Regular Operations

Based on slides of Aaron Deever

Closure Properties

- A collection of objects is closed under a particular operation if applying that operation to members of the collection returns an object still in that collection
- Examples:

Yes
  - Is the set of integers closed under multiplication?

No
  - Is the set of integers closed under division?

Yes
  - Is the set of positive integers closed under addition?

No
  - Is the set of positive integers closed under subtraction?

Closure Properties of Regular Languages

- Regular Languages are closed under each of these operations:
  - Union
    - If A is regular and B is regular then $A \cup B$ is regular.
  - Concatenation
    - If A is regular and B is regular then AB is regular.
  - Kleene Star
    - If A is regular, then A^* is regular.
  - (also Intersection, Complement, and Difference!)

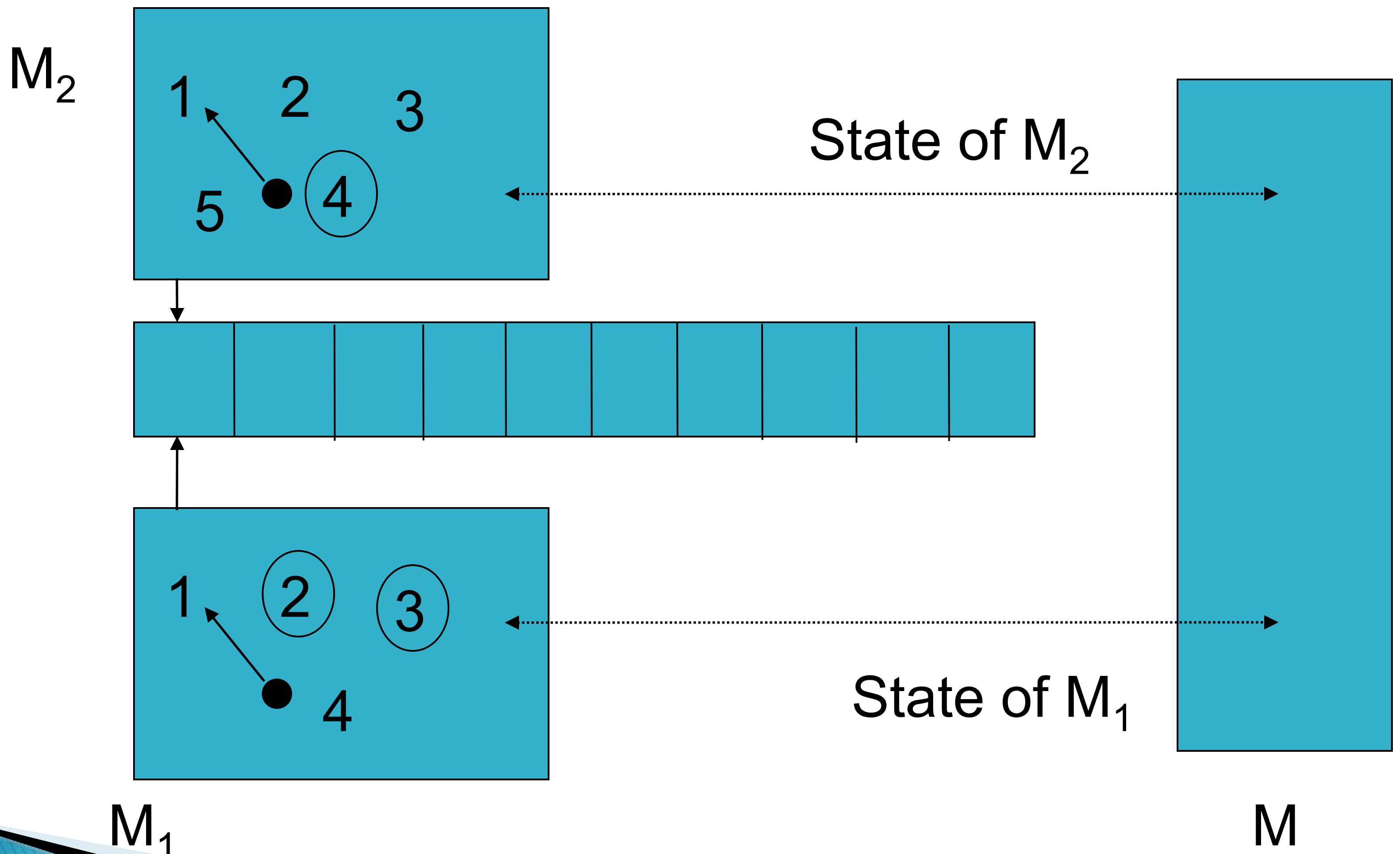
- Let's prove for union
  - By construction – we will build a DFA that accepts $A \cup B$

Closure of Union Operation

- Basic idea
  - If L_1 and L_2 are regular, then by definition, there exist DFAs, M_1 and M_2 such that
    - $L_1 = L(M_1)$
    - $L_2 = L(M_2)$
  - We will build a DFA, M, that for a single input w, will keep track of the current states of both M_1 and M_2 as they read w.
    - Any time ***either one*** of the machines M_1 and M_2 accepts w, the machine M will accept w.

- We will refer to this as the *Cartesian Product Construction*

Closure of Union



Closure of Union

- Let L_1 and L_2 be regular languages and let
- $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ be a DFA that accepts L_1
- $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ be a DFA that accepts L_2

Closure of Union

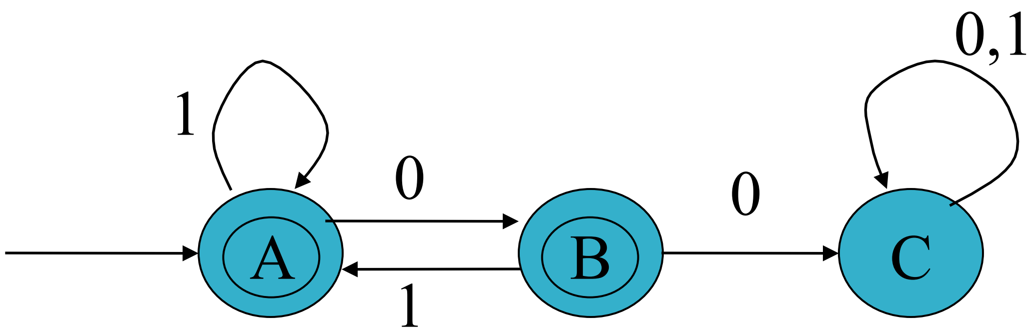
- We will build a new DFA, M, such that
  - Each state of M is an ordered pair (p, q) where $p \in Q_1$ and $q \in Q_2$
  - Informally, the states of M will represent the current states of M_1 and M_2 at each simultaneous move of the machines.
  - How many states will machine M have?
    - Given that M_1 has 4 states in our example
    - Given that M_2 has 5 states in our example

Closure of Union

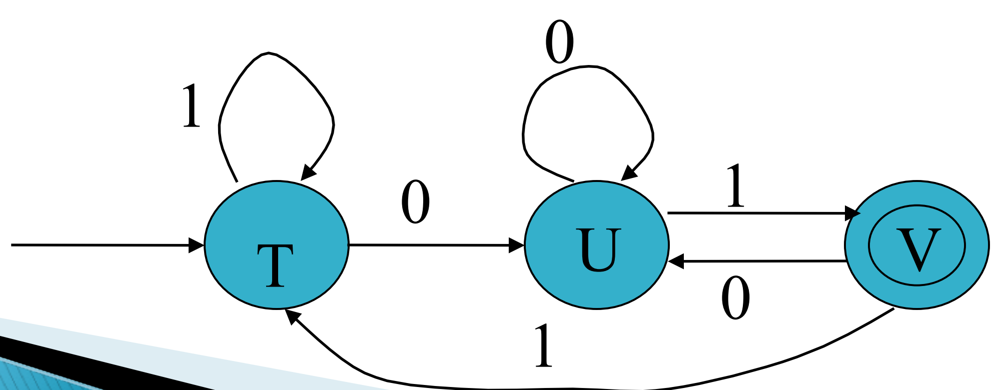
- Formally…
  - $M = (Q, \Sigma, \delta, q_0, F)$ where
    - $Q = Q_1 \times Q_2$
    - $q_o = (q_1, q_2)$
    - $\delta: (Q_1 \times Q_2) \times \Sigma \to (Q_1 \times Q_2)$
      - $\delta((p,q), a) = (\delta_1(p,a), \delta_2(q,a))$
    - $F = \{ (r_1, r_2) \mid r_1 \in F_1 \text{ or } r_2 \in F_2 \}$

Union: Example

- $L_1 = \{ w \mid 00 \text{ is not a substring of } w\}$
- $L_2 = \{ w \mid w \text{ ends in } 01\}$



M_1



M_2

Union: Example

- $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$
  - $Q_1 = \{A, B, C\}$
  - $q_1 = A$
  - $F_1 = \{A, B\}$

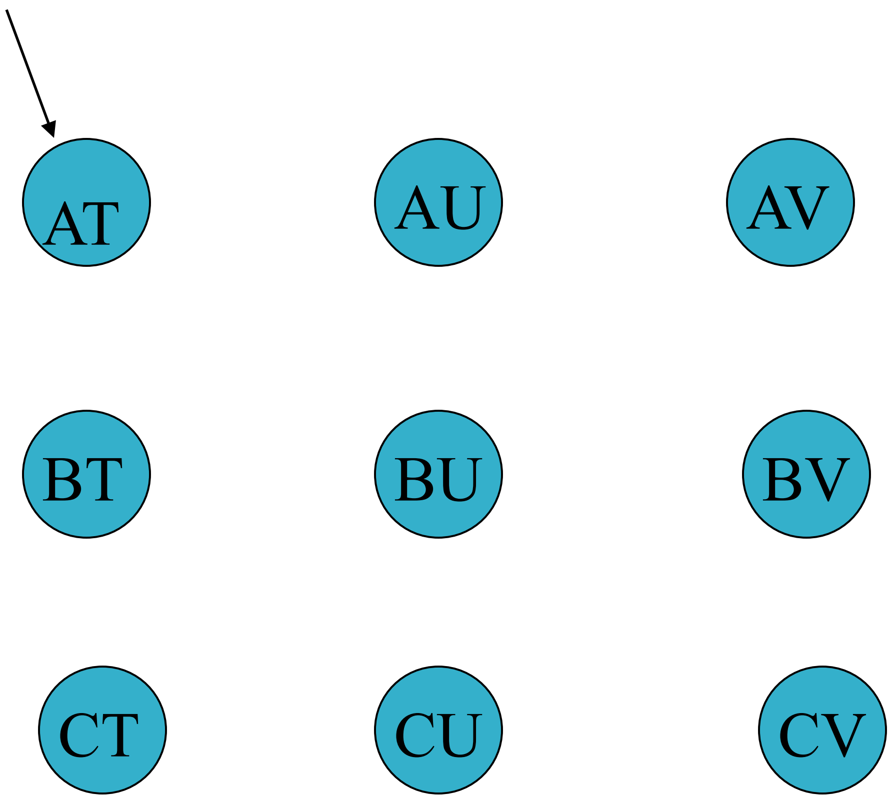
- $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$
  - $Q_2 = \{T, U, V\}$
  - $q_2 = T$
  - $F_2 = \{V\}$

Union: Example

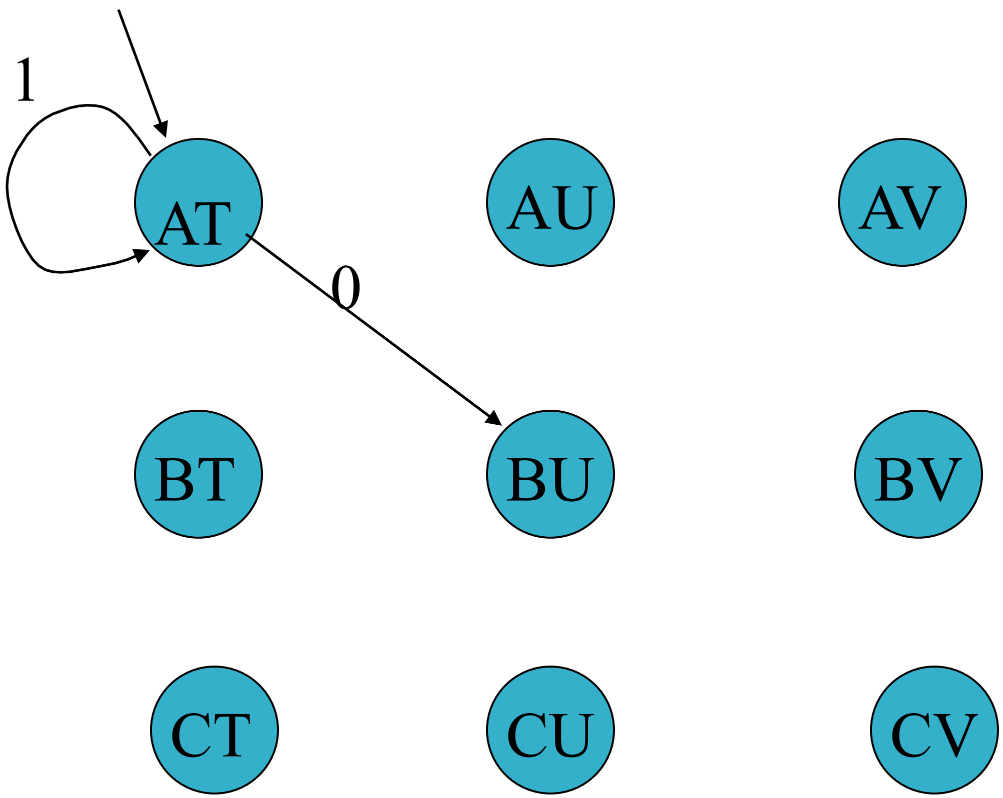
- $M = (Q, \Sigma, \delta, q_0, F)$
  - $Q = \{AT, AU, AV, BT, BU, BV, CT, CU, CV\}$
  - $q_0 = AT$
  - $F = \{AT, AU, AV, BT, BU, BV, CV\}$

  - Note that AT corresponds to the ordered pair (A,T), but we're just naming states here, so we can call them whatever we want

Union: Example



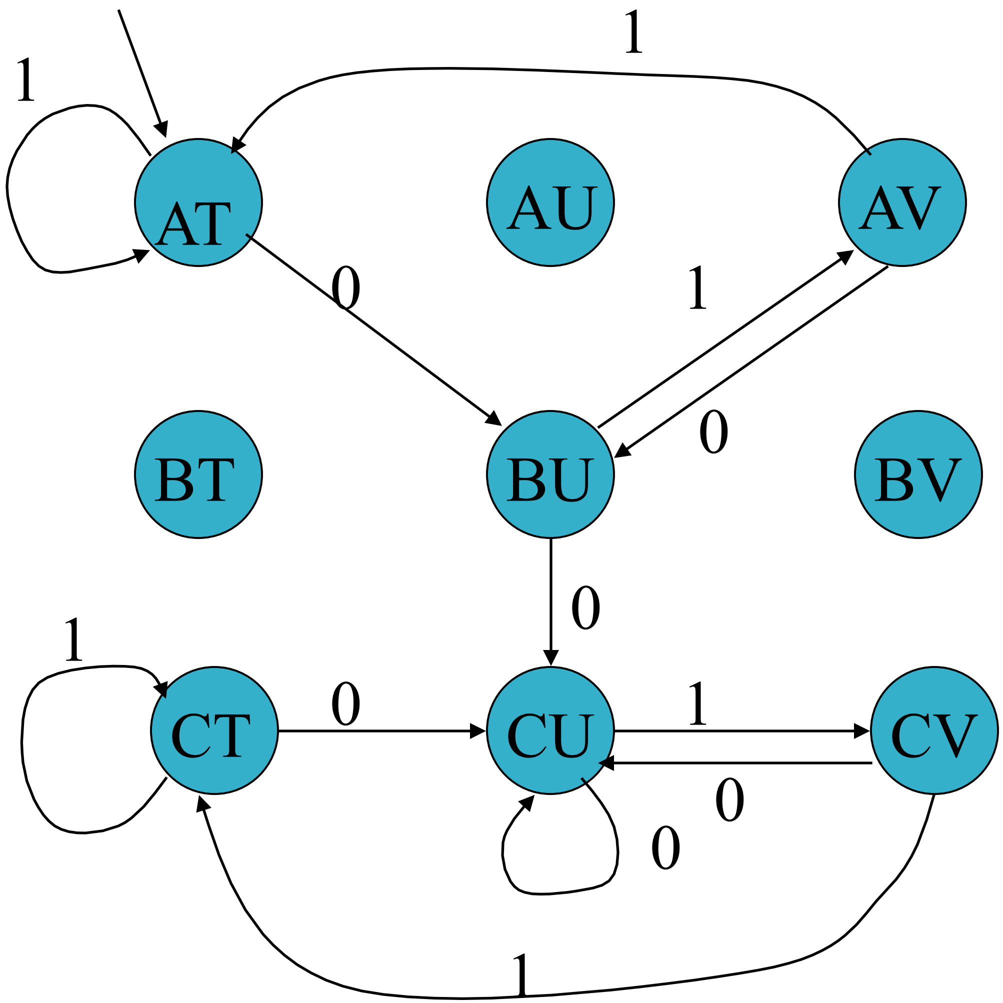
Union: Example



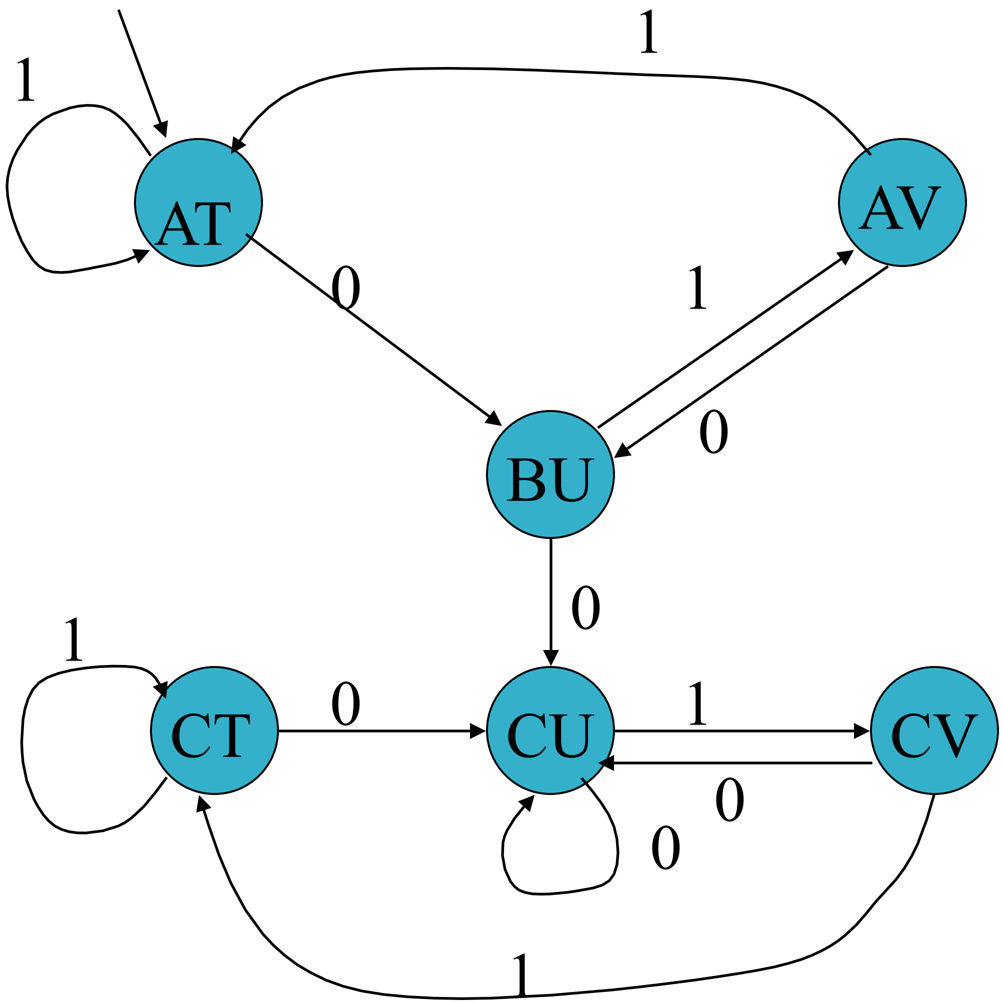
$\delta((A,T), 1) = (\delta_1(A,1), \delta_2(T,1)) = (A, T)$

$\delta((A,T), 0) = (\delta_1(A,0), \delta_2(T,0)) = (B, U)$

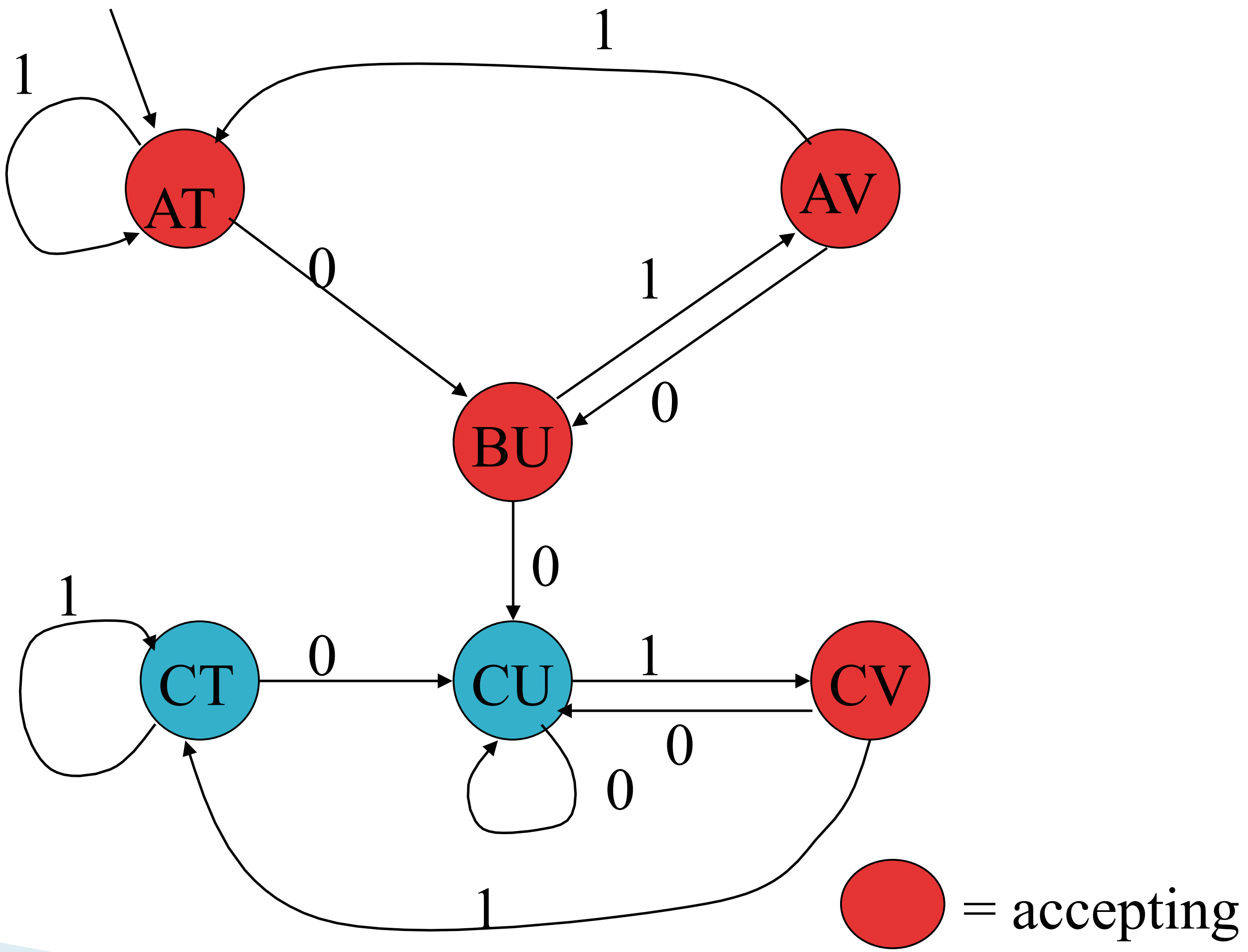
Union: Example



Union: Example



Union: Example



Closure of Union

- We've shown regular languages to be closed under Union
  - By definition
    - $L_1 = L(M_1)$ and $L_2 = L(M_2)$
  - Built a DFA, M that accepts $L_1 \cup L_2$
    - M simulates the simultaneous running of M_1 and M_2 on the same string.
    - Defined accepting states of M based on the operation.

Intersection and Difference

- Note that the same argument can be used to show that regular languages are closed under:
  - Intersection
  - Difference

  - What needs to change in the previous construction?

Intersection and Difference

- Set of accepting states
  - Union
    - $F = \{(p,q) \mid p \in F_1 \text{ or } q \in F_2\}$
  - Intersection
    - $F = \{(p,q) \mid p \in F_1 \text{ and } q \in F_2\}$
  - Difference
    - $F = \{(p,q) \mid p \in F_1 \text{ and } q \notin F_2\}$

Complementation

- How can we show regular languages are closed under complementation?
- Given machine $M = (Q, \Sigma, \delta, q_0, F)$

  - One way: construct $M' = (Q, \Sigma, \delta, q_0, Q-F)$

  - Another way: Consider language of all strings (Σ^*).
    - Just an accepting start state that loops to itself
    - This is a regular language
    - Complement: $A' = \Sigma^* - A$
      - Regular languages closed under difference

What about Concatenation and Kleene Star?

- Use same approach (constructive proof)
- Will need to introduce a new finite automata technique called NON-DETERMINISM…
- But…before we get to that, a couple additional concepts related to what we've seen so far.

Using Closure Properties

- Using closure properties of regular languages to show that a particular language, A, is not regular
  - Proof by contradiction
  - Need to use an additional language, B, that is already known/proven to be not regular
  - Assume A is regular
  - Show that language A when combined (via union, intersection, difference, or complementation) with some known regular language, yields language B
    - Contradicts closure properties of regular languages

A few language classes

- Regular languages
- Non-regular languages (all languages that aren't regular)

- Finite languages
  - languages containing a finite number of strings
- Infinite languages (all languages that aren't finite)

- Question: Are all finite languages regular?

Recall – Specifying Languages

- How do we specify languages?
  - If language is finite, we can list all of its strings.
    - L = {a, aa, aba, aca}
  - Using basic language operations
    - $L = \{aa, ab\}^* \cup \{b\}\{bb\}^*$
  - Descriptive:
    - $L = \{w \mid n_a(w) = n_b(w)\}$
    - L = {w | w begins with an a and ends with a b}

Regular Operations

- Some important operations over regular languages:
  - Union:
    - $A \cup B = \{ w \mid w \in A \text{ or } w \in B \}$
  - Concatenation:
    - $AB = \{ wx \mid w \in A \text{ and } x \in B \}$
  - Kleene Star
    - $A^* = \bigcup_{i=0}^{\infty} A^i = A^0 \cup A^1 \cup A^2 \cup A^3 \cup A^4 \ldots$

Examples Describing Languages using Regular Operations

- Consider the languages $\{0\}$, $\{1\}$, $\{\epsilon\}$, $\emptyset$ and use regular operations (union, concatenation, Kleene star) on these languages to represent:

- The language of all strings over $\{0,1\}$ that end with a 0
  - ANSWER: $(\{0\} \cup \{1\})^*\{0\}$

Note we're combining languages, so everything must be expressed as a language (i.e. a *set* of strings in set notation, even when the set is just the individual string 0 or 1)

Examples Describing Languages using Regular Operations

- Consider the languages {0}, {1}, {ϵ}, $\varnothing$ and use regular operations (union, concatenation, Kleene star) on these languages to represent:
  - The language of all strings over {0,1}:
    - With exactly one 0
    - With exactly one 0 or exactly one 1
    - With exactly two 0's
    - With an even number of 0's (zero 0's should be included)
    - With 1110 as a substring
    - Of even length