Jumping Finite Automata

Alexander Meduna

Brno University of Technology, Faculty of Information Technology
Božetěchova 2, 612 00 Brno, Czech Republic
http://www.fit.vutbr.cz/∼meduna

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Prepared in cooperation with Petr Zemek based on

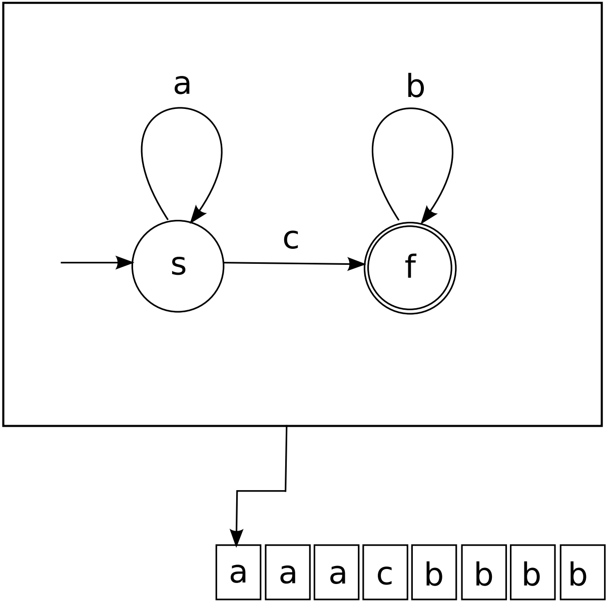
Alexander Meduna and Petr Zemek
Jumping Finite Automata
International Journal of Foundations of Computer Science
Vol. 49, No. 2, p. 1555–1578, 2012

Supported by the IT4I Centre of Excellence CZ.1.05/1.1.00/02.0070.

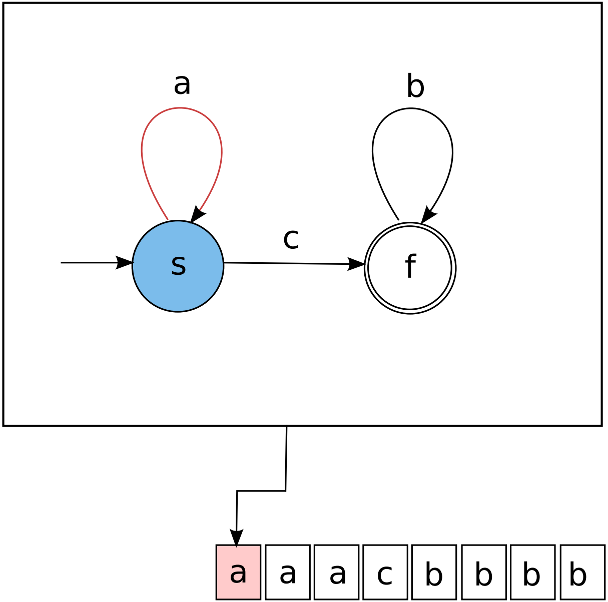
Contents

- **Introduction**
- **Definitions and Examples**
- **Results**
- **Concluding Remarks and Discussion**

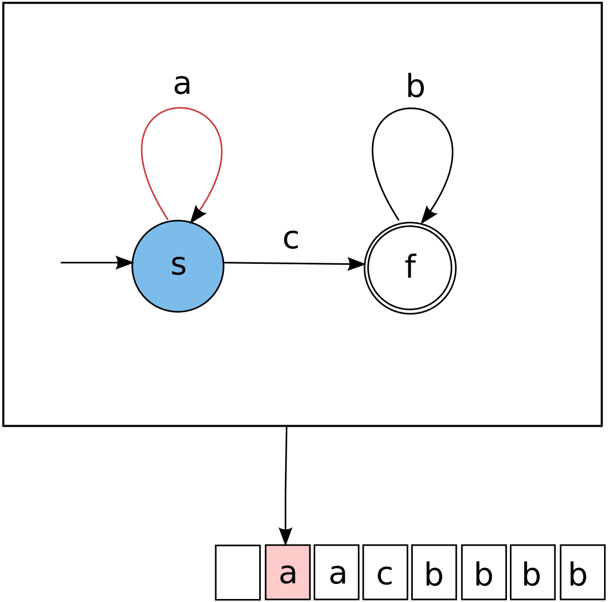
Finite Automata



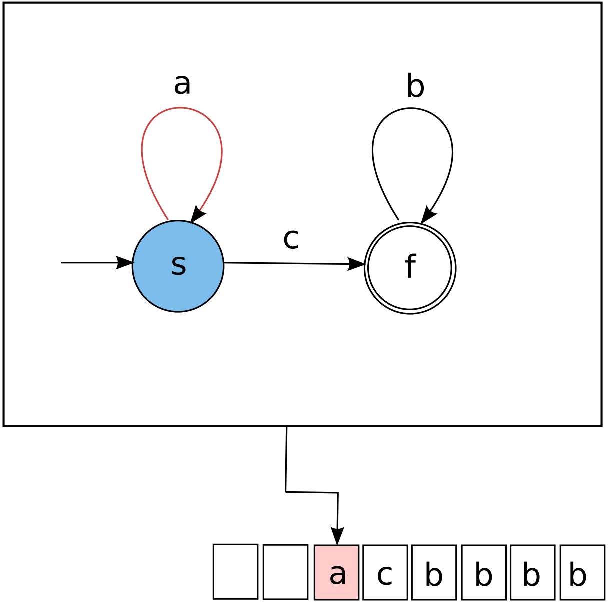
Finite Automata



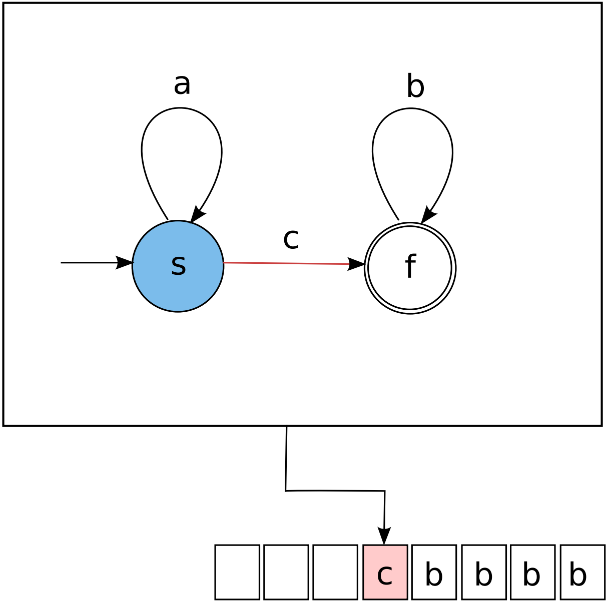
Finite Automata



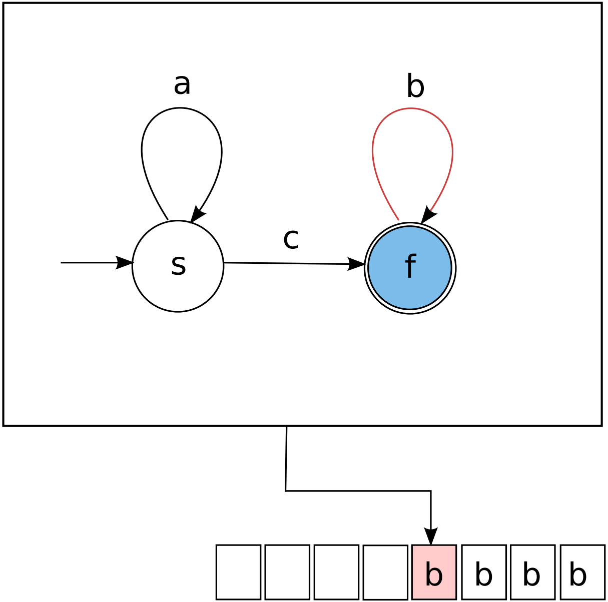
Finite Automata



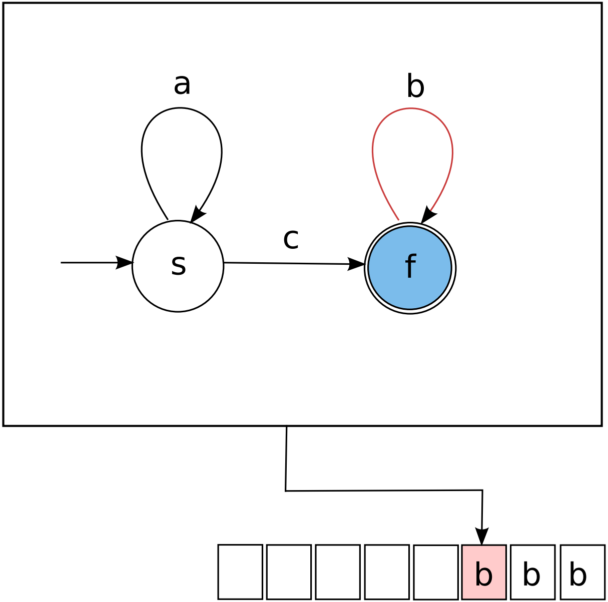
Finite Automata



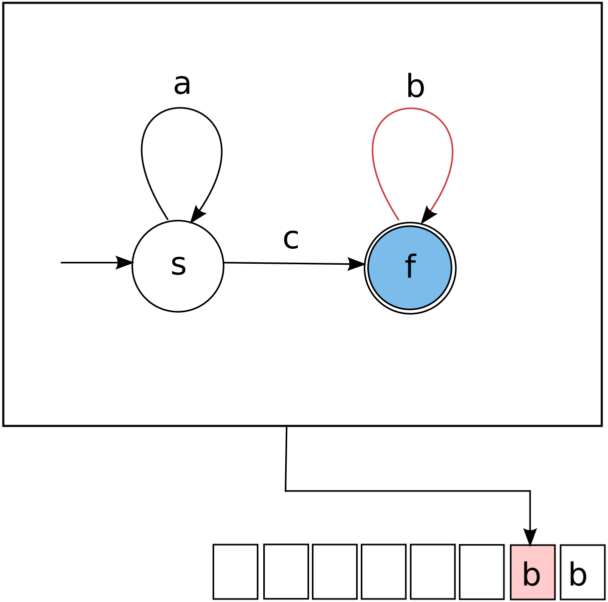
Finite Automata



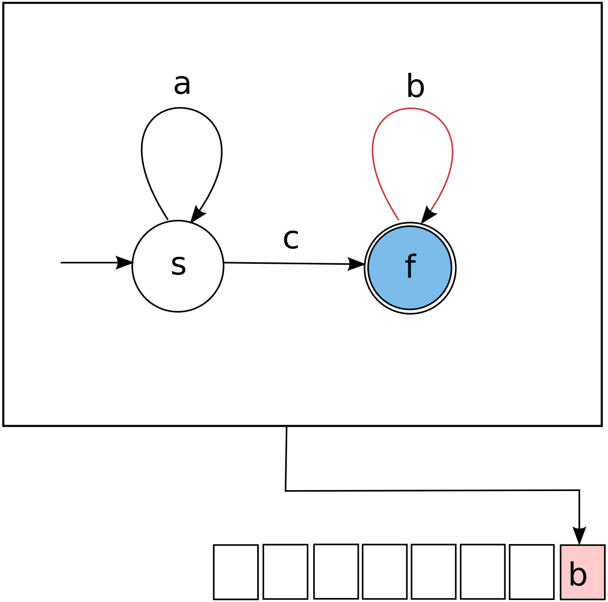
Finite Automata



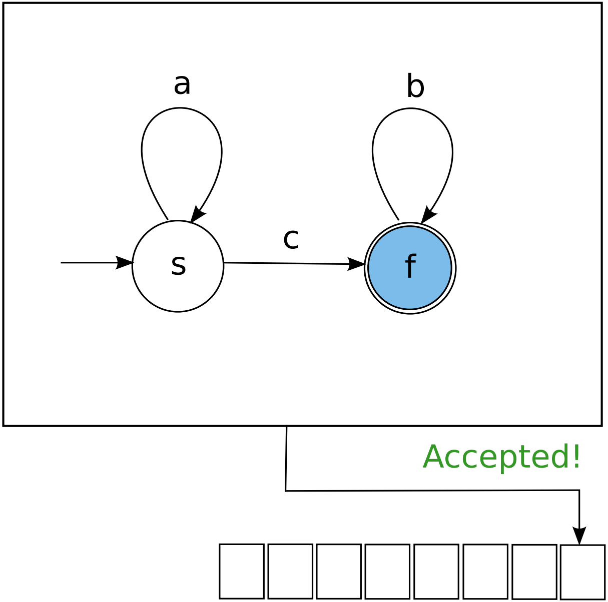
Finite Automata

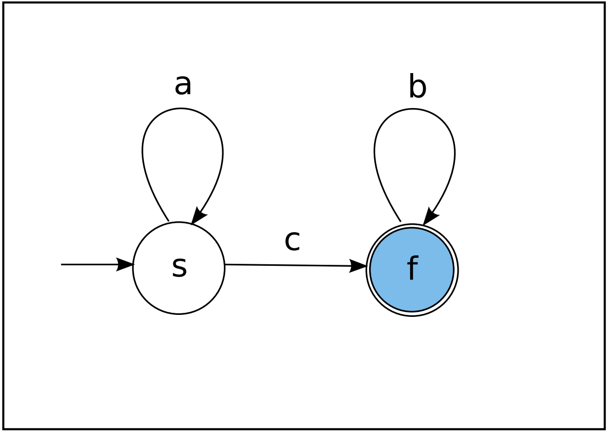


Finite Automata

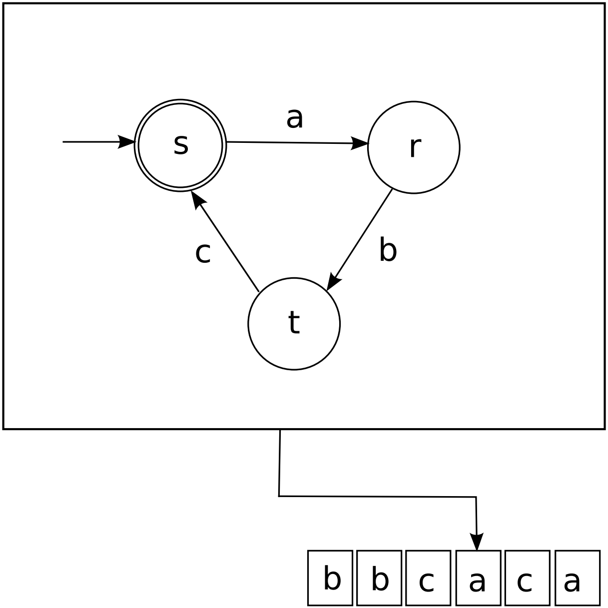


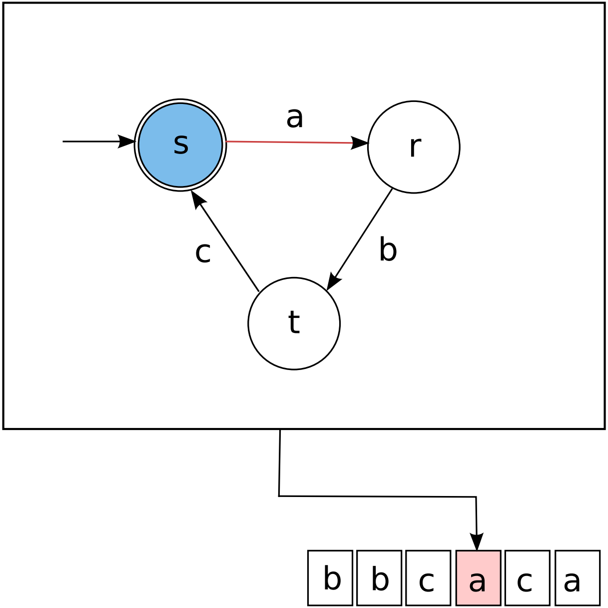
Finite Automata



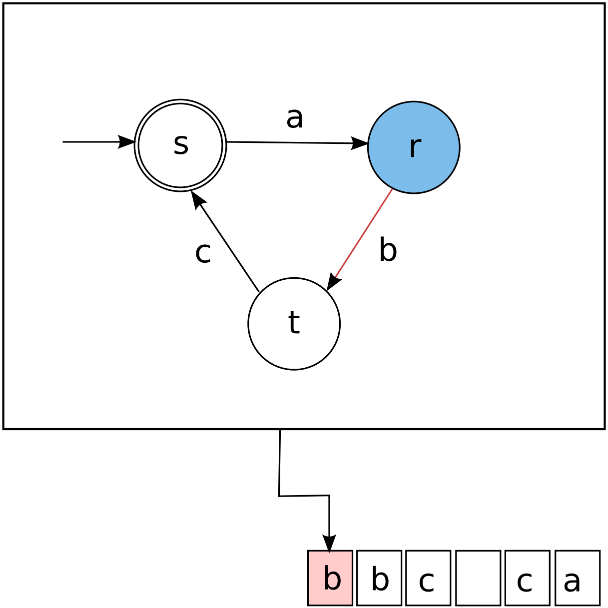


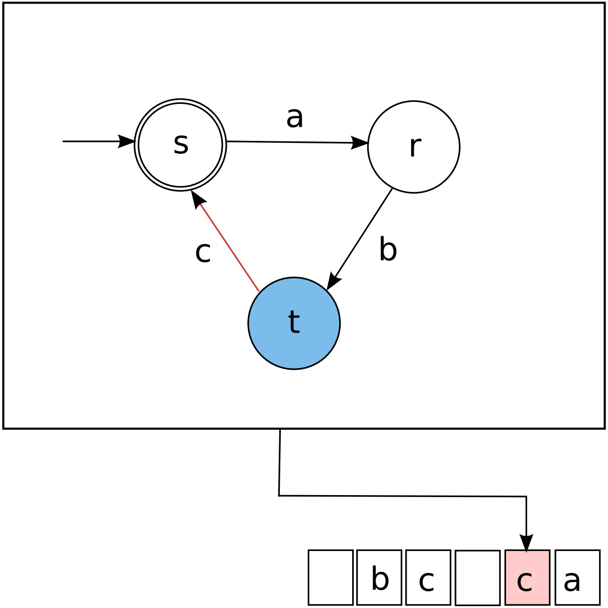
Accepted language: $\{a\}^*\{c\}\{b\}^*$


a
s
r
b
c
t
b
b
c
a
c
a

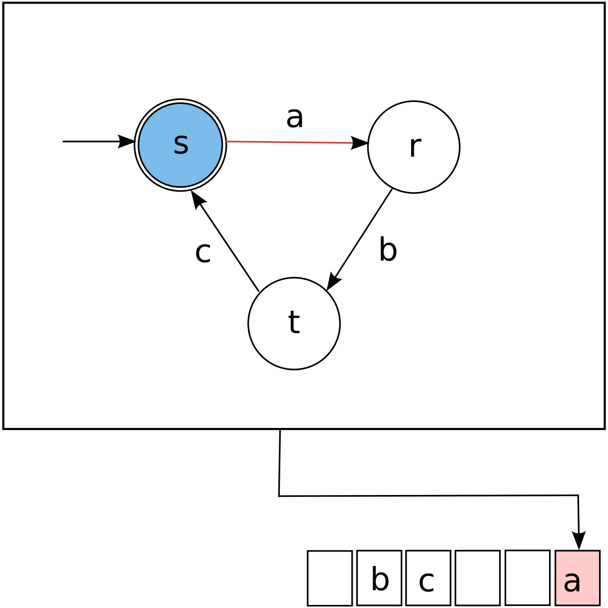


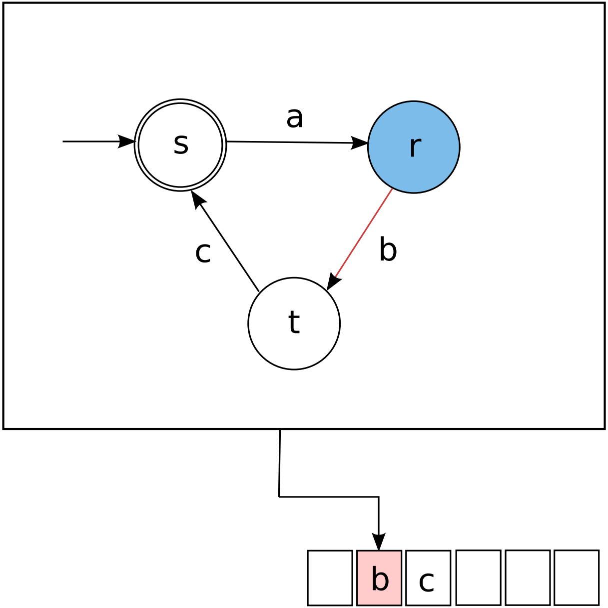
a
s
r
b
c
t
b
b
c
a
c
a

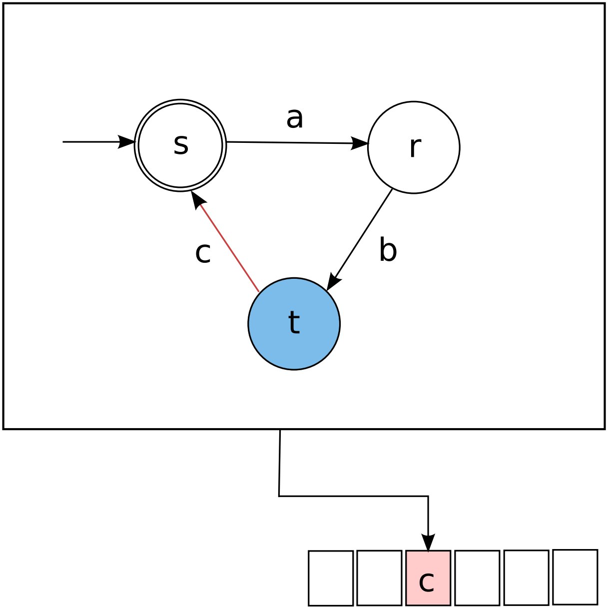

a
s
r
c
b
t
b
b
c
c
a

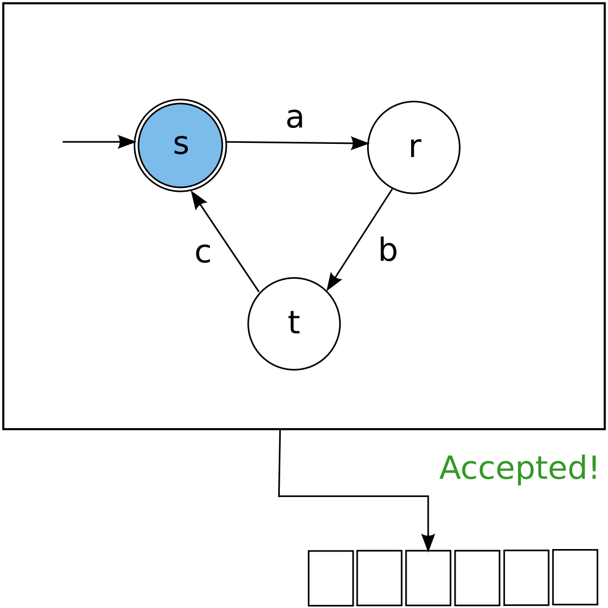

a
s
r
c
b
t
b
c
c
a

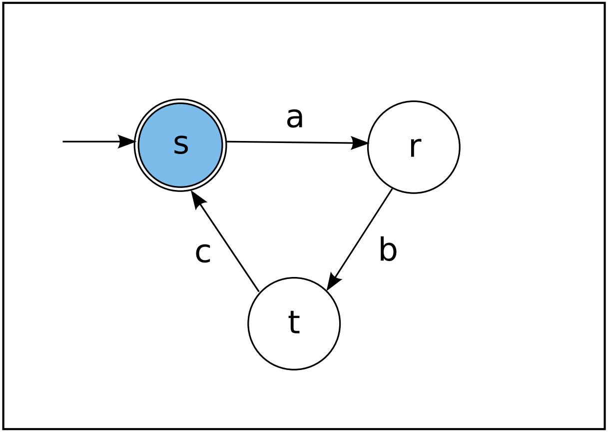
Jumping Finite Automata




a
s
r
c
b
t
b
c


a
s
r
c
b
t
c


a
s
r
c
b
t
Accepted!



Accepted language: $\{w \in \{a, b, c\}^* : |w|_a = |w|_b = |w|_c\}$

Definition

A *general jumping finite automaton* (*GJFA*) is a quintuple

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

where

- Q is a finite set of *states*;
- Σ is the *input alphabet*;
- $R \subseteq Q \times \Sigma^* \times Q$ is a finite ternary relation, called the *set of rules*; in what follows, every rule $(p, y, q) \in R$ is written as $py \rightarrow q$
- s is the *start state*;
- F is a set of *final states*.

Definition

A *general jumping finite automaton* (*GJFA*) is a quintuple

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

where

- Q is a finite set of *states*;
- Σ is the *input alphabet*;
- $R \subseteq Q \times \Sigma^* \times Q$ is a finite ternary relation, called the *set of rules*; in what follows, every rule $(p, y, q) \in R$ is written as $py \to q$
- s is the *start state*;
- F is a set of *final states*.

Definition

If all rules $py \to q \in R$ satisfy $|y| \leq 1$, then M is a *jumping finite automaton* (*JFA*).

Definitions – Continued

Definition

If $x, z, x', z', y \in \Sigma^*$ such that $xz = x'z'$ and $py \rightarrow q \in R$, then M makes a *jump* from $xpyz$ to $x'qz'$, symbolically written as

$$x\underline{p}yz \curvearrowright x'\underline{q}z'$$

Definition

If $x, z, x', z', y \in \Sigma^*$ such that $xz = x'z'$ and $py \to q \in R$, then M makes a *jump* from $xpyz$ to $x'qz'$, symbolically written as

$$x\underline{p}yz \curvearrowright x'\underline{q}z'$$

- $\curvearrowright^n$ intuitively, a sequence of n jumps ($n \geq 0$); mathematically, the nth power of $\curvearrowright$
- $\curvearrowright^*$ intuitively, a sequence of jumps (possibly empty); mathematically, the reflexive-transitive closure of $\curvearrowright$

Definitions – Continued

Definition

If $x, z, x', z', y \in \Sigma^*$ such that $xz = x'z'$ and $py \rightarrow q \in R$, then M makes a *jump* from $xpyz$ to $x'qz'$, symbolically written as

$$x\underline{p}yz \curvearrowright x'\underline{q}z'$$

- $\curvearrowright^n$ intuitively, a sequence of n jumps ($n \geq 0$); mathematically, the nth power of $\curvearrowright$
- $\curvearrowright^*$ intuitively, a sequence of jumps (possibly empty); mathematically, the reflexive-transitive closure of $\curvearrowright$

Definition

The *language accepted by* M, denoted by $L(M)$, is defined as

$$L(M) = \{uv : u, v \in \Sigma^*, u\underline{s}v \curvearrowright^* \underline{f}, f \in F\}$$

Example

The JFA

$$M = (\{s, r, t\}, \{a, b, c\}, R, s, \{s\})$$

with

$$R = \{sa \to r, rb \to t, tc \to s\}$$

accepts

$$L(M) = \{w \in \{a, b, c\}^* : |w|_a = |w|_b = |w|_c\}$$

Example

The JFA

$$M = (\{s, r, t\}, \{a, b, c\}, R, s, \{s\})$$

with

$$R = \{sa \to r, rb \to t, tc \to s\}$$

accepts

$$L(M) = \{w \in \{a, b, c\}^* : |w|_a = |w|_b = |w|_c\}$$

For instance:

$$bacbc\underline{s}a$$

Example

The JFA

$$M = (\{s, r, t\}, \{a, b, c\}, R, s, \{s\})$$

with

$$R = \{sa \to r, rb \to t, tc \to s\}$$

accepts

$$L(M) = \{w \in \{a, b, c\}^* : |w|_a = |w|_b = |w|_c\}$$

For instance:

$$bacbc\underline{s}a \curvearrowright bac\underline{r}bc \quad [sa \to r]$$

Example

The JFA

$$M = (\{s, r, t\}, \{a, b, c\}, R, s, \{s\})$$

with

$$R = \{sa \to r, rb \to t, tc \to s\}$$

accepts

$$L(M) = \{w \in \{a, b, c\}^* : |w|_a = |w|_b = |w|_c\}$$

For instance:

$$\begin{array}{llll} bacbc\underline{s}a & \curvearrowright & bac\underline{r}bc & [sa \to r] \\ & \curvearrowright & bac\underline{t}c & [rb \to t] \end{array}$$

Example

The JFA

$$M = (\{s, r, t\}, \{a, b, c\}, R, s, \{s\})$$

with

$$R = \{sa \rightarrow r, rb \rightarrow t, tc \rightarrow s\}$$

accepts

$$L(M) = \{w \in \{a, b, c\}^* : |w|_a = |w|_b = |w|_c\}$$

For instance:

$$\begin{array}{llll} bacbc\underline{s}a & \curvearrowright & bac\underline{r}bc & [sa \rightarrow r] \\ & \curvearrowright & bac\underline{t}c & [rb \rightarrow t] \\ & \curvearrowright & b\underline{s}ac & [tc \rightarrow s] \end{array}$$

Example

The JFA

$$M = (\{s, r, t\}, \{a, b, c\}, R, s, \{s\})$$

with

$$R = \{sa \to r, rb \to t, tc \to s\}$$

accepts

$$L(M) = \{w \in \{a, b, c\}^* : |w|_a = |w|_b = |w|_c\}$$

For instance:

$$\begin{array}{llll} bacbc\underline{s}a & \curvearrowright & bac\underline{r}bc & [sa \to r] \\ & \curvearrowright & bac\underline{t}c & [rb \to t] \\ & \curvearrowright & b\underline{s}ac & [tc \to s] \\ & \curvearrowright & \underline{r}bc & [sa \to r] \end{array}$$

Example

The JFA

$$M = (\{s, r, t\}, \{a, b, c\}, R, s, \{s\})$$

with

$$R = \{sa \to r, rb \to t, tc \to s\}$$

accepts

$$L(M) = \{w \in \{a, b, c\}^* : |w|_a = |w|_b = |w|_c\}$$

For instance:

$$\begin{array}{lcll} bacbc\underline{s}a & \curvearrowright & bac\underline{r}bc & [sa \to r] \\ & \curvearrowright & bac\underline{t}c & [rb \to t] \\ & \curvearrowright & b\underline{s}ac & [tc \to s] \\ & \curvearrowright & \underline{r}bc & [sa \to r] \\ & \curvearrowright & \underline{t}c & [rb \to t] \end{array}$$

Example

The JFA

$$M = (\{s, r, t\}, \{a, b, c\}, R, s, \{s\})$$

with

$$R = \{sa \to r, rb \to t, tc \to s\}$$

accepts

$$L(M) = \{w \in \{a, b, c\}^* : |w|_a = |w|_b = |w|_c\}$$

For instance:

$$\begin{array}{lcll} bacbc\underline{s}a & \curvearrowright & bac\underline{r}bc & [sa \to r] \\ & \curvearrowright & bac\underline{t}c & [rb \to t] \\ & \curvearrowright & b\underline{s}ac & [tc \to s] \\ & \curvearrowright & \underline{r}bc & [sa \to r] \\ & \curvearrowright & \underline{t}c & [rb \to t] \\ & \curvearrowright & \underline{s} & [tc \to s] \end{array}$$

Example

The GJFA

$$H = (\{s, f\}, \{a, b\}, R, s, \{f\}),$$

with

$$R = \{sba \rightarrow f, fa \rightarrow f, fb \rightarrow f\}$$

accepts

$$L(H) = \{a, b\}^* \{ba\} \{a, b\}^*$$

Example

The GJFA

$$H = (\{s, f\}, \{a, b\}, R, s, \{f\}),$$

with

$$R = \{sba \rightarrow f, fa \rightarrow f, fb \rightarrow f\}$$

accepts

$$L(H) = \{a, b\}^* \{ba\} \{a, b\}^*$$

For instance:

$$bb\underline{s}baa$$

Example #2

Example

The GJFA

$$H = (\{s, f\}, \{a, b\}, R, s, \{f\}),$$

with

$$R = \{sba \to f, fa \to f, fb \to f\}$$

accepts

$$L(H) = \{a, b\}^* \{ba\} \{a, b\}^*$$

For instance:

$$bb\underline{s}baa \curvearrowright bb\underline{f}a \quad [sba \to f]$$

Example

The GJFA

$$H = (\{s, f\}, \{a, b\}, R, s, \{f\}),$$

with

$$R = \{sba \to f, fa \to f, fb \to f\}$$

accepts

$$L(H) = \{a, b\}^* \{ba\} \{a, b\}^*$$

For instance:

$$\begin{array}{lcll} bb\underline{s}baa & \curvearrowright & bb\underline{f}a & [sba \to f] \\ & \curvearrowright & \underline{f}bb & [fa \to f] \end{array}$$

Example #2

Example

The GJFA

$$H = (\{s, f\}, \{a, b\}, R, s, \{f\}),$$

with

$$R = \{sba \to f, fa \to f, fb \to f\}$$

accepts

$$L(H) = \{a, b\}^* \{ba\} \{a, b\}^*$$

For instance:

$$\begin{array}{llll} bb\underline{s}baa & \curvearrowright & bb\underline{f}a & [sba \to f] \\ & \curvearrowright & \underline{f}bb & [fa \to f] \\ & \curvearrowright & \underline{f}b & [fb \to f] \end{array}$$

Example

The GJFA

$$H = (\{s, f\}, \{a, b\}, R, s, \{f\}),$$

with

$$R = \{sba \rightarrow f, fa \rightarrow f, fb \rightarrow f\}$$

accepts

$$L(H) = \{a, b\}^* \{ba\} \{a, b\}^*$$

For instance:

$$\begin{array}{llll} bb\underline{s}baa & \curvearrowright & bb\underline{f}a & [sba \rightarrow f] \\ & \curvearrowright & \underline{f}bb & [fa \rightarrow f] \\ & \curvearrowright & \underline{f}b & [fb \rightarrow f] \\ & \curvearrowright & \underline{f} & [fb \rightarrow f] \end{array}$$

For any string w, $perm(w)$ denotes the set of all its permutations.

For an arbitrary language L, set

$$perm(L) = \{perm(w) : w \in L\}$$

For any string w, $perm(w)$ denotes the set of all its permutations.

For an arbitrary language L, set

$$perm(L) = \{perm(w) : w \in L\}$$

Theorem

Let L be an arbitrary language. L is accepted by a JFA if and only if $L = perm(K)$, where K is a regular language.

Power of GJFAs and JFAs

For any string w, $perm(w)$ denotes the set of all its permutations.

For an arbitrary language L, set

$$perm(L) = \{perm(w) : w \in L\}$$

Theorem

Let L be an arbitrary language. L is accepted by a JFA if and only if $L = perm(K)$, where K is a regular language.

Proof Idea

I. Let M be a JFA. Consider M as an FA M'. Set $K = L(M')$. K is regular, and $L(M) = perm(K)$.

II. Take $perm(K)$, where K is any regular language. Let $K = L(M)$, where M is an FA. Consider M as a JFA M'. $L(M') = perm(K)$. □

Corollary

There is no JFA that accepts $\{a, b\}^*\{ba\}\{a, b\}^*$.

Power of GJFAs and JFAs (Continued)

Corollary

There is no JFA that accepts $\{a, b\}^*\{ba\}\{a, b\}^*$.

Theorem

GJFAs are strictly stronger than JFAs.

Power of GJFAs and JFAs (Continued)

Corollary

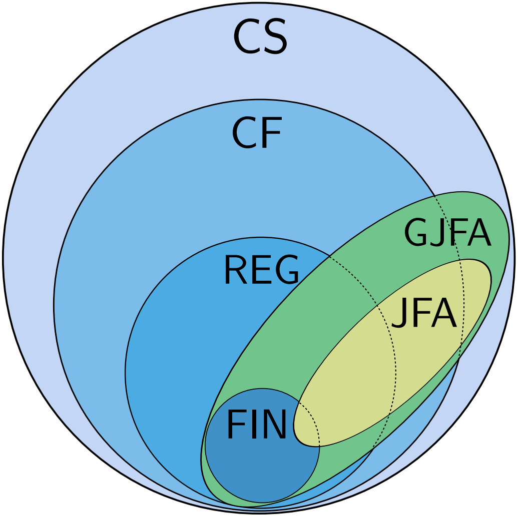
There is no JFA that accepts $\{a, b\}^*\{ba\}\{a, b\}^*$.

Theorem

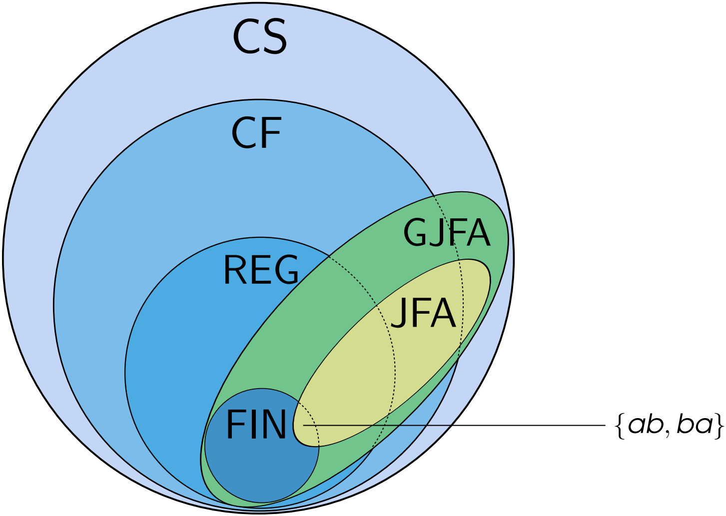
GJFAs are strictly stronger than JFAs.

Proof Idea

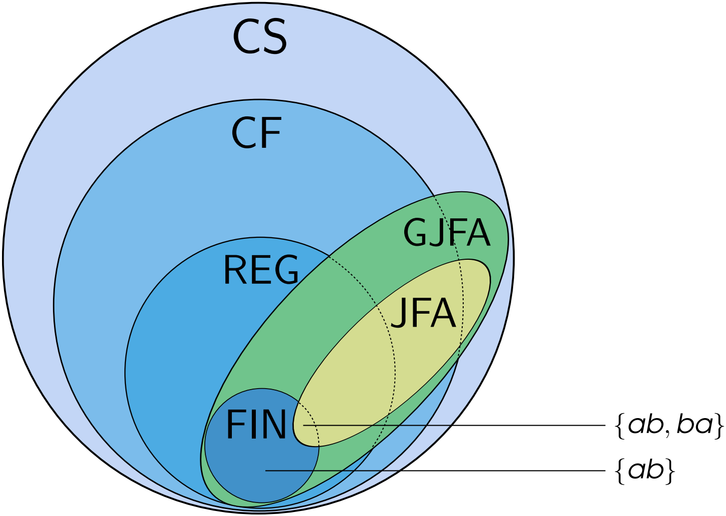
The language $\{a, b\}^*\{ba\}\{a, b\}^*$ is accepted by the GJFA from Example #2. □


CS
CF
GJFA
REG
JFA
FIN

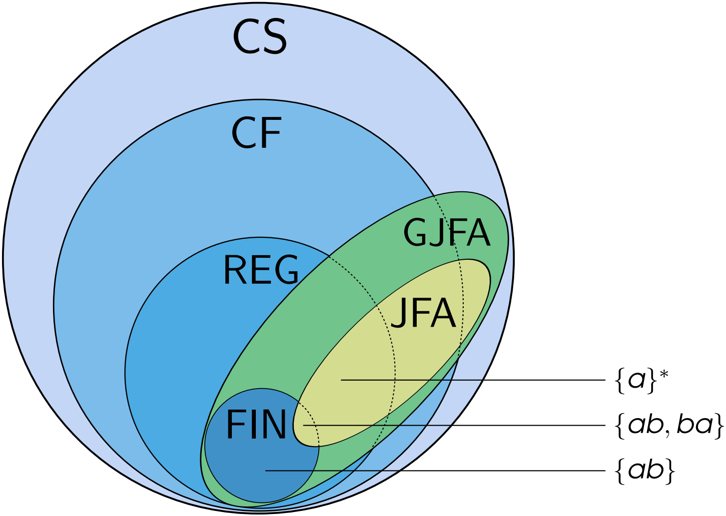
Relations Between Language Families



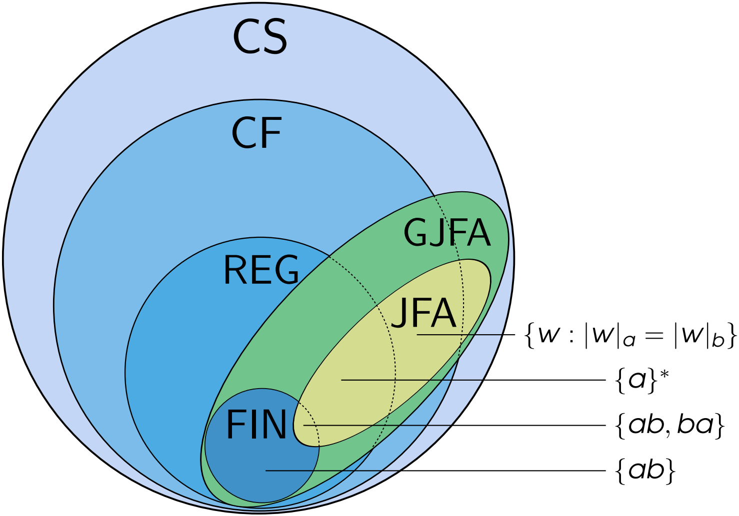
Relations Between Language Families



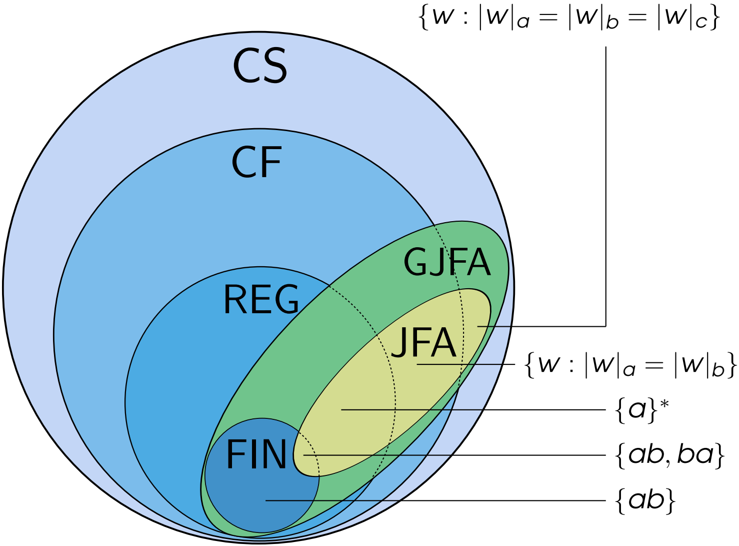
Relations Between Language Families

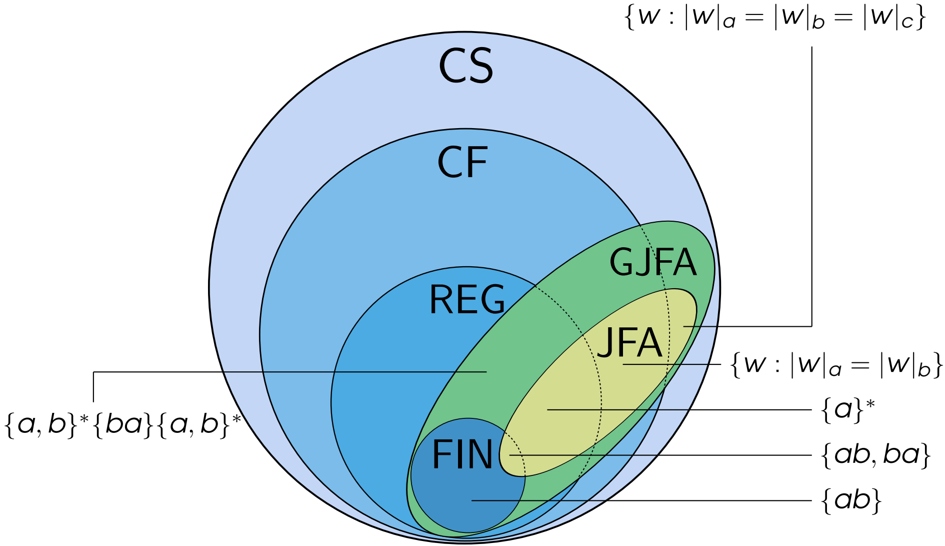


Relations Between Language Families

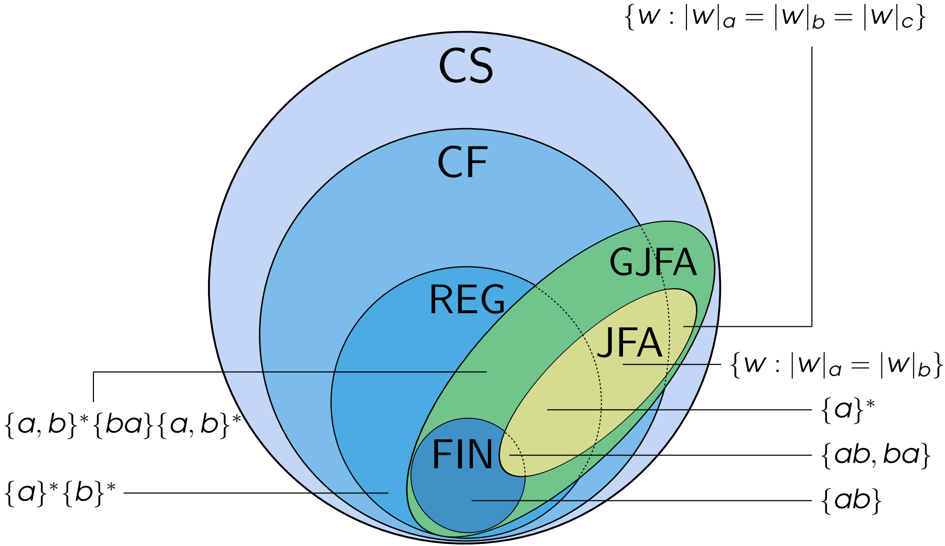


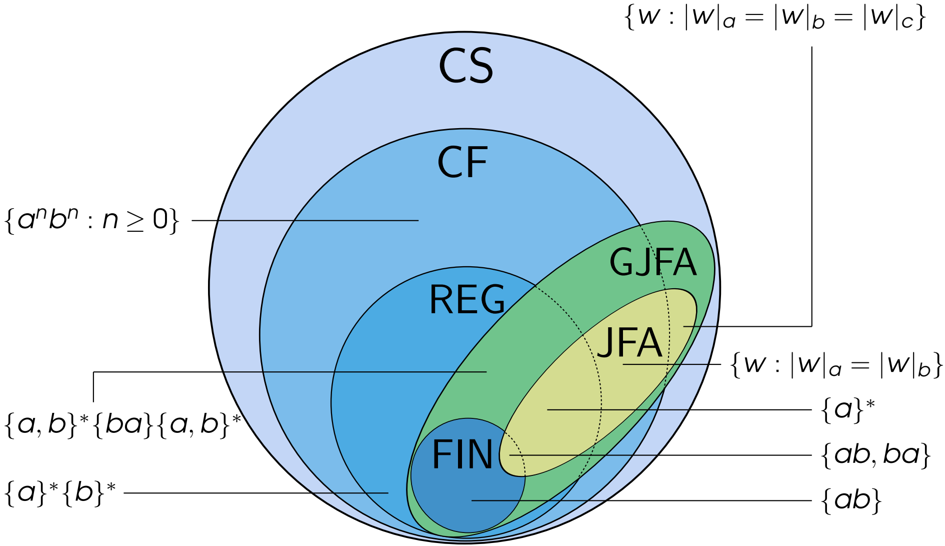
Relations Between Language Families

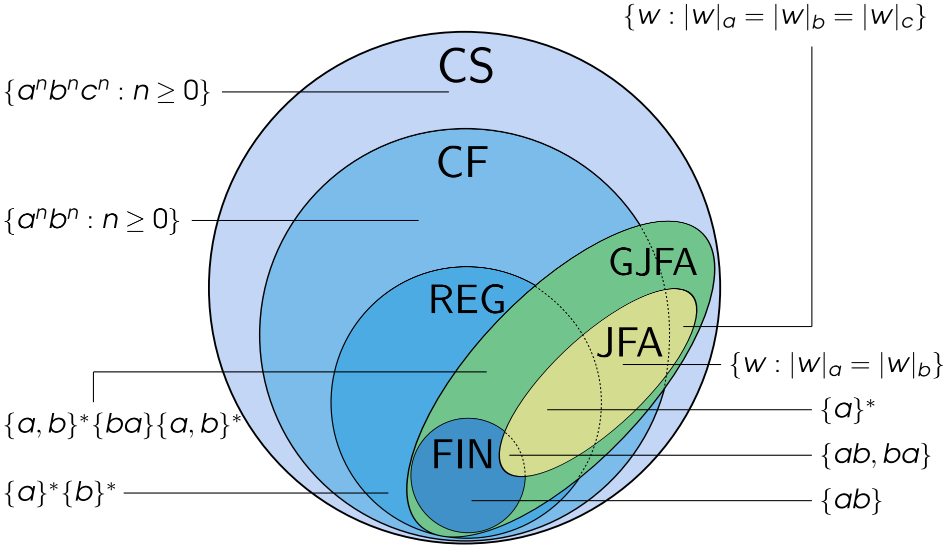



$\{w : |w|_a = |w|_b = |w|_c\}$
CS
CF
GJFA
REG
JFA
$\{w : |w|_a = |w|_b\}$
$\{a, b\}^*\{ba\}\{a, b\}^*$
$\{a\}^*$
FIN
$\{ab, ba\}$
$\{ab\}$

Relations Between Language Families




$\{w : |w|_a = |w|_b = |w|_c\}$
CS
CF
$\{a^n b^n : n \geq 0\}$
GJFA
REG
JFA
$\{w : |w|_a = |w|_b\}$
$\{a, b\}^*\{ba\}\{a, b\}^*$
$\{a\}^*$
FIN
$\{ab, ba\}$
$\{a\}^*\{b\}^*$
$\{ab\}$


$\{w : |w|_a = |w|_b = |w|_c\}$
CS
$\{a^n b^n c^n : n \geq 0\}$
CF
$\{a^n b^n : n \geq 0\}$
GJFA
REG
JFA
$\{w : |w|_a = |w|_b\}$
$\{a\}^*$
$\{a, b\}^*\{ba\}\{a, b\}^*$
FIN
$\{ab, ba\}$
$\{a\}^*\{b\}^*$
$\{ab\}$

Further Topics of Investigation

By analogy with finite automata:

- removal of ε-*moves* $(p \rightarrow q$ and $qa \rightarrow r \quad \Rightarrow \quad pa \rightarrow r)$
- making JFAs deterministic

By analogy with finite automata:

- removal of ε-*moves* $(p \to q$ and $qa \to r \;\Rightarrow\; pa \to r)$
- making JFAs deterministic

Theorem

Every unary language accepted by a JFA is regular.

By analogy with finite automata:

- removal of ε-*moves* $\quad (p \rightarrow q$ and $qa \rightarrow r \quad \Rightarrow \quad pa \rightarrow r)$
- making JFAs deterministic

Theorem

Every unary language accepted by a JFA is regular.

Proof Idea

In unary languages, it does not matter where the automaton jumps. □

By analogy with finite automata:

- removal of ε-*moves* $(p \to q$ and $qa \to r \quad \Rightarrow \quad pa \to r)$
- making JFAs deterministic

Theorem

Every unary language accepted by a JFA is regular.

Proof Idea

In unary languages, it does not matter where the automaton jumps. □

Corollary

The language of primes

$$\{a^p : p \text{ is a prime number}\}$$

cannot be accepted by any JFA.

Theorem

JFA *is closed under union.*

Theorem

JFA *is closed under union.*

Proof

We have: Two JFAs

- $M_1 = (Q_1, \Sigma_1, R_1, s_1, F_1)$
- $M_2 = (Q_2, \Sigma_2, R_2, s_2, F_2)$ $\quad (Q_1 \cap Q_2 = \emptyset)$

We need: JFA $H = (Q, \Sigma, R, s, F)$ such that $L(H) = L(M_1) \cup L(M_2)$

Construction:

$$
\begin{aligned}
Q &= Q_1 \cup Q_2 \cup \{s\} \qquad (s \notin Q_1 \cup Q_2) \\
\Sigma &= \Sigma_1 \cup \Sigma_2 \\
R &= R_1 \cup R_2 \cup \{s \to s_1, s \to s_2\} \\
F &= F_1 \cup F_2
\end{aligned}
$$

□

Closure Properties – Continued

Theorem

JFA *is not closed under concatenation.*

Theorem

JFA *is not closed under concatenation.*

Proof

- Consider $K_1 = \{a\}$ and $K_2 = \{b\}$.
- The JFA $M_1 = (\{s, f\}, \{a\}, \{sa \rightarrow f\}, s, \{f\})$ accepts K_1.
- The JFA $M_2 = (\{s, f\}, \{b\}, \{sb \rightarrow f\}, s, \{f\})$ accepts K_2.
- However, there is no JFA that accepts $K_1K_2 = \{ab\}$. □

	GJFA	JFA	REG
union	+	+	+
intersection	−	+	+
concatenation	−	−	+
intersection with reg. lang.	−	−	+
complement	−	+	+
shuffle	?	+	+
mirror image	?	+	+
Kleene star	?	−	+
Kleene plus	?	−	+
substitution	−	−	+
regular substitution	−	−	+
finite substitution	+	−	+
homomorphism	+	−	+
ε-free homomorphism	+	−	+
inverse homomorphism	+	+	+

Decidability – Summary

	GJFA	JFA
membership	+	+
emptiness	+	+
finiteness	+	+
infiniteness	+	+

Definition

A GJFA $M = (Q, \Sigma, R, s, F)$ is of *degree n*, where $n \geq 0$, if $py \rightarrow q \in R$ implies that $|y| \leq n$.

Definition

A GJFA $M = (Q, \Sigma, R, s, F)$ is of *degree n*, where $n \geq 0$, if $py \rightarrow q \in R$ implies that $|y| \leq n$.

Example

The GJFA $M = (\{s, p, f\}, \{a, b, c\}, R, s, \{f\})$ with

$$R = \{sabc \rightarrow p, pcc \rightarrow f, fa \rightarrow f\}$$

is of degree 3.

Definition

A GJFA $M = (Q, \Sigma, R, s, F)$ is of *degree n*, where $n \geq 0$, if $py \to q \in R$ implies that $|y| \leq n$.

Example

The GJFA $M = (\{s, p, f\}, \{a, b, c\}, R, s, \{f\})$ with

$$R = \{sabc \to p, pcc \to f, fa \to f\}$$

is of degree 3.

GJFA$_n$ the family of languages accepted by GJFAs of degree n

Definition

A GJFA $M = (Q, \Sigma, R, s, F)$ is of *degree n*, where $n \geq 0$, if $py \rightarrow q \in R$ implies that $|y| \leq n$.

Example

The GJFA $M = (\{s, p, f\}, \{a, b, c\}, R, s, \{f\})$ with

$$R = \{sabc \rightarrow p, pcc \rightarrow f, fa \rightarrow f\}$$

is of degree 3.

GJFA$_n$ the family of languages accepted by GJFAs of degree n

Theorem

GJFA$_n \subset$ **GJFA**$_{n+1}$ *for all* $n \geq 0$

Definition

A GJFA makes a *left jump* from $wxpyz$ to $wqxz$ by $py \rightarrow q$:

$$wx\underline{p}yz \;{}_{l}\!\curvearrowright\; w\underline{q}xz$$

where $w, x, y, z \in \Sigma^*$.

Left and Right Jumps

Definition

A GJFA makes a *left jump* from $wxpyz$ to $wqxz$ by $py \rightarrow q$:

$$wx\underline{p}yz \ {}_{l}\!\curvearrowright \ w\underline{q}xz$$

where $w, x, y, z \in \Sigma^*$.

Definition

A GJFA makes a *right jump* from $wpyxz$ to $wxqz$ by $py \rightarrow q$:

$$w\underline{p}yxz \ {}_{r}\!\curvearrowright \ wx\underline{q}z$$

where $w, x, y, z \in \Sigma^*$.

Definition

A GJFA makes a *left jump* from $wxpyz$ to $wqxz$ by $py \rightarrow q$:

$$w x \underline{p} y z \;{}_{l}\!\curvearrowright\; w \underline{q} x z$$

where $w, x, y, z \in \Sigma^*$.

Definition

A GJFA makes a *right jump* from $wpyxz$ to $wxqz$ by $py \rightarrow q$:

$$w \underline{p} y x z \;{}_{r}\!\curvearrowright\; w x \underline{q} z$$

where $w, x, y, z \in \Sigma^*$.

${}_{l}\mathbf{GJFA}$	GJFAs using only left jumps
${}_{l}\mathbf{JFA}$	JFAs using only left jumps
${}_{r}\mathbf{GJFA}$	GJFAs using only right jumps
${}_{r}\mathbf{JFA}$	JFAs using only right jumps

Left and Right Jumps – Results

Theorem

$_r\mathbf{GJFA} = {}_r\mathbf{JFA} = \mathbf{REG}$

Left and Right Jumps – Results

Theorem

$_r\mathbf{GJFA} = {_r\mathbf{JFA}} = \mathbf{REG}$

Proof Idea

- $_r\mathbf{JFA} = \mathbf{REG}$ simulating a finite automaton
- $_r\mathbf{GJFA} = \mathbf{REG}$ simulating a *general finite automaton* □

Left and Right Jumps – Results

Theorem

$_r\mathbf{GJFA} = {_r\mathbf{JFA}} = \mathbf{REG}$

Proof Idea

- $_r\mathbf{JFA} = \mathbf{REG}$ simulating a finite automaton
- $_r\mathbf{GJFA} = \mathbf{REG}$ simulating a *general finite automaton* □

Theorem

$_l\mathbf{JFA} - \mathbf{REG} \neq \emptyset$

Theorem

$_r\mathbf{GJFA} = {_r\mathbf{JFA}} = \mathbf{REG}$

Proof Idea

- $_r\mathbf{JFA} = \mathbf{REG}$ simulating a finite automaton
- $_r\mathbf{GJFA} = \mathbf{REG}$ simulating a *general finite automaton* □

Theorem

$_l\mathbf{JFA} - \mathbf{REG} \neq \emptyset$

Proof Idea

$$M = (\{s, p, q\}, \{a, b\}, R, s, \{s\})$$

with

$$R = \{sa \to p, pb \to s, sb \to q, qa \to s\}$$

accepts

$$_lL(M) = \{w : |w|_a = |w|_b\}$$ □

A Variety of Start Configurations

Definition

Let $M = (Q, \Sigma, R, s, F)$ be a GJFA. Set

$$
\begin{array}{lcll}
{}^{b}L(M) & = & \{w \in \Sigma^* : \underline{s}w \curvearrowright^* \underline{f} \text{ with } f \in F\} & \text{(beginning)} \\
{}^{a}L(M) & = & \{uv : u, v \in \Sigma^*, u\underline{s}v \curvearrowright^* \underline{f} \text{ with } f \in F\} & \text{(anywhere)} \\
{}^{e}L(M) & = & \{w \in \Sigma^* : w\underline{s} \curvearrowright^* \underline{f} \text{ with } f \in F\} & \text{(end)}
\end{array}
$$

Definition

Let $M = (Q, \Sigma, R, s, F)$ be a GJFA. Set

$$
\begin{array}{lcll}
{}^{b}L(M) & = & \{w \in \Sigma^* : \underline{s}w \curvearrowright^* \underline{f} \text{ with } f \in F\} & \text{(beginning)} \\
{}^{a}L(M) & = & \{uv : u, v \in \Sigma^*, u\underline{s}v \curvearrowright^* \underline{f} \text{ with } f \in F\} & \text{(anywhere)} \\
{}^{e}L(M) & = & \{w \in \Sigma^* : w\underline{s} \curvearrowright^* \underline{f} \text{ with } f \in F\} & \text{(end)}
\end{array}
$$

^{b}**GJFA**	GJFAs starting at the beginning
^{a}**GJFA**	GJFAs starting anywhere
^{e}**GJFA**	GJFAs starting at the end
^{b}**JFA**	JFAs starting at the beginning
^{a}**JFA**	JFAs starting anywhere
^{e}**JFA**	JFAs starting at the end

Definition

Let $M = (Q, \Sigma, R, s, F)$ be a GJFA. Set

$$
\begin{aligned}
{}^{b}L(M) &= \{w \in \Sigma^* : \underline{s}w \curvearrowright^* \underline{f} \text{ with } f \in F\} && \text{(beginning)}\\
{}^{a}L(M) &= \{uv : u, v \in \Sigma^*, u\underline{s}v \curvearrowright^* \underline{f} \text{ with } f \in F\} && \text{(anywhere)}\\
{}^{e}L(M) &= \{w \in \Sigma^* : w\underline{s} \curvearrowright^* \underline{f} \text{ with } f \in F\} && \text{(end)}
\end{aligned}
$$

^{b}**GJFA**	GJFAs starting at the beginning
^{a}**GJFA**	GJFAs starting anywhere
^{e}**GJFA**	GJFAs starting at the end
^{b}**JFA**	JFAs starting at the beginning
^{a}**JFA**	JFAs starting anywhere
^{e}**JFA**	JFAs starting at the end

Observations:

- ${}^{a}L(M) = L(M)$
- ^{a}**GJFA** $=$ **GJFA** and ^{a}**JFA** $=$ **JFA**

A Variety of Start Configurations – Results

Theorem

${}^{a}\mathbf{JFA} \subset {}^{b}\mathbf{JFA}$

Theorem

${}^{a}\mathbf{JFA} \subset {}^{b}\mathbf{JFA}$

Proof Idea

The JFA

$$M = (\{s, f\}, \{a, b\}, \{sa \rightarrow f, fb \rightarrow f\}, s, \{f\})$$

satisfies ${}^{b}L(M) = \{a\}\{b\}^* \ (\{a\}\{b\}^* \notin {}^{a}\mathbf{JFA})$. □

Theorem

$^{a}\mathbf{JFA} \subset {}^{b}\mathbf{JFA}$

Proof Idea

The JFA

$$M = (\{s, f\}, \{a, b\}, \{sa \to f, fb \to f\}, s, \{f\})$$

satisfies $^{b}L(M) = \{a\}\{b\}^{*}$ $(\{a\}\{b\}^{*} \notin {}^{a}\mathbf{JFA})$. □

Theorem

$^{a}\mathbf{GJFA} \subset {}^{b}\mathbf{GJFA}$

Theorem

${}^{a}\mathbf{JFA} \subset {}^{b}\mathbf{JFA}$

Proof Idea

The JFA

$$M = (\{s, f\}, \{a, b\}, \{sa \to f, fb \to f\}, s, \{f\})$$

satisfies ${}^{b}L(M) = \{a\}\{b\}^{*}$ $(\{a\}\{b\}^{*} \notin {}^{a}\mathbf{JFA})$. □

Theorem

${}^{a}\mathbf{GJFA} \subset {}^{b}\mathbf{GJFA}$

Theorem

${}^{e}\mathbf{GJFA} = {}^{a}\mathbf{GJFA}$ *and* ${}^{e}\mathbf{JFA} = {}^{a}\mathbf{JFA}$

Conclusion and Open Problem Areas

- closure properties of **GJFA** (shuffle, Kleene star, Kleene plus, and mirror image)
- other decision problems of **GJFA** and **JFA**, like equivalence, universality, inclusion, or regularity
- the effect of left jumps to the power of JFAs and GJFAs (we only know that $_{l}\mathbf{JFA} - \mathbf{REG} \neq \emptyset$)
- strict determinism (precisely determine where to jump)
- applications: verification of a relation concerning the number of symbol occurrences (genetics)

$\Box$ a blank
V the alphabet of letters
$W = V \cup \{\Box\}$

Definition

A string $w \in W^*$ is *useful* if it contains more letters than blanks.

$\square$ a blank
V the alphabet of letters
$W = V \cup \{\square\}$

Definition

A string $w \in W^*$ is *useful* if it contains more letters than blanks.

Objective: Acceptance of all useful strings

Illustration

Acceptance: this□is□usefull□information
Rejection: □□□□□□□use□less□info□□□□□□

The next JFA performs this acceptance:

$$M = (\{s, p, f\}, W, R, s, \{f\})$$

where R contains the following rules:

$$\begin{array}{rcll} s\square & \rightarrow & p & \\ pa & \rightarrow & s & \text{for each } a \in V \\ sa & \rightarrow & f & \text{for each } a \in V \\ fa & \rightarrow & f & \text{for each } a \in V \end{array}$$

The next JFA performs this acceptance:

$$M = (\{s, p, f\}, W, R, s, \{f\})$$

where R contains the following rules:

$$\begin{array}{rcll} s\Box & \rightarrow & p & \\ pa & \rightarrow & s & \text{for each } a \in V \\ sa & \rightarrow & f & \text{for each } a \in V \\ fa & \rightarrow & f & \text{for each } a \in V \end{array}$$

Implementation: Where to jump? Jump to the leftmost possible symbol that can be read.

Discussion