Speech Recognition
Lecture 2: Finite Automata and Finite-State Transducers

Mehryar Mohri
Courant Institute and Google Research
mohri@cims.nyu.com

Preliminaries

- Finite alphabet Σ, empty string ϵ.
- Set of all strings over Σ: Σ^* (free monoid).
- Length of a string $x \in \Sigma^*$: $|x|$.
- Mirror image or reverse of a string $x = x_1 \cdots x_n$:
$$x^R = x_n \cdots x_1.$$
- A language L: subset of Σ^*.

Rational Operations

- Rational operations over languages:
  - union: also denoted $L_1 + L_2$,

$$L_1 \cup L_2 = \{x \in \Sigma^* : x \in L_1 \vee x \in L_2\}.$$

  - concatenation:

$$L_1 \cdot L_2 = \{x = uv \in \Sigma^* : u \in L_1 \vee v \in L_2\}.$$

  - closure:

$$L^* = \bigcup_{n=0}^{\infty} L^n, \quad \text{where } L^n = \underbrace{L \cdots L}_{n}.$$

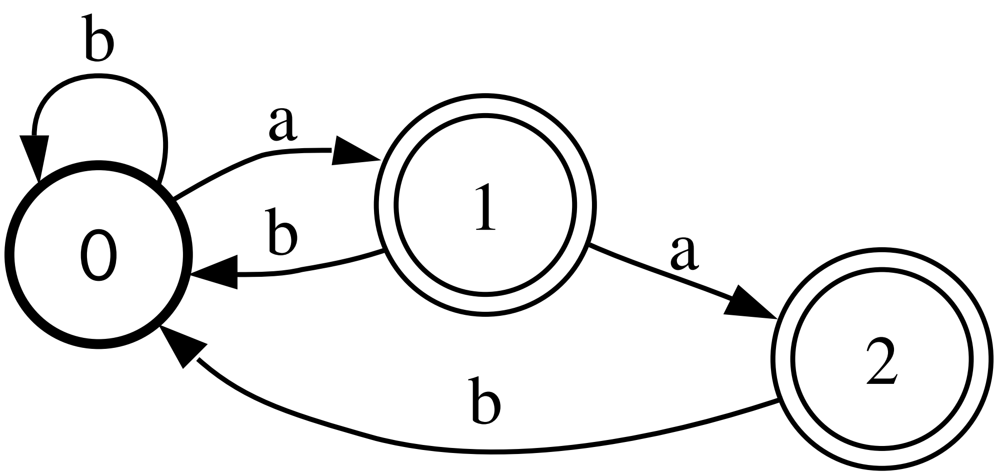
Regular or Rational Languages

- Definition: closure under rational operations of Σ^*. Thus, $\mathrm{Rat}(\Sigma^*)$ is the smallest subset $\mathcal{L}$ of 2^{Σ^*} verifying
  - $\emptyset \in \mathcal{L}$;
  - $\forall x \in \Sigma^*, \{x\} \in \mathcal{L}$;
  - $\forall L_1, L_2 \in \mathcal{L}, L_1 \cup L_2 \in \mathcal{L}, L_1 \cdot L_2 \in \mathcal{L}, L_1^* \in \mathcal{L}.$
- Examples of regular languages over $\Sigma = \{a, b, c\}$:
  - Σ^*, $(a+b)^*c$, ab^nc, $(a+(b+c)^*ba)^*cb$.

Finite Automata

- Definition: a finite automaton A over the alphabet Σ is 4-tuple (Q, I, F, E) where Q is a finite set of states, $I \subseteq Q$ a set of initial states, $F \subseteq Q$ a set of final states, and E a multiset of transitions which are elements of $Q \times (\Sigma \cup \{\epsilon\}) \times Q$.
  - a path π in an automaton $A = (Q, I, F, E)$ is an element of E^*.
  - a path from a state in I to a state in F is called an accepting path. Language $L(A)$ accepted by A: set of strings labeling accepting paths.

Finite Automata - Example

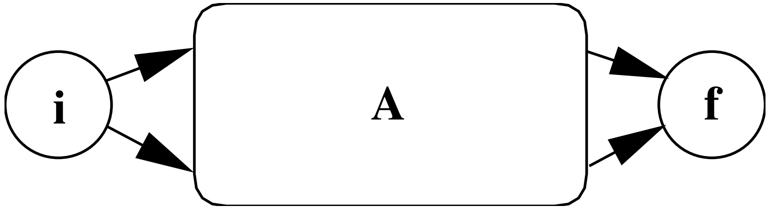


Finite Automata - Some Properties

- Trim: any state lies on some accepting path.
- Unambiguous: no two accepting paths have the same label.
- Deterministic: unique initial state, two transitions leaving the same state have different labels.
- Complete: at least one outgoing transition labeled with any alphabet element at any state.
- Acyclic: no path with a cycle.

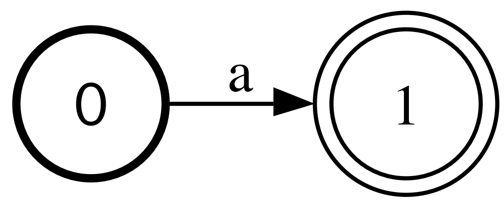
Normalized Automata

- Definition: a finite automaton is normalized if
  - it has a unique initial state with no incoming transition.
  - it has a unique final state with no outgoing transition.



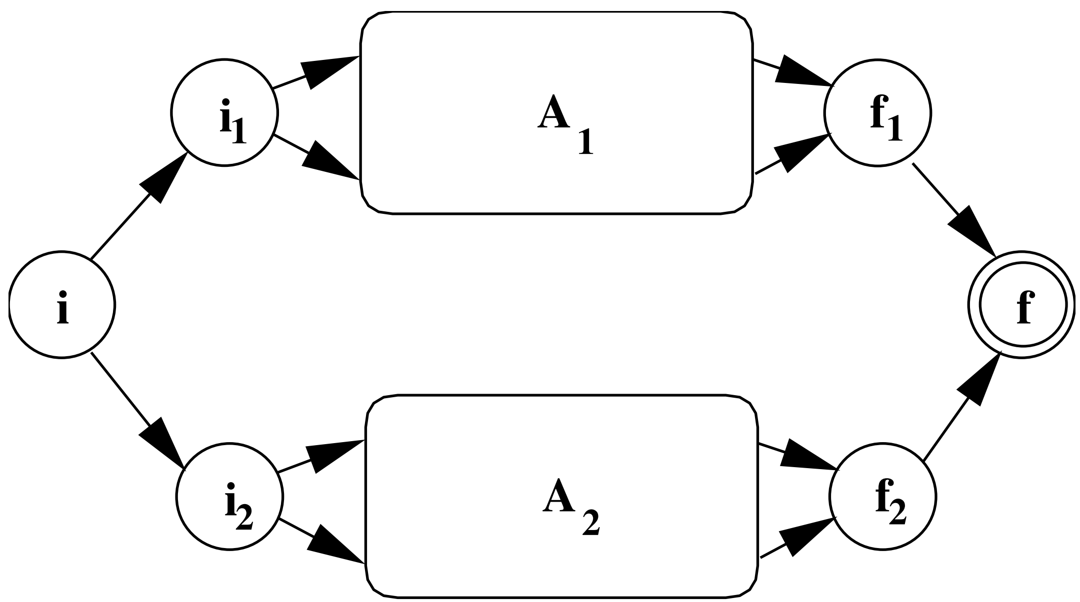
Elementary Normalized Automaton

- Definition: normalized automaton accepting an element $a \in \Sigma \cup \{\epsilon\}$ constructed as follows.



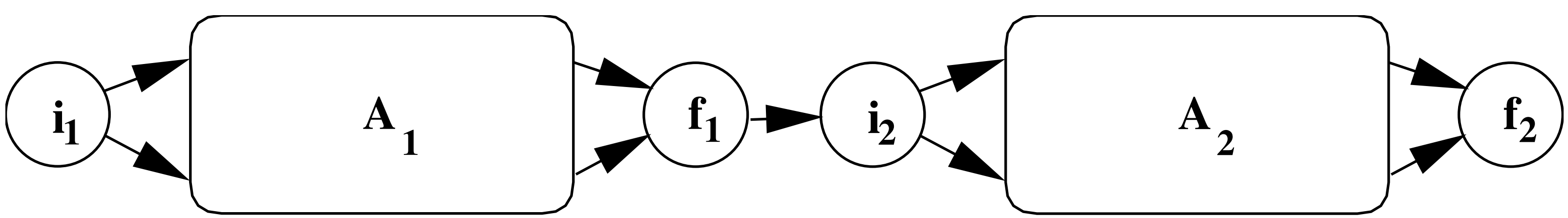
Normalized Automata: Union

- Construction: the union of two normalized automata is a normalized automaton constructed as follows.



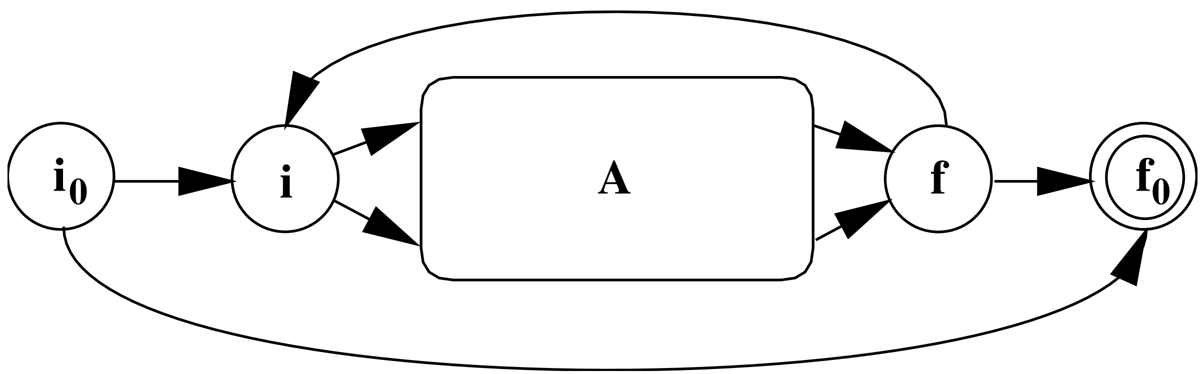
Normalized Automata: Concatenation

- Construction: the concatenation of two normalized automata is a normalized automaton constructed as follows.



Normalized Automata: Closure

- Construction: the closure of a normalized automaton is a normalized automaton constructed as follows.



Normalized Automata - Properties

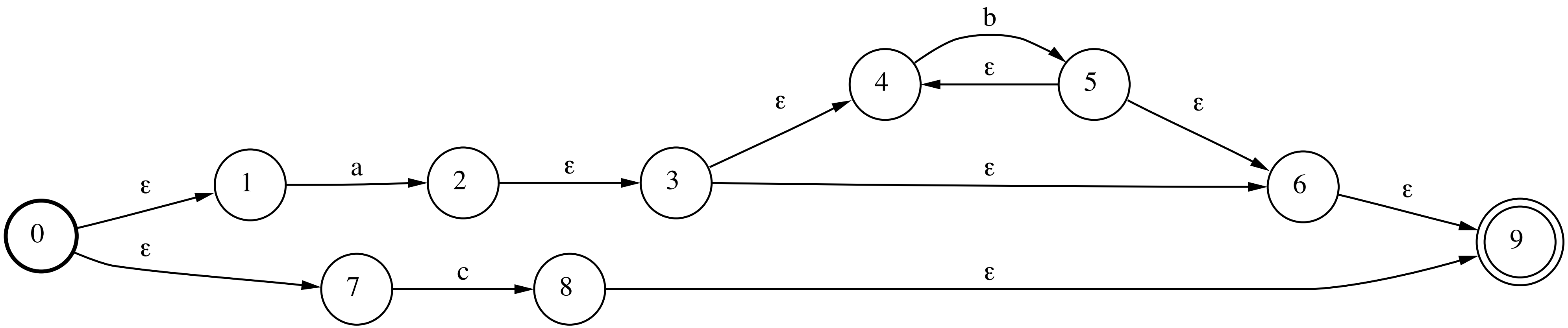
- Construction properties:
  - each rational operation require creating at most two states.
  - each state has at most two outgoing transitions.
  - the complexity of each operation is linear.

Thompson's Construction

(Thompson, 1968)

- Proposition: let r be a regular expression over the alphabet Σ. Then, there exists a normalized automaton A with at most $2|r|$ states representing r .
- Proof:
  - linear-time context-free parser to parse regular expression.
  - construction of normalized automaton starting from elementary expressions and following operations of the tree.

Thompson's Construction - Example



Normalized automaton for regular expression $ab^* + c$.

Regular Languages and Finite Automata

(Kleene, 1956)

- Theorem: A language is regular iff it can be accepted by a finite automaton.
- Proof: Let $A = (Q, I, F, E)$ be a finite automaton.
  - for $(i, j, k) \in [1, |Q|] \times [1, |Q|] \times [0, |Q|]$ define
    $$X_{ij}^k = \{i \to q_1 \to q_2 \to \ldots \to q_n \to j : n \geq 0, q_i \leq k\}.$$
  - X_{ij}^0 is regular for all (i, j) since E is finite.
  - by recurrence X_{ij}^k for all (i, j, k) since
    $$X_{ij}^{k+1} = X_{ij}^k + X_{i,k+1}^k (X_{k+1,k+1}^k)^* X_{k+1,j}^k.$$
  - $L(A) = \bigcup_{i \in I, f \in F} X_{if}^{|Q|}$ is thus regular.

Regular Languages and Finite Automata

- Proof: the converse holds by Thompson's construction.
- Notes:
  - a more general theorem (Schützenberger, 1961) holds for weighted automata.
  - not all languages are regular, e.g., $L = \{a^n b^n : n \in \mathbb{N}\}$ is not regular. Let A be an automaton. If $L \subseteq L(A)$, then for large enough n, $a^n b^n$ corresponds to a path with a cycle: $a^n b^n = a^p u b^q$, $a^p u^* b^q \subseteq L(A)$, which implies $L(A) \neq L$.

Left Syntactic Congruence

- Definition: for any language $L \subseteq \Sigma^*$, the left syntactic congruence is the equivalence relation defined by

$$u \equiv_L v \Leftrightarrow u^{-1}L = v^{-1}L,$$

where for any $u \in \Sigma^*$, $u^{-1}L$ is defined by

$$u^{-1}L = \{w \colon uw \in L\}.$$

  - $u^{-1}L$ is sometimes called the partial derivative of L with respect to u and denoted $\frac{\partial L}{\partial u}$.

Regular Languages - Characterization

- Theorem: a language L is regular iff the set of $u^{-1}L$ is finite ($\equiv_L$ has a finite index).
- Proof: let $A = (Q, I, F, E)$ be a trim deterministic automaton accepting L (existence seen later).
  - let δ the partial transition function. Then,

    $$u \; R \; v \Leftrightarrow \delta(i, u) = \delta(i, v).$$

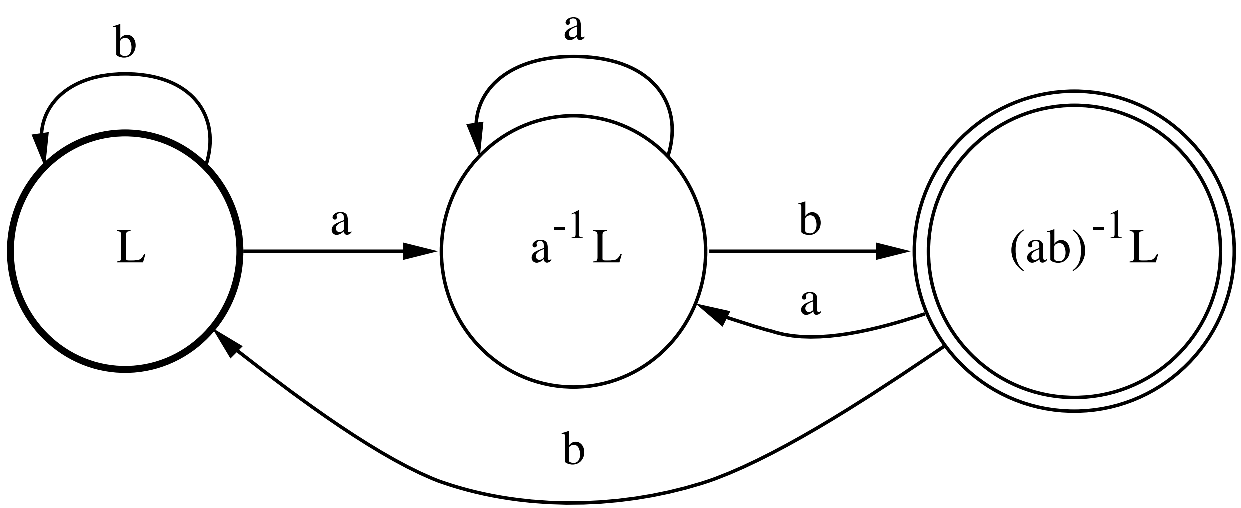
    also defines an eq. relation with index $|Q|$.
  - since $\delta(i, u) = \delta(i, v) \Rightarrow u^{-1}L = v^{-1}L$, the index of $\equiv_L$ is at most $|Q|$, thus finite.

Regular Languages - Characterization

- Proof: conversely, if the set of $u^{-1}L$ is finite, then the automaton $A = (Q, I, F, E)$ defined by
  - $Q = \{u^{-1}L \colon u \in \Sigma^*\}$;
  - $i = \epsilon^{-1}L = L$, $I = \{i\}$;
  - $F = \{u^{-1}L \colon u \in L\}$;
  - $E = \{(u^{-1}L, a, (ua)^{-1}L) \colon u \in \Sigma^*\}$; is well defined since $u^{-1}L = v^{-1}L \Rightarrow (ua)^{-1}L = (va)^{-1}L$ and accepts exactly L.

Illustration

- Minimal deterministic automaton for $(a+b)^*ab$:

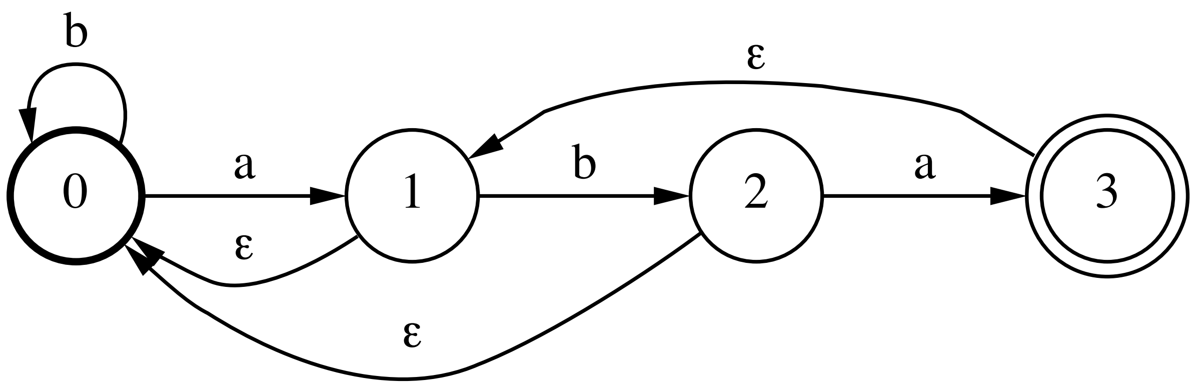


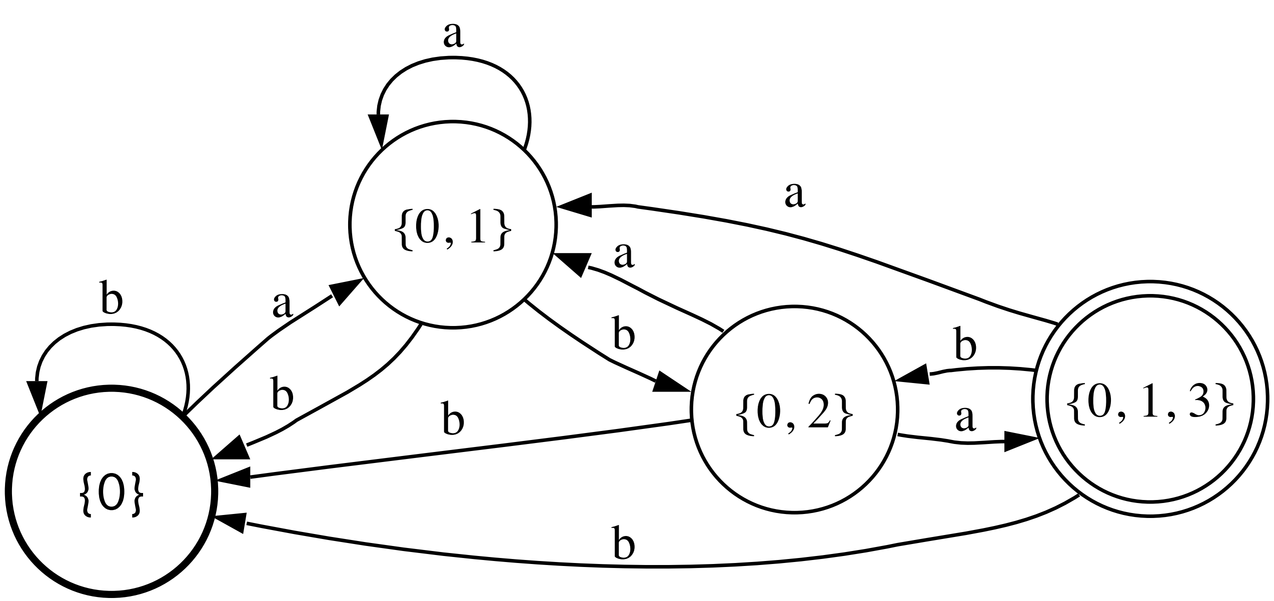
ε-Removal

- **Theorem**: any finite automaton $A = (Q, I, F, E)$ admits an equivalent automaton with no ε-transition.

- **Proof**: for any state $q \in Q$, let $\epsilon[q]$ denote the set of states reached from q by paths labeled with ϵ. Define $A' = (Q', I', F', E')$ by
  - $Q' = \{\epsilon[q] \colon q \in Q\}$, $I' = \bigcup_{q \in I} \epsilon[q]$, $F' = \{\epsilon[q] \colon \epsilon[q] \cap F \neq \emptyset\}$.
  - $E' = \{(\epsilon[p], a, \epsilon[q]) : \exists (p', a, q') \in E, p' \in \epsilon[p], q' \in \epsilon[q]\}$.

ε-Removal - Illustration



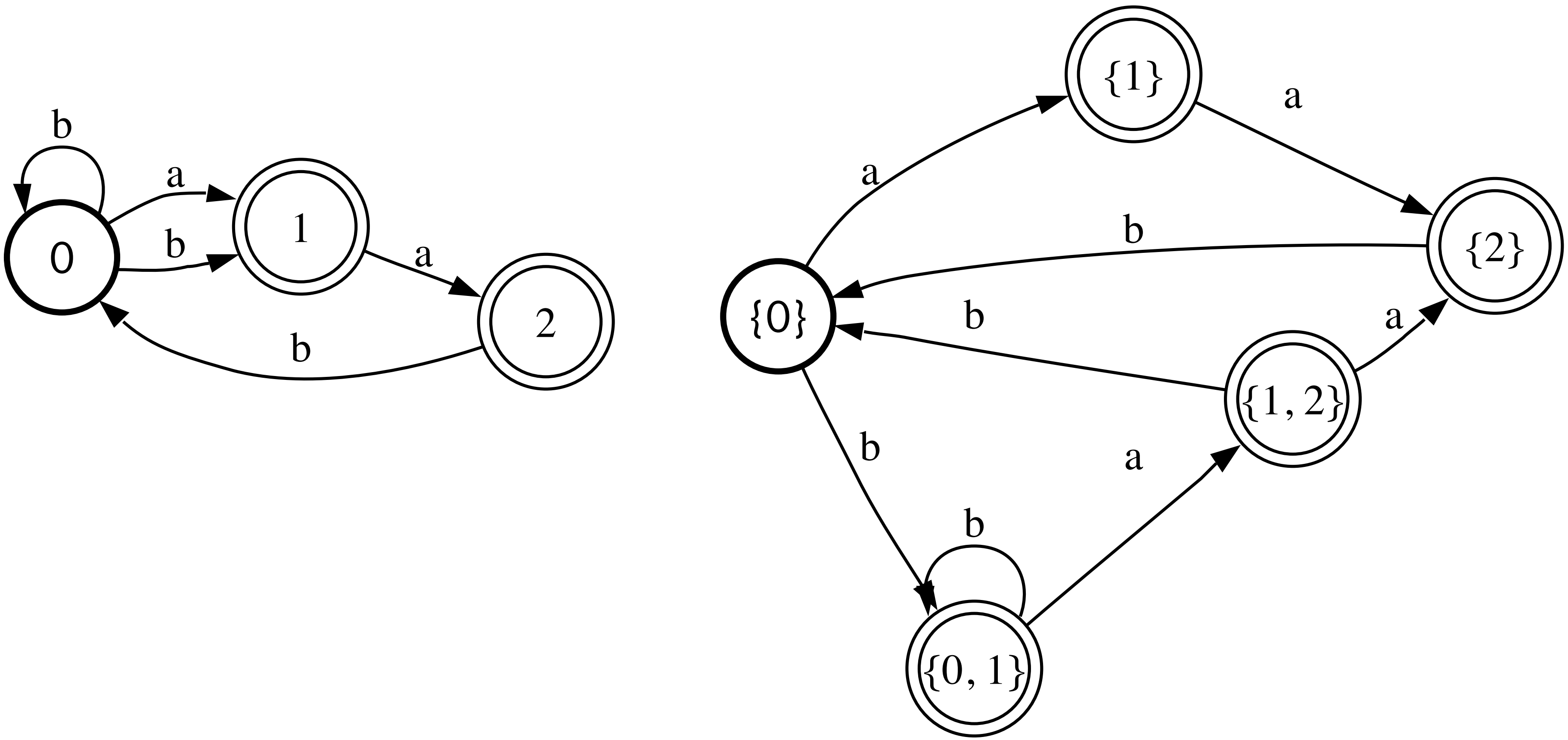


Determinization

- **Theorem**: any automaton $A = (Q, I, F, E)$ without ϵ-transitions admits an equivalent deterministic automaton.

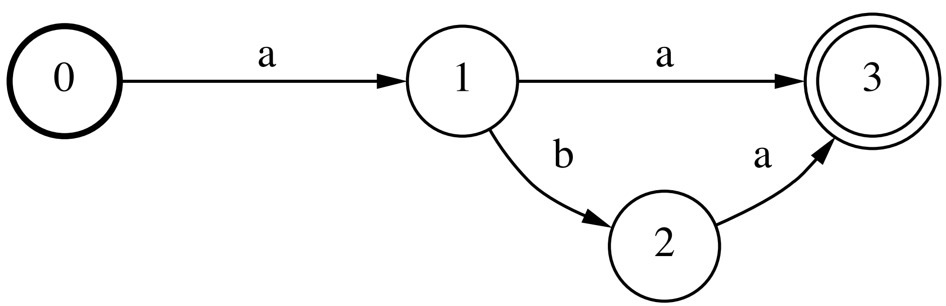
- **Proof**: Subset construction: $A' = (Q', I', F', E')$ with
  - $Q' = 2^Q$.
  - $I' = \{s \in Q' \colon s \cap I \neq \emptyset\}$.
  - $F' = \{s \in Q' \colon s \cap F \neq \emptyset\}$.
  - $E' = \{(s, a, s') \colon \exists (q, a, q') \in E, q \in s, q' \in s'\}$.

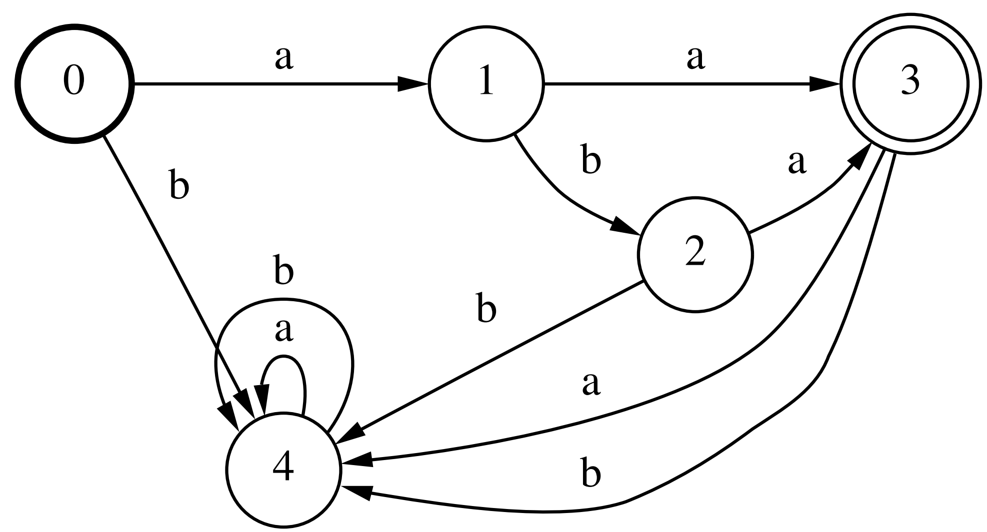
Determinization - Illustration



Completion

- Theorem: any deterministic automaton admits an equivalent complete deterministic automaton.
- Proof: constructive, see example.



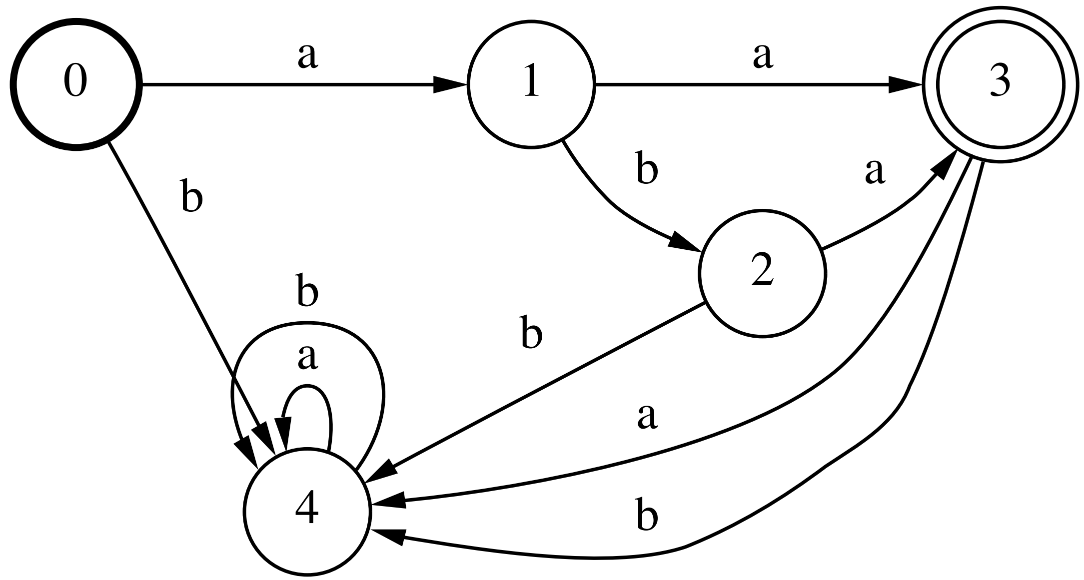


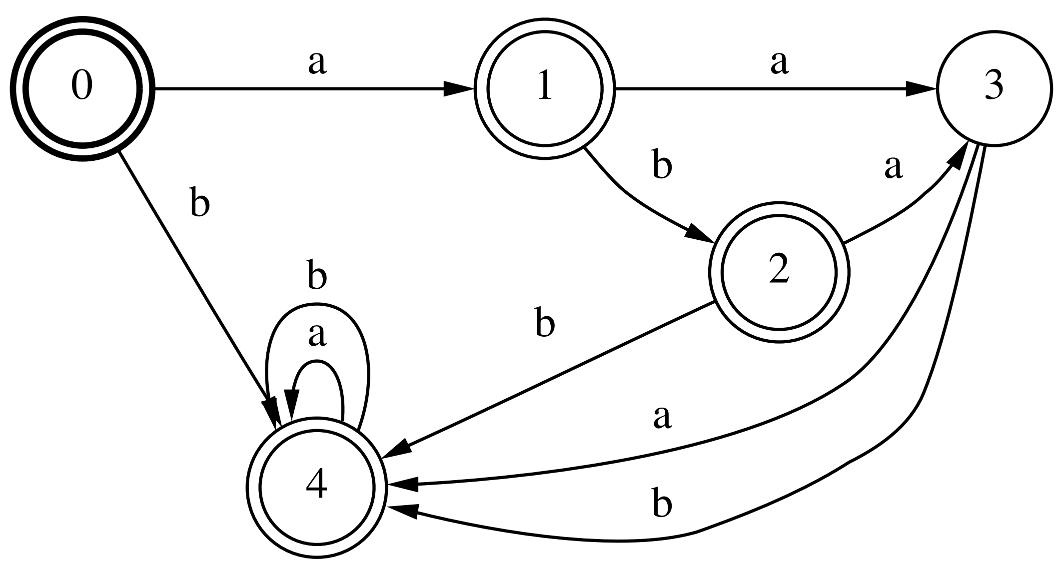
Complementation

- **Theorem**: let $A = (Q, I, F, E)$ be a deterministic automaton, then there exists a deterministic automaton accepting $\overline{L(A)}$.

- **Proof**: by a previous theorem, we can assume A complete. The automaton $B = (\Sigma, Q, I, Q - F, E)$ obtained from A by making non-final states final and final states non-final exactly accepts $\overline{L(A)}$.

Complementation - Ilustration





Regular Languages - Properties

- Theorem: regular languages are closed under rational operations, intersection, complementation, reversal, morphism, inverse morphism, and quotient with any set.
- Proof: closure under rational operations holds by definition.
  - intersection: use De Morgan's law.
  - complementation: use algorithm.
  - others: algorithms and equivalence relation.

Rational Relations

- Definition: closure under rational operations of the monoid $\Sigma^* \times \Delta^*$, where Σ and Δ are finite alphabets, denoted by $\mathrm{Rat}(\Sigma^* \times \Delta^*)$.
    - examples: $(a, b)^*$, $(a, b)^*(bb, a) + (b, a)$.

Rational Relations - Characterization

(Nivat, 1968)

- **Theorem**: $R \subseteq \mathrm{Rat}(\Sigma^* \times \Delta^*)$ is a rational relation iff there exists a regular language $L \subseteq (\Sigma \cup \Delta)^*$ such that

$$R = \{(\pi_\Sigma(x), \pi_\Delta(x)) : x \in L\}$$

where π_Σ is the projection of $(\Sigma \cup \Delta)^*$ over Σ^* and π_Δ the projection over Δ^*.

- **Proof**: use surjective morphism

$$\begin{aligned} \pi : (\Sigma \cup \Delta)^* &\rightarrow (\Sigma^* \times \Delta^*) \\ x &\rightarrow (\pi_\Sigma(x), \pi_\Delta(x)). \end{aligned}$$

Transductions

- Definition: a function $\tau\colon \Sigma^* \to 2^{\Delta^*}$ is called a transduction from Σ^* to Δ^*.
  - relation associate to τ:

$$R(\tau) = \{(x, y) \in \Sigma^* \times \Delta^* : y \in \tau(x)\}.$$

  - transduction associated to a relation:

$$\forall x \in \Sigma^*, \tau(x) = \{y : (x, y) \in R\}.$$

  - rational transductions: transductions with rational relations.

Finite-State Transducers

- **Definition:** a finite-state transducer T over the alphabets Σ and Δ is 4-tuple where Q is a finite set of states, $I \subseteq Q$ a set of initial states, $F \subseteq Q$ a set of final states, and E a multiset of transitions which are elements of $Q \times (\Sigma \cup \{\epsilon\}) \times (\Delta \cup \{\epsilon\}) \times Q$.

  - T defines a relation via the pair of input and output labels of its accepting paths,

$$R(T) = \{(x, y) \in \Sigma^* \times \Delta^* : I \xrightarrow{x:y} F\}.$$

Rational Relations and Transducers

- Theorem: a transduction is rational iff it can be realized by a finite-state transducer.
- Proof: Nivat's theorem combined with Kleene's theorem, and construction of a normalized transducer from a finite-state transducer.

References

- Kleene, S. C.1956. Representation of events in nerve nets and finite automata. *Automata Studies*.
- Nivat, Maurice. 968. Transductions des langages de Chomsky. Annales 18, Institut Fourier.
- Schützenberger, Marcel~Paul. 1961. On the definition of a family of automata. *Information and Control*, 4
- Thompson, K. 1968. Regular expression search algorithm. *Comm. ACM*, 11.