Properties of Regular Languages

Chapter 4 (Last Part)

Closure Properties

- Union
- Intersection
- Difference
- Concatenation
- Kleene Closure
- Reversal
- Homomorphism
- Inverse Homomorphism

Closure Properties

- Recall a closure property is a statement that a certain operation on languages, when applied to languages in a class (e.g., the regular languages), produces a result that is also in that class.
- For regular languages, we can use any of its representations to prove a closure property.

Closure Under Union

- If L and M are regular languages, so is $L \cup M$.
- Proof: Let L and M be the languages of regular expressions R and S, respectively.
- Then R+S is a regular expression whose language is $L \cup M$.

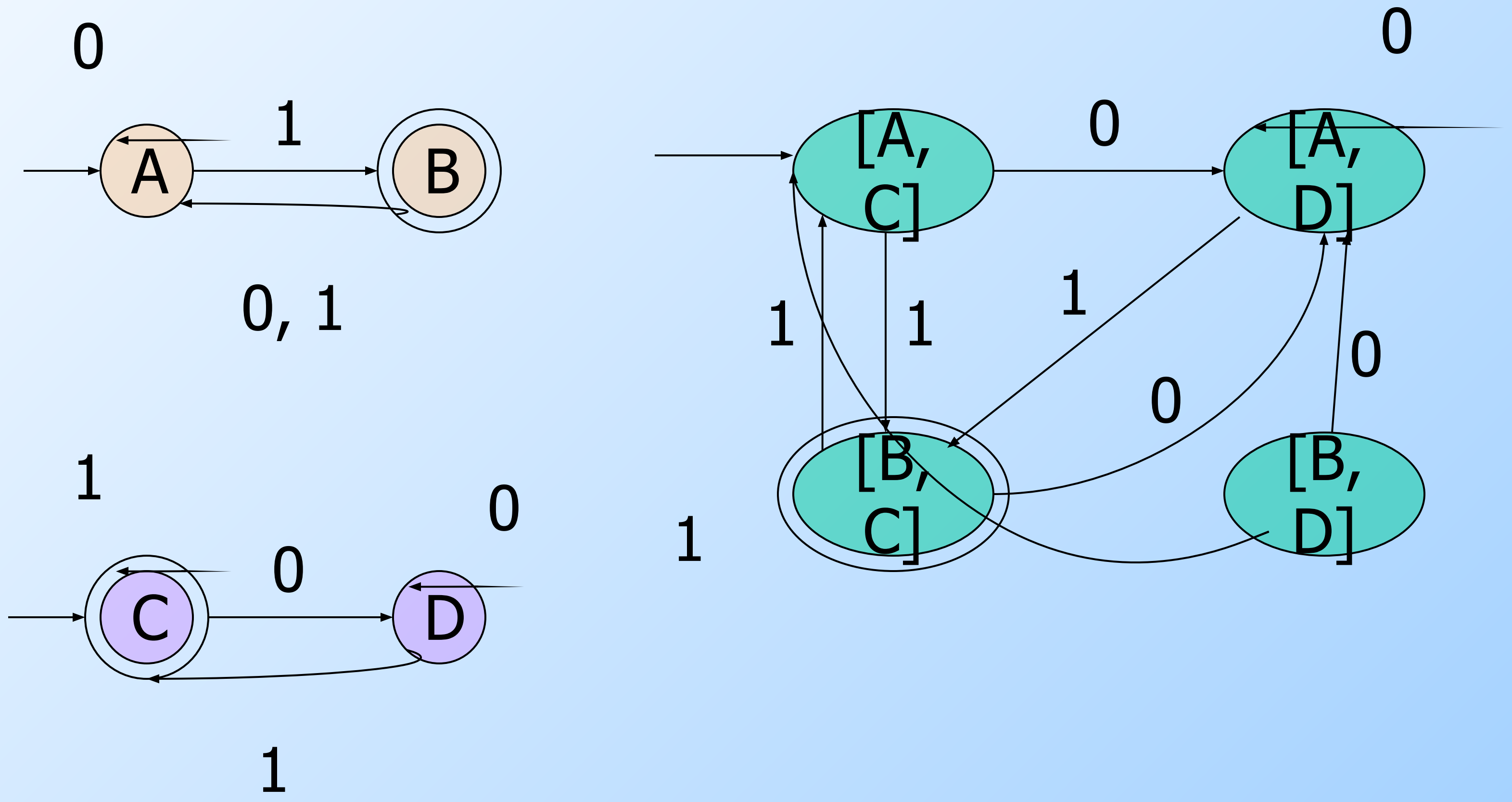
Closure Under Concatenation and Kleene Closure

- Same idea:
  - RS is a regular expression whose language is LM.
  - R* is a regular expression whose language is L*.

Closure Under Intersection

- If L and M are regular languages, then so is $L \cap M$.
- Proof: Let A and B be DFA's whose languages are L and M, respectively.
- Construct C, the product automaton of A and B.
- Make the final states of C be the pairs consisting of final states of both A and B.

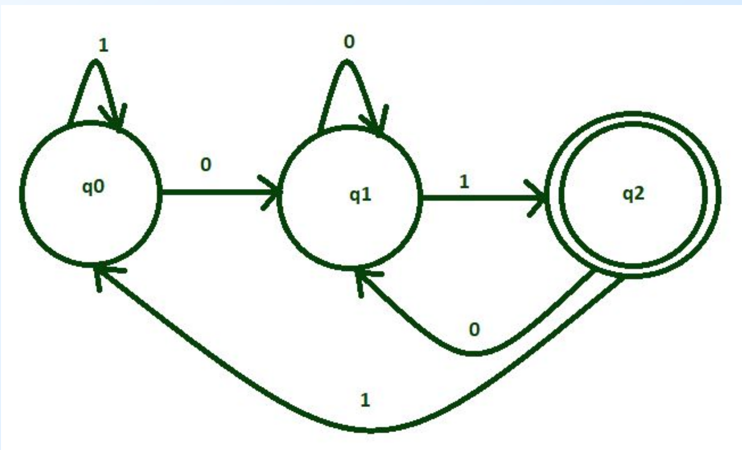
Example: Product DFA for Intersection

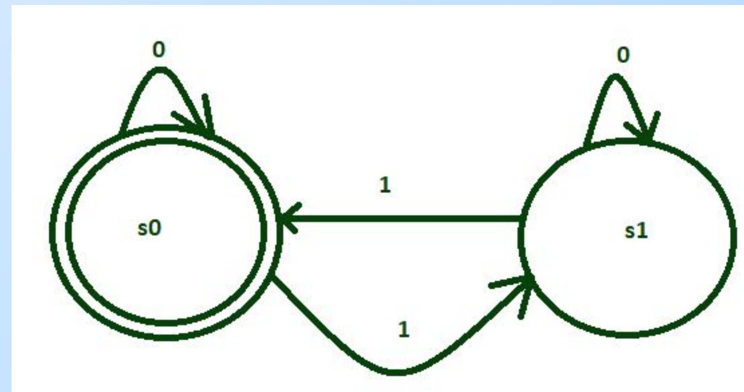


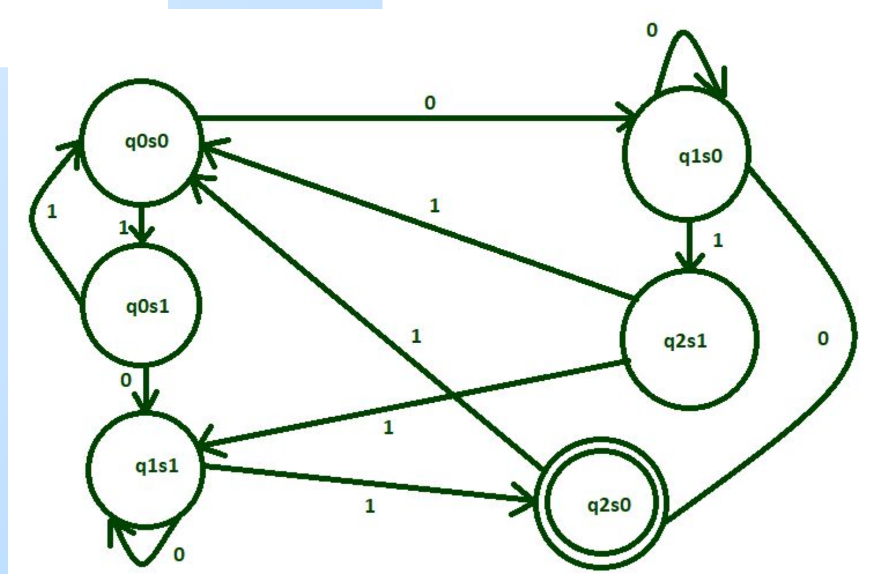
Contd…

$L_1 = \{01, 001, 101, 0101, 1001, 1101, ...\}$
$L_2 = \{11, 011, 101, 110, 0011, 1100, ...\}$

$L = L_1 \cap L_2 = \{1001, 0101, 01001, 10001, ...\}$



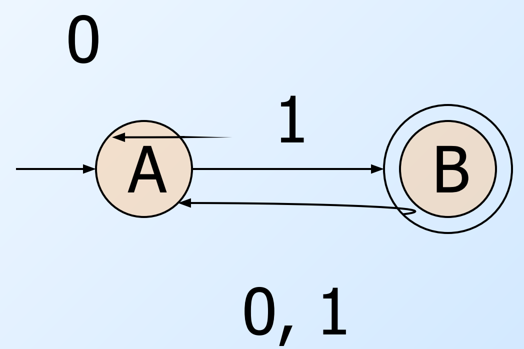


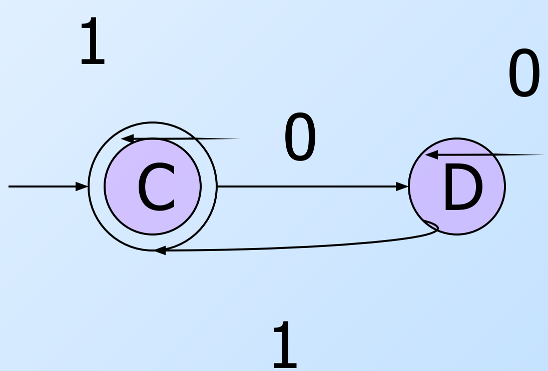


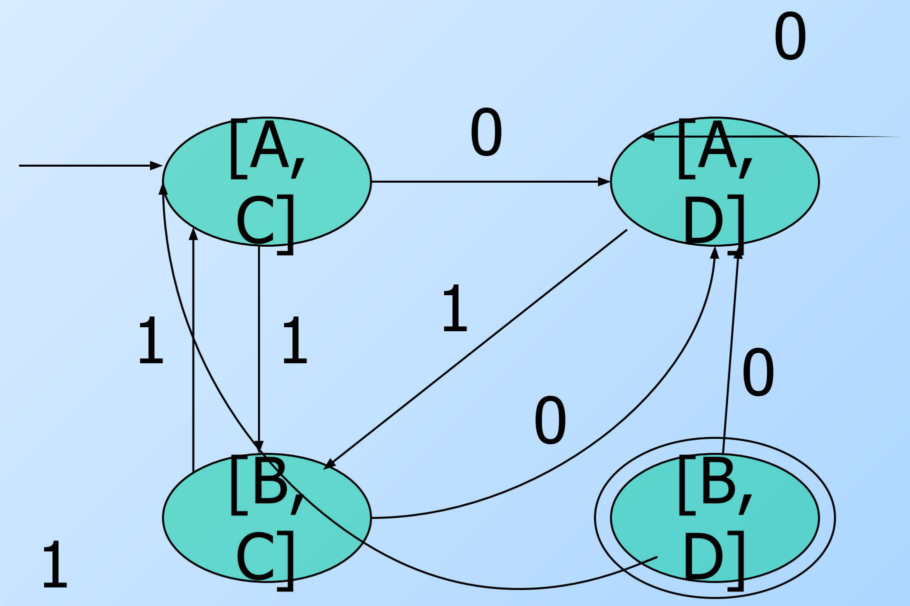
Closure Under Difference

- If L and M are regular languages, then so is $L - M$ = strings in L but not M.
- Proof: Let A and B be DFA's whose languages are L and M, respectively.
- Construct C, the product automaton of A and B.
- Make the final states of C be the pairs where A-state is final but B-state is not.

Example: Product DFA for Difference







Notice: difference is the empty language

Closure Under Complementation

- The *complement* of a language L (with respect to an alphabet Σ such that Σ^* contains L) is $\Sigma^* - L$.
- Since Σ^* is surely regular, the complement of a regular language is always regular.

Closure Under Reversal

- Given language L, L^R is the set of strings whose reversal is in L.
- Example: L = {0, 01, 100};
  L^R = {0, 10, 001}.
- Proof: Let E be a regular expression for L.
- We show how to reverse E, to provide a regular expression E^R for L^R.

Reversal of a Regular Expression

- Basis: If E is a symbol a, ε, or $\varnothing$, then $E^R = E$.
- Induction: If E is
  - F+G, then $E^R = F^R + G^R$.
  - FG, then $E^R = G^R F^R$
  - F*, then $E^R = (F^R)^*$.

Example: Reversal of a RE

- Let E = $\mathbf{01}^* + \mathbf{10}^*$.
- $E^R = (\mathbf{01}^* + \mathbf{10}^*)^R = (\mathbf{01}^*)^R + (\mathbf{10}^*)^R$
- $= (\mathbf{1}^*)^R\mathbf{0}^R + (\mathbf{0}^*)^R\mathbf{1}^R$
- $= (\mathbf{1}^R)^*\mathbf{0} + (\mathbf{0}^R)^*\mathbf{1}$
- $= \mathbf{1}^*\mathbf{0} + \mathbf{0}^*\mathbf{1}$.

Reversal of a RE

Given a language L that is $L(A)$ for some finite automaton, perhaps with nondeterminism and ϵ-transitions, we may construct an automaton for L^R by:

1. Reverse all the arcs in the transition diagram for A.
2. Make the start state of A be the only accepting state for the new automaton.
3. Create a new start state p_0 with transitions on ϵ to all the accepting states of A.

The result is an automaton that simulates A "in reverse," and therefore accepts a string w if and only if A accepts w^R. Now, we prove the reversal theorem formally.

Theorem:

The reverse L^R of a regular language L is a regular language

Proof idea:

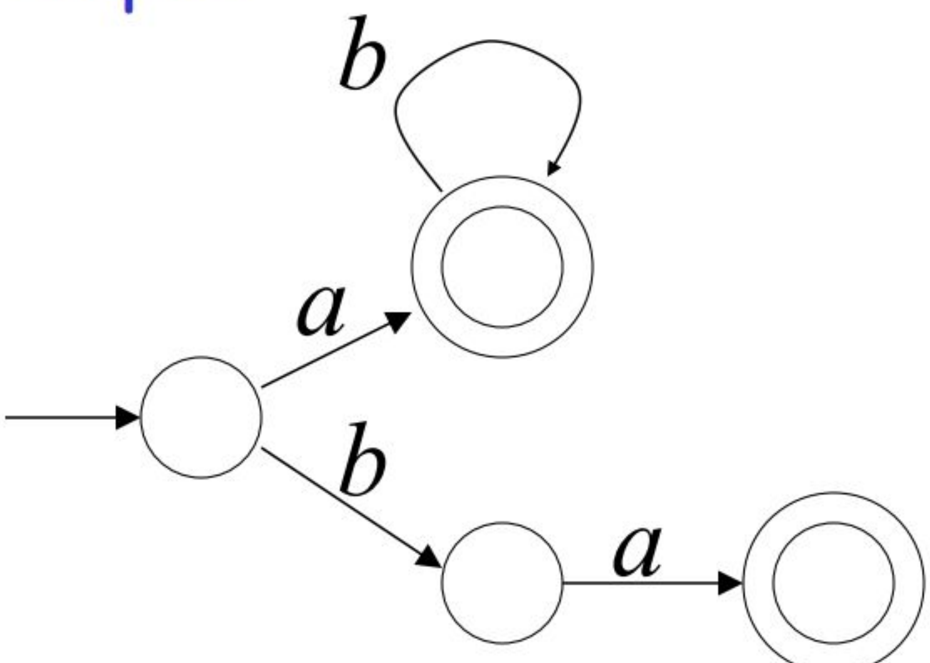
Construct NFA that accepts L^R :

invert the transitions of the NFA that accepts L

Proof

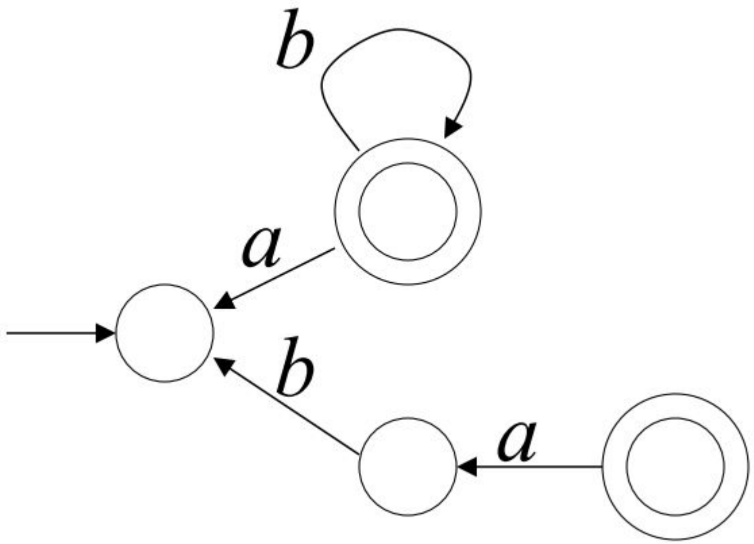
Since L is regular,
there is NFA that accepts L

Example:

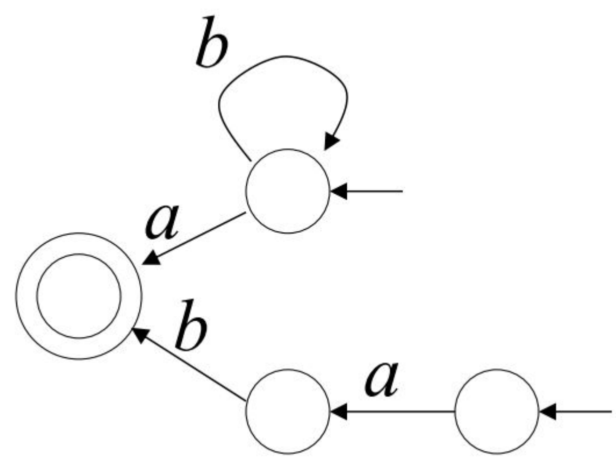


$$L = ab^* + ba$$

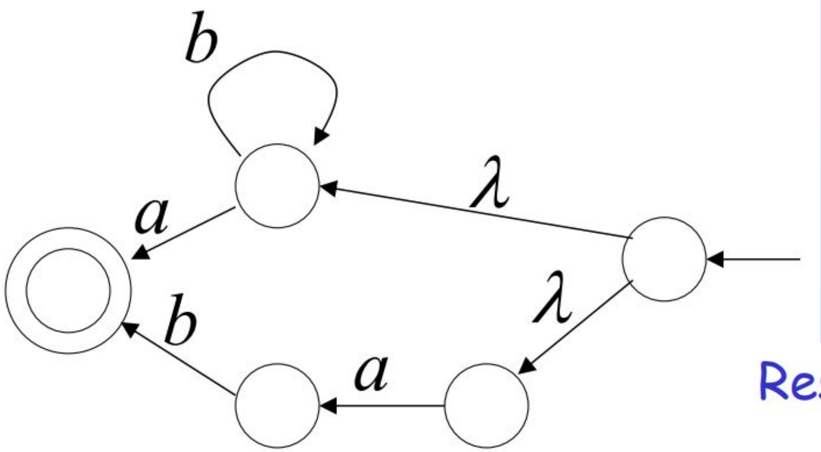
Invert Transitions



Make old initial state a final state



Add a new initial state

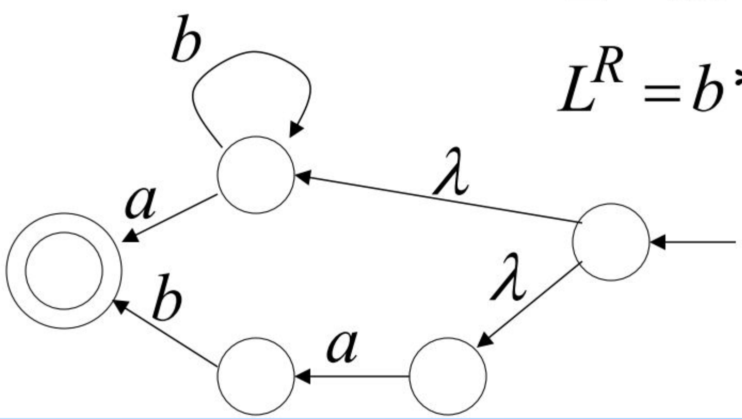


Resulting machine accepts L^R

L^R is regular

$$L = ab^* + ba$$

$$L^R = b^* a + ab$$



Homomorphisms

- A *homomorphism* on an alphabet is a function that gives a string for each symbol in that alphabet.
- Example: h(0) = ab; h(1) = ε.
- Extend to strings by $h(a_1 \ldots a_n) = h(a_1) \ldots h(a_n)$.
- Example: h(01010) = ababab.

Closure Under Homomorphism

- If L is a regular language, and h is a homomorphism on its alphabet, then h(L) = {h(w) | w is in L} is also a regular language.
- Proof: Let E be a regular expression for L.
- Apply h to each symbol in E.
- Language of resulting RE is h(L).

Example: Closure under Homomorphism

- Let h(0) = ab; h(1) = ε.
- Let L be the language of regular expression **01*** + **10***.
- Then h(L) is the language of regular expression **ab**ε* + ε(**ab**)*.

Note: use parentheses to enforce the proper grouping.

Example – Continued

- **ab**ε* + ε(**ab**)* can be simplified.
- ε* = ε, so **ab**ε* = **ab**ε.
- ε is the identity under concatenation.
  - That is, εE = Eε = E for any RE *E*.
- Thus, **ab**ε* + ε(**ab**)* = **ab**ε + ε(**ab**)* = **ab** + (**ab**)*.
- Finally, L(**ab**) is contained in L((**ab**)*), so a RE for h(L) is (**ab**)*.

Example – Continued

- Define h as follows:

$$\begin{aligned} h(0) &= hello \\ h(1) &= goodbye \end{aligned}$$

- Then, $h(010) = hellogoodbyehello$.
- The homomorphic image of $L = \{00, 010\}$ is

$$h(L) = \{hellohello, hellogoodbyehello\}.$$

Inverse Homomorphisms

- Let h be a homomorphism and L a language whose alphabet is the output language of h.
- $h^{-1}(L)$ = {w | h(w) is in L}.

Example: Inverse Homomorphism

- Let $h(0) = ab$; $h(1) = \varepsilon$.
- Let $L = \{abab, baba\}$.
- $h^{-1}(L)$ = the language with two 0's and any number of 1's = $L(\mathbf{1}*\mathbf{01}*\mathbf{01}*)$.

Notice: no string maps to baba; any string with exactly two 0's maps to abab.

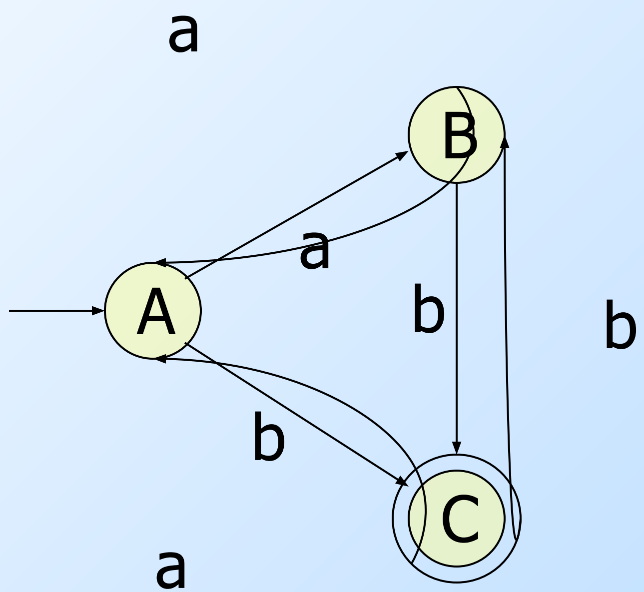
Closure Proof for Inverse Homomorphism

- Start with a DFA A for L.
- Construct a DFA B for $h^{-1}(L)$ with:
  - The same set of states.
  - The same start state.
  - The same final states.
  - Input alphabet = the symbols to which homomorphism h applies.

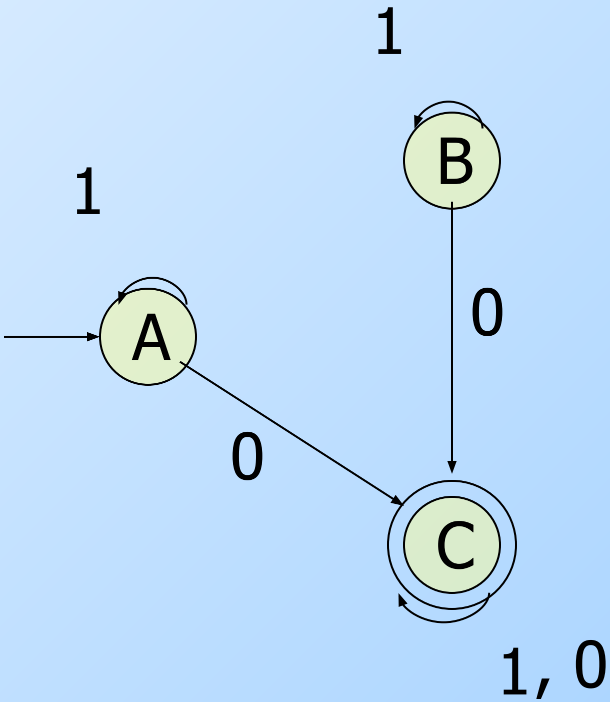
Proof – (2)

- The transitions for B are computed by applying h to an input symbol *a* and seeing where A would go on sequence of input symbols h(a).
- Formally, $\delta_B(q, a) = \delta_A(q, h(a))$.

Example: Inverse Homomorphism Construction



$h(0) = ab$
$h(1) = \varepsilon$



Since
$h(1) = \varepsilon$

Since
$h(0) = ab$

Proof – (3)

- Induction on $|w|$ shows that $\delta_B(q_0, w) = \delta_A(q_0, h(w))$.
- Basis: $w = \varepsilon$.
- $\delta_B(q_0, \varepsilon) = q_0$, and $\delta_A(q_0, h(\varepsilon)) = \delta_A(q_0, \varepsilon) = q_0$.

Proof – (4)

- Induction: Let w = xa; assume IH for x.
- $\delta_B(q_0, w) = \delta_B(\delta_B(q_0, x), a)$.
- $= \delta_B(\delta_A(q_0, h(x)), a)$ by the IH.
- $= \delta_A(\delta_A(q_0, h(x)), h(a))$ by definition of the DFA B.
- $= \delta_A(q_0, h(x)h(a))$ by definition of the extended delta.
- $= \delta_A(q_0, h(w))$ by def. of homomorphism.

IH - Example

Exercise 4.2.1 : Suppose h is the homomorphism from the alphabet $\{0, 1, 2\}$ to the alphabet $\{a, b\}$ defined by: $h(0) = a$; $h(1) = ab$, and $h(2) = ba$.

* e) Suppose L is the language $\{ababa\}$, that is, the language consisting of only the one string $ababa$. What is $h^{-1}(L)$?

- $h^{-1}(a)=0$, $h^{-1}(ab)=1$, $h^{-1}(ba)=2$
- $h^{-1}(L)=\{022, 102, 110\}$

- ababa = 022
- ababa = 102
- ababa = 110