Part III.
Models for Regular Languages

Regular Expressions (RE): Definition

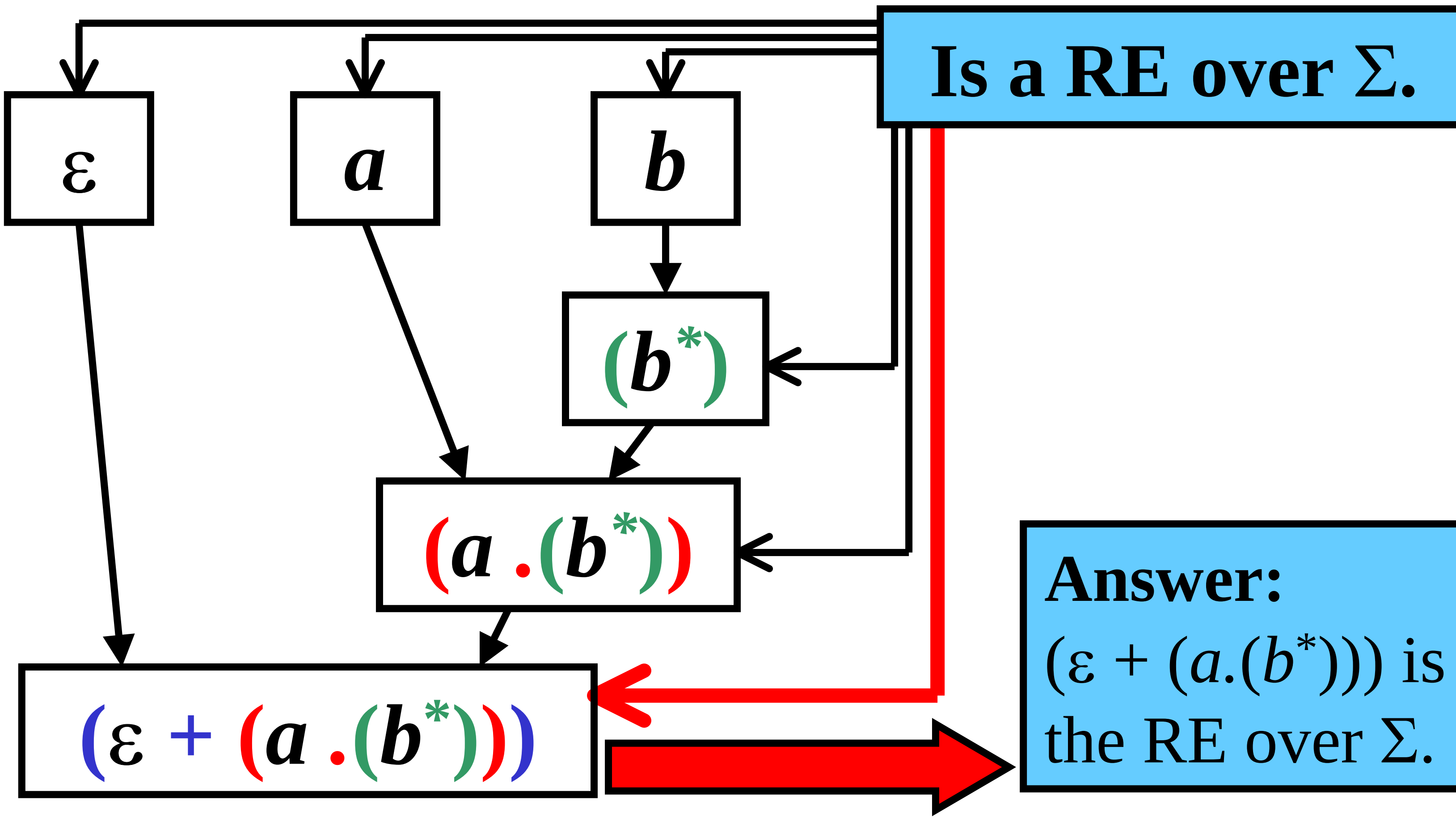
Gist: Expressions with operators ., +, and * that denote concatenation, union, and iteration, respectively.

Definition: Let Σ be an alphabet. The ***regular expressions*** over Σ and the ***languages they denote*** are defined as follows:

- ∅ is a RE denoting the empty set
- ε is a RE denoting {ε}
- a, where $a \in \Sigma$, is a RE denoting $\{a\}$
- Let r and s be regular expressions denoting the languages L_r and L_s, respectively; then
  - $(r.s)$ is a RE denoting $L = L_r L_s$
  - $(r + s)$ is a RE denoting $L = L_r \cup L_s$
  - (r^*) is a RE denoting $L = L_r^*$

Regular Expressions: Example

Question: Is $(\varepsilon + (a.(b^*)))$ the regular expression over $\Sigma = \{a, b\}$ **?**



Simplification

1) Reduction of the number of parentheses by

Precedences: $^{*} > . > +$

2) Expression $r.s$ is simplified to rs

3) Expression rr^{*} or $r^{*}r$ is simplified to r^{+}

Example:

$((\boldsymbol{a}.(\boldsymbol{a}^{*})) + ((\boldsymbol{b}^{*}).\boldsymbol{b}))$ can be written as $\boldsymbol{a} . \boldsymbol{a}^{*} + \boldsymbol{b}^{*}.\boldsymbol{b}$,

and $\boldsymbol{a} . \boldsymbol{a}^{*} + \boldsymbol{b}^{*}.\boldsymbol{b}$ can be written as $\boldsymbol{a}^{+} + \boldsymbol{b}^{+}$

Regular Language (RL)

Gist: Every RE denotes a regular language

Definition: Let L be a language. L is a *regular language* (RL) if there exists a regular expression r that denotes L.

Denotation: $L(r)$ means the language denoted by r.

Examples:

$r_1 = ab + ba$ denotes $L_1 = \{ab, ba\}$

$r_2 = a^+b^*$ denotes $L_2 = \{a^nb^m: n \geq 1, m \geq 0\}$

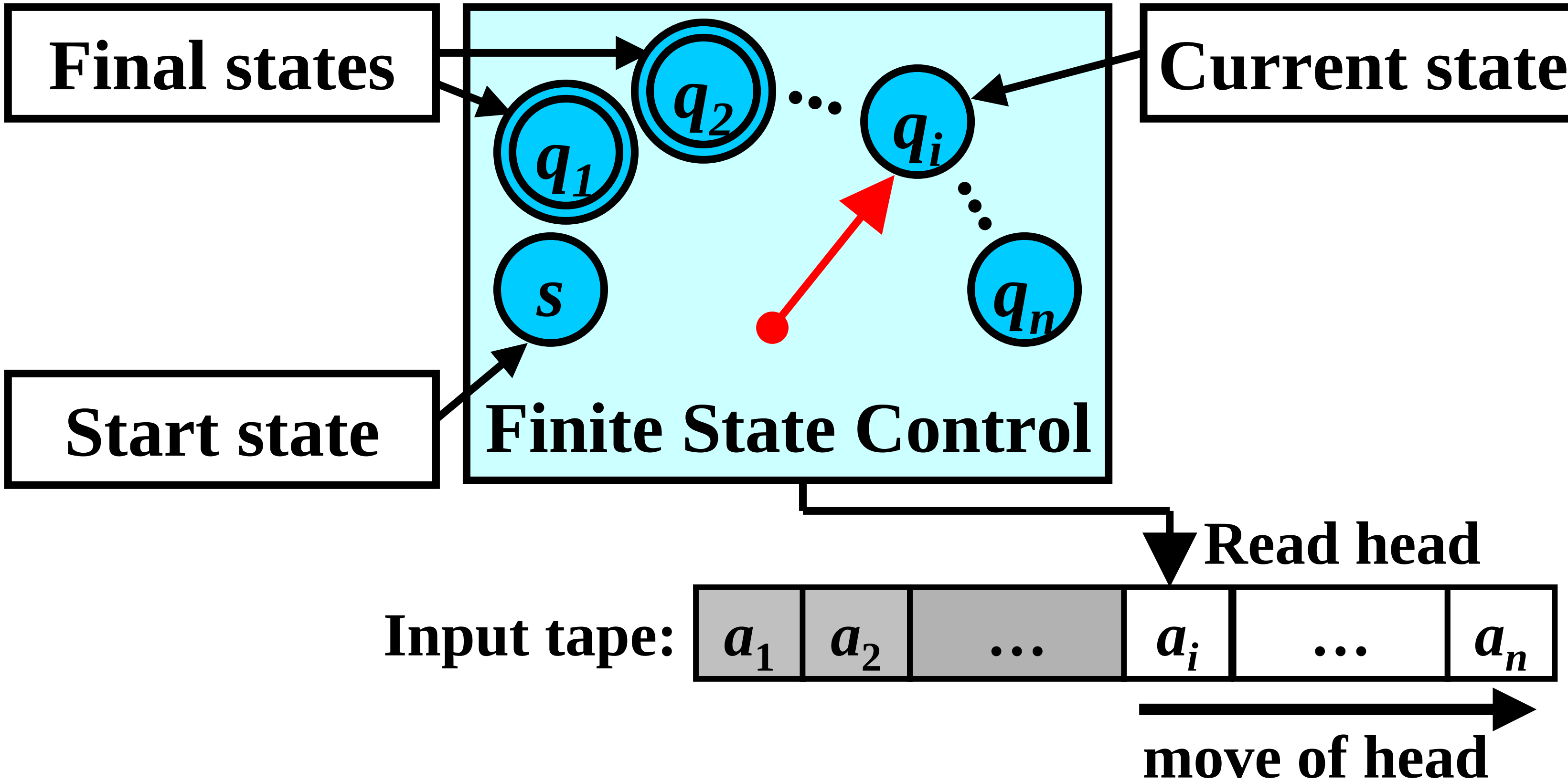
$r_3 = ab(a + b)^*$ denotes $L_3 = \{x: ab \text{ is prefix of } x\}$

$r_4 = (a + b)^*ab(a + b)^*$ denotes $L_4 = \{x: ab \text{ is substring of } x\}$

L_1, L_2, L_3, L_4 are regular languages over Σ

Finite Automata (FA)

Gist: The simplest model of computation based on a finite set of states and computational rules.



Finite Automata: Definition

Definition: *A finite automaton* (FA) is a 5-tuple:

$$M = (Q, \Sigma, R, s, F), \text{ where}$$

- Q is a *finite set of states*
- Σ is an *input alphabet*
- R is a *finite set of rules* of the form: $pa \rightarrow q$,
  where $p, q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$
- $s \in Q$ is the *start state*
- $F \subseteq Q$ is a set of *final states*

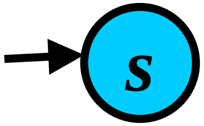
Mathematical note on rules:

- Strictly mathematically, R is a relation from $Q \times (\Sigma \cup \{\varepsilon\})$ to Q
- Instead of (pa, q), however, we write the rule as $pa \rightarrow q$

- $pa \rightarrow q$ means that with a, M can move from p to q
- if $a = \varepsilon$, no symbol is read

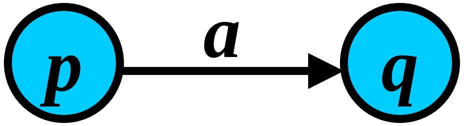
Graphical Representation

denotes a state $q \in Q$



denotes the start state $s \in Q$

denotes a final state $f \in F$

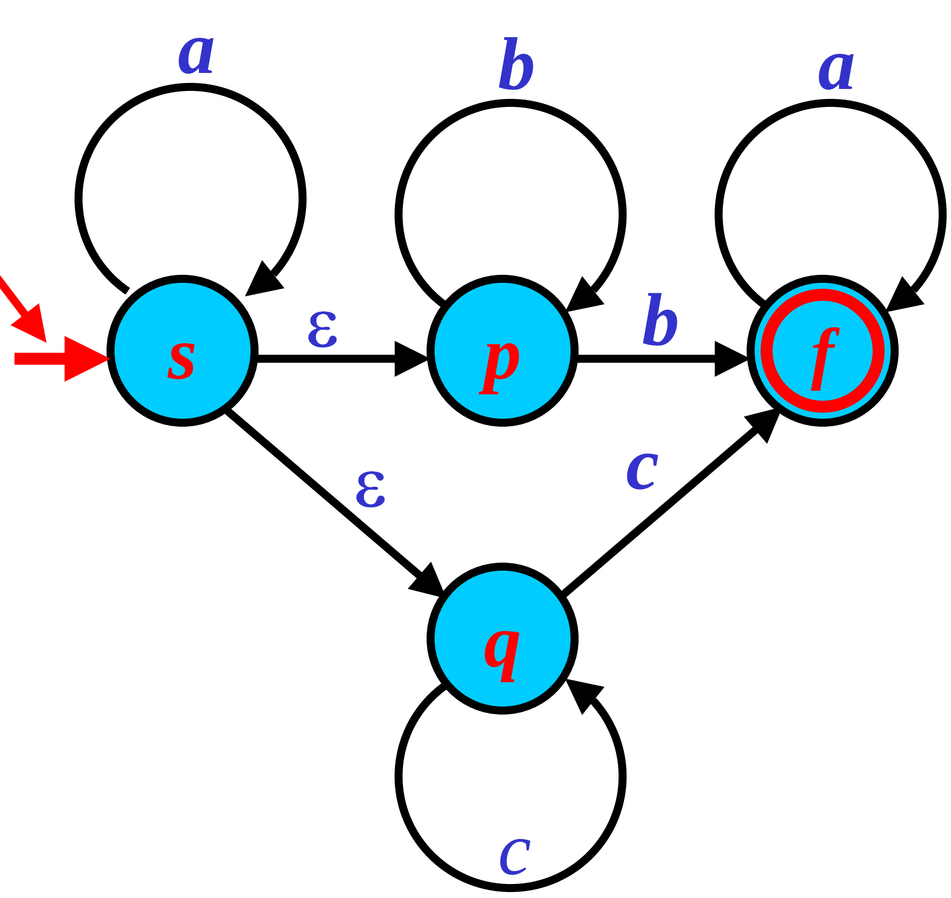


denotes $pa \rightarrow q \in R$

Graphical Representation: Example

$M = (Q, \Sigma, R, s, F)$,
where:

- $Q = \{s, p, q, f\}$;
- $\Sigma = \{a, b, c\}$;
- $R = \{sa \rightarrow s,$
  $s \rightarrow p,$
  $pb \rightarrow p,$
  $pb \rightarrow f,$
  $s \rightarrow q,$
  $qc \rightarrow q,$
  $qc \rightarrow f,$
  $fa \rightarrow f\}$;
- $F = \{f\}$



Tabular Representation

- **Columns:** Member of $\Sigma \cup \{\varepsilon\}$
- **Rows:** States of Q
- **First row:** The start state
- **Underscored:** Final states

	…	a	…	ε
s				
…				
p		$t(p, a)$		
…				
$\underline{f}$				

$$t(p, a) = \{q : pa \rightarrow q \in R\}$$

Tabular Representation: Example

$M = (Q, \Sigma, R, s, F)$, where:

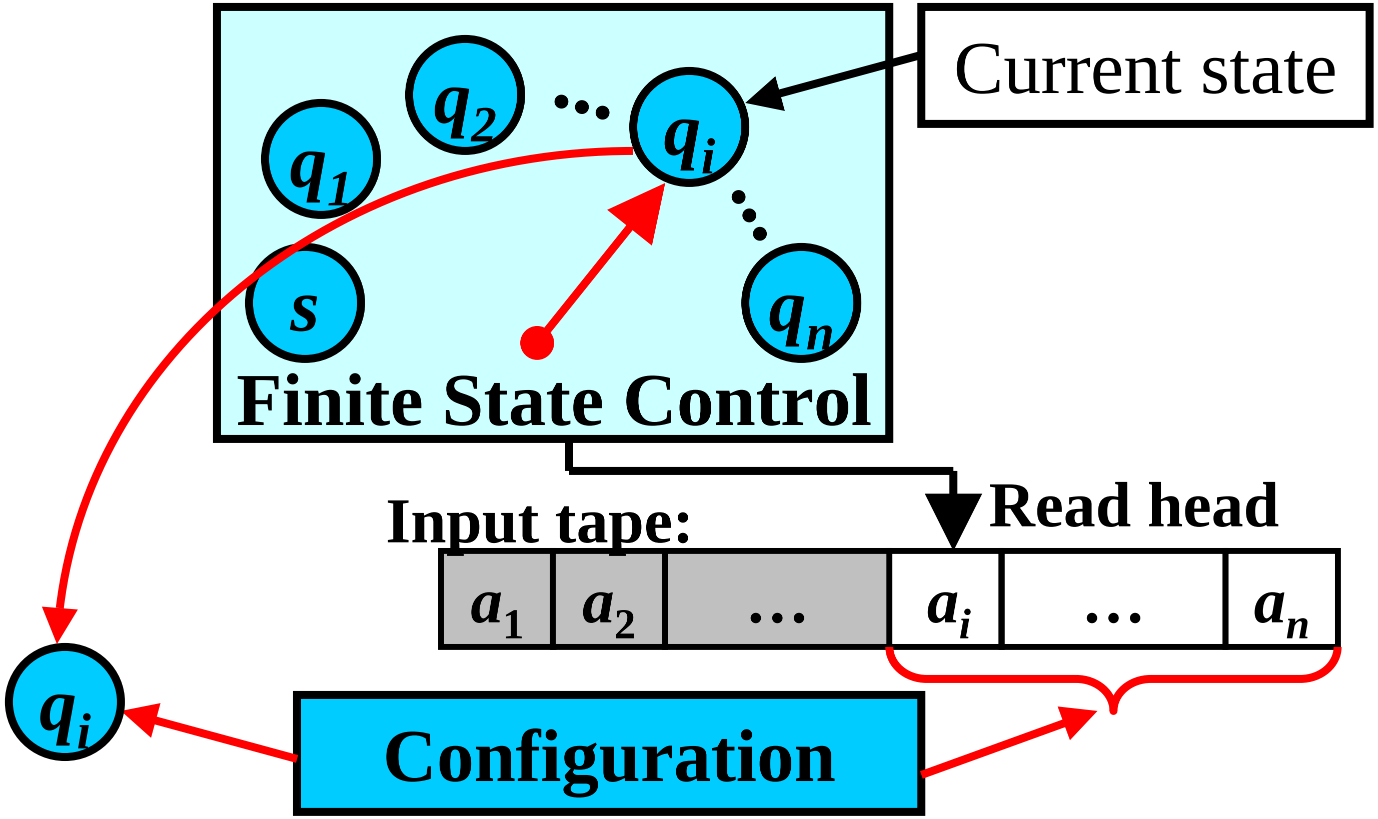
- $Q = \{s, p, q, f\}$;
- $\Sigma = \{a, b, c\}$;
- $R = \{sa \rightarrow s,$
  $s \rightarrow p,$
  $pb \rightarrow p,$
  $pb \rightarrow f,$
  $s \rightarrow q,$
  $qc \rightarrow q,$
  $qc \rightarrow f,$
  $fa \rightarrow f\}$;
- $F = \{f\}$

	a	b	c	ε
s	$\{s\}$	$\emptyset$	$\emptyset$	$\{p, q\}$
p	$\emptyset$	$\{p, f\}$	$\emptyset$	$\emptyset$
q	$\emptyset$	$\emptyset$	$\{q, f\}$	$\emptyset$
$\underline{f}$	$\{f\}$	$\emptyset$	$\emptyset$	$\emptyset$

Configuration

Gist: Instance description of FA

Definition: Let $M = (Q, \Sigma, R, s, F)$ be a FA.
A configuration of M is a string $\chi \in Q\Sigma^*$



Move

Gist: Computational step of FA

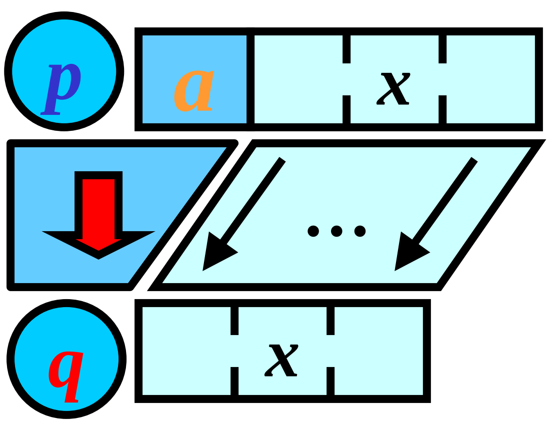
Definition: Let pax and qx be two configurations of M, where $p, q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, and $x \in \Sigma^*$. Let $r = pa \rightarrow q \in R$ be a rule. Then M makes a *move* from pax to qx according to r, written as $pax \vdash qx$ [r] or, simply, $pax \vdash qx$

Note: if $a = \varepsilon$, no input symbol is read

Configuration:

Rule: $pa \rightarrow q$

New configuration:



Sequence of Moves 1/2

Gist: Several consecutive computational steps

Definition: Let χ be a configuration. M makes *zero moves* from χ to χ; in symbols,

$$\chi \vdash^{0} \chi\ [\varepsilon] \text{ or, simply, } \chi \vdash^{0} \chi$$

Definition: Let $\chi_0, \chi_1, ..., \chi_n$ be a sequence of configurations, $n \geq 1$, and $\chi_{i-1} \vdash \chi_i\ [r_i]$, $r_i \in R$, for all $i = 1, ..., n$; that is,

$$\chi_0 \vdash \chi_1\ [r_1] \vdash \chi_2\ [r_2] \ldots \vdash \chi_n\ [r_n]$$

Then M makes *n moves* from χ_0 to χ_n:

$$\chi_0 \vdash^{n} \chi_n\ [r_1 ... r_n] \text{ or, simply, } \chi_0 \vdash^{n} \chi_n$$

Sequence of Moves 2/2

If $\chi_0 \vdash^n \chi_n\,[\rho]$ for some $n \geq 1$, then

$$\chi_0 \vdash^+ \chi_n\,[\rho].$$

If $\chi_0 \vdash^n \chi_n\,[\rho]$ for some $n \geq 0$, then

$$\chi_0 \vdash^* \chi_n\,[\rho].$$

Example: Consider

$pabc \vdash qbc\ [\mathbf{1}\!: pa \to q]$, and $qbc \vdash rc\ [\mathbf{2}\!: qb \to r]$.

Then,

$$pabc \vdash^2 rc\ [\mathbf{1\ 2}],$$

$$pabc \vdash^+ rc\ [\mathbf{1\ 2}],$$

$$pabc \vdash^* rc\ [\mathbf{1\ 2}]$$

Accepted Language

Gist: ***M*** **accepts** ***w*** **if it can completely read** ***w*** **by a sequence of moves from** ***s*** **to a final state**

Definition: Let $M = (Q, \Sigma, R, s, F)$ be a FA. The *language accepted by M*, $L(M)$, is defined as:

$$L(M) = \{w: w \in \Sigma^*, sw \vdash^* f, f \in F\}$$

$M = (Q, \Sigma, R, s, F)$:

if $q_n \in F$ then $w \in L(M)$;
otherwise, $w \notin L(M)$

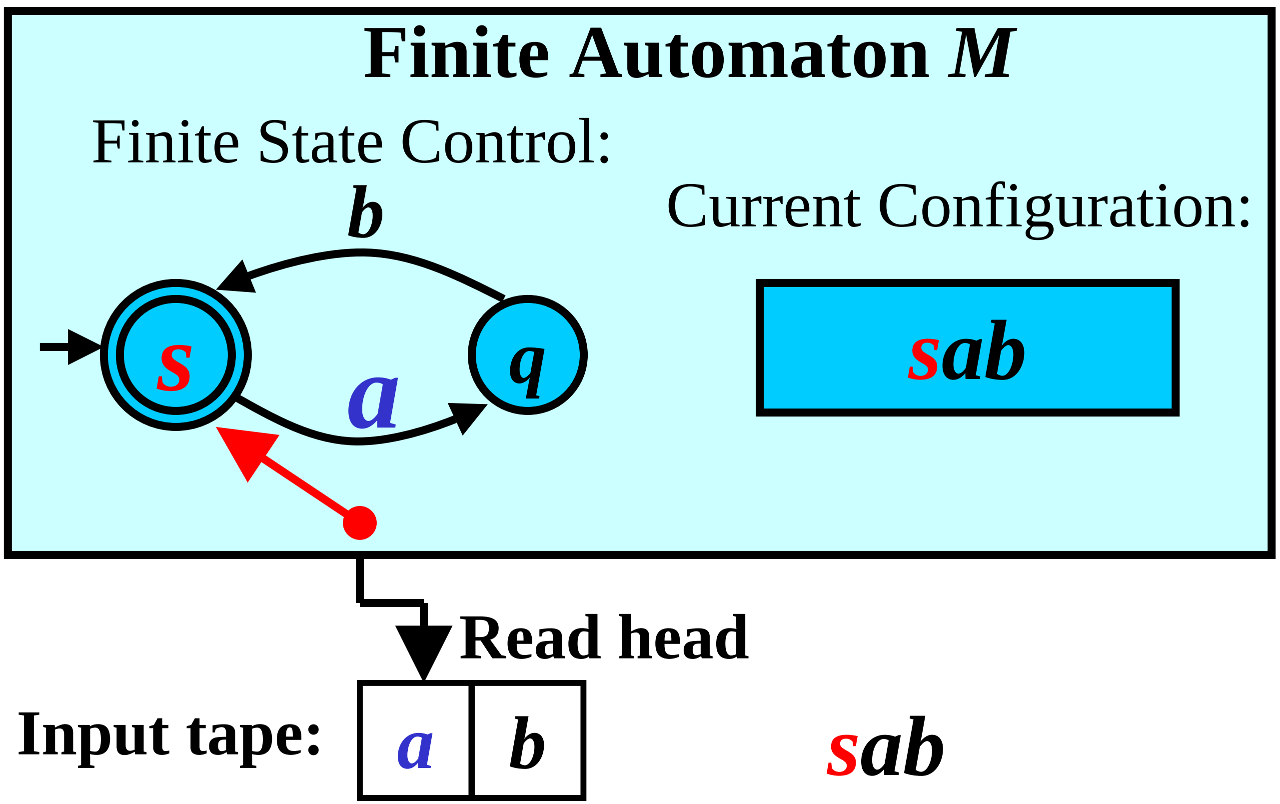
$$\underbrace{s a_1 a_2 \ldots a_n}_{w} \vdash q_1 a_2 \ldots a_n \vdash \ldots \vdash q_{n-1} a_n \vdash q_n$$

FA: Example 1/3

$M = (Q, \Sigma, R, s, F)$, where:

$Q = \{s, q\}$, $\Sigma = \{a, b\}$, $R = \{sa \rightarrow q, qb \rightarrow s\}$, $F = \{s\}$

Question: $ab \in L(M)$ **?**

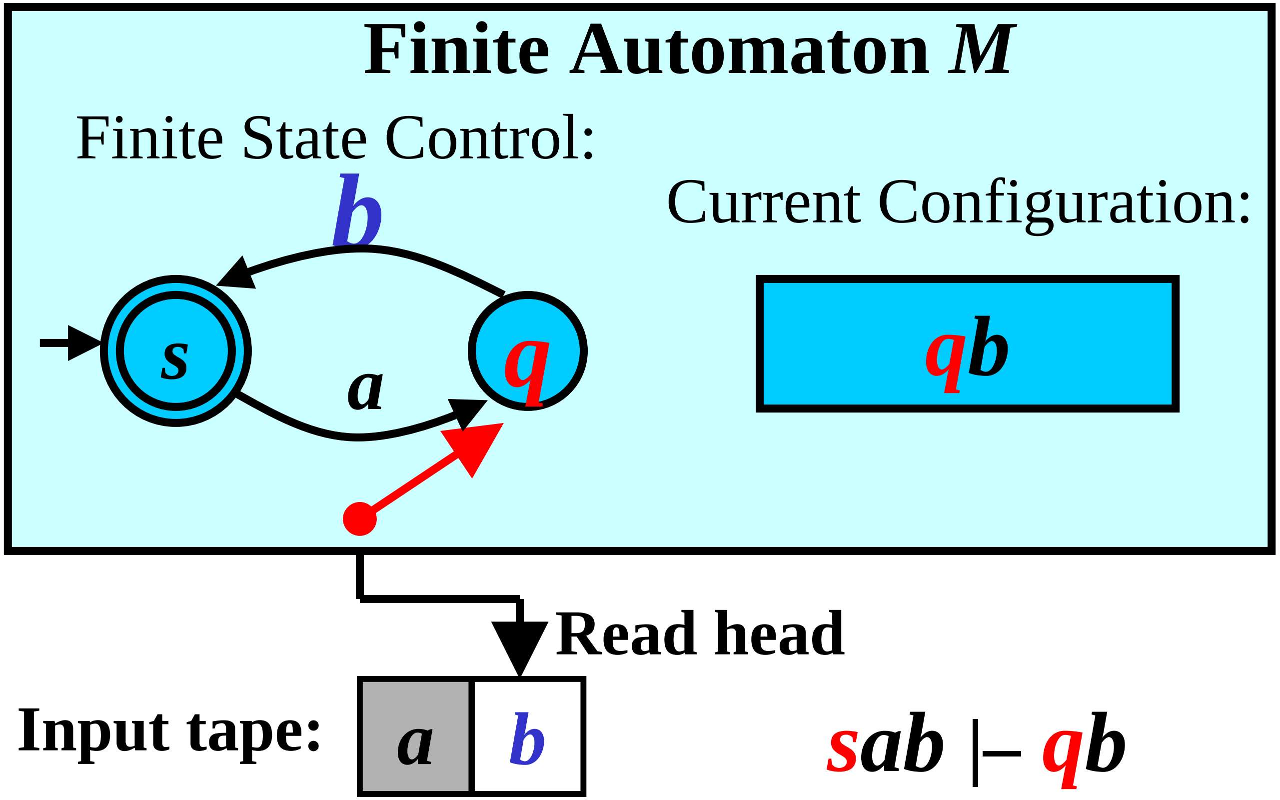


FA: Example 2/3

$M = (Q, \Sigma, R, s, F)$, where:

$Q = \{s, q\}$, $\Sigma = \{a, b\}$, $R = \{sa \rightarrow q, qb \rightarrow s\}$, $F = \{s\}$

Question: $ab \in L(M)$ **?**

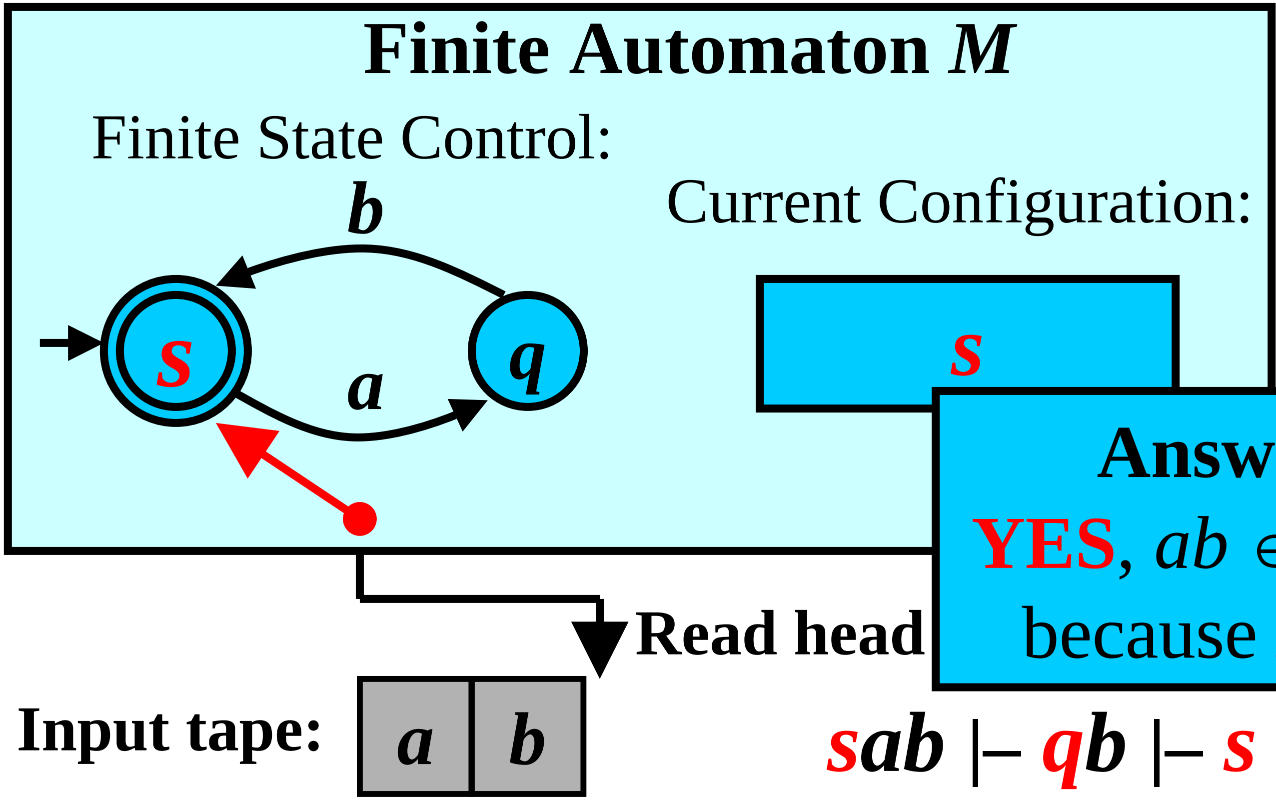


FA: Example 3/3

$M = (Q, \Sigma, R, s, F)$, where:

$Q = \{s, q\}$, $\Sigma = \{a, b\}$, $R = \{sa \rightarrow q, qb \rightarrow s\}$, $F = \{s\}$

Question: $ab \in L(M)$ **?**



Answer:
YES, $ab \in L(M)$, because $s \in F$

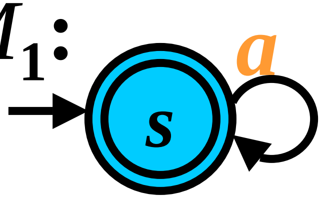
$sab \vdash qb \vdash s$

Equivalent Models

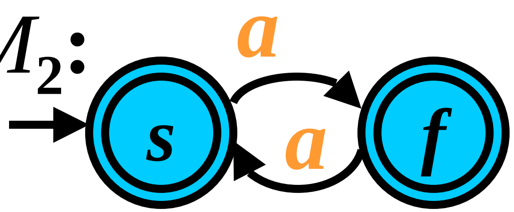
Definition: Two models for languages, such as FAs, are equivalent if they both specify the same language.

Example:

M_1:



M_2:



Question: Is M_1 equivalent to M_2 ?

Answer: M_1 and M_2 **are equivalent** because

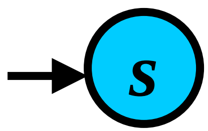
$$L(M_1) = L(M_2) = \{a^n : n \geq 0\}$$

Conversion of RE to FA: Basics 1/5

Gist: Algorithm that converts any RE to an equivalent FA (lex in UNIX).

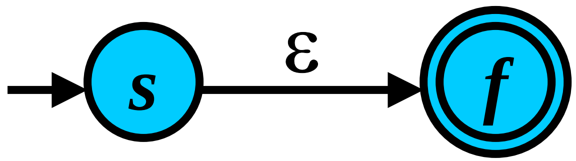
- For a RE $r = \emptyset$, there is an equivalent FA $M_\emptyset$.

Proof: $M_\emptyset$:



- For a RE $r = \varepsilon$, there is an equivalent FA M_ε .

Proof: M_ε :



- For a RE $r = a$, $a \in \Sigma$, there is an equivalent FA M_a .

Proof: M_a :

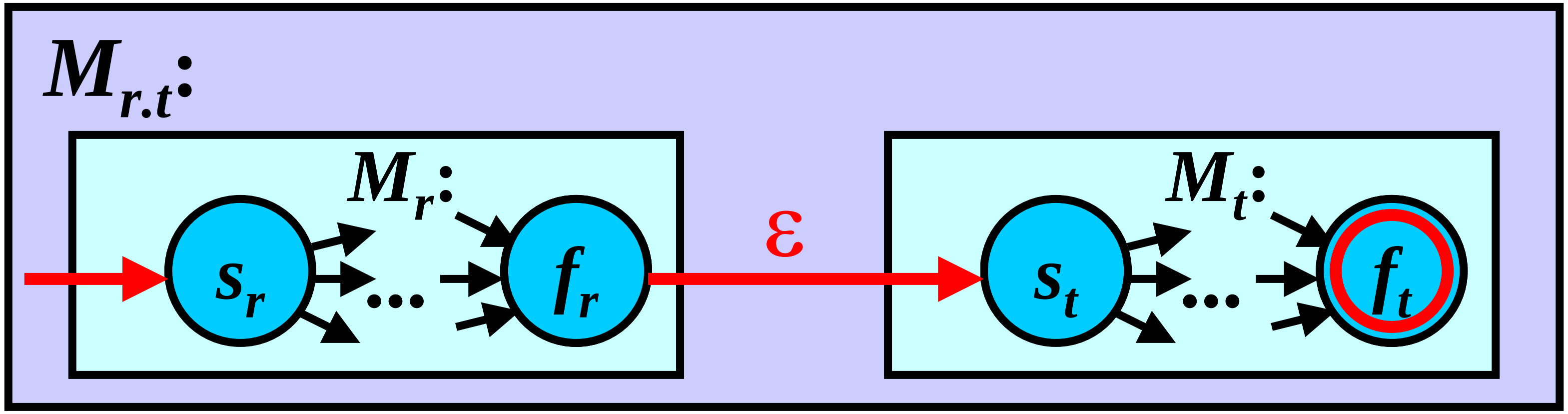
RE to FA: Concatenation 2/5

- Let $\boldsymbol{r}$ be a RE over Σ and $\boldsymbol{M_r} = (Q_r, \Sigma, R_r, s_r, \{f_r\})$ be an FA such that $L(M_r) = L(r)$.
- Let $\boldsymbol{t}$ be a RE over Σ and $\boldsymbol{M_t} = (Q_t, \Sigma, R_t, s_t, \{f_t\})$ be an FA such that $L(M_t) = L(t)$.
- Then, for the RE $\boldsymbol{r.t}$, there exists an equivalent FA $\boldsymbol{M_{r.t}}$

Proof: Let $Q_r \cap Q_t = \emptyset$.

Construction:

$$M_{r.t} = (Q_r \cup Q_t, \Sigma, R_r \cup R_t \cup \{f_r \rightarrow s_t\}, s_r, \{f_t\})$$



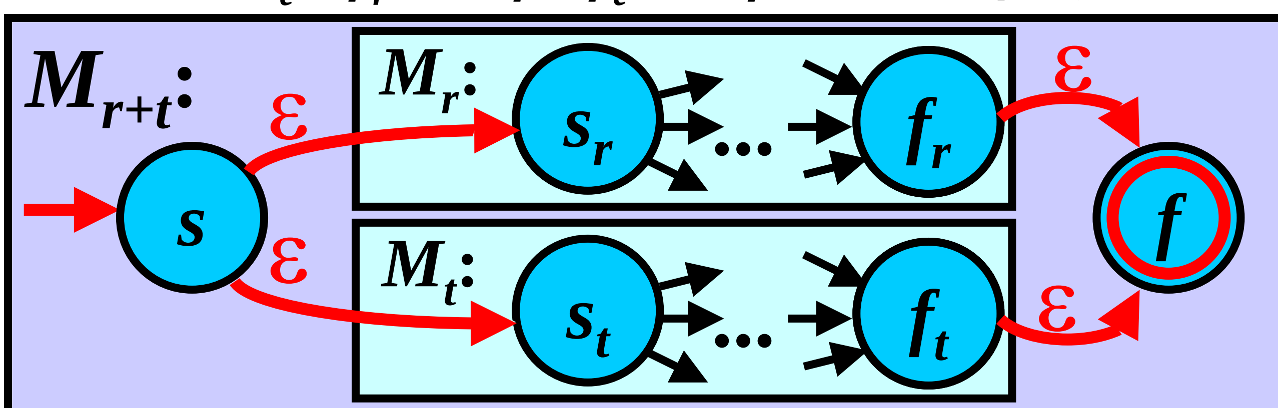
RE to FA: Union 3/5

- Let $\boldsymbol{r}$ be a RE over Σ and $\boldsymbol{M_r} = (Q_r, \Sigma, R_r, s_r, \{f_r\})$ be an FA such that $L(M_r) = L(r)$.
- Let $\boldsymbol{t}$ be RE over Σ and $\boldsymbol{M_t} = (Q_t, \Sigma, R_t, s_t, \{f_t\})$ be an FA such that $L(M_t) = L(t)$.
- For a RE $\boldsymbol{r + t}$, there exists an equivalent FA $\boldsymbol{M_{r+t}}$

Proof: Let $Q_r \cap Q_t = \emptyset$, $s, f \notin Q_r \cup Q_t$.

Construction

$M_{r+t} = (Q_r \cup Q_t \cup \{s, f\}, \Sigma, R_r \cup R_t \cup \{s \rightarrow s_r, s \rightarrow s_t, f_r \rightarrow f, f_t \rightarrow f\}, s, \{f\})$



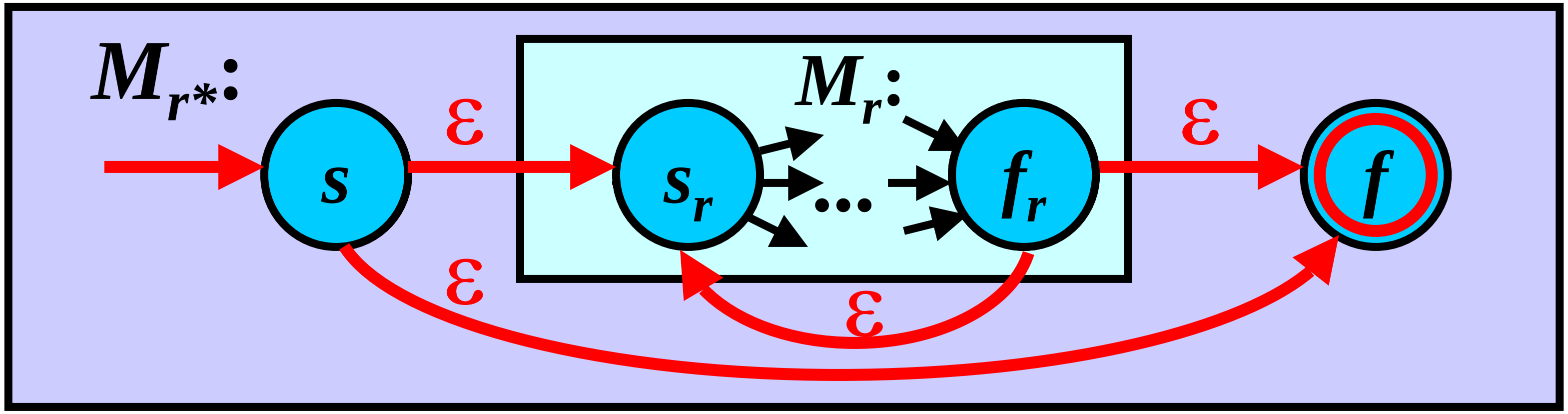
RE to FA: Iteration 4/5

- Let r be a RE over Σ and $M_r = (Q_r, \Sigma, R_r, s_r, \{f_r\})$ be an FA such that $L(M_r) = L(r)$.
- For the RE r^*, there exists an equivalent FA M_{r*}

Proof: Let $s, f \notin Q_r$.

Construction:

$$M_{r*} = (Q_r \cup \{s, f\}, \Sigma, R_r \cup \{s \rightarrow s_r, f_r \rightarrow f, f_r \rightarrow s_r, s \rightarrow f\}, s, \{f\})$$



RE to FA: Completion 5/5

- **Input:** RE r over Σ
- **Output:** FA M such that $L(r) = L(M)$

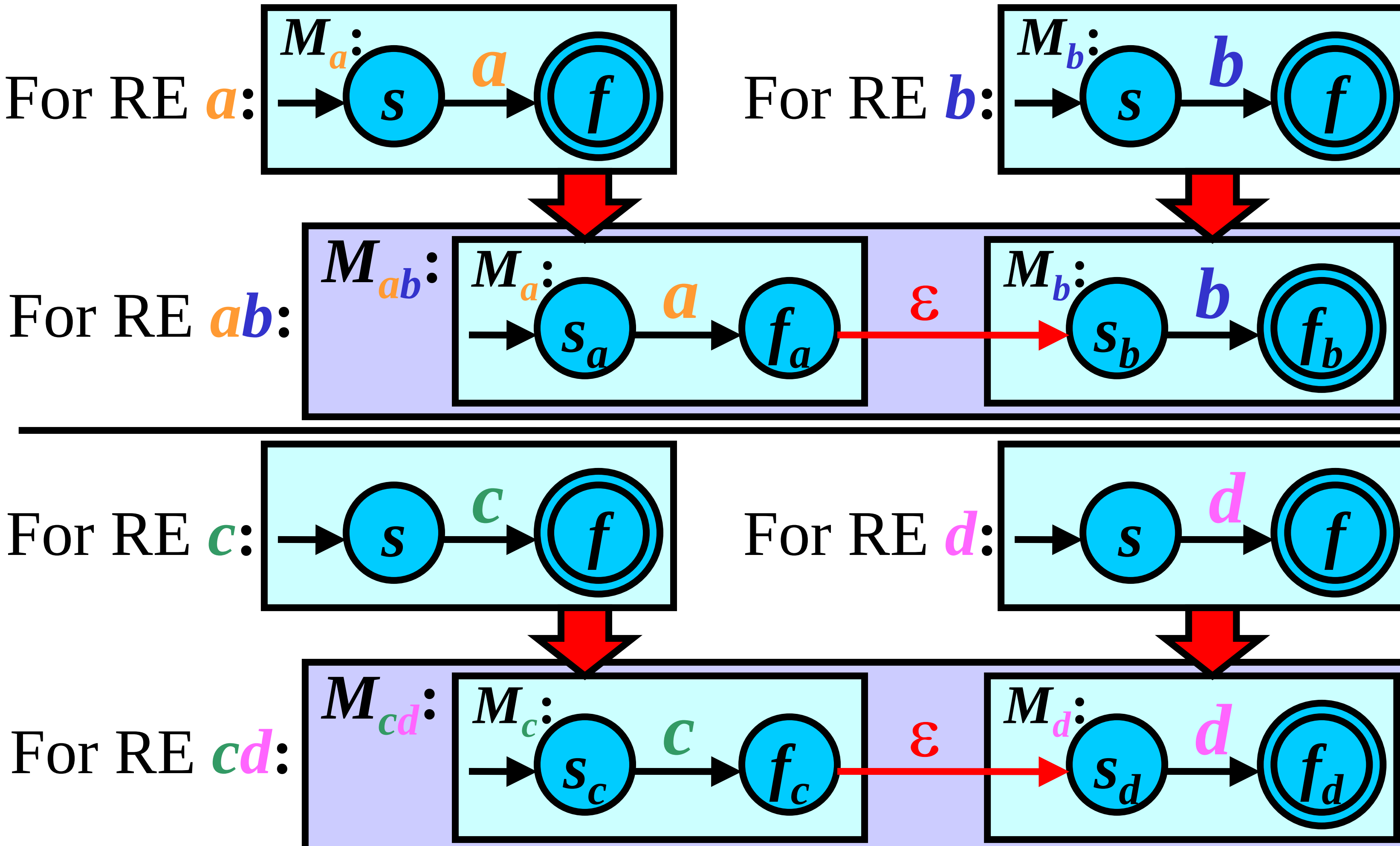
- **Method:**
- **From "inside" of r, repeatedly use the next rules to construct M:**
  - for RE $\varnothing$, construct FA $M_{\varnothing}$
  - for RE ε, construct FA M_{ε}
  - for RE $a \in \Sigma$, construct FA M_a

  → (see 1/5)

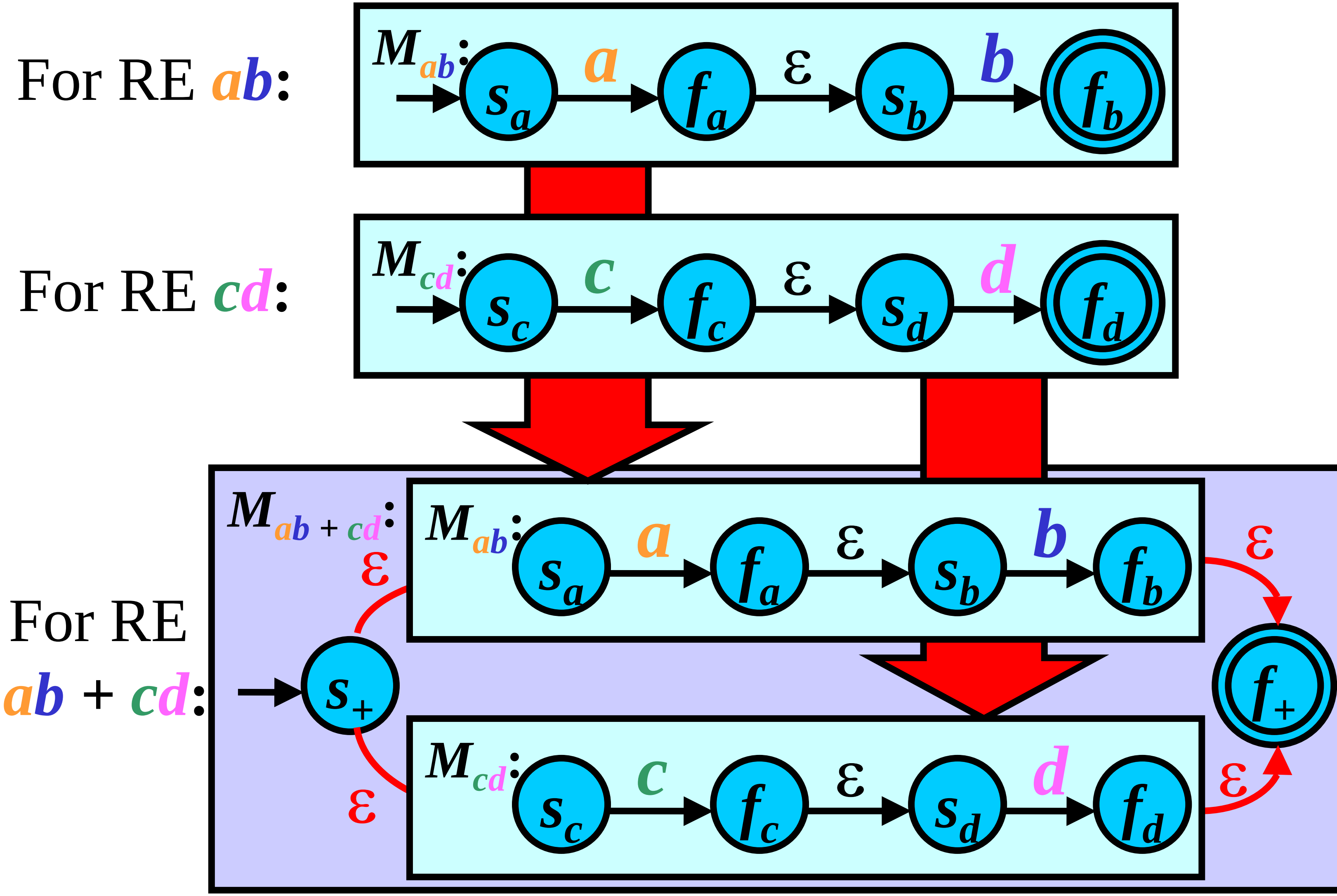
  - **let** for REs r and t, there already exist FAs M_r and M_t, respectively; **then**,
    - for RE $r.t$, construct FA $M_{r.t}$ (see 2/5)
    - for RE $r + t$, construct FA M_{r+t} (see 3/5)
    - for RE r^* construct FA M_{r^*} (see 4/5)

RE to FA: Example 1/3

Transform RE $r = ((ab) + (cd))^*$ to an equivalent FA M

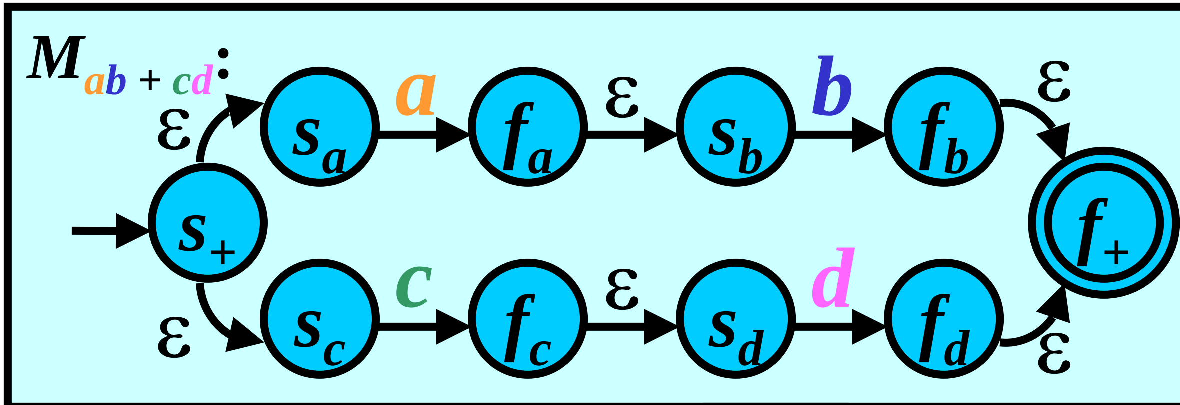


RE to FA: Example 2/3

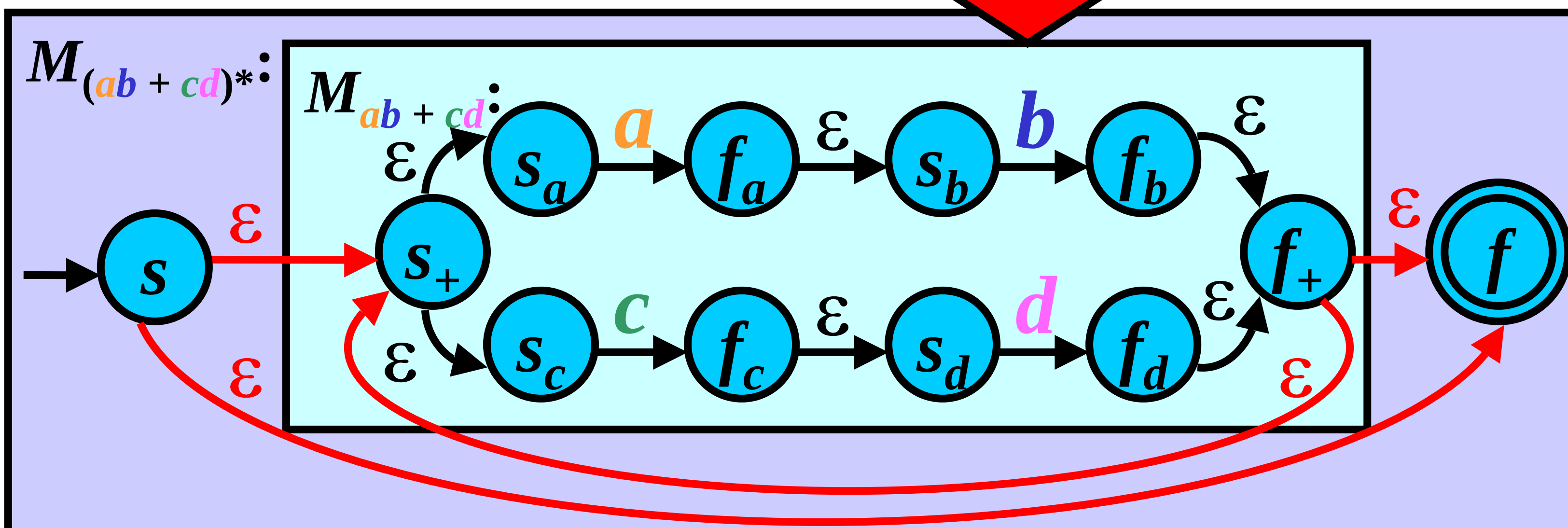


RE to FA: Example 3/3

For RE
ab + *cd*:



For a final RE (*ab* + *cd*)*:



Models for Regular Languages

Theorem: For every RE r, there is an FA M such that $L(r) = L(M)$.

Proof is based on the previous algorithm.

Theorem: For every FA M, there is an RE r such that $L(M) = L(r)$.

Proof: See page 210 in [Meduna: Automata and Languages]

Conclusion: The fundamental models for regular languages are

1) Regular expressions **2) Finite Automata**