MANCHESTER 1824
The University of Manchester

Context-Free Graphs in Inverse Semigroup Theory

Jan Philipp **Wächter**

Department of Mathematics
University of Manchester

This research was supported by EPSRC

19 May 2026

Schützenberger Graphs

Schützenberger Graphs

Let $M = \mathrm{Inv}\langle A \mid \mathcal{R} \rangle$ be an inverse monoid

Schützenberger Graphs

Let $M = \mathrm{Inv}\langle A \mid \mathcal{R} \rangle$ be an inverse monoid,
let $A^{-1} = \{a^{-1} \mid a \in A\}$ be the disjoint set of formal inverses of A with $(a^{-1})^{-1} = a$

Schützenberger Graphs

Let $M = \mathrm{Inv}\langle A \mid \mathcal{R} \rangle$ be an inverse monoid,
let $A^{-1} = \{a^{-1} \mid a \in A\}$ be the disjoint set of formal inverses of A with $(a^{-1})^{-1} = a$ and
let $A^{\pm *} = (A \uplus A^{-1})^*$ be the set of words over A and A^{-1}.

Schützenberger Graphs

Let $M = \mathrm{Inv}\langle A \mid \mathcal{R} \rangle$ be an inverse monoid,
let $A^{-1} = \{a^{-1} \mid a \in A\}$ be the disjoint set of formal inverses of A with $(a^{-1})^{-1} = a$ and
let $A^{\pm *} = (A \uplus A^{-1})^*$ be the set of words over A and A^{-1}.

The Cayley graph $\mathscr{C}(M)$ (for the generating set A) is the directed graph with

Schützenberger Graphs

Let $M = \mathrm{Inv}\langle A \mid \mathcal{R} \rangle$ be an inverse monoid,
let $A^{-1} = \{a^{-1} \mid a \in A\}$ be the disjoint set of formal inverses of A with $(a^{-1})^{-1} = a$ and
let $A^{\pm *} = (A \uplus A^{-1})^*$ be the set of words over A and A^{-1}.

The Cayley graph $\mathscr{C}(M)$ (for the generating set A) is the directed graph with
nodes M

Schützenberger Graphs

Let $M = \mathrm{Inv}\langle A \mid \mathcal{R}\rangle$ be an inverse monoid,
let $A^{-1} = \{a^{-1} \mid a \in A\}$ be the disjoint set of formal inverses of A with $(a^{-1})^{-1} = a$ and
let $A^{\pm *} = (A \uplus A^{-1})^*$ be the set of words over A and A^{-1}.

The Cayley graph $\mathscr{C}(M)$ (for the generating set A) is the directed graph with
nodes M and edges $E = \{s \xrightarrow{a} sa \mid s \in M, a \in A^{\pm 1}\}$.

Schützenberger Graphs

Let $M = \mathrm{Inv}\langle A \mid \mathcal{R}\rangle$ be an inverse monoid,
let $A^{-1} = \{a^{-1} \mid a \in A\}$ be the disjoint set of formal inverses of A with $(a^{-1})^{-1} = a$ and
let $A^{\pm *} = (A \uplus A^{-1})^*$ be the set of words over A and A^{-1}.

The Cayley graph $\mathscr{C}(M)$ (for the generating set A) is the directed graph with
nodes M and edges $E = \{s \xrightarrow{a} sa \mid s \in M, a \in A^{\pm 1}\}$.

Definition (Schützenberger Graph)

The Schützenberger graph $S\Gamma(s)$ of $s \in M$ is the strongly connected component of $\mathscr{C}(M)$ containing s.

Schützenberger Graphs

Let $M = \mathrm{Inv}\langle A \mid \mathcal{R}\rangle$ be an inverse monoid,
let $A^{-1} = \{a^{-1} \mid a \in A\}$ be the disjoint set of formal inverses of A with $(a^{-1})^{-1} = a$ and
let $A^{\pm *} = (A \uplus A^{-1})^*$ be the set of words over A and A^{-1}.

The Cayley graph $\mathscr{C}(M)$ (for the generating set A) is the directed graph with
nodes M and edges $E = \{s \xrightarrow{a} sa \mid s \in M, a \in A^{\pm 1}\}$.

Definition (Schützenberger Graph)

The Schützenberger graph $S\Gamma(s)$ of $s \in M$ is the strongly connected component of $\mathscr{C}(M)$ containing s. This is the $\mathcal{R}$-class of s!

Schützenberger Graphs

Let $M = \mathrm{Inv}\langle A \mid \mathcal{R}\rangle$ be an inverse monoid,
let $A^{-1} = \{a^{-1} \mid a \in A\}$ be the disjoint set of formal inverses of A with $(a^{-1})^{-1} = a$ and
let $A^{\pm *} = (A \uplus A^{-1})^*$ be the set of words over A and A^{-1}.

The Cayley graph $\mathscr{C}(M)$ (for the generating set A) is the directed graph with
nodes M and edges $E = \{s \xrightarrow{a} sa \mid s \in M, a \in A^{\pm 1}\}$.

Definition (Schützenberger Graph)

The Schützenberger graph $S\Gamma(s)$ of $s \in M$ is the strongly connected component of $\mathscr{C}(M)$ containing s. This is the $\mathcal{R}$-class of s!

An inverse monoid is "fully determined" by its Schützenberger graphs.

Schützenberger Graphs

Let $M = \mathrm{Inv}\langle A \mid \mathcal{R} \rangle$ be an inverse monoid,
let $A^{-1} = \{a^{-1} \mid a \in A\}$ be the disjoint set of formal inverses of A with $(a^{-1})^{-1} = a$ and
let $A^{\pm *} = (A \uplus A^{-1})^*$ be the set of words over A and A^{-1}.

The Cayley graph $\mathscr{C}(M)$ (for the generating set A) is the directed graph with
nodes M and edges $E = \{s \xrightarrow{a} sa \mid s \in M, a \in A^{\pm 1}\}$.

Definition (Schützenberger Graph)

The Schützenberger graph $S\Gamma(s)$ of $s \in M$ is the strongly connected component of $\mathscr{C}(M)$ containing s. This is the $\mathcal{R}$-class of s!

An inverse monoid is "fully determined" by its Schützenberger graphs.

Schützenberger graphs have many "desirable" properties of group Cayley graphs!

Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

Properties of Schützenberger Graphs

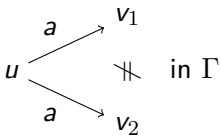
Any Schützenberger graphs Γ is...

- deterministic:

Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

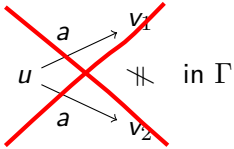
- deterministic:



Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

- deterministic:

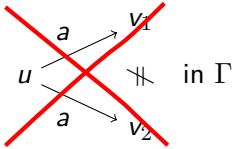


in Γ

Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

- deterministic:



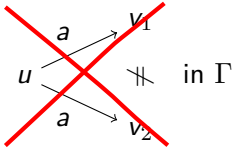
in Γ

- involutive:

Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

- deterministic:



$\nexists$ in Γ

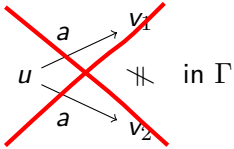
- involutive:

$u \xrightarrow{a} v$ in $\Gamma \iff u \xleftarrow{a^{-1}} v$ in Γ

Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

- deterministic:



in Γ

- involutive:

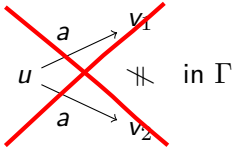
$u \xrightarrow{a} v$ in $\Gamma \iff u \xleftarrow{a^{-1}} v$ in Γ

- co-deterministic:

Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

- deterministic:

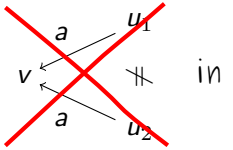


$\nexists$ in Γ

- involutive:

$u \xrightarrow{a} v$ in $\Gamma \iff u \xleftarrow{a^{-1}} v$ in Γ

- co-deterministic:

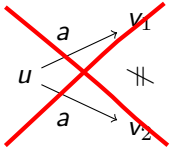


$\nexists$ in Γ

Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

- deterministic:

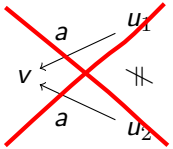


$\nexists$ in Γ

- involutive:

$u \xrightarrow{a} v$ in $\Gamma \iff u \xleftarrow{a^{-1}} v$ in Γ

- co-deterministic: implied by the above!

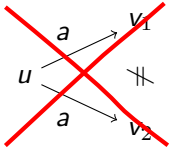


$\nexists$ in Γ

Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

- deterministic:

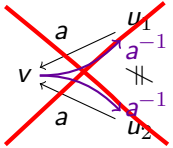


$\nexists$ in Γ

- involutive:

$u \xrightarrow{a} v$ in $\Gamma \iff u \xleftarrow{a^{-1}} v$ in Γ

- co-deterministic: implied by the above!

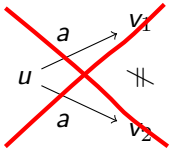


in Γ

Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

- deterministic:

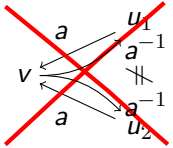


$\nexists$ in Γ

- involutive:

$u \xrightarrow{a} v$ in $\Gamma \iff u \xleftarrow{a^{-1}} v$ in Γ

- co-deterministic: implied by the above!



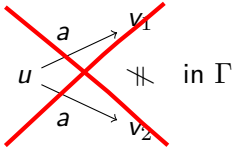
$\nexists$ in Γ

- (strongly) connected

Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

- deterministic:

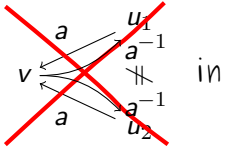


$\nexists$ in Γ

- involutive:

$u \xrightarrow{a} v$ in $\Gamma \iff u \xleftarrow{a^{-1}} v$ in Γ

- co-deterministic: implied by the above!



$\nexists$ in Γ

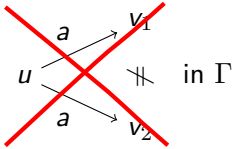
- (strongly) connected

inverse graph

Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

- deterministic:

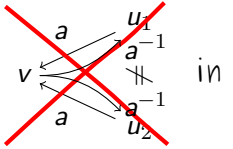


$\nexists$ in Γ

- involutive:

$u \xrightarrow{a} v$ in $\Gamma \iff u \xleftarrow{a^{-1}} v$ in Γ

- co-deterministic: implied by the above!



$\nexists$ in Γ

- (strongly) connected

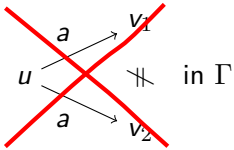
inverse graph

What can we use Schützenberger graphs for?

Properties of Schützenberger Graphs

Any Schützenberger graphs Γ is...

- deterministic:

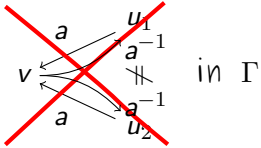


$\not\in$ in Γ

- involutive:

$u \xrightarrow{a} v$ in $\Gamma \iff u \xleftarrow{a^{-1}} v$ in Γ

- co-deterministic: implied by the above!



$\not\in$ in Γ

- (strongly) connected

inverse graph

Intermezzo!

~~What can we use Schützenberger graphs for?~~

Intermezzo: Homomorphisms and Isomorphism of Directed Graphs

Intermezzo: Homomorphisms and Isomorphism of Directed Graphs

Consider directed $A^{\pm 1}$-graphs $\Gamma = (V, E)$ and $\Delta = (W, F)$.

Intermezzo: Homomorphisms and Isomorphism of Directed Graphs

Consider directed $A^{\pm 1}$-graphs $\Gamma = (V, E)$ and $\Delta = (W, F)$.

A homomorphism $\varphi\colon \Gamma \to \Delta$ is a map $V \to W$ with

$$v_1 \xrightarrow{a} v_2 \in E \implies \varphi(v_1) \xrightarrow{a} \varphi(v_2) \in F \quad \text{for all } a \in A^{\pm 1}.$$

Intermezzo: Homomorphisms and Isomorphism of Directed Graphs

Consider directed $A^{\pm 1}$-graphs $\Gamma = (V, E)$ and $\Delta = (W, F)$.

A homomorphism $\varphi\colon \Gamma \to \Delta$ is a map $V \to W$ with

$$v_1 \xrightarrow{a} v_2 \in E \implies \varphi(v_1) \xrightarrow{a} \varphi(v_2) \in F \quad \text{for all } a \in A^{\pm 1}.$$

An isomorphism $\varphi\colon \Gamma \to \Delta$ is a bijection $V \to W$ with

$$v_1 \xrightarrow{a} v_2 \in E \iff \varphi(v_1) \xrightarrow{a} \varphi(v_2) \in F \quad \text{for all } a \in A^{\pm 1}.$$

Intermezzo: Homomorphisms and Isomorphism of Directed Graphs

Consider directed $A^{\pm 1}$-graphs $\Gamma = (V, E)$ and $\Delta = (W, F)$.

A homomorphism $\varphi\colon \Gamma \to \Delta$ is a map $V \to W$ with

$$v_1 \xrightarrow{a} v_2 \in E \implies \varphi(v_1) \xrightarrow{a} \varphi(v_2) \in F \quad \text{for all } a \in A^{\pm 1}.$$

An isomorphism $\varphi\colon \Gamma \to \Delta$ is a bijection $V \to W$ with

$$v_1 \xrightarrow{a} v_2 \in E \iff \varphi(v_1) \xrightarrow{a} \varphi(v_2) \in F \quad \text{for all } a \in A^{\pm 1}.$$

An automorphism of Γ is an isomorphism $\Gamma \to \Gamma$.

Intermezzo: Homomorphisms and Isomorphism of Directed Graphs

Consider directed $A^{\pm 1}$-graphs $\Gamma = (V, E)$ and $\Delta = (W, F)$.

A homomorphism $\varphi\colon \Gamma \to \Delta$ is a map $V \to W$ with

$$v_1 \xrightarrow{a} v_2 \in E \implies \varphi(v_1) \xrightarrow{a} \varphi(v_2) \in F \quad \text{for all } a \in A^{\pm 1}.$$

An isomorphism $\varphi\colon \Gamma \to \Delta$ is a bijection $V \to W$ with

$$v_1 \xrightarrow{a} v_2 \in E \iff \varphi(v_1) \xrightarrow{a} \varphi(v_2) \in F \quad \text{for all } a \in A^{\pm 1}.$$

An automorphism of Γ is an isomorphism $\Gamma \to \Gamma$.

Now: What can we use Schützenberger graphs for?

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\mathrm{Aut}\, S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\operatorname{Aut} S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1}$

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\mathrm{Aut}\, S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1}$
- $s \mathcal{L}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $s \mapsto t$

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\mathrm{Aut}\, S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1}$
- $s \mathcal{L}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $s \mapsto t$
- $s \mathcal{D}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\mathrm{Aut}\, S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1}$
- $s \mathcal{L}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $s \mapsto t$
- $s \mathcal{D}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$
- $s \geq_{\mathcal{J}} t$

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\mathrm{Aut}\, S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1}$
- $s \mathcal{L}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $s \mapsto t$
- $s \mathcal{D}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$
- $s \geq_{\mathcal{J}} t \iff MsM \supseteq MtM$

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\operatorname{Aut} S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1}$
- $s \mathcal{L}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $s \mapsto t$
- $s \mathcal{D}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$
- $s \geq_{\mathcal{J}} t \iff MsM \supseteq MtM$
  $\iff$ there is a homomorphism $S\Gamma(s) \to S\Gamma(t)$

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\mathrm{Aut}\, S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1}$
- $s \mathcal{L}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $s \mapsto t$
- $s \mathcal{D}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$
- $s \geq_{\mathcal{J}} t \iff MsM \supseteq MtM$
  $\iff$ there is a homomorphism $S\Gamma(s) \to S\Gamma(t)$

If we want to decide these algorithmically, we need to

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\mathrm{Aut}\,S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1}$
- $s \mathcal{L}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $s \mapsto t$
- $s \mathcal{D}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$
- $s \geq_{\mathcal{J}} t \iff MsM \supseteq MtM$
  $\iff$ there is a homomorphism $S\Gamma(s) \to S\Gamma(t)$

If we want to decide these algorithmically, we need to

1. compute/describe $S\Gamma(s)$

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\mathrm{Aut}\, S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1}$
- $s \mathcal{L}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $s \mapsto t$
- $s \mathcal{D}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$
- $s \geq_{\mathcal{J}} t \iff MsM \supseteq MtM$
  $\iff$ there is a homomorphism $S\Gamma(s) \to S\Gamma(t)$

If we want to decide these algorithmically, we need to

1. compute/describe $S\Gamma(s)$ and
2. decide homomorphism/isomorphism problems.

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\mathrm{Aut}\, S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1} \iff ss^{-1} = tt^{-1}$
- $s \mathcal{L}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $s \mapsto t \iff s^{-1}s = t^{-1}t$
- $s \mathcal{D}\ t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$
- $s \geq_{\mathcal{J}} t \iff MsM \supseteq MtM$
  $\iff$ there is a homomorphism $S\Gamma(s) \to S\Gamma(t)$

We could use the word problem…

If we want to decide these algorithmically, we need to

1. compute/describe $S\Gamma(s)$ and
2. decide homomorphism/isomorphism problems.

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\operatorname{Aut} S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R} t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1} \iff ss^{-1} = tt^{-1}$
- $s \mathcal{L} t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $s \mapsto t \iff s^{-1}s = t^{-1}t$
- $s \mathcal{D} t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$
- $s \geq_{\mathcal{J}} t \iff MsM \supseteq MtM$
  $\iff$ there is a homomorphism $S\Gamma(s) \to S\Gamma(t)$

We could use the word problem…

If we want to decide these algorithmically, we need to

1. compute/describe $S\Gamma(s)$ and
2. decide homomorphism/isomorphism problems.

We may use Stephen's procedure to "compute" $S\Gamma(s)$.

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\operatorname{Aut} S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R} t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1} \iff ss^{-1} = tt^{-1}$
- $s \mathcal{L} t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $s \mapsto t \iff s^{-1}s = t^{-1}t$
- $s \mathcal{D} t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$
- $s \geq_{\mathcal{J}} t \iff MsM \supseteq MtM$
  $\iff$ there is a homomorphism $S\Gamma(s) \to S\Gamma(t)$

We could use the word problem…

If we want to decide these algorithmically, we need to

1. compute/describe $S\Gamma(s)$ and
2. decide homomorphism/isomorphism problems.

We may use Stephen's procedure to "compute" $S\Gamma(s)$. But: this quickly yields undecidability!

Schützenberger Graphs, Maximal Subgroups and Green's Relations

- $\operatorname{Aut} S\Gamma(s)$ is isomorphic to the maximal subgroup of M containing ss^{-1}. (Stephen, 1990)
- $s \mathcal{R} t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $ss^{-1} \mapsto tt^{-1} \iff ss^{-1} = tt^{-1}$
- $s \mathcal{L} t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$ mapping $s \mapsto t \iff s^{-1}s = t^{-1}t$
- $s \mathcal{D} t \iff$ there is an isomorphism $S\Gamma(s) \to S\Gamma(t)$
- $s \geq_{\mathcal{J}} t \iff MsM \supseteq MtM$
  $\iff$ there is a homomorphism $S\Gamma(s) \to S\Gamma(t)$

We could use the word problem…

If we want to decide these algorithmically, we need to

1. compute/describe $S\Gamma(s)$ and
2. decide homomorphism/isomorphism problems.

We may use Stephen's procedure to "compute" $S\Gamma(s)$. But: this quickly yields undecidability!

What else can we do?

Schützenberger Graphs as Automata

Schützenberger Graphs as Automata

Definition

A deterministic automaton $\mathcal{A}$ is a inverse $A^{\pm 1}$-graph

Schützenberger Graphs as Automata

Definition

A deterministic automaton $\mathcal{A}$ is a inverse $A^{\pm 1}$-graph with some node q_0 marked as initial

Schützenberger Graphs as Automata

Definition

A deterministic automaton $\mathcal{A}$ is a inverse $A^{\pm 1}$-graph with some node q_0 marked as initial and possibly multiple final nodes F.

Schützenberger Graphs as Automata

Definition

A deterministic automaton $\mathcal{A}$ is a inverse $A^{\pm 1}$-graph with some node q_0 marked as initial and possibly multiple final nodes F.
It accepts the language $\mathscr{L}(\mathcal{A}) = \{w \in A^{\pm *} \mid q_0 \overset{w}{\rightsquigarrow} f \in F\}$.

A language is a subset of $A^{\pm *}$.

Schützenberger Graphs as Automata

Definition

A deterministic automaton $\mathcal{A}$ is a inverse $A^{\pm 1}$-graph with some node q_0 marked as initial and possibly multiple final nodes F.
It accepts the language $\mathscr{L}(\mathcal{A}) = \{w \in A^{\pm *} \mid q_0 \rightsquigarrow_w f \in F\}$.

A language is a subset of $A^{\pm *}$.

Every automaton accepts a unique language

Schützenberger Graphs as Automata

Definition

A deterministic automaton $\mathcal{A}$ is a inverse $A^{\pm 1}$-graph with some node q_0 marked as initial and possibly multiple final nodes F.
It accepts the language $\mathscr{L}(\mathcal{A}) = \{w \in A^{\pm *} \mid q_0 \rightsquigarrow_w f \in F\}$.

A language is a subset of $A^{\pm *}$.

Every automaton accepts a unique language and
every language $L \subseteq A^{\pm *}$ has a unique minimal automaton.

Schützenberger Graphs as Automata

Definition

A deterministic automaton $\mathcal{A}$ is a inverse $A^{\pm 1}$-graph with some node q_0 marked as initial and possibly multiple final nodes F.
It accepts the language $\mathscr{L}(\mathcal{A}) = \{w \in A^{\pm *} \mid q_0 \rightsquigarrow_w f \in F\}$.

A language is a subset of $A^{\pm *}$.

Every automaton accepts a unique language and
every language $L \subseteq A^{\pm *}$ has a unique minimal automaton.

Theorem (Stephen, 1990)

$S\Gamma(s)$ *with initial node* ss^{-1}

Schützenberger Graphs as Automata

Definition

A deterministic automaton $\mathcal{A}$ is a inverse $A^{\pm 1}$-graph with some node q_0 marked as initial and possibly multiple final nodes F.
It accepts the language $\mathscr{L}(\mathcal{A}) = \{w \in A^{\pm *} \mid q_0 \overset{w}{\rightsquigarrow} f \in F\}$.

A language is a subset of $A^{\pm *}$.

Every automaton accepts a unique language and
every language $L \subseteq A^{\pm *}$ has a unique minimal automaton.

Theorem (Stephen, 1990)

$S\Gamma(s)$ with initial node ss^{-1} and final node s

Schützenberger Graphs as Automata

Definition

A deterministic automaton $\mathcal{A}$ is a inverse $A^{\pm 1}$-graph with some node q_0 marked as initial and possibly multiple final nodes F.
It accepts the language $\mathscr{L}(\mathcal{A}) = \{w \in A^{\pm *} \mid q_0 \overset{w}{\rightsquigarrow} f \in F\}$.

A language is a subset of $A^{\pm *}$.

Every automaton accepts a unique language and
every language $L \subseteq A^{\pm *}$ has a unique minimal automaton.

Theorem (Stephen, 1990)

$S\Gamma(s)$ with initial node ss^{-1} and final node s is the minimal automaton for the language

Schützenberger Graphs as Automata

Definition

A deterministic automaton $\mathcal{A}$ is a inverse $A^{\pm 1}$-graph with some node q_0 marked as initial and possibly multiple final nodes F.
It accepts the language $\mathscr{L}(\mathcal{A}) = \{w \in A^{\pm *} \mid q_0 \rightsquigarrow_w f \in F\}$.

A language is a subset of $A^{\pm *}$.

Every automaton accepts a unique language and
every language $L \subseteq A^{\pm *}$ has a unique minimal automaton.

Theorem (Stephen, 1990)

$S\Gamma(s)$ with initial node ss^{-1} and final node s is the minimal automaton for the language

$$\mathscr{L}(S\Gamma(s)) = \mathscr{U}(s) = \{u \in A^{\pm *} \mid u \geq s \text{ in } M\}.$$

Schützenberger Graphs as Automata

Definition

A deterministic automaton $\mathcal{A}$ is a inverse $A^{\pm 1}$-graph with some node q_0 marked as initial and possibly multiple final nodes F.
It accepts the language $\mathscr{L}(\mathcal{A}) = \{w \in A^{\pm *} \mid q_0 \overset{w}{\rightsquigarrow} f \in F\}$.

A language is a subset of $A^{\pm *}$.

Every automaton accepts a unique language and
every language $L \subseteq A^{\pm *}$ has a unique minimal automaton.

Theorem (Stephen, 1990)

$S\Gamma(s)$ with initial node ss^{-1} and final node s is the minimal automaton for the language

$$\mathscr{L}(S\Gamma(s)) = \mathscr{U}(s) = \{u \in A^{\pm *} \mid u \geq s \text{ in } M\}.$$

Thus: $\mathscr{L}(S\Gamma(s)) = \mathscr{U}(s)$ fully describes $S\Gamma(s)$!

Schützenberger Graphs as Automata

Definition

A deterministic automaton $\mathcal{A}$ is a inverse $A^{\pm 1}$-graph with some node q_0 marked as initial and possibly multiple final nodes F.
It accepts the language $\mathscr{L}(\mathcal{A}) = \{w \in A^{\pm *} \mid q_0 \overset{w}{\rightsquigarrow} f \in F\}$.

A language is a subset of $A^{\pm *}$.

Every automaton accepts a unique language and
every language $L \subseteq A^{\pm *}$ has a unique minimal automaton.

Theorem (Stephen, 1990)

$S\Gamma(s)$ with initial node ss^{-1} and final node s is the minimal automaton for the language

$$\mathscr{L}(S\Gamma(s)) = \mathscr{U}(s) = \{u \in A^{\pm *} \mid u \geq s \text{ in } M\}.$$

Thus: $\mathscr{L}(S\Gamma(s)) = \mathscr{U}(s)$ fully describes $S\Gamma(s)$! But: undecidable in general

Tree-Like Graphs

Tree-Like Graphs

A directed graph is tree-like

Tree-Like Graphs

A directed graph is tree-like
$\iff$ it is quasi-isometric to a tree

Tree-Like Graphs

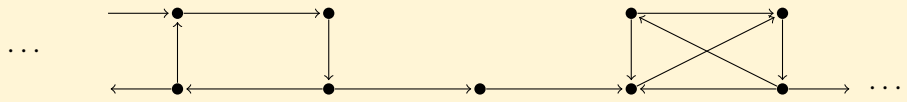
A directed graph is tree-like
$\iff$ it is quasi-isometric to a tree
$\iff$ it has a strong tree decomposition

Example

Tree-Like Graphs

A directed graph is tree-like
$\iff$ it is quasi-isometric to a tree
$\iff$ it has a strong tree decomposition

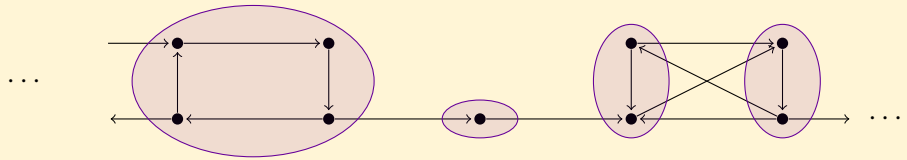
Example



Tree-Like Graphs

A directed graph is tree-like
$\iff$ it is quasi-isometric to a tree
$\iff$ it has a strong tree decomposition

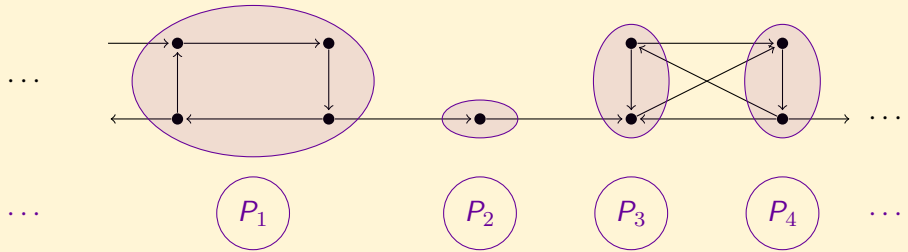
Example



Tree-Like Graphs

A directed graph is tree-like
$\iff$ it is quasi-isometric to a tree
$\iff$ it has a strong tree decomposition

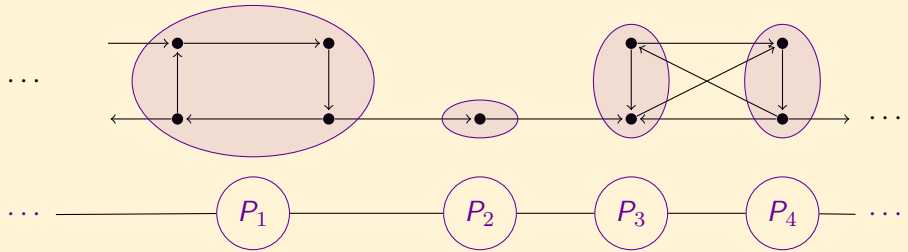
Example



Tree-Like Graphs

A directed graph is tree-like
$\iff$ it is quasi-isometric to a tree
$\iff$ it has a strong tree decomposition

Example



Tree-Like Inverse Monoids

Tree-Like Inverse Monoids

Theorem (Gray, Silva, Szakács, 2022)

Let $M = \mathrm{Inv}\langle A \mid \ell_1 = r_1, \ldots, \ell_n = r_n \rangle$ *be finitely presented*

Tree-Like Inverse Monoids

Theorem (Gray, Silva, Szakács, 2022)

Let $M = \mathrm{Inv}\langle A \mid \ell_1 = r_1, \ldots, \ell_n = r_n \rangle$ *be finitely presented and* $s \in M$. *Then:*

$$S\Gamma(s) \text{ is tree-like} \implies \mathscr{L}(S\Gamma(s)) \text{ is context-free}$$

Tree-Like Inverse Monoids

Theorem (Gray, Silva, Szakács, 2022)

Let $M = \mathrm{Inv}\langle A \mid \ell_1 = r_1, \ldots, \ell_n = r_n \rangle$ *be finitely presented and* $s \in M$. *Then:*

$$S\Gamma(s) \text{ is tree-like} \implies \mathscr{L}(S\Gamma(s)) \text{ is context-free}$$

In fact: This is true for certain tree-like subgraphs of Schützenberger graphs as well!

Tree-Like Inverse Monoids

Theorem (Gray, Silva, Szakács, 2022)

Let $M = \mathrm{Inv}\langle A \mid \ell_1 = r_1, \ldots, \ell_n = r_n \rangle$ *be finitely presented and* $s \in M$. *Then:*

$$S\Gamma(s) \text{ is tree-like} \implies \mathscr{L}(S\Gamma(s)) \text{ is context-free}$$

In fact: This is true for certain tree-like subgraphs of Schützenberger graphs as well!

Used, for example, in:

Theorem (Gray, Kambites, W., WIP)

G, H: finitely presented groups Then:
There is a finitely presented special inverse monoid with

- *G as group of units and*
- *H as the maximal subgroup of some idempotent.*

Tree-Like Inverse Monoids

Theorem (Gray, Silva, Szakács, 2022)

Let $M = \mathrm{Inv}\langle A \mid \ell_1 = r_1, \ldots, \ell_n = r_n \rangle$ *be finitely presented and* $s \in M$. *Then:*

$$S\Gamma(s) \text{ is tree-like} \implies \mathscr{L}(S\Gamma(s)) \text{ is context-free}$$

In fact: This is true for certain tree-like subgraphs of Schützenberger graphs as well!

Used, for example, in:

Theorem (Gray, Kambites, W., WIP)

G, H: finitely presented groups Then:
There is a finitely presented special inverse monoid with

- *G as group of units and*
- *H as the maximal subgroup of some idempotent.*

What is a context-free language?

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free } \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free } \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free } \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet
- Γ: stack alphabet

Context-Free Languages

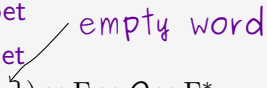
For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet
- Γ: stack alphabet
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$



Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet
- Γ: stack alphabet
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$: transition relation

empty word

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet
- Γ: stack alphabet
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$: transition relation with elements $\tau \in \delta$ written as $\tau = p, a_0, A \to q, \gamma$

empty word

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- $q_0 \in Q$: initial state
- A: input alphabet
- Γ: stack alphabet
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$: transition relation with elements $\tau \in \delta$ written as $\tau = p, a_0, A \to q, \gamma$

empty word

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet
- Γ: stack alphabet
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$: transition relation with elements $\tau \in \delta$ written as $\tau = p, a_0, A \to q, \gamma$
- $q_0 \in Q$: initial state
- $\bot \in \Gamma$: stack bottom symbol

empty word

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet
- Γ: stack alphabet
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$: transition relation with elements $\tau \in \delta$ written as $\tau = p, a_0, A \to q, \gamma$
- $q_0 \in Q$: initial state
- $\bot \in \Gamma$: stack bottom symbol
- $F \subseteq Q$: final states

empty word

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet
- Γ: stack alphabet
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$: transition relation with elements $\tau \in \delta$ written as $\tau = p, a_0, A \to q, \gamma$
- $q_0 \in Q$: initial state
- $\bot \in \Gamma$: stack bottom symbol
- $F \subseteq Q$: final states

empty word

Configuration:

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet
- Γ: stack alphabet
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$: transition relation with elements $\tau \in \delta$ written as $\tau = p, a_0, A \to q, \gamma$
- $q_0 \in Q$: initial state
- $\bot \in \Gamma$: stack bottom symbol
- $F \subseteq Q$: final states

empty word

Configuration: $w \in A^*$

remaining input word: w

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet
- Γ: stack alphabet
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$: transition relation with elements $\tau \in \delta$ written as $\tau = p, a_0, A \to q, \gamma$
- $q_0 \in Q$: initial state
- $\bot \in \Gamma$: stack bottom symbol
- $F \subseteq Q$: final states

empty word

Configuration: $w \in A^*$, $p \in Q$

remaining input word: w

state: p

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet
- Γ: stack alphabet
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$: transition relation with elements $\tau \in \delta$ written as $\tau = p, a_0, A \to q, \gamma$
- $q_0 \in Q$: initial state
- $\bot \in \Gamma$: stack bottom symbol
- $F \subseteq Q$: final states

empty word

Configuration: $w \in A^*$, $p \in Q$, $\gamma' \in \Gamma^*$

remaining input word: w

state: p

stack: γ'

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet
- Γ: stack alphabet
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$: transition relation with elements $\tau \in \delta$ written as $\tau = p, a_0, A \to q, \gamma$
- $q_0 \in Q$: initial state
- $\bot \in \Gamma$: stack bottom symbol
- $F \subseteq Q$: final states

empty word

Configuration: $w \in A^*$, $p \in Q$, $\gamma' \in \Gamma^*$

remaining input word:	$a_0 w$
state:	p
stack:	$A\gamma'$

Context-Free Languages

For us:

$$L \subseteq A^* \text{ is context-free} \iff L = \mathscr{L}(\mathcal{C}) \text{ for some pushdown automaton } \mathcal{C}$$

Definition

A pushdown automaton $\mathcal{C} = (Q, A, \Gamma, \delta, q_0, \bot, F)$ consists of:

- Q: finite set of states
- A: input alphabet
- Γ: stack alphabet
- $q_0 \in Q$: initial state
- $\bot \in \Gamma$: stack bottom symbol
- $F \subseteq Q$: final states
- $\delta \subseteq Q \times (A \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$: transition relation with elements $\tau \in \delta$ written as $\tau = p, a_0, A \to q, \gamma$

empty word

Configuration: $w \in A^*$, $p \in Q$, $\gamma' \in \Gamma^*$

remaining input word:	$a_0 w$		w
state:	p	$\vdash_\tau$	q
stack:	$A\gamma'$		$\gamma\gamma'$

Example PDA

Example

Example PDA

Example

states: p (initial), q, f (final)

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\perp$ (bottom symbol)

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\perp$ (bottom symbol)
transitions: $p, a, \perp \to p, A\perp$

Example PDA

Example

states: p (initial), q, f (final)
alphabet: $\{a, b\}$
stack symbols: A, $\perp$ (bottom symbol)
transitions: $p, a, \perp \to p, A\perp$
$p, a, A \to p, AA$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\perp$ (bottom symbol)

transitions: $p, a, \perp \to p, A\perp$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\perp$ (bottom symbol)

transitions: $p, a, \perp \to p, A\perp$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\perp$ (bottom symbol)

transitions: $p, a, \perp \to p, A\perp$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \perp \to f, \varepsilon$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$

remaining input: $aabb$

state: p

stack: $\bot$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$

remaining input: $aabb$

state: p

stack: $\bot$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$

remaining input:	$aabb$		abb
state:	p	$\vdash$	p
stack:	$\bot$		$A\bot$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$

remaining input:	$aabb$		abb
state:	p	$\vdash$	p
stack:	$\bot$		$A\bot$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$

remaining input:	$aabb$		abb	bb
state:	p	$\vdash$	p	$\vdash$ p
stack:	$\bot$		$A\bot$	$AA\bot$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$

remaining input:	$aabb$		abb	bb
state:	p	$\vdash$	p $\vdash$	p
stack:	$\bot$		$A\bot$	$AA\bot$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$

remaining input:	$aabb$		abb		bb		b
state:	p	$\vdash$	p	$\vdash$	p	$\vdash$	q
stack:	$\bot$		$A\bot$		$AA\bot$		$A\bot$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$

remaining input:	$aabb$		abb		bb	b
state:	p	$\vdash$	p	$\vdash$	p	$\vdash$ q
stack:	$\bot$		$A\bot$		$AA\bot$	$A\bot$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\perp$ (bottom symbol)

transitions: $p, a, \perp \to p, A\perp$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \perp \to f, \varepsilon$

remaining input:	$aabb$		abb		bb		b		ε
state:	p	$\vdash$	p	$\vdash$	p	$\vdash$	q	$\vdash$	q
stack:	$\perp$		$A\perp$		$AA\perp$		$A\perp$		$\perp$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\perp$ (bottom symbol)

transitions: $p, a, \perp \to p, A\perp$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \perp \to f, \varepsilon$

remaining input:	$aabb$		abb		bb		b		ε
state:	p	$\vdash$	p	$\vdash$	p	$\vdash$	q	$\vdash$	q
stack:	$\perp$		$A\perp$		$AA\perp$		$A\perp$		$\perp$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\perp$ (bottom symbol)

transitions: $p, a, \perp \to p, A\perp$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \perp \to f, \varepsilon$

remaining input:	$aabb$		abb		bb		b		ε		ε
state:	p	$\vdash$	p	$\vdash$	p	$\vdash$	q	$\vdash$	q	$\vdash$	f
stack:	$\perp$		$A\perp$		$AA\perp$		$A\perp$		$\perp$		ε

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$

remaining input:	$aabb$		abb		bb		b		ε		ε
state:	p	$\vdash$	p	$\vdash$	p	$\vdash$	q	$\vdash$	q	$\vdash$	f
stack:	$\bot$		$A\bot$		$AA\bot$		$A\bot$		$\bot$		ε

accept

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\perp$ (bottom symbol)

transitions: $p, a, \perp \to p, A\perp$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \perp \to f, \varepsilon$

remaining input:	$aabb$		abb		bb		b		ε		ε
state:	p	$\vdash$	p	$\vdash$	p	$\vdash$	q	$\vdash$	q	$\vdash$	f
stack:	$\perp$		$A\perp$		$AA\perp$		$A\perp$		$\perp$		ε

accept

$$\mathscr{L}(\mathcal{C}) = \{w \in A^* \mid \begin{pmatrix} w \\ q_0 \\ \perp \end{pmatrix} \vdash^* \begin{pmatrix} \varepsilon \\ f \\ \gamma \end{pmatrix} \text{ for } f \in F, \gamma \in \Gamma^*\}$$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$

remaining input:	$aabb$		abb		bb		b		ε		ε
state:	p	$\vdash$	p	$\vdash$	p	$\vdash$	q	$\vdash$	q	$\vdash$	f
stack:	$\bot$		$A\bot$		$AA\bot$		$A\bot$		$\bot$		ε

accept

$$\mathscr{L}(\mathcal{C}) = \{w \in A^* \mid \begin{pmatrix} w \\ q_0 \\ \bot \end{pmatrix} \vdash^* \begin{pmatrix} \varepsilon \\ f \\ \gamma \end{pmatrix} \text{ for } f \in F, \gamma \in \Gamma^*\}$$

$$= \{a^n b^n \mid n \geq 1\}$$

Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$

remaining input:	$aabb$		abb		bb		b		ε		ε
state:	p	$\vdash$	p	$\vdash$	p	$\vdash$	q	$\vdash$	q	$\vdash$	f
stack:	$\bot$		$A\bot$		$AA\bot$		$A\bot$		$\bot$		ε

accept

$$\mathscr{L}(\mathcal{C}) = \{w \in A^* \mid \begin{pmatrix} w \\ q_0 \\ \bot \end{pmatrix} \vdash^* \begin{pmatrix} \varepsilon \\ f \\ \gamma \end{pmatrix} \text{ for } f \in F, \gamma \in \Gamma^*\}$$

$$= \{a^n b^n \mid n \geq 1\}$$

$\rightsquigarrow$ Configuration graph $\Gamma(\mathcal{C})$

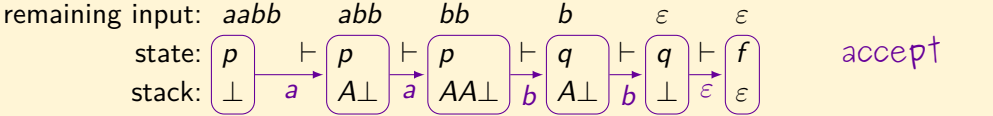
Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$



$$\mathscr{L}(\mathcal{C}) = \{w \in A^* \mid \begin{pmatrix} w \\ q_0 \\ \bot \end{pmatrix} \vdash^* \begin{pmatrix} \varepsilon \\ f \\ \gamma \end{pmatrix} \text{ for } f \in F, \gamma \in \Gamma^*\}$$

$$= \{a^n b^n \mid n \geq 1\}$$

$\rightsquigarrow$ Configuration graph $\Gamma(\mathcal{C})$

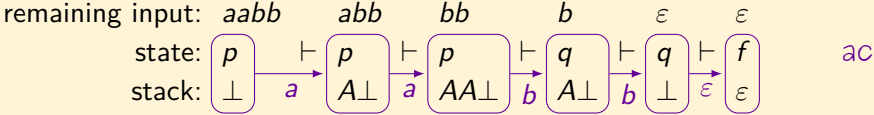
Example PDA

Example

states: p (initial), q, f (final) **alphabet**: $\{a, b\}$ **stack symbols**: A, $\bot$ (bottom symbol)

transitions: $p, a, \bot \to p, A\bot$ $p, a, A \to p, AA$ $p, b, A \to q, \varepsilon$

$q, b, A \to q, \varepsilon$ $q, \varepsilon, \bot \to f, \varepsilon$



$$\mathscr{L}(\mathcal{C}) = \{w \in A^* \mid \begin{pmatrix} w \\ q_0 \\ \bot \end{pmatrix} \vdash^* \begin{pmatrix} \varepsilon \\ f \\ \gamma \end{pmatrix} \text{ for } f \in F, \gamma \in \Gamma^*\}$$

$$= \{a^n b^n \mid n \geq 1\}$$

$\rightsquigarrow$ Configuration graph $\Gamma(\mathcal{C})$ with edges $\{(p, A\gamma') \xrightarrow{a_0} (q, \gamma\gamma') \mid (p, a_0, A \to q, \gamma) \in \delta\}$

Context-Free Graphs

Γ: inverse $A^{\pm *}$-graph Then:

Γ is context-free

$\iff$ Γ is the configuration graph of some PDA (Muller, Schupp, 1990)

Context-Free Graphs

Γ: inverse $A^{\pm *}$-graph Then:

Γ is context-free $\iff$ Γ has finitely many end-isomorphism classes (definition)

$\iff$ Γ is the configuration graph of some PDA (Muller, Schupp, 1990)

Context-Free Graphs

Γ: inverse $A^{\pm *}$-graph with root u Then:

Γ is context-free $\iff$ Γ has finitely many end-isomorphism classes (definition)

$\iff$ Γ is the configuration graph of some PDA (Muller, Schupp, 1990)

$\iff$ $\mathscr{L}(\Gamma; u, u)$ is deterministic context-free (Rodaro, 2024)

Context-Free Graphs

Some details on ε-transitions and $\perp$ omitted

Γ: inverse $A^{\pm *}$-graph with root u Then:

Γ is context-free $\iff$ Γ has finitely many end-isomorphism classes (definition)

$\iff$ Γ is the configuration graph of some PDA (Muller, Schupp, 1990)

$\iff$ $\mathscr{L}(\Gamma; u, u)$ is deterministic context-free (Rodaro, 2024)

Context-Free Graphs

Some details on ε-transitions and $\perp$ omitted

Γ: inverse $A^{\pm *}$-graph with root u Then:

$$
\begin{aligned}
\Gamma \text{ is context-free} &\iff \Gamma \text{ has finitely many end-isomorphism classes} && \text{(definition)} \\
&\iff \Gamma \text{ is the configuration graph of some PDA} && \text{(Muller, Schupp, 1990)} \\
&\iff \mathscr{L}(\Gamma; u, u) \text{ is deterministic context-free} && \text{(Rodaro, 2024)} \\
&\implies \Gamma \text{ is tree-like}
\end{aligned}
$$

Context-Free Graphs

Some details on ε-transitions and $\perp$ omitted

Γ: inverse $A^{\pm *}$-graph with root u Then:

Γ is context-free $\iff$ Γ has finitely many end-isomorphism classes (definition)

$\iff$ Γ is the configuration graph of some PDA (Muller, Schupp, 1990)

$\iff$ $\mathscr{L}(\Gamma; u, u)$ is deterministic context-free (Rodaro, 2024)

$\implies$ Γ is tree-like

"$\impliedby$" if Schützenberger graph of a f.p. inverse semigroup (Gray, Silva, Szakács)

Context-Free Graphs

Some details on ε-transitions and $\perp$ omitted

Γ: inverse $A^{\pm *}$-graph with root u Then:

$$
\begin{aligned}
\Gamma \text{ is context-free} &\iff \Gamma \text{ has finitely many end-isomorphism classes} && \text{(definition)} \\
&\iff \Gamma \text{ is the configuration graph of some PDA} && \text{(Muller, Schupp, 1990)} \\
&\iff \mathscr{L}(\Gamma; u, u) \text{ is deterministic context-free} && \text{(Rodaro, 2024)} \\
&\implies \Gamma \text{ is tree-like} \\
\text{“}\impliedby\text{”} &\text{ if Schützenberger graph of a f.p. inverse semigroup (Gray, Silva, Szakács)}
\end{aligned}
$$

Now: We may use deterministic PDAs to describe Schützenberger graphs.

Context-Free Graphs

Some details on ε-transitions and $\perp$ omitted

Γ: inverse $A^{\pm *}$-graph with root u Then:

$$
\begin{aligned}
\Gamma \text{ is context-free} &\iff \Gamma \text{ has finitely many end-isomorphism classes} && \text{(definition)} \\
&\iff \Gamma \text{ is the configuration graph of some PDA} && \text{(Muller, Schupp, 1990)} \\
&\iff \mathscr{L}(\Gamma; u, u) \text{ is deterministic context-free} && \text{(Rodaro, 2024)} \\
&\implies \Gamma \text{ is tree-like} \\
&\text{“}\impliedby\text{”} \text{ if Schützenberger graph of a f.p. inverse semigroup (Gray, Silva, Szakács)}
\end{aligned}
$$

Now: We may use deterministic PDAs to describe Schützenberger graphs.

But: PDAs are algorithmically not “nice”.

Context-Free Graphs

Some details on ε-transitions and $\perp$ omitted

Γ: inverse $A^{\pm *}$-graph with root u Then:

$$
\begin{aligned}
\Gamma \text{ is context-free} &\iff \Gamma \text{ has finitely many end-isomorphism classes} && \text{(definition)} \\
&\iff \Gamma \text{ is the configuration graph of some PDA} && \text{(Muller, Schupp, 1990)} \\
&\iff \mathscr{L}(\Gamma; u, u) \text{ is deterministic context-free} && \text{(Rodaro, 2024)} \\
&\implies \Gamma \text{ is tree-like} \\
&\text{“}\impliedby\text{”} \text{ if Schützenberger graph of a f.p. inverse semigroup (Gray, Silva, Szakács)}
\end{aligned}
$$

Now: We may use deterministic PDAs to describe Schützenberger graphs.

But: PDAs are algorithmically not “nice”. Can we do better?

Context-Free Graphs

Some details on ε-transitions and $\perp$ omitted

Γ: inverse $A^{\pm *}$-graph with root u Then:

$$
\begin{aligned}
\Gamma \text{ is context-free} &\iff \Gamma \text{ has finitely many end-isomorphism classes} && \text{(definition)}\\
&\iff \Gamma \text{ is the configuration graph of some PDA} && \text{(Muller, Schupp, 1990)}\\
&\iff \mathscr{L}(\Gamma; u, u) \text{ is deterministic context-free} && \text{(Rodaro, 2024)}\\
&\implies \Gamma \text{ is tree-like}\\
&\text{“}\impliedby\text{”} \text{ if Schützenberger graph of a f.p. inverse semigroup (Gray, Silva, Szakács)}
\end{aligned}
$$

Now: We may use deterministic PDAs to describe Schützenberger graphs.

But: PDAs are algorithmically not “nice”. Can we do better? Probably…

Finite Automata Generating Trees

Finite Automata Generating Trees

Definition (pDFA)

A pDFA is a finite, deterministic $A^{\pm 1}$-graph

Finite Automata Generating Trees

Definition (pDFA)

A pDFA is a finite, deterministic $A^{\pm 1}$-graph whose nodes we call states

Finite Automata Generating Trees

Definition (pDFA)

A pDFA is a finite, deterministic $A^{\pm 1}$-graph whose nodes we call states and whose edges we call transitions.

Finite Automata Generating Trees

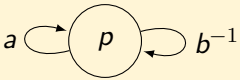
Definition (pDFA)

A pDFA is a finite, deterministic $A^{\pm 1}$-graph whose
nodes we call states and whose edges we call transitions.

Every state p of a pDFA generates an involutive tree via its tree of runs/path space.

Example

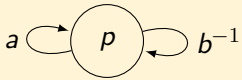
Example



Example

Example

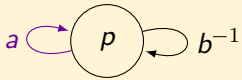
$p\ \varepsilon$ ← This is the root!



Example

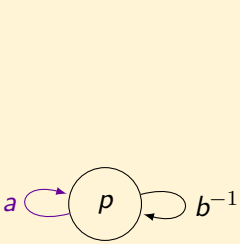
Example

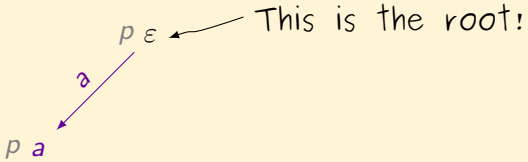
$p\ \varepsilon$ ← This is the root!



Example

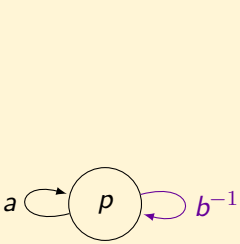
Example

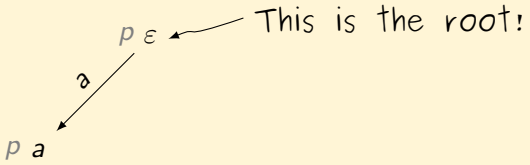




Example

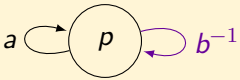
Example

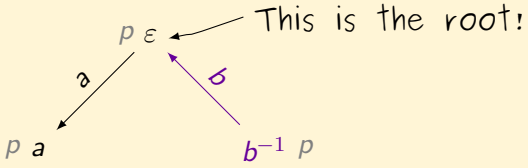




Example

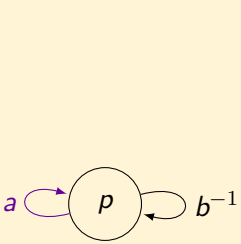
Example

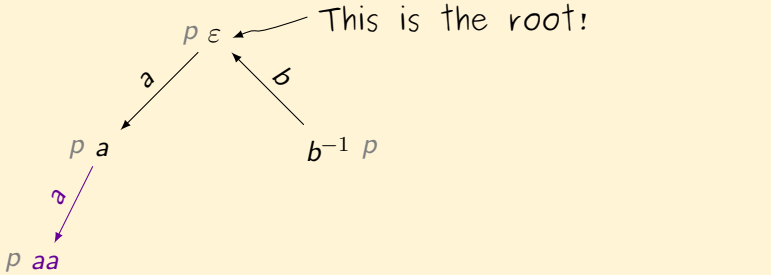




Example

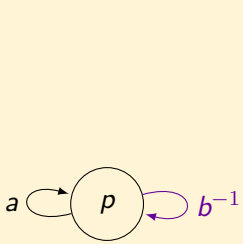
Example

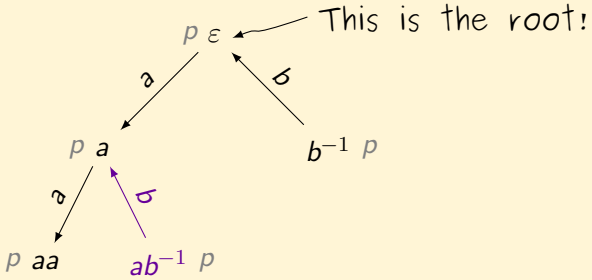




Example

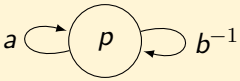
Example

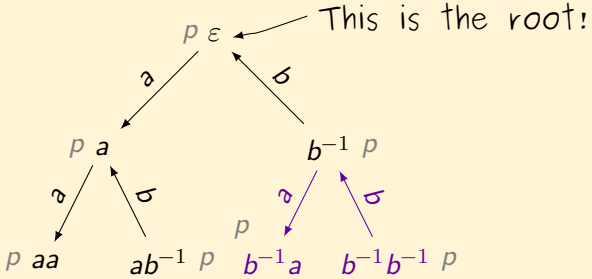




Example

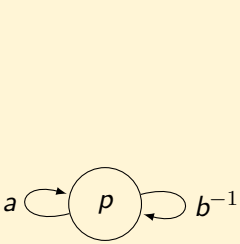
Example

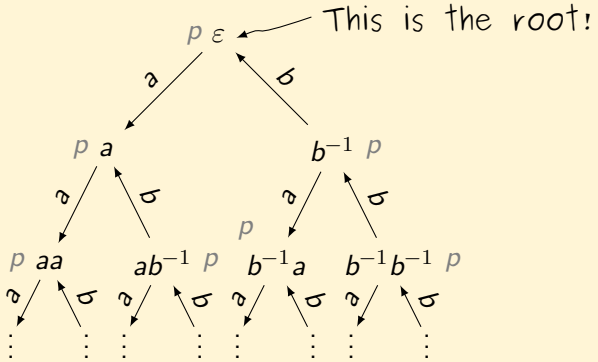




Example

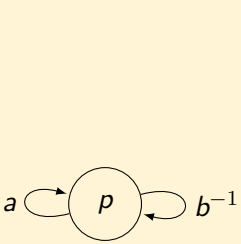
Example

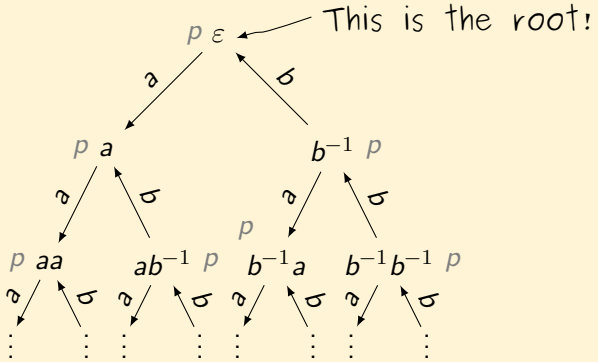




Example

Example

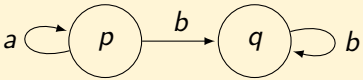




$\Gamma(p)$

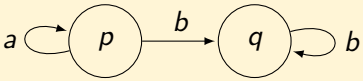
Example

Example



Example

Example

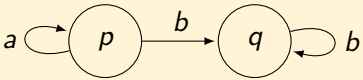


$\Gamma(p)$

Example

Example

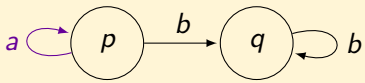
p
ε

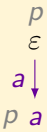


$\Gamma(p)$

Example

Example

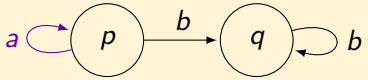


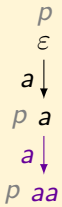


$\Gamma(p)$

Example

Example

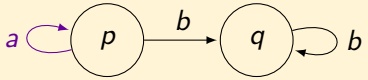


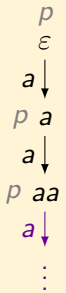


$\Gamma(p)$

Example

Example

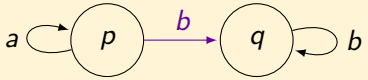


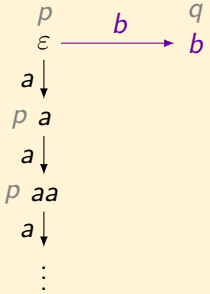


$\Gamma(p)$

Example

Example

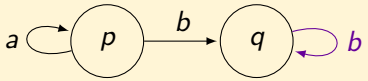


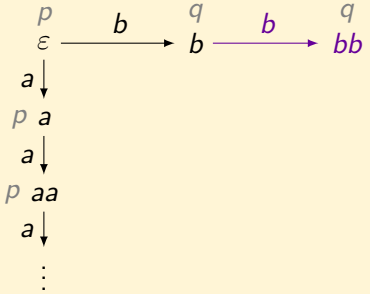


$\Gamma(p)$

Example

Example

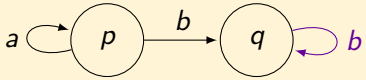


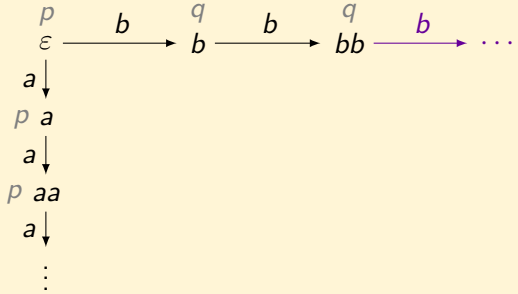


$\Gamma(p)$

Example

Example

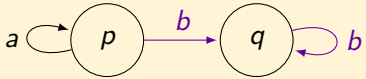


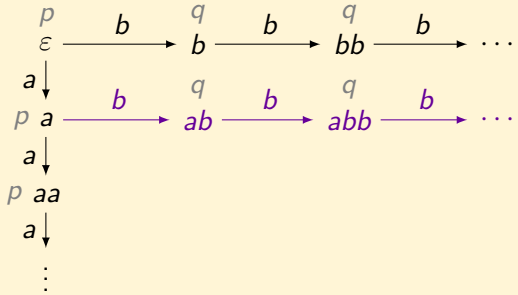


$\Gamma(p)$

Example

Example

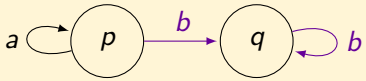


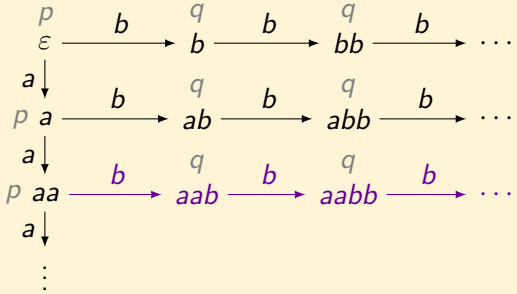


$\Gamma(p)$

Example

Example

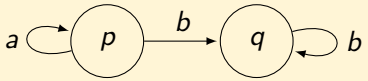


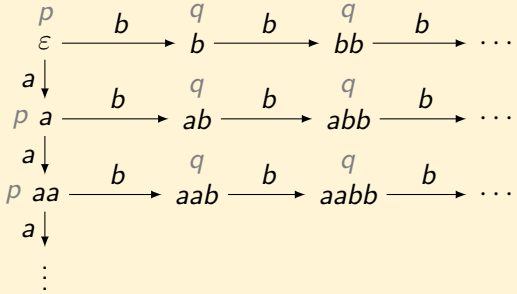


$\Gamma(p)$

Example

Example

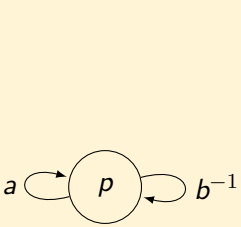


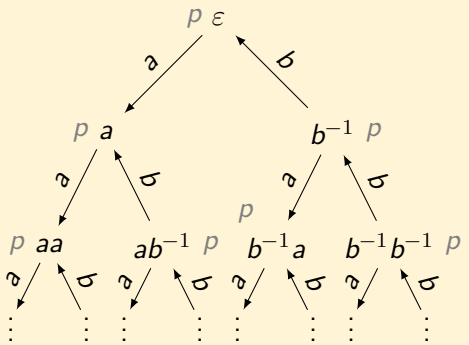


$\Gamma(p)$

Example

Example

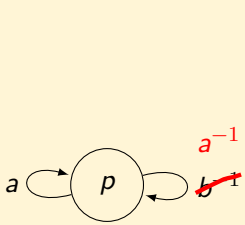


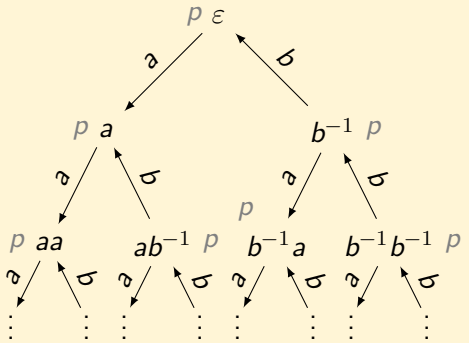


$\Gamma(p)$

Example

Example

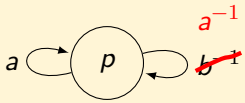


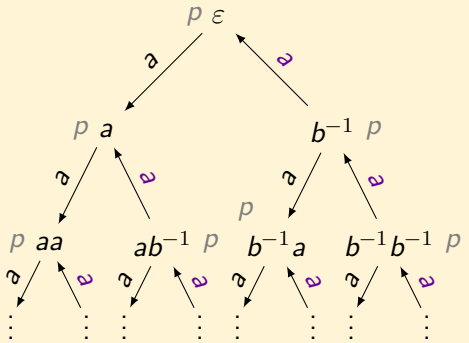


$\Gamma(p)$

Example

Example

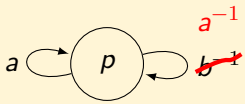


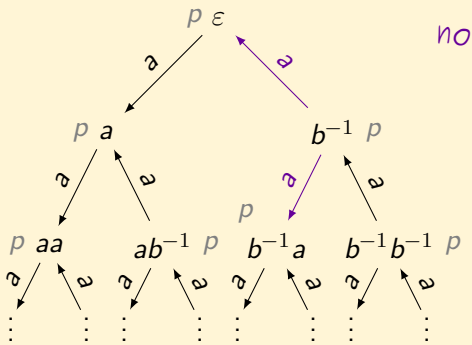


$\Gamma(p)$

Example

Example



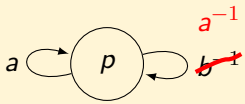


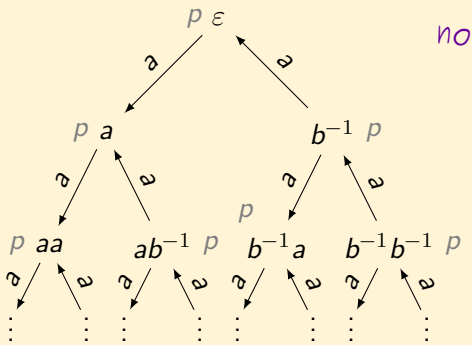
not deterministic!

$\Gamma(p)$

Example

Example





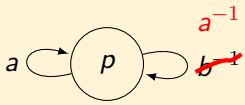
not deterministic!

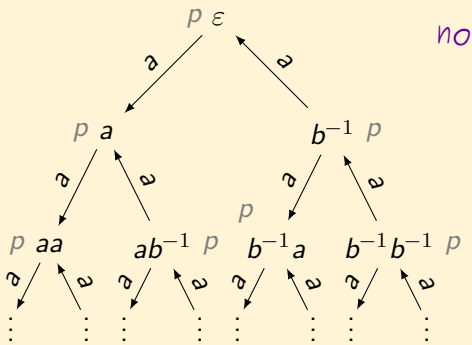
$\Gamma(p)$

$\Gamma(p)$ is inverse and deterministic $\iff$ aa^{-1} cannot be read for any $a \in A^{\pm 1}$

Example

Example





not deterministic!

$\Gamma(p)$

$\Gamma(p)$ is inverse and deterministic $\iff$ aa^{-1} cannot be read for any $a \in A^{\pm 1}$

$\iff$ the pDFA is reduced

Context-Free Trees

Γ: inverse $A^{\pm 1}$-tree, i. e. a tree as an unlabeled, undirected graph

Context-Free Trees

Γ: inverse $A^{\pm 1}$-tree, i. e. a tree as an unlabeled, undirected graph

Theorem (W., arXiv 2026)

Γ *is context-free*

Context-Free Trees

Γ: inverse $A^{\pm 1}$-tree, i. e. a tree as an unlabeled, undirected graph

Theorem (W., arXiv 2026)

Γ *is context-free* $\iff$ Γ *arises as* $\Gamma(p)$ *for a state* p *of a reduced pDFA*

Context-Free Trees

Γ: inverse $A^{\pm 1}$-tree, i. e. a tree as an unlabeled, undirected graph

Theorem (W., arXiv 2026)

Γ *is context-free* $\iff$ Γ *arises as* $\Gamma(p)$ *for a state* p *of a reduced pDFA*

Context-free trees

- arise naturally as Schützenberger graphs of

Context-Free Trees

Γ: inverse $A^{\pm 1}$-tree, i. e. a tree as an unlabeled, undirected graph

Theorem (W., arXiv 2026)

Γ *is context-free* $\iff$ Γ *arises as* $\Gamma(p)$ *for a state* p *of a reduced pDFA*

Context-free trees

- arise naturally as Schützenberger graphs of
  - free inverse monoids (Munn trees).

Context-Free Trees

Γ: inverse $A^{\pm 1}$-tree, i. e. a tree as an unlabeled, undirected graph

Theorem (W., arXiv 2026)

Γ *is context-free* $\iff$ Γ *arises as* $\Gamma(p)$ *for a state* p *of a reduced pDFA*

Context-free trees

- arise naturally as Schützenberger graphs of
  - free inverse monoids (Munn trees).
  - inverse monoids with idempotent relators.

Context-Free Trees

Γ: inverse $A^{\pm 1}$-tree, i. e. a tree as an unlabeled, undirected graph

Theorem (W., arXiv 2026)

Γ *is context-free* $\iff$ Γ *arises as* $\Gamma(p)$ *for a state* p *of a reduced pDFA*

Context-free trees

- arise naturally as Schützenberger graphs of
  - free inverse monoids (Munn trees).
  - inverse monoids with idempotent relators. $\rightsquigarrow$ more on next slide

Context-Free Trees

Γ: inverse $A^{\pm 1}$-tree, i. e. a tree as an unlabeled, undirected graph

Theorem (W., arXiv 2026)

Γ *is context-free* $\iff$ Γ *arises as* $\Gamma(p)$ *for a state* p *of a reduced pDFA*

Context-free trees

- arise naturally as Schützenberger graphs of
  - free inverse monoids (Munn trees).
  - inverse monoids with idempotent relators. $\rightsquigarrow$ more on next slide
- hopefully serve as a stepping stone for general context-free graphs.

Inverse Monoids with Idempotent Relators

Inverse Monoids with Idempotent Relators

Lemma (Margolis, Meakin, 1993)

$M = \mathrm{Inv}\langle A \mid \ell_i = r_i, i \in I \rangle$ *for* $\ell_i = 1 = r_i$ *in* $F(A)$

Inverse Monoids with Idempotent Relators

Lemma (Margolis, Meakin, 1993)

$M = \mathrm{Inv}\langle A \mid \ell_i = r_i, i \in I \rangle$ *for* $\ell_i = 1 = r_i$ *in* $F(A)$

$\iff$ *all Schützenberger graphs of* M *are trees*

Inverse Monoids with Idempotent Relators

Lemma (Margolis, Meakin, 1993)

$M = \mathrm{Inv}\langle A \mid \ell_i = r_i, i \in I \rangle$ *for* $\ell_i = 1 = r_i$ *in* $F(A)$

$\iff$ *all Schützenberger graphs of M are trees and, thus, context-free if* $|I| < \infty$

Inverse Monoids with Idempotent Relators

Lemma (Margolis, Meakin, 1993)

$M = \mathrm{Inv}\langle A \mid \ell_i = r_i, i \in I \rangle$ *for* $\ell_i = 1 = r_i$ *in* $F(A)$

$\iff$ *all Schützenberger graphs of* M *are trees and, thus, context-free if* $|I| < \infty$

Theorem (Margolis, Meakin, 1993)

The word problem

Constant: $M = \mathrm{Inv}\langle A \mid \ell_i = r_i \rangle$ *for* $\ell_i = 1 = r_i$ *in* $F(A)$
Input: $u, v \in A^{\pm *}$
Question: *is* $u = v$ *in* M*?*

is decidable.

Inverse Monoids with Idempotent Relators

Lemma (Margolis, Meakin, 1993)

$M = \mathrm{Inv}\langle A \mid \ell_i = r_i, i \in I \rangle$ *for* $\ell_i = 1 = r_i$ *in* $F(A)$

$\iff$ *all Schützenberger graphs of* M *are trees and, thus, context-free if* $|I| < \infty$

Theorem (Margolis, Meakin, 1993)

The word problem

Constant: $M = \mathrm{Inv}\langle A \mid \ell_i = r_i \rangle$ *for* $\ell_i = 1 = r_i$ *in* $F(A)$
Input: $u, v \in A^{\pm *}$
Question: *is* $u = v$ *in* M*?*

is decidable.

This solves $\mathcal{R}$ and $\mathcal{L}$.

Inverse Monoids with Idempotent Relators

Lemma (Margolis, Meakin, 1993)

$M = \mathrm{Inv}\langle A \mid \ell_i = r_i, i \in I \rangle$ *for* $\ell_i = 1 = r_i$ *in* $F(A)$

$\iff$ *all Schützenberger graphs of* M *are trees* and, thus, context-free *if* $|I| < \infty$

Theorem (Margolis, Meakin, 1993)

The word problem

Constant: $M = \mathrm{Inv}\langle A \mid \ell_i = r_i \rangle$ *for* $\ell_i = 1 = r_i$ *in* $F(A)$
Input: $u, v \in A^{\pm *}$
Question: *is* $u = v$ *in* M*?*

is decidable.

This solves $\mathcal{R}$ and $\mathcal{L}$. What about $\mathcal{D}$ and $\mathcal{J}$?

Inverse Monoids with Idempotent Relators

Lemma (Margolis, Meakin, 1993)

$M = \mathrm{Inv}\langle A \mid \ell_i = r_i, i \in I \rangle$ *for* $\ell_i = 1 = r_i$ *in* $F(A)$

$\iff$ *all Schützenberger graphs of* M *are trees* and, thus, context−free *if* $|I| < \infty$

Theorem (Margolis, Meakin, 1993)

The word problem

Constant: $M = \mathrm{Inv}\langle A \mid \ell_i = r_i \rangle$ *for* $\ell_i = 1 = r_i$ *in* $F(A)$
Input: $u, v \in A^{\pm *}$
Question: *is* $u = v$ *in* M*?*

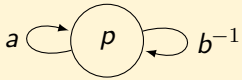
is decidable.

This solves $\mathcal{R}$ and $\mathcal{L}$. What about $\mathcal{D}$ and $\mathcal{J}$?
$\rightsquigarrow$ Let's look at the homomorphism/isomorphism problem of context−free trees!

The Rooted Isomorphism Problem

The Rooted Isomorphism Problem

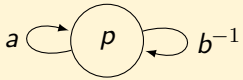
Example



The Rooted Isomorphism Problem

$\mathscr{L}(p)$: words readable from state p in its pDFA

Example

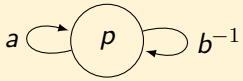


$\mathscr{L}(p) = \{a, b^{-1}\}^*$

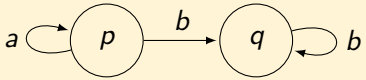
The Rooted Isomorphism Problem

$\mathscr{L}(p)$: words readable from state p in its pDFA

Example



$\mathscr{L}(p) = \{a, b^{-1}\}^*$

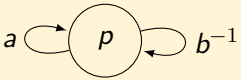


$\mathscr{L}(p) = a^* b^*$

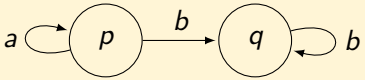
The Rooted Isomorphism Problem

$\mathscr{L}(p)$: words readable from state p in its pDFA

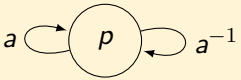
Example



$\mathscr{L}(p) = \{a, b^{-1}\}^*$



$\mathscr{L}(p) = a^* b^*$

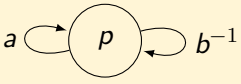


$\mathscr{L}(p) = \{a, a^{-1}\}^*$

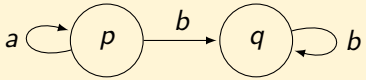
The Rooted Isomorphism Problem

$\mathscr{L}(p)$: words readable from state p in its pDFA

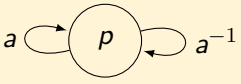
Example



$\mathscr{L}(p) = \{a, b^{-1}\}^*$



$\mathscr{L}(p) = a^* b^*$



$\mathscr{L}(p) = \{a, a^{-1}\}^*$

Fact

Let p and q be states of reduced pDFAs. Then:

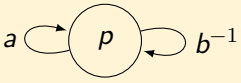
$$\Gamma(p) \cong \Gamma(q) \iff \mathscr{L}(p) = \mathscr{L}(q)$$

isomorphic as **rooted** *graphs*

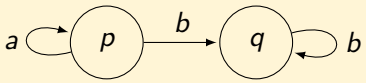
The Rooted Isomorphism Problem

$\mathscr{L}(p)$: words readable from state p in its pDFA

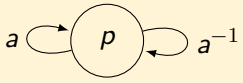
Example



$\mathscr{L}(p) = \{a, b^{-1}\}^*$



$\mathscr{L}(p) = a^* b^*$



$\mathscr{L}(p) = \{a, a^{-1}\}^*$

Fact

Let p and q be states of reduced pDFAs. Then:

$$\Gamma(p) \stackrel{\sim}{=} \Gamma(q) \iff \mathscr{L}(p) = \mathscr{L}(q)$$

isomorphic as **rooted** *graphs*

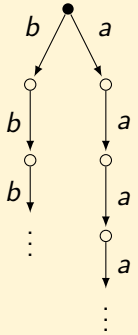
It is well known that the latter is in NL and, thus, polynomial time.

The Non-Rooted Case: The Problem

The Non-Rooted Case: The Problem

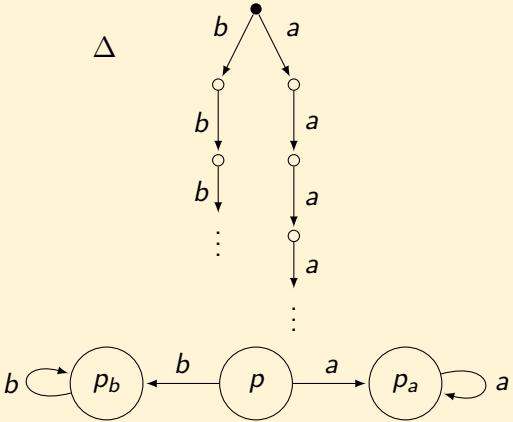
Example

Δ



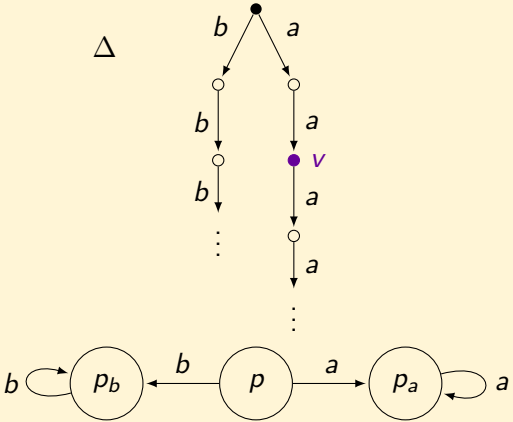
The Non-Rooted Case: The Problem

Example



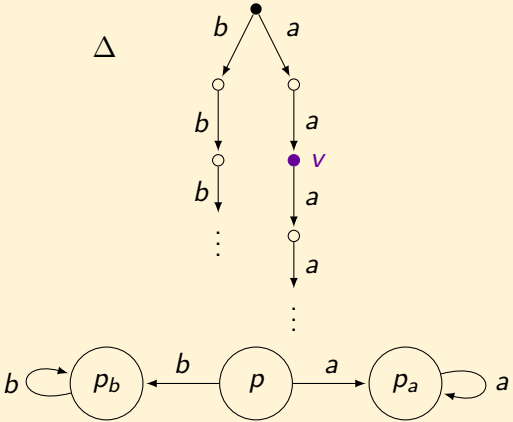
The Non-Rooted Case: The Problem

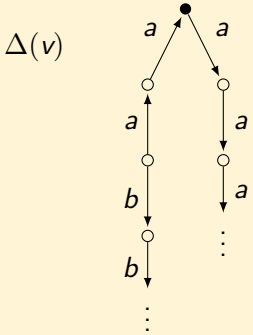
Example



The Non-Rooted Case: The Problem

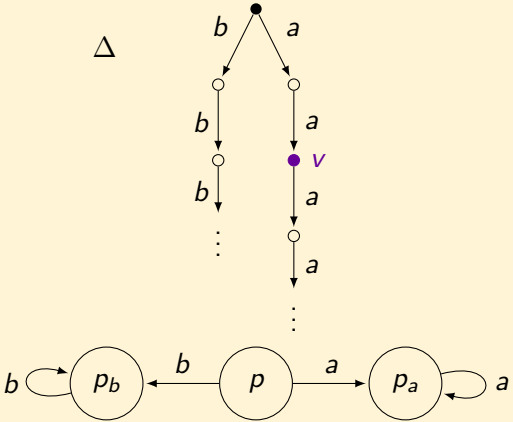
Example

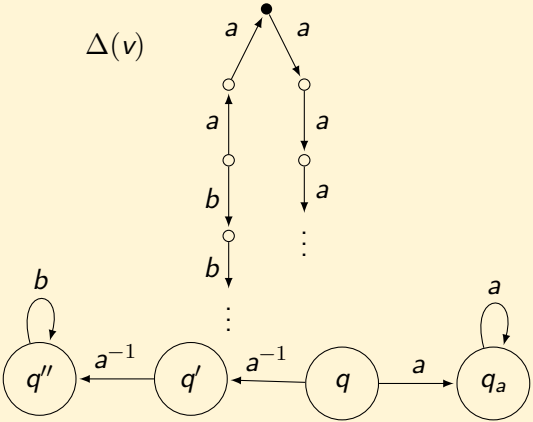




The Non-Rooted Case: The Problem

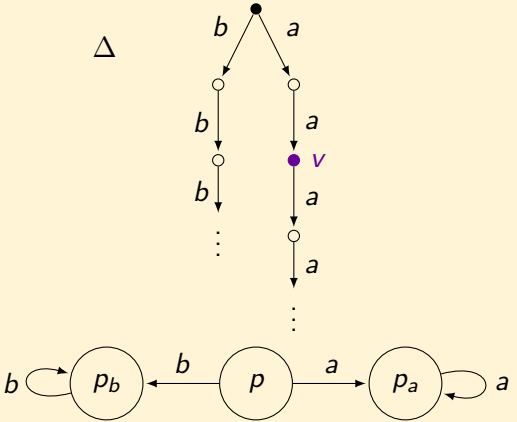
Example

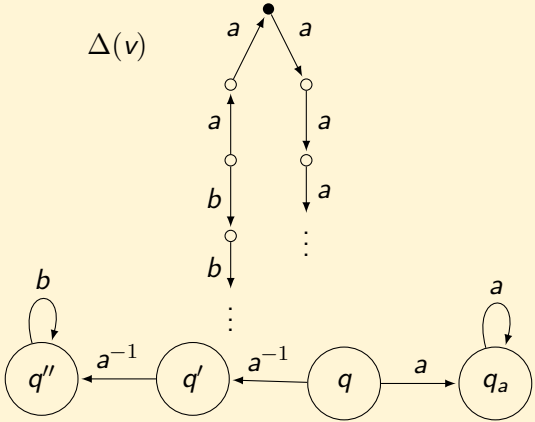




The Non-Rooted Case: The Problem

Example





The languages don't seem connected!

The Non-Rooted Isomorphism Problem

Theorem (W., arXiv 2026)

The non-rooted isomorphism problem for inverse context-free trees

Input: *states $\check{p}$ and $\check{q}$ of reduced pDFAs*

Question: *are $\Gamma(\check{p})$ and $\Gamma(\check{q})$ isomorphic as non-rooted graphs*

is in NL and, thus, in particular, decidable in deterministic polynomial time.

Outlook

Outlook

- We still need to compute a generating reduced pDFA for $S\Gamma(s)$ from s

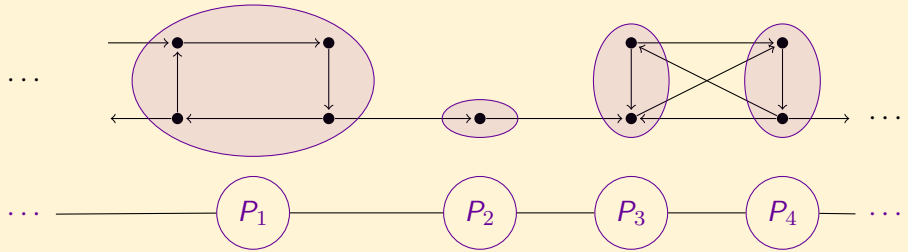
Outlook

- We still need to compute a generating reduced pDFA for $S\Gamma(s)$ from s and the presentation.

Outlook

- We still need to compute a generating reduced pDFA for $S\Gamma(s)$ from s and the presentation.
- General context-free graphs are tree-like:

Example



Thank you!