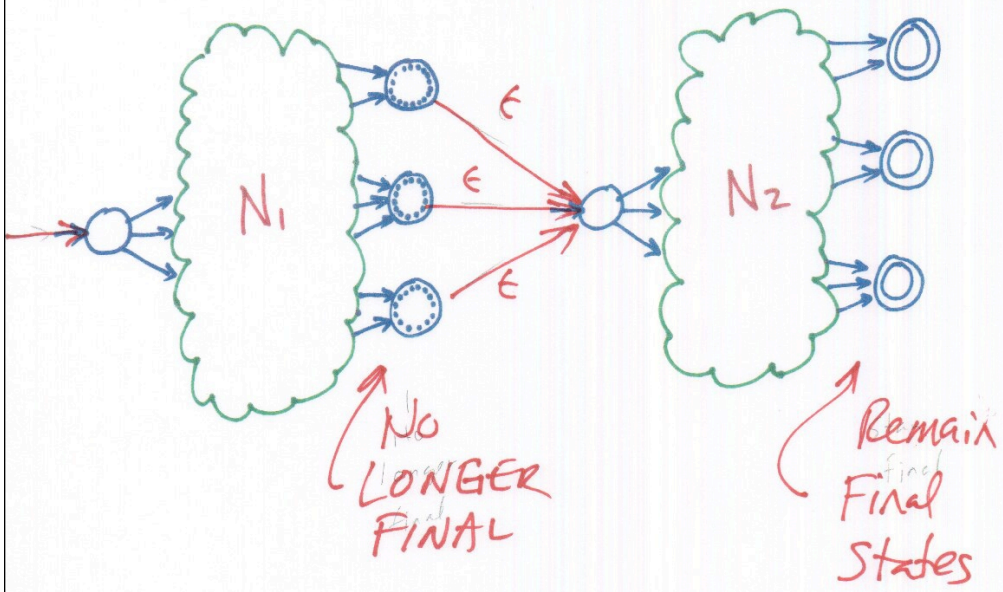
$F = F_1 \cup F_2$

$$\delta(q,a) = \begin{cases} \delta_1(q,a) & \text{if } q \in Q_1 \\ \delta_2(q,a) & \text{if } q \in Q_2 \\ \{q_1, q_2\} & \text{if } q = q_0 \text{ and } a = \epsilon \\ \{\} & \text{if } q = q_0 \text{ and } a \neq \epsilon \end{cases}$$

Closed under Concatenation

THEOREM

THE CLASS OF REGULAR LANGUAGES IS CLOSED UNDER CONCATENATION.



FORMALLY

Let $N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$

$N_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$

Construct

$N = (Q, \Sigma, \delta, q_0, F)$

$Q = Q_1 \cup Q_2$

$q_0 = q_1$ the start state of N_1

$F = F_2$ the final states of N_2

$$\delta(q,a) = \begin{cases} \delta_1(q,a) & \text{if } q \in Q_1 \\ \delta_2(q,a) & \text{if } q \in Q_2 \\ \delta_1(q,a) \cup \{q_2\} & \text{if } q \in F_1 \text{ and } a = \epsilon \\ \delta_1(q,a) & \text{if } q \in F_1 \text{ and } a \neq \epsilon \end{cases}$$

Closed under Star

THEOREM

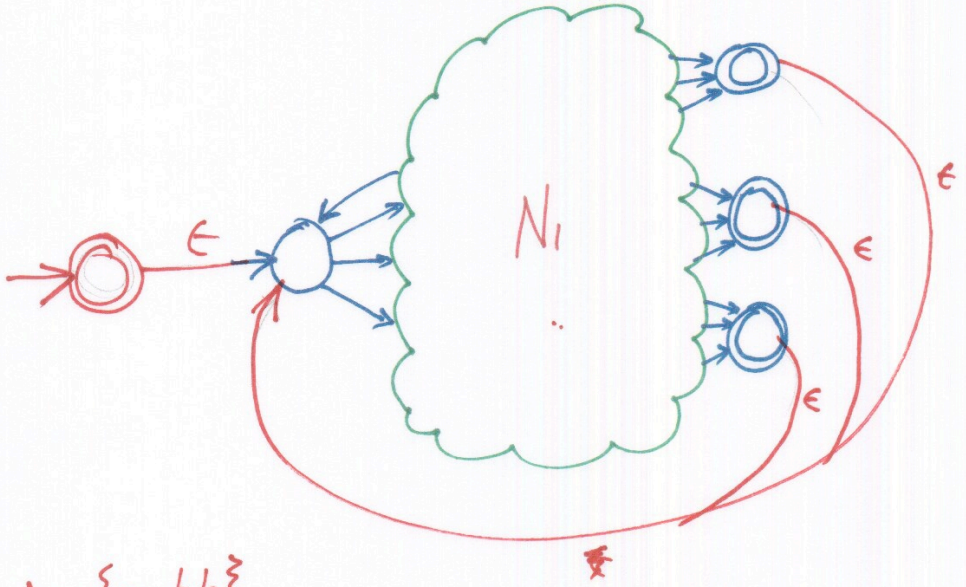
The class of regular languages is closed under "star."

CLOSURE

A^*

PROOF

Same idea.



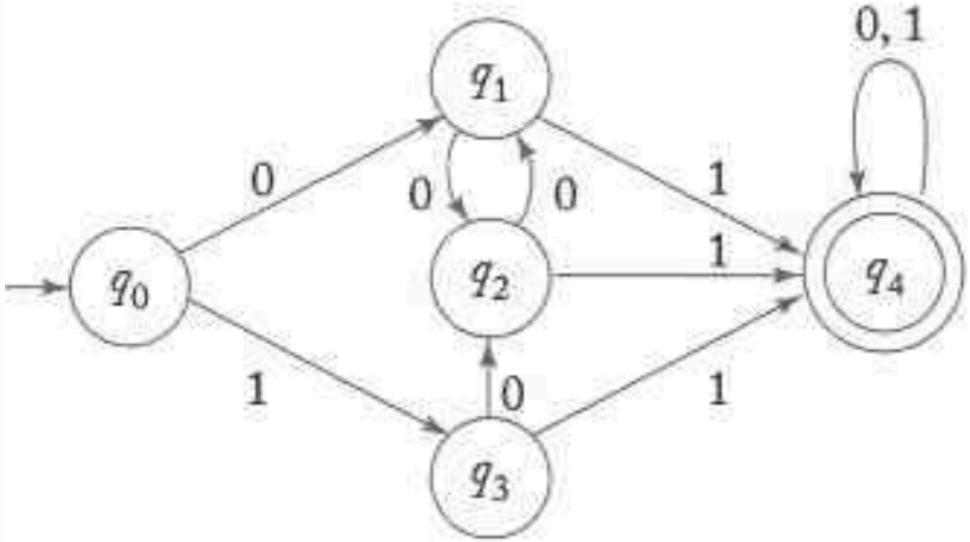
$A = \{a, bb\}$

$A^* = \{\epsilon, a, aa, aaa, bb, bbbb, abb, bba, bbaabbabb, \ldots\}$

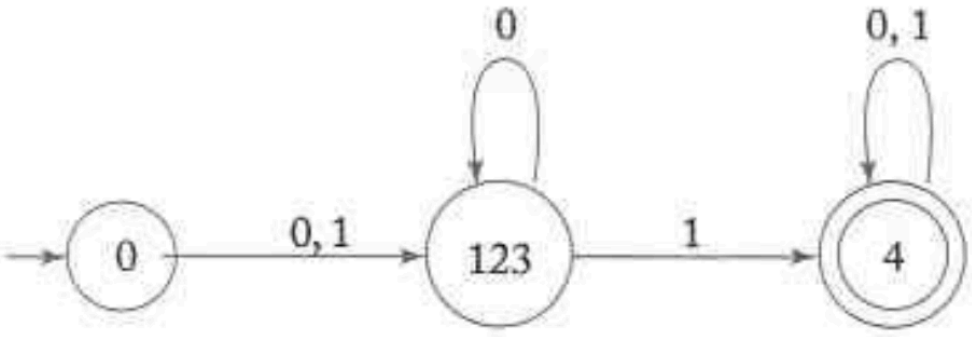
42

DFA minimization (Linz 2.4)

- NFA => DFA=> minimal DFA (b/c some nodes are equivalent or "mergeable")
- first compute equivalence classes of nodes
  - initially, two classes: F and Q - F, and keep splitting:
  - if $\delta(p, a)$ and $\delta(q, a)$ fall into different classes, then p and q aren't equivalent
- merge states in the same class, and form the minimal DFA



initial: F={q4}, A={q0,q1,q2,q3}
step 1: δ(q0, 1) in A
but δ(q1, 1)= δ(q2, 1)= δ(q3, 1) in F
so split: F= {q4}, A={q0}, B={q1,q2,q3}
no further splitting possible



this partition is unique.

Definition of Regular Expression

DEFINITION 1.52

Say that R is a ***regular expression*** if R is

1. a for some a in the alphabet Σ,
2. ε,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, where R_1 and R_2 are regular expressions,
5. $(R_1 \circ R_2)$, where R_1 and R_2 are regular expressions, or
6. (R_1^*), where R_1 is a regular expression.

also notated: $R_1 \mid R_2$ or $R_1 + R_2$

also $R_1 R_2$

let R^+ be shorthand for RR^*. $R^+ \cup \varepsilon = R^*$

$R \cup \emptyset = R$.
Adding the empty language to any other language will not change it.

$R \circ \varepsilon = R$.
Joining the empty string to any string will not change it.

$R \cup \varepsilon$ may not equal R.
For example, if $R = 0$, then $L(R) = \{0\}$ but $L(R \cup \varepsilon) = \{0, \varepsilon\}$.

$R \circ \emptyset$ may not equal R.
For example, if $R = 0$, then $L(R) = \{0\}$ but $L(R \circ \emptyset) = \emptyset$.

EXAMPLE 1.53

In the following instances, we assume that the alphabet Σ is $\{0,1\}$.

1. $0^*10^* = \{w \mid w \text{ contains a single } 1\}$.
2. $\Sigma^*1\Sigma^* = \{w \mid w \text{ has at least one } 1\}$.
3. $\Sigma^*001\Sigma^* = \{w \mid w \text{ contains the string } 001 \text{ as a substring}\}$.
4. $1^*(01^+)^* = \{w \mid \text{every } 0 \text{ in } w \text{ is followed by at least one } 1\}$.
5. $(\Sigma\Sigma)^* = \{w \mid w \text{ is a string of even length}\}$.[5]
6. $(\Sigma\Sigma\Sigma)^* = \{w \mid \text{the length of } w \text{ is a multiple of } 3\}$.
7. $01 \cup 10 = \{01, 10\}$.
8. $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1 = \{w \mid w \text{ starts and ends with the same symbol}\}$.
9. $(0 \cup \varepsilon)1^* = 01^* \cup 1^*$.
   The expression $0 \cup \varepsilon$ describes the language $\{0, \varepsilon\}$, so the concatenation operation adds either 0 or ε before every string in 1^*.
10. $(0 \cup \varepsilon)(1 \cup \varepsilon) = \{\varepsilon, 0, 1, 01\}$.
11. $1^*\emptyset = \emptyset$.
    Concatenating the empty set to any set yields the empty set.
12. $\emptyset^* = \{\varepsilon\}$.
    The star operation puts together any number of strings from the language to get a string in the result. If the language is empty, the star operation can put together 0 strings, giving only the empty string.

$$(+ \cup - \cup \varepsilon)(D^+ \cup D^+ . D^* \cup D^* . D^+)$$

where $D = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ is the alphabet of decimal digits. Examples of generated strings are: 72, 3.14159, +7., and -.01.

Examples

Parsing Practice

aa bUc aa bUc aa = ?

(aab) U (caab) U (caa)

---

aab | caab | caa

= (aab) | (caab) | (caa)

---

d U ab* cd* = ?

d | ab* cd* = ?

= (d) U (a(b*)c(d*))

= d | (a(b*)c(d*))

EXAMPLE REGULAR EXPRESSIONS

Assume $\Sigma = \{a, b, c, d\}$

a

$\{a\}$

abccb

$\{abccb\}$

ab U cd = ab | cd

$\{ab, cd\}$

a(bUc)d = a(b|c)d

$\{abd, acd\}$

ab*c

$\{ac, abc, abbc, abbbc, \ldots\}$

a(bUϵ)c = a(b|ϵ)c = a[b]c

$\{abc, ac\}$

$\emptyset$

$\{\}$

a(bUc)$\emptyset$

$\{\}$

$\emptyset$*

$\{\epsilon\}$

Regular Expression == Regular Language

Regular Expression

Each Regular Expression describes a Language. Which Language?

$L(R) = ?$

$L(a) = \{a\}$

$L(R_1 | R_2) = L(R_1) \cup L(R_2)$

$L(R_1 \circ R_2) = L(R_1) \circ L(R_2)$

$L(R_1^*) = L(R_1)^*$

$L(\epsilon) = \{\epsilon\}$

$L(\emptyset) = \{\}$

$L((R_1)) = L(R_1)$

Regular Languages are closed under Union, Concatenation, and Star.

$\Rightarrow$ Regular Expressions describe Regular Languages.

Theorem

A language is Regular iff some Regular Expression describes it.

Previous Definition

A language is Regular iff it is ~~described~~ recognized by some Finite State Machine. (NFA=DFA).

Given a Regular Expression, E, the language it describes

$L(E) =$ See previous slide

So we are saying that the class of languages recognized by DFA's, NFAs, and Reg. Expressions is the same! All have equivalent "power"!

Regular Expression == Regular Language

LEMMA 1:

IF A LANGUAGE IS DESCRIBED BY A REGULAR EXPRESSION, THEN IT IS REGULAR.

PROOF #1: USE THE CLOSURE OF $\cup$ $*$ and $\circ$.

PROOF #2: FROM A REGULAR EXPRESSION, BUILD AN NFA TO RECOGNIZE IT.

LEMMA 2:

IF A LANGUAGE IS REGULAR, THEN IT CAN BE DESRIBED BY A REGULAR EXPRESSION.

PROOF APPROACH:

- START WITH A DFA THAT RECOGNIZES IT.
- BUILD A GNFA (GENERALIZED NONDETERMINISTIC FINITE STATE AUTOMATON).
- REDUCE IT (details to follow).
- THIS YIELDS A REGULAR EXPRESSION.

INDUCTIVE PROOF! (STRUCTURAL INDUCTION)

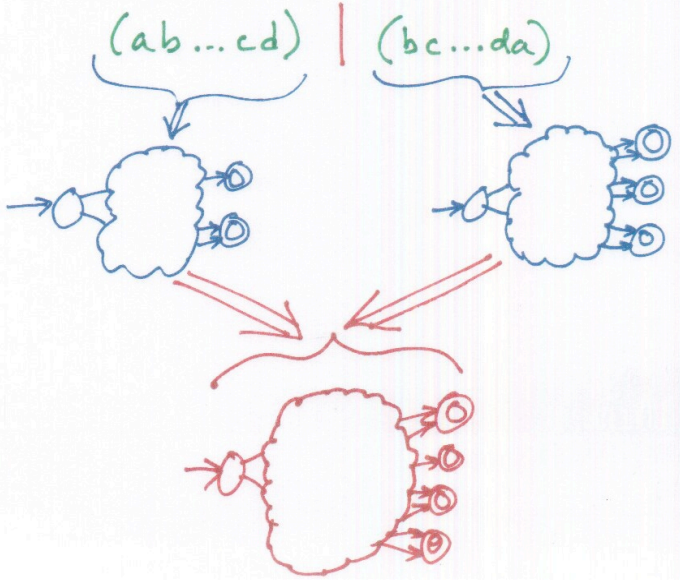
LEMMA 1

- WE HAVE A REGULAR EXPRESSION. CALL IT R.
- CONVERT R TO A NFA.
- CONCLUDE THAT THE LANGUAGE MUST BE REGULAR.

EVERY LARGE REGULAR EXPRESSION IS MADE OF SMALLER REGULAR EXPRESSIONS.

ASSUME WE CAN BUILD THE NFAs FOR SMALLER REGULAR EXPRESSIONS.

SHOW HOW TO BUILD THE NFA FOR LARGER REGULAR EXPRESSIONS.



RE => NFA

Definition of Regular Expressions

$R = a$

$R = R_1 | R_2$

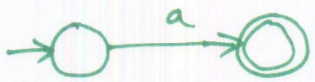
$R = R_1 \circ R_2$

$R = R_1^*$

$R = \epsilon$

$R = \emptyset$

$R = a$



$R = \epsilon$

$R = \emptyset$

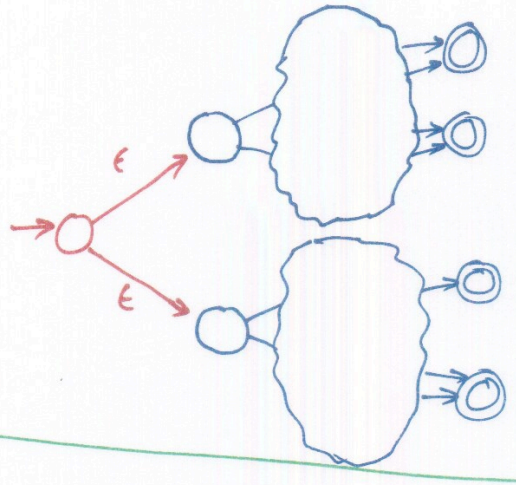
$R = R_1 | R_2$

$R = R_1 \circ R_2$

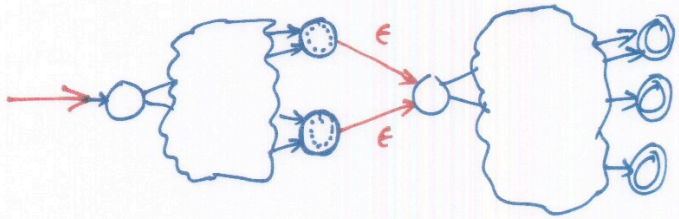
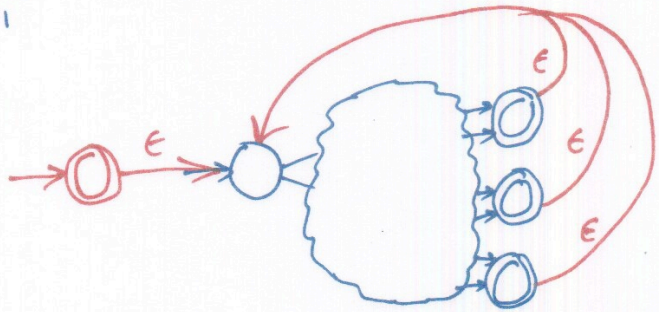
$R = R_1^*$

Same construction used in proof of closure.

$R_1 | R_2$



$R_1 \circ R_2$



R_1^*