Properties of Regular Languages

Closure Properties

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

Closure Properties

- 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 the union $L \cup M$ is also regular.

Proof: Let L and M be the languages of regular expressions R and S, respectively.

- By the definition of regular expression $R+S$ is a regular expression whose language is $L \cup M$.
- So $L \cup M$ is regular.

Closure Under Concatenation and Kleene Closure

- **Same idea:** By the definition of RE
- If L and M are language represented by regular expression R and S respectively then,
  - RS is a regular expression whose language is LM - Concatenation
  - R* is a regular expression whose language is L* - Kleen Closure.

Closure Under Intersection

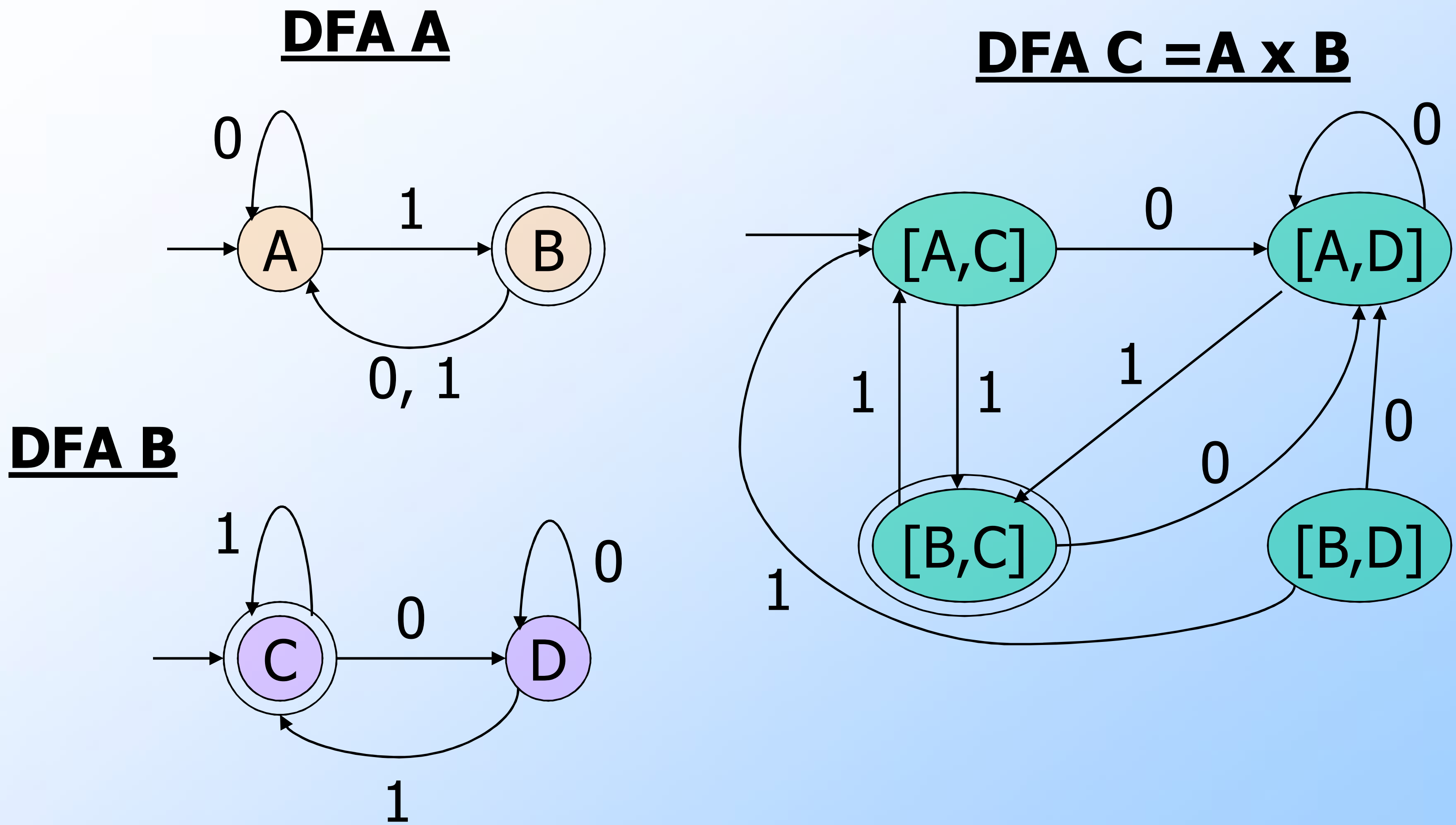
- If L and M are regular languages, then $L \cap M$ is also regular.

Proof: Since L and M are regular languages, then there exist DFA's accepting languages L and M.

- Let A and B be DFA's whose languages are L and M, respectively.
- Construct C, the product automaton of A and B whose states are order pairs of states of A and B.
- Make the start state of C as pair of start states of A and B.
- Make the final states of C be the pairs consisting of final states of both A and B (pair of final states).
- Define the transition function of C as:

$$\delta_c((p,q),a)=(\delta_A(p,a), \delta_{\mathbf{B}}(q,a)) \text{ for } p \in Q_A,\ q \in Q_B \text{ and } a \in \Sigma$$

Example: Product DFA for Intersection



- Hence we can construct DFA C that accepts what A and B both accepts. So intersection of Regular language is regular.

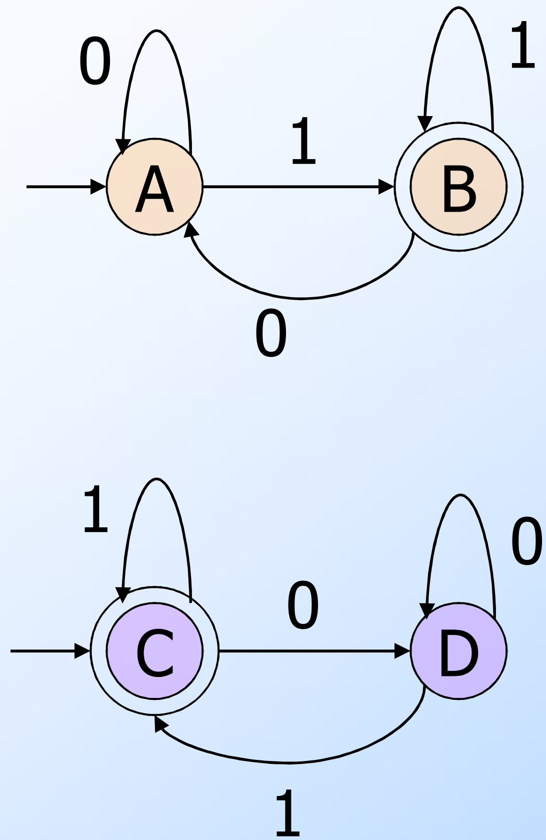
Closure Under Difference

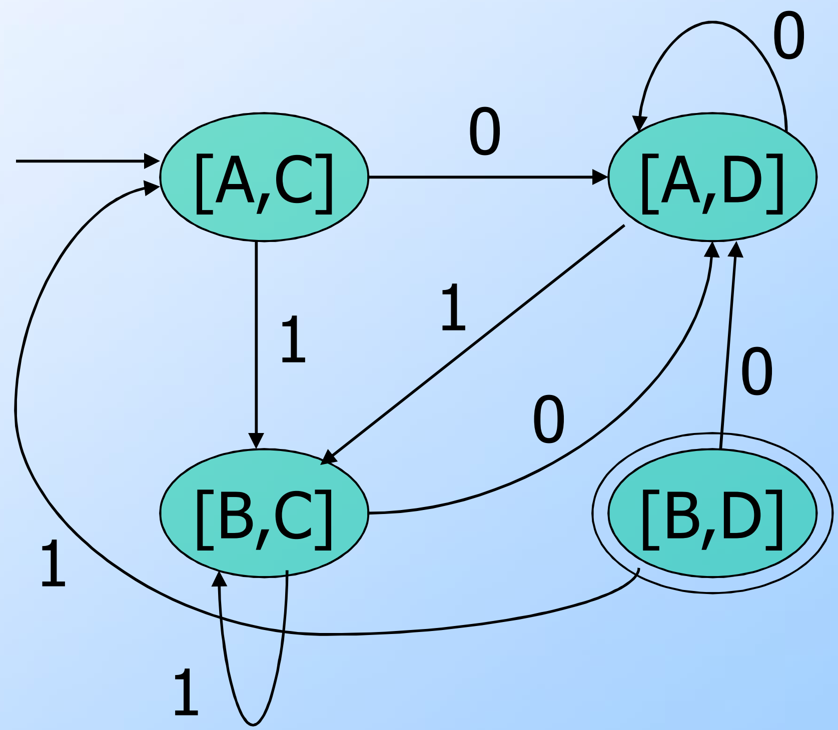
- If L and M are regular languages, then so is $L - M$ = strings in L but not M.

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 start state of C as pair of start states 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 regular language L (with respect to an alphabet Σ such that Σ^* contains L) is $\Sigma^* - L$.
- Since Σ^* is surely regular, the above statement shows that the complement of L is the difference two regular languages Σ^* and L.
- Hence the complement of a regular language is always regular.

Finding complement of regular language

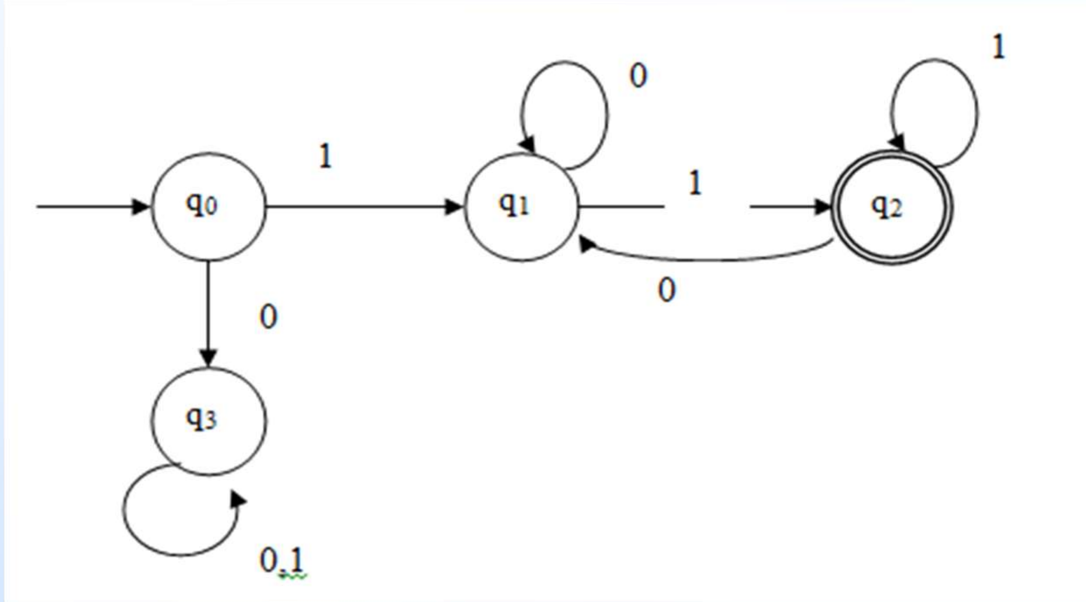
- Obtain the regular expression for language.
- Construct ϵ–NFA from regular expression.
- Convert ϵ–NFA into DFA.
- Complement the states of DFA i.e. convert accepting states into non accepting states and vice-versa.
- Turn the complement DFA back into R.E.

Finding complement of regular language

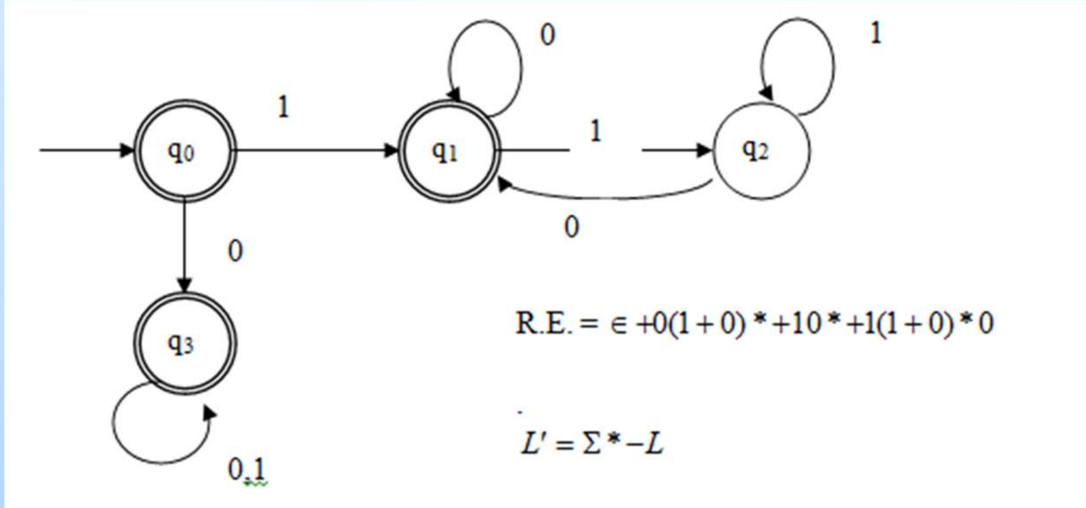
- Let the DFA constructed accepting Regular Language is.

R.E. = $1^+(0+1)^*1^+$

= Accepts all strings starting and ending with 1.



The complement DFA of above DFA



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 ∅, 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 $E^R = (F^R)^*$.

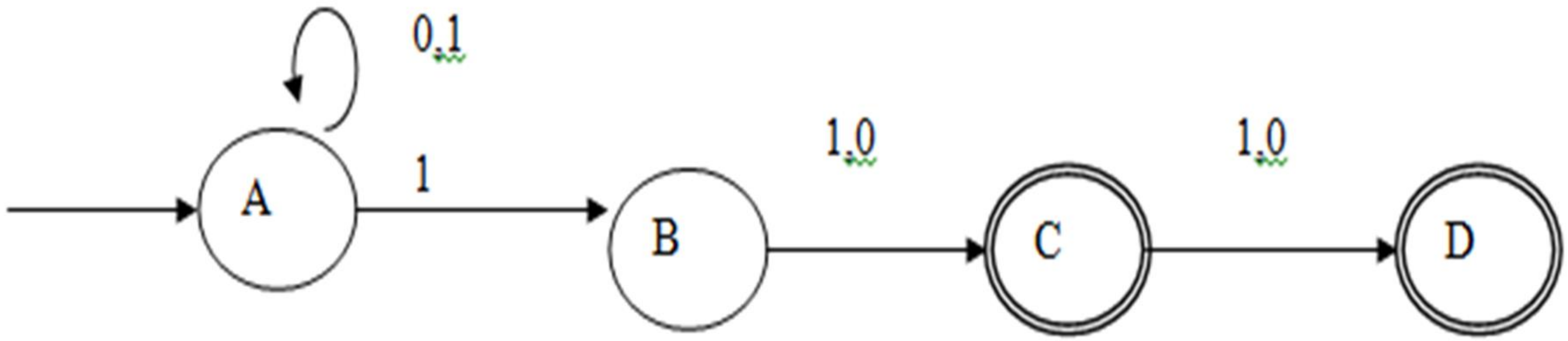
Example: Reversal of a RE

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

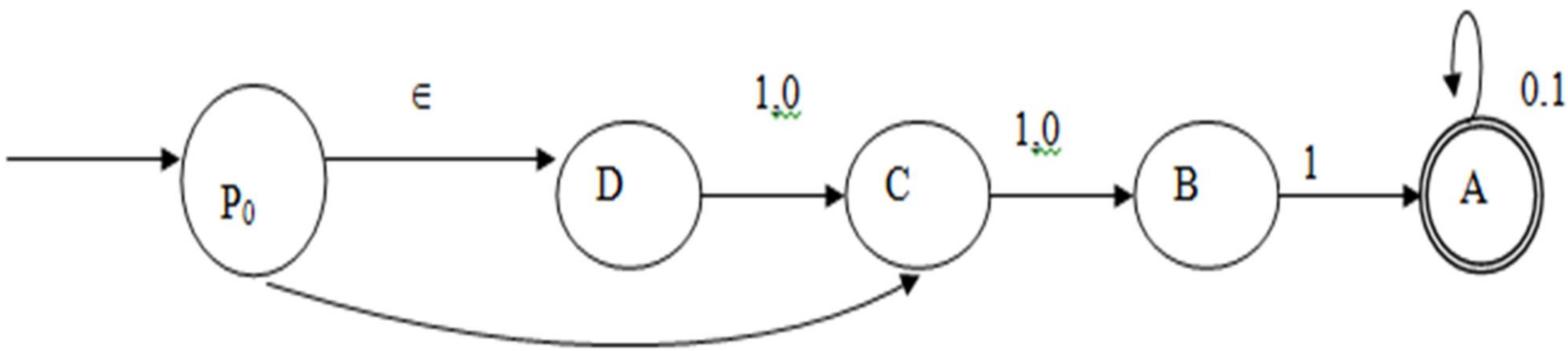
Creating DFA accepting Reversal of Language

- Given a language L that is L(M) for some FA M, we can construct Automaton for L^R as:
  - Reverse all the arcs in the transition diagram of M
  - Make the start state only accepting state for new automaton
  - Create a new start state p_0 with transition on   to all the accepting states of M.

Reversal FA: Example



M=NFA accepting string having a 1 in 3rd position from end.



Reversal of FA for M ∈

Homomorphisms

- A *homomorphism* on an alphabet is a function that gives a string for each symbol in that alphabet.
- Example: $h(0) = ab$; $h(1) = \epsilon$.
- 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 $h(L) = \{h(w) \mid w \text{ is in } L\}$ 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

- $\mathbf{ab}\epsilon^* + \epsilon(\mathbf{ab})^*$ can be simplified.
- $\epsilon^* = \epsilon$, so $\mathbf{ab}\epsilon^* = \mathbf{ab}\epsilon$.
- ϵ is the identity under concatenation.
  - That is, $\epsilon E = E\epsilon = E$ for any RE E.
- Thus, $\mathbf{ab}\epsilon^* + \epsilon(\mathbf{ab})^* = \mathbf{ab}\epsilon + \epsilon(\mathbf{ab})^* = \mathbf{ab} + (\mathbf{ab})^*$.
- Finally, $L(\mathbf{ab})$ is contained in $L((\mathbf{ab})^*)$, so a RE for $h(L)$ is $(\mathbf{ab})^*$.

Inverse Homomorphisms

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

Example: Inverse Homomorphism

- Let $h(0) = ab$; $h(1) = \epsilon$.
- 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.

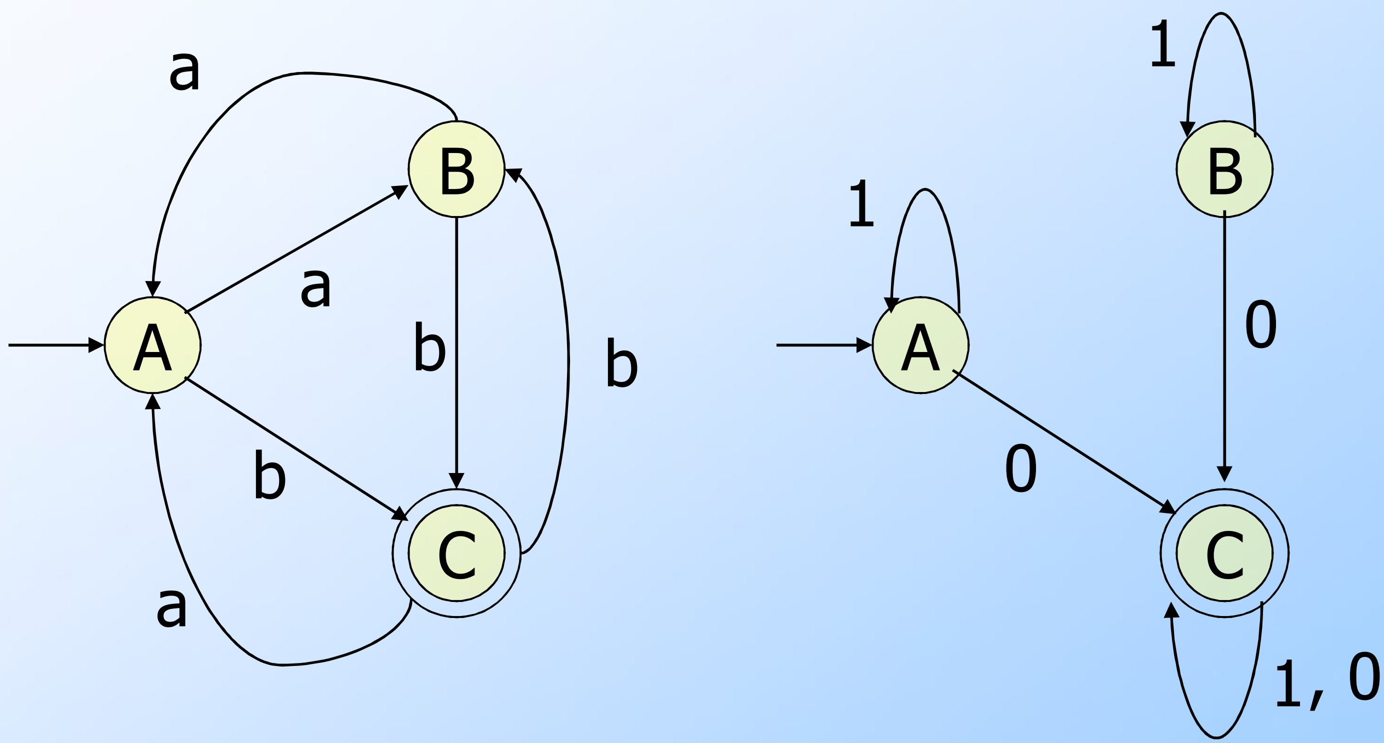
Closure Proof for Inverse Homomorphism

- 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 – Continued

- 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



Since
$h(1) = \epsilon$

Since
$h(0) = ab$

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

Proof -Continued

- Induction on |w| shows that $\delta_B(q_0, w) = \delta_A(q_0, h(w))$.
- Basis: $w = \epsilon$.
- $\delta_B(q_0, \epsilon) = q_0$, and $\delta_A(q_0, h(\epsilon)) = \delta_A(q_0, \epsilon) = q_0$.
- Induction: Let $w = xa$; where x is prefix without last symbol 'a' of w and **$\delta_B(q_0, x)=\delta_A(q_0, h(x))$**
  - $\delta_B(q_0, w) = \delta_B(\delta_B(q_0, x), a)$. (Induction Hypothesis)
  - $= \delta_B(\delta_A(q_0, h(x)), a)$ by the Induction Hypothesis.
  - $= \delta_A(\delta_A(q_0, h(x)), h(a))$ 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.
- This Completes the proof.