Equivalence between RE and DFA/NFA

- The second part is,
- Given DFA/NFA convert this to equivalent RE.

- There are various ways for this.
  - The Ullman's book gives a rigorous algorithm
  - Same thing in essence is achieved by the Sipser's book in a different way.
    - We follow Sipser's book.

DFA/NFA ➔RE

- Go for GNFA (Generalized Nondeterministic Finite Automata)
- RE can be on arrows.

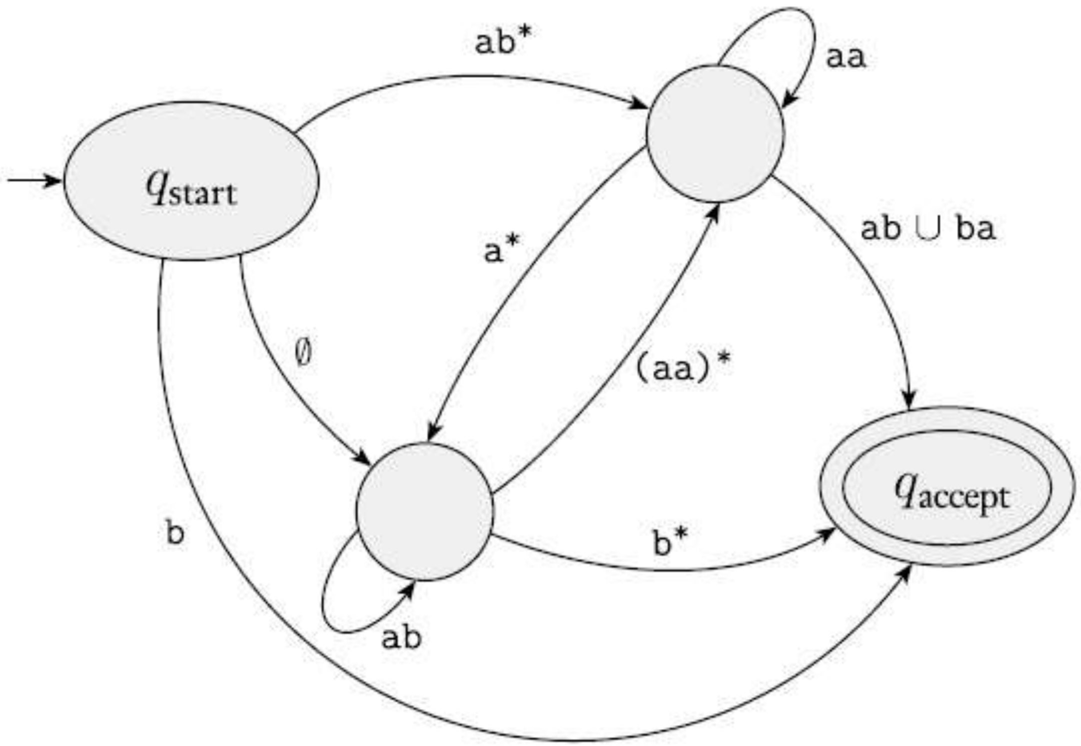
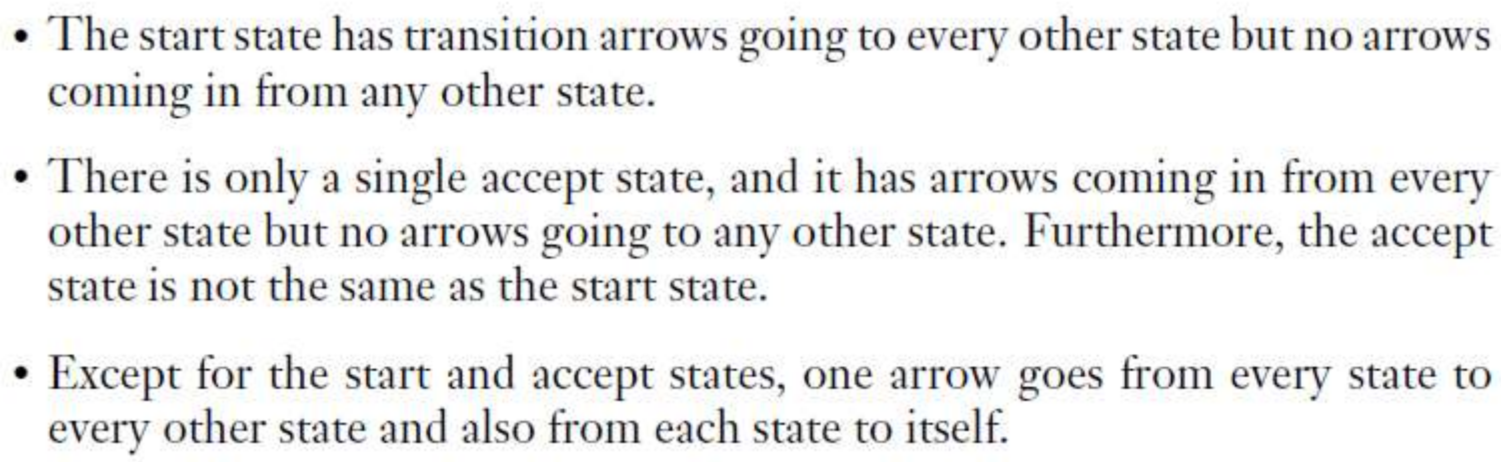


FIGURE 1.61
A generalized nondeterministic finite automaton

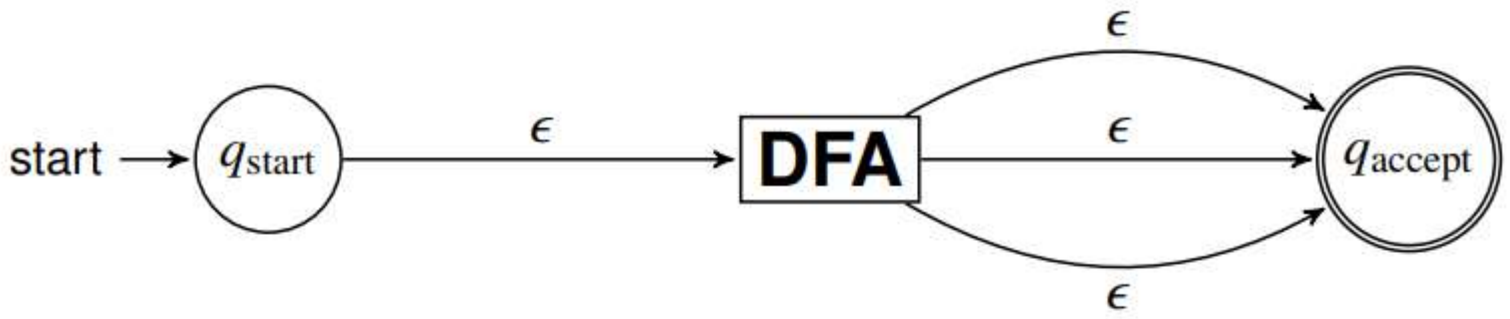
GNFA should -



- The start state has transition arrows going to every other state but no arrows coming in from any other state.
- There is only a single accept state, and it has arrows coming in from every other state but no arrows going to any other state. Furthermore, the accept state is not the same as the start state.
- Except for the start and accept states, one arrow goes from every state to every other state and also from each state to itself.

- For more details, consult the Sipser’s book.

Step 1: DFA to GNFA



- Add a new start state with an ε transition to the DFA start state.
- Add a new accept state with ε transitions from the DFA accept states to the new accept state. (Change the DFA accept states to non-accept states.)
- A transition should contain the union of the DFA transition labels.
- Add the Ø transition to pairs of states in the DFA that had no transition between them.

What we do?

- We convert the given DFA/NFA into a GNFA.
- Then, progressively we remove all states, one by one, except for the start and accept states.

Step 2: GNFA to RE

- For example we are given a 3 state DFA

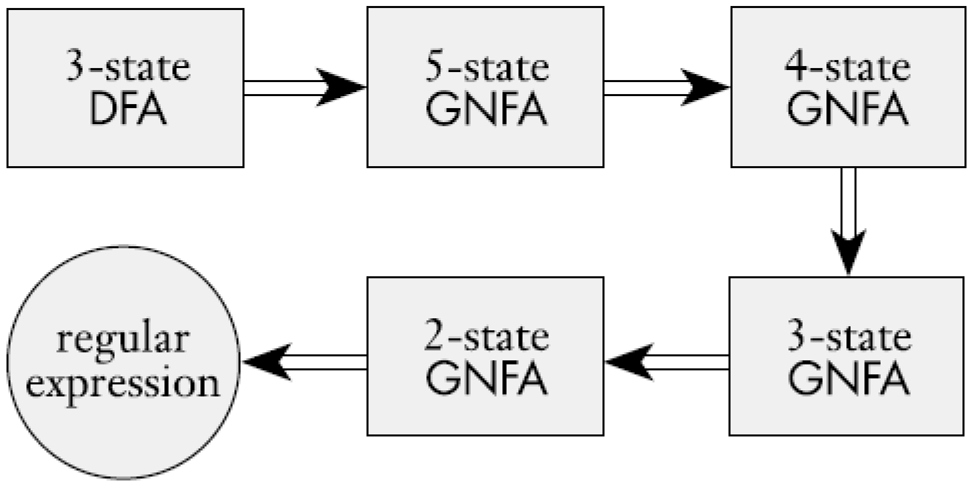
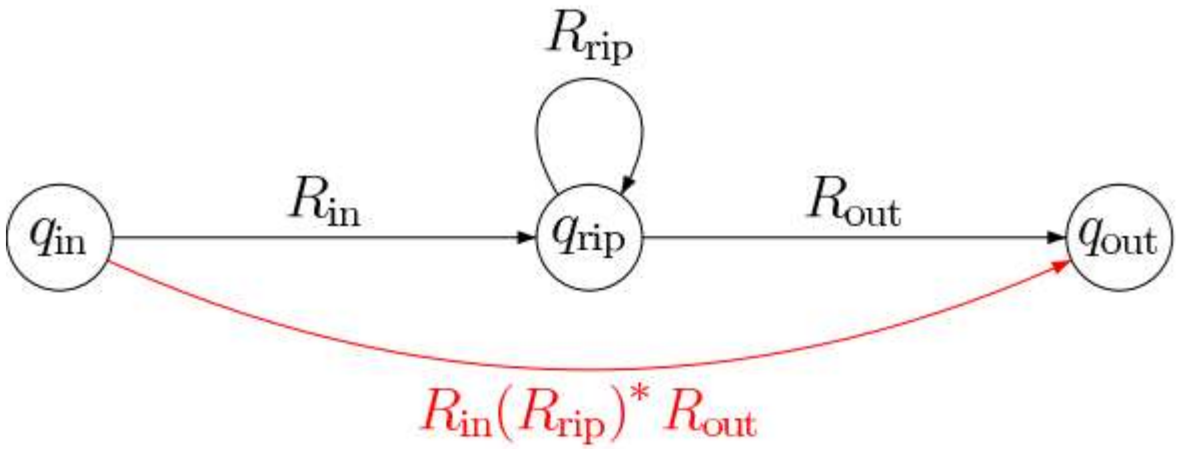


FIGURE 1.62
Typical stages in converting a DFA to a regular expression

Step 2: GNFA to RE

- While machine has more than 2 states:
  - Pick any internal state, rip it out
  - Re-label the arrows with regular expressions to account for the missing state



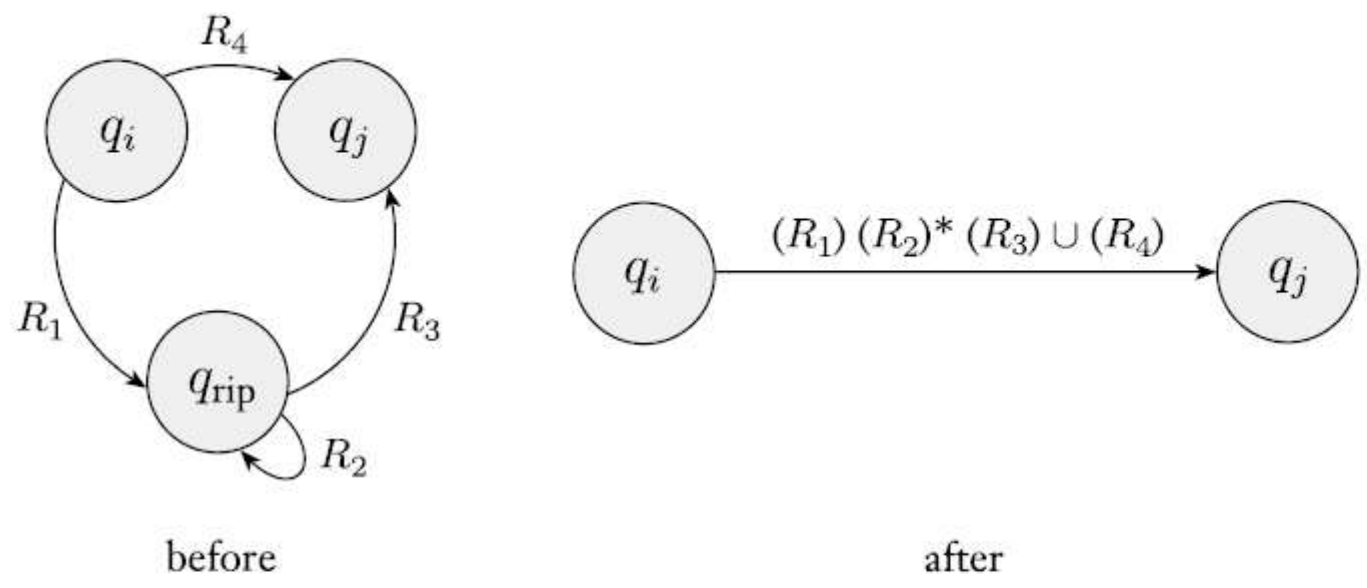
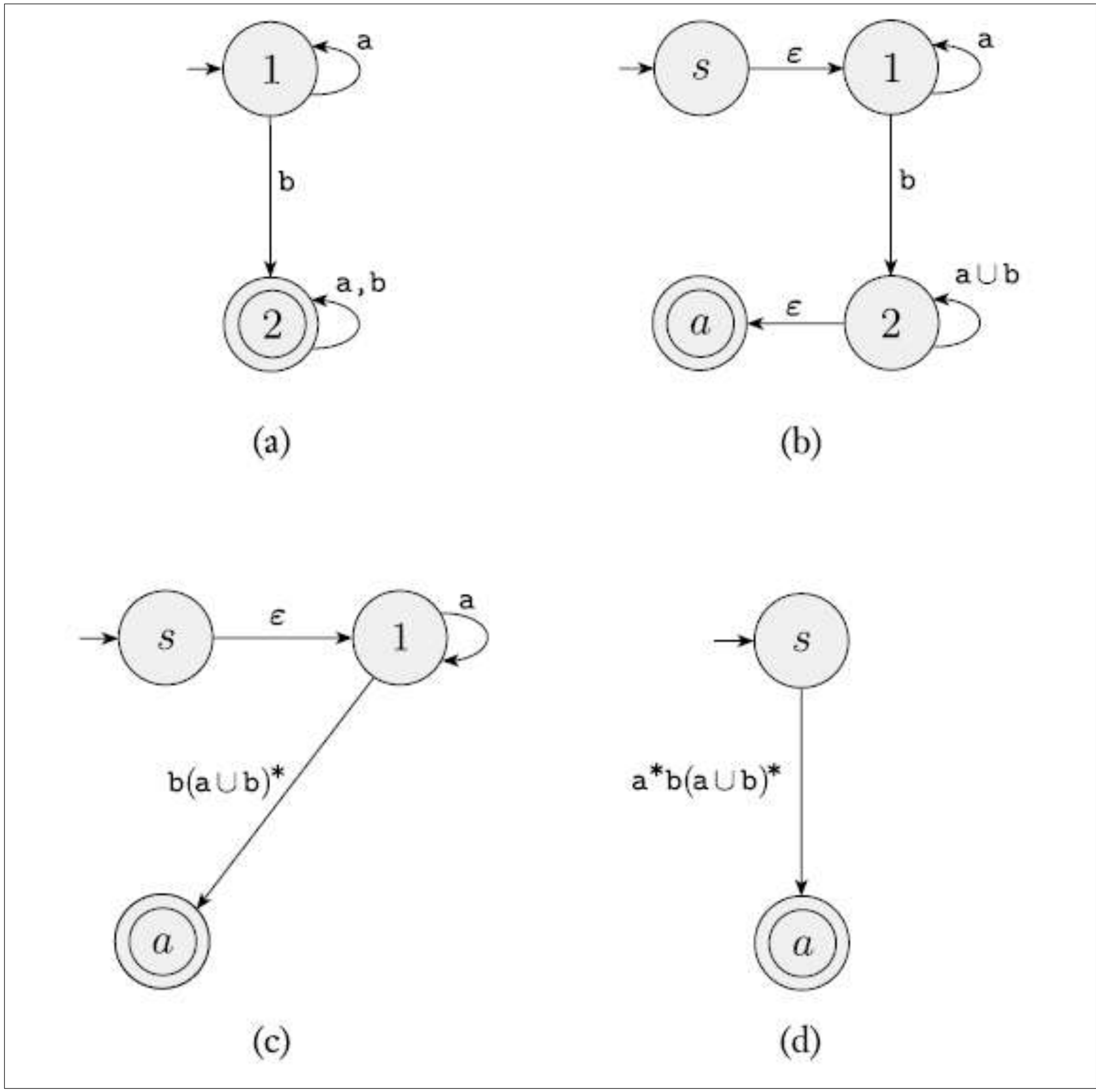


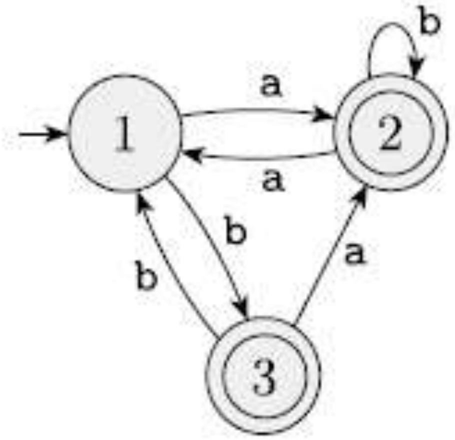
FIGURE 1.63
Constructing an equivalent GNFA with one fewer state

Example 1

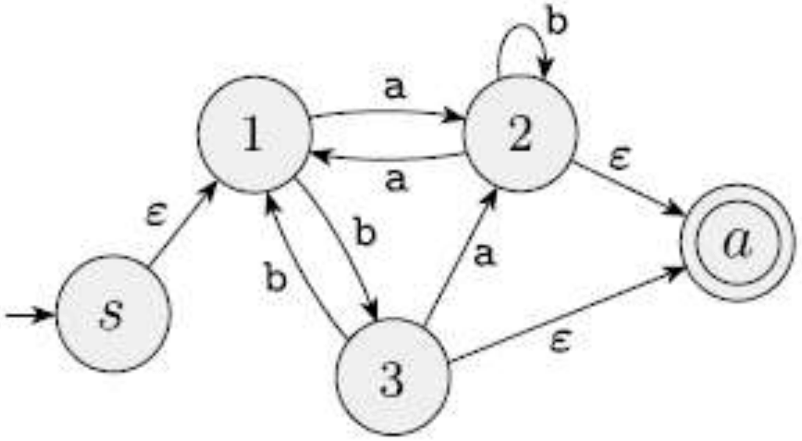


To avoid cluttering up the figure, we do not draw the arrows labeled ∅, even though they are present.

Example 2

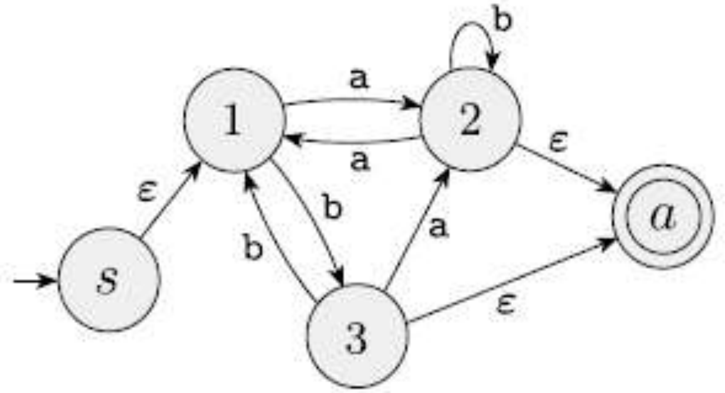


(a)

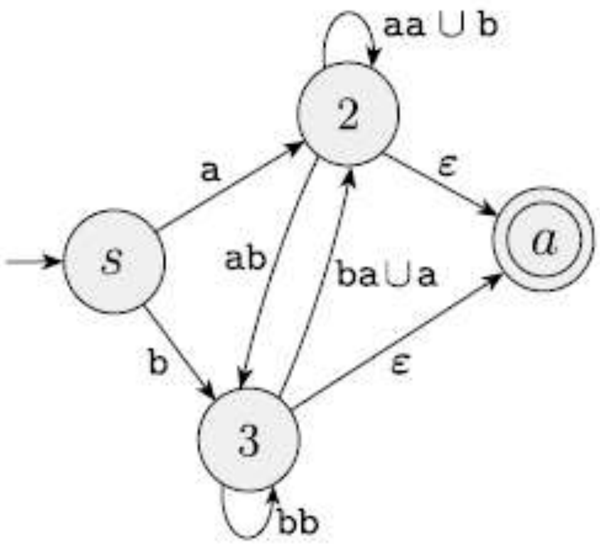


(b)

Example 2…

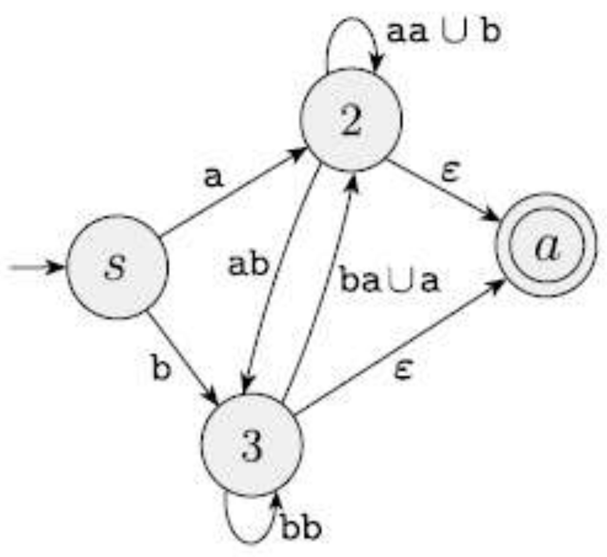


(b)

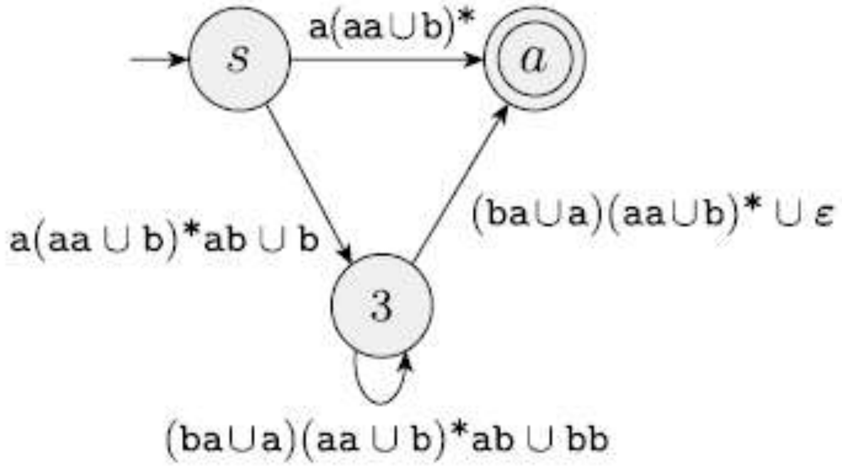


(c)

Example 2…

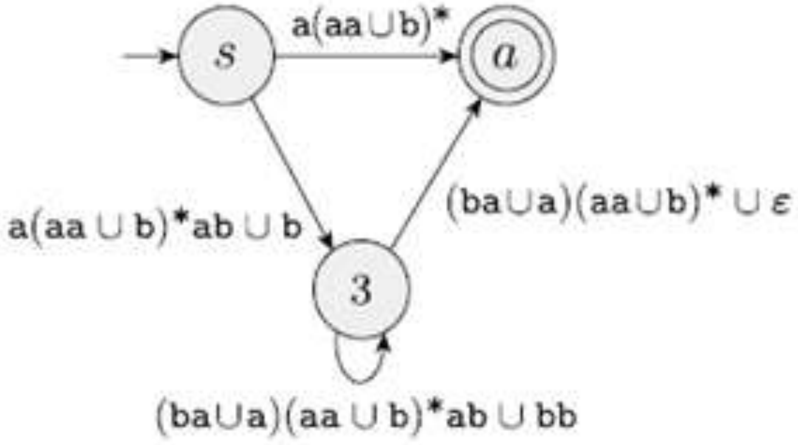


(c)

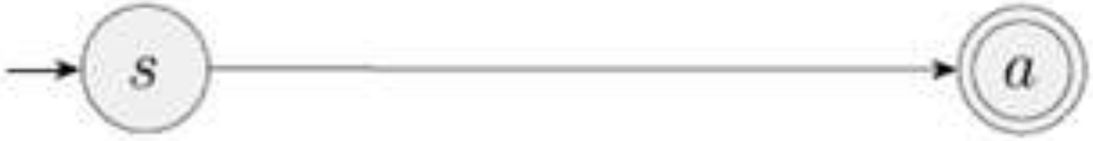


(d)

Example 2…



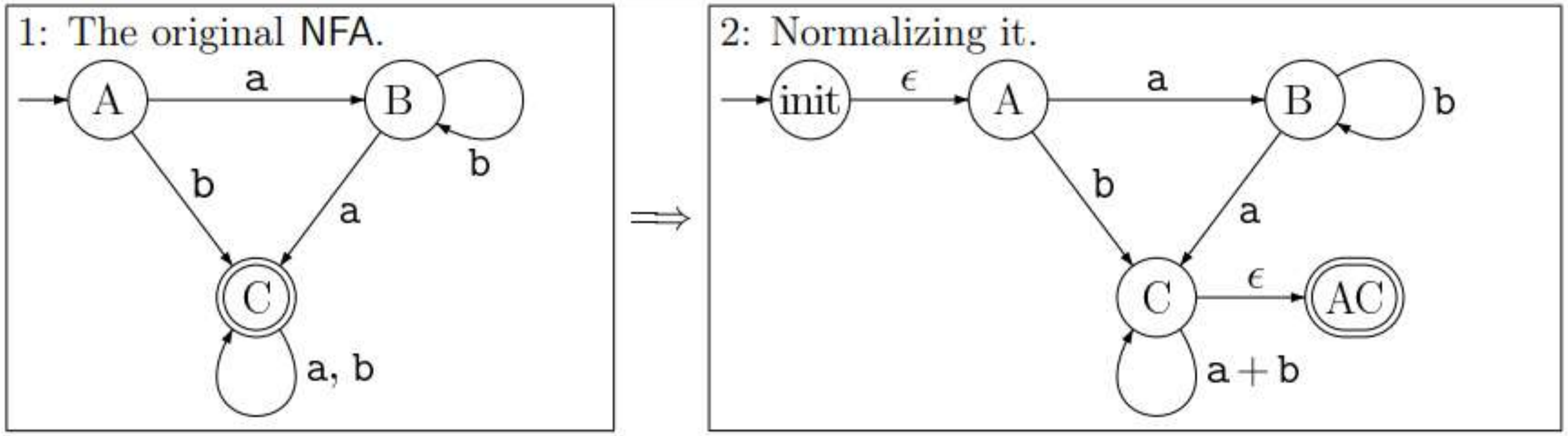
(d)



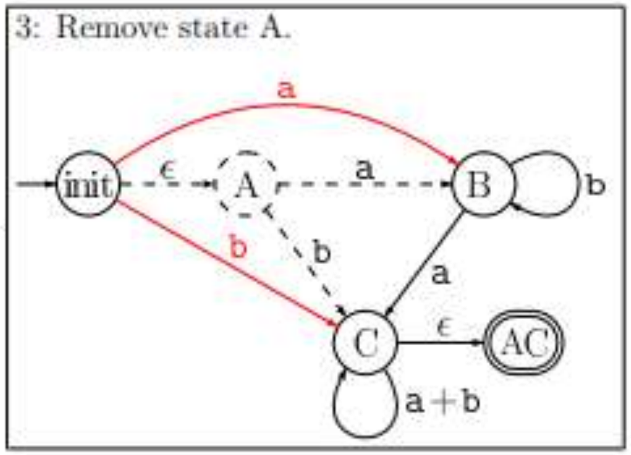
$(a(aa\cup b)^{*}ab\cup b)((ba\cup a)(aa\cup b)^{*}ab\cup bb)^{*}((ba\cup a)(aa\cup b)^{*}\cup \varepsilon)\cup a(aa\cup b)^{*}$

(e)

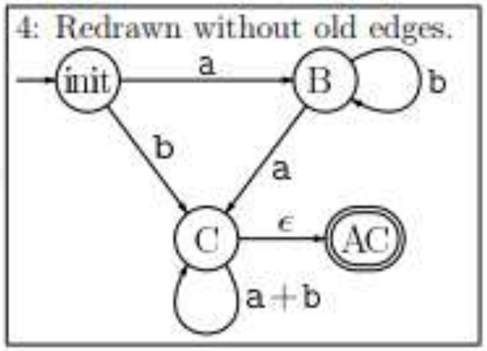
Example 3


1: The original NFA.
A
B
C
a
b
b
a
a, b
⟹
2: Normalizing it.
init
A
B
C
AC
ϵ
a
b
b
a
ϵ
a + b

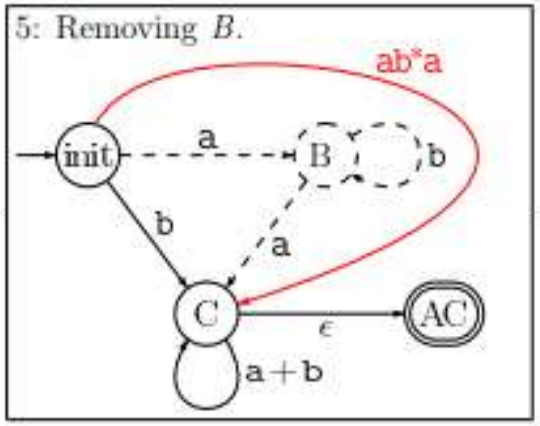
$\Longrightarrow$


3: Remove state A.

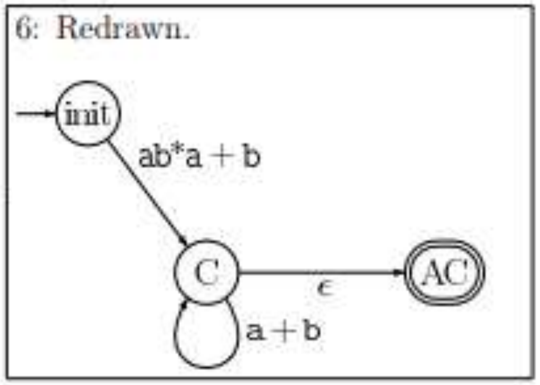
$\Longrightarrow$


4: Redrawn without old edges.

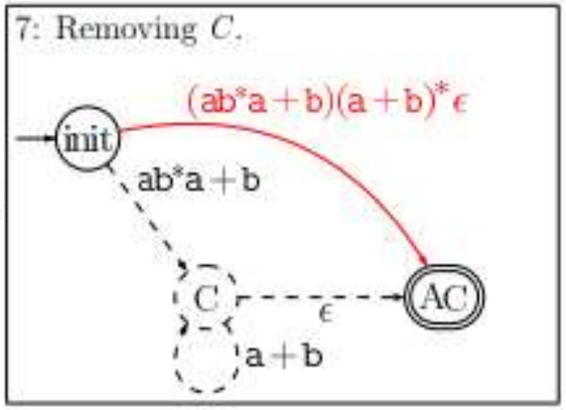
$\Longrightarrow$


5: Removing *B*.

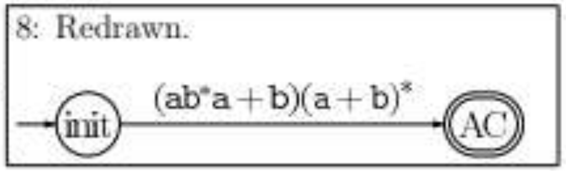
$\Longrightarrow$


6: Redrawn.

$\Longrightarrow$


7: Removing *C*.

$\Longrightarrow$


8: Redrawn.

Thus, this automata is equivalent to the regular expression $(ab^*a + b)(a + b)^*$.

Laws concerning R.E.

- We overload + to mean U also.
  - You have to live with this notational abuse between Sipser and Ullman.
- a+b = a U b

3.4 Algebraic Laws for Regular Expressions

- Commutativity and Associativity
  - L + M = M + L (Remember, L U M = M U L)
  - (L + M) + R = L + (M + R) = L + M + R
  - (LM)R = L(MR) = LMR
- Left distribution and right distribution
  - L(M + N) = LM + LN
  - (M + N)L = ML + NL

Identities and Annihalors

- $\emptyset + L = L + \emptyset = L$. This law asserts that $\emptyset$ is the identity for union.
- $\epsilon L = L\epsilon = L$. This law asserts that ϵ is the identity for concatenation.
- $\emptyset L = L\emptyset = \emptyset$. This law asserts that $\emptyset$ is the annihilator for concatenation.

Idempotent Laws

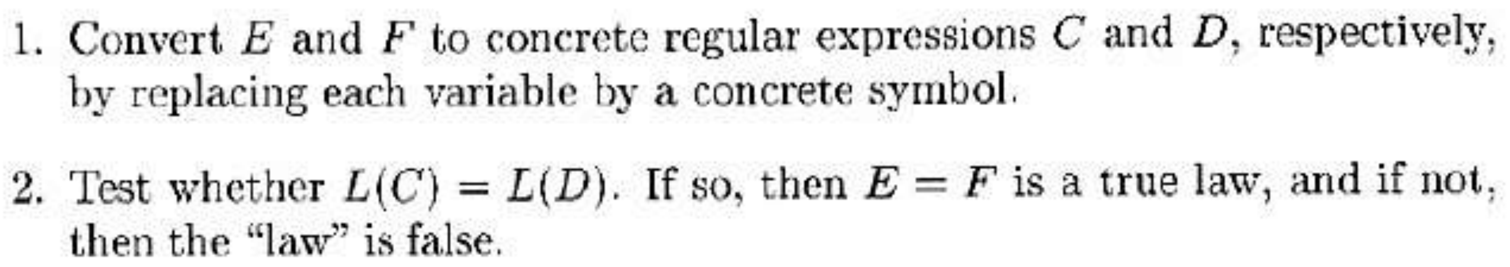
- $L+L = L$
- $(L^*)^* = L^*$

How to prove?

- Inequality can be easily proved by a counter example.
- But, equality, to be proved is cumbersome.
  - You can follow your set theory knowledge to deduct that from LHS, RHS is deductible.
  - These is an established simple way of doing this.

Assuming only three regular operators, viz., +, . * are only used.

- To test whether E = F is true or false.



1. Convert E and F to concrete regular expressions C and D, respectively, by replacing each variable by a concrete symbol.
2. Test whether $L(C) = L(D)$. If so, then $E = F$ is a true law, and if not, then the "law" is false.

- What do you mean by concrete R.E. ?

Concretizing a R.E.

Let $E = P + Q(RS^*)$ be a regular expression where P, Q, R, S are some regular expressions.

Here, we say E has variables P, Q, R, S.

Concretizing E means replacing each variable in E by a distinct symbol.

In this example, we can concretize $P + Q(RS^*)$ to $a + b(cd^*)$

To prove, $P(M+N) = PM + PN$.

Concretizing L.H.S will give us $a(b+c)$

Concretizing R.H.S will give us $ab+ac$

Now, one has to show $L\big(a(b+c)\big) = L(ab+ac)$, which can be done easily.

Now, verify whether $PR = RP$ is true or false.

Concretize L.H.S in to ab

Concretize R.H.S in to ba

Now, it is clear that $L(ab) = \{ab\}$ is not equal to $L(ba) = \{ba\}$.

! **Exercise 3.4.2:** Prove or disprove each of the following statements about regular expressions.

* a) $(R+S)^* = R^* + S^*$.

  b) $(RS+R)^*R = R(SR+R)^*$.

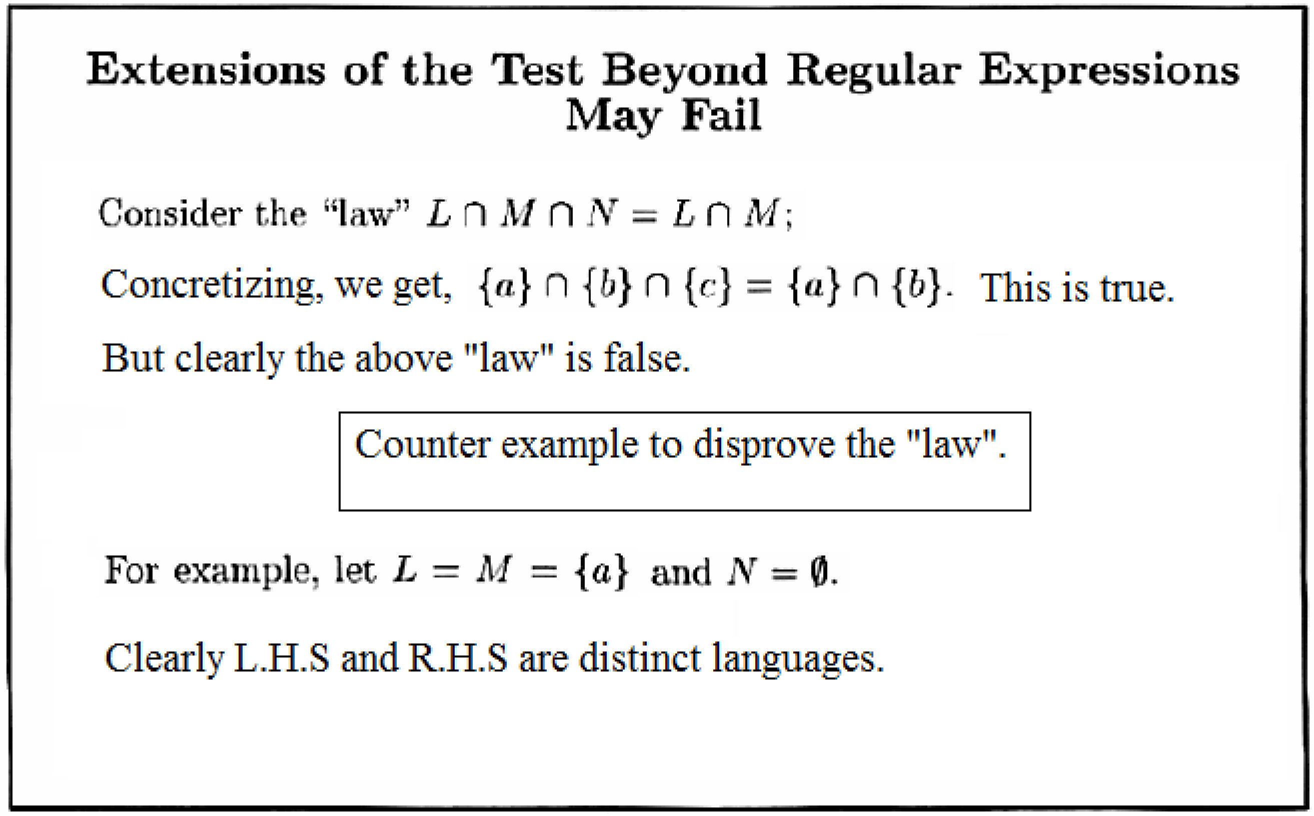
* c) $(RS+R)^*RS = (RR^*S)^*$.

  d) $(R+S)^*S = (R^*S)^*$.

  e) $S(RS+S)^*R = RR^*S(RR^*S)^*$.

Don't use operators beyond $+ \quad \cdot \quad *$

- That is, stick to, regular operators only.



Extensions of the Test Beyond Regular Expressions May Fail

Consider the "law" $L \cap M \cap N = L \cap M$;

Concretizing, we get, $\{a\} \cap \{b\} \cap \{c\} = \{a\} \cap \{b\}$. This is true.

But clearly the above "law" is false.

Counter example to disprove the "law".

For example, let $L = M = \{a\}$ and $N = \emptyset$.

Clearly L.H.S and R.H.S are distinct languages.