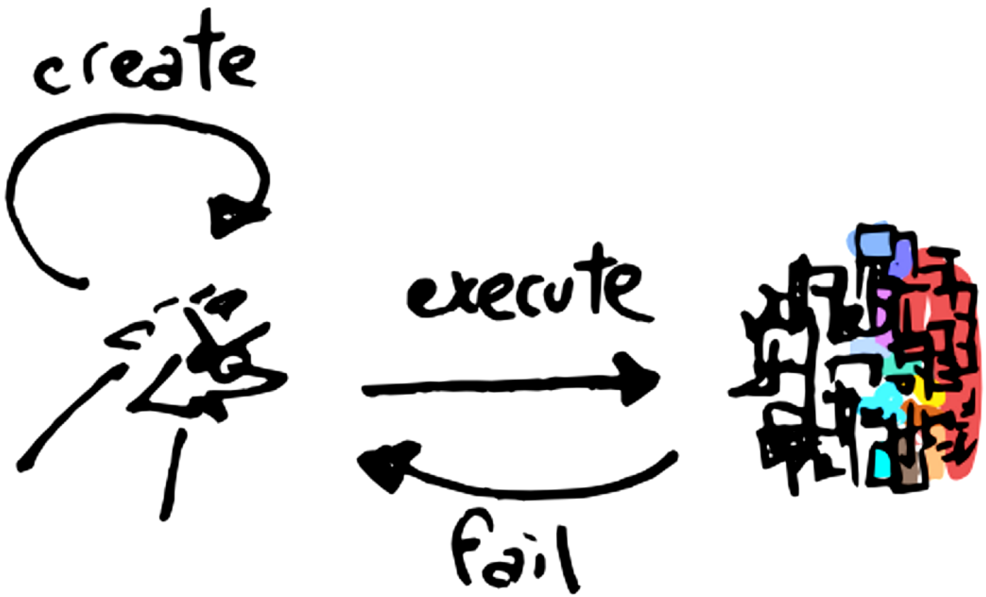
Verification of infinite state systems

Gabriele Puppis

LaBRI / CNRS



Outline of the course

1. **Warm-up**

   (transition systems, automata, logics)

2. **First-order theories**

   (undecidability, Presburger logic, automatic structures)

3. **The monadic theory of one successor**

   (contraction and composition methods, factorization forests)

4. **The monadic theory of two successors**

   (Rabin's complementation, application examples)

5. **The transformational approach**

   (interpretations, context-free and prefix-rewriting graphs, unfoldings, Caucal hierarchy, recursive program schemes)

6. **Reachability via saturation**

   (pushdown systems, VAS / Petri nets, lossy counter machines)

Goal

Automatic verification of **properties** of **systems**.

Goal

Automatic verification of **properties** of **systems**.

- which properties?

safety	*"something bad never happens"*	reachability
liveness	*"something good eventually happens"*	reachability
fairness	*"if something happens infinitely often then something else eventually happens"*	
formulas	" $\forall t.\ \exists t'.\ t \leq t' \wedge a(t')$ "	

Goal

Automatic verification of **properties** of **systems**.

- which properties?

safety	*"something bad never happens"*	reachability
liveness	*"something good eventually happens"*	reachability
fairness	*"if something happens infinitely often then something else eventually happens"*	
formulas	" $\forall t.\ \exists t'.\ t \leq t' \wedge a(t')$ "	

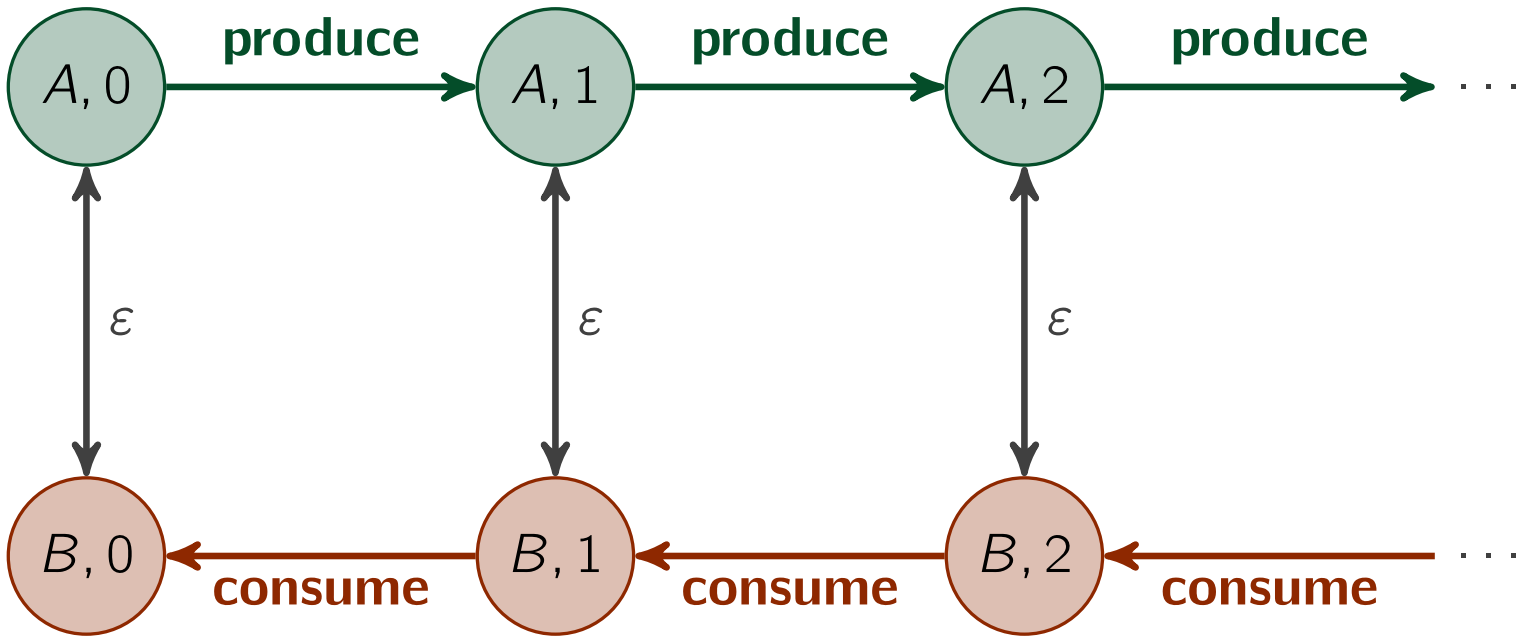
- which systems?

reactive	*"transitions enabled on the basis of input"*
infinite	*stacks* (or *recursion*) *variables* *queues* *lists*

A : repeat forever
do atomically
produce
$count := count + 1$

$\|$

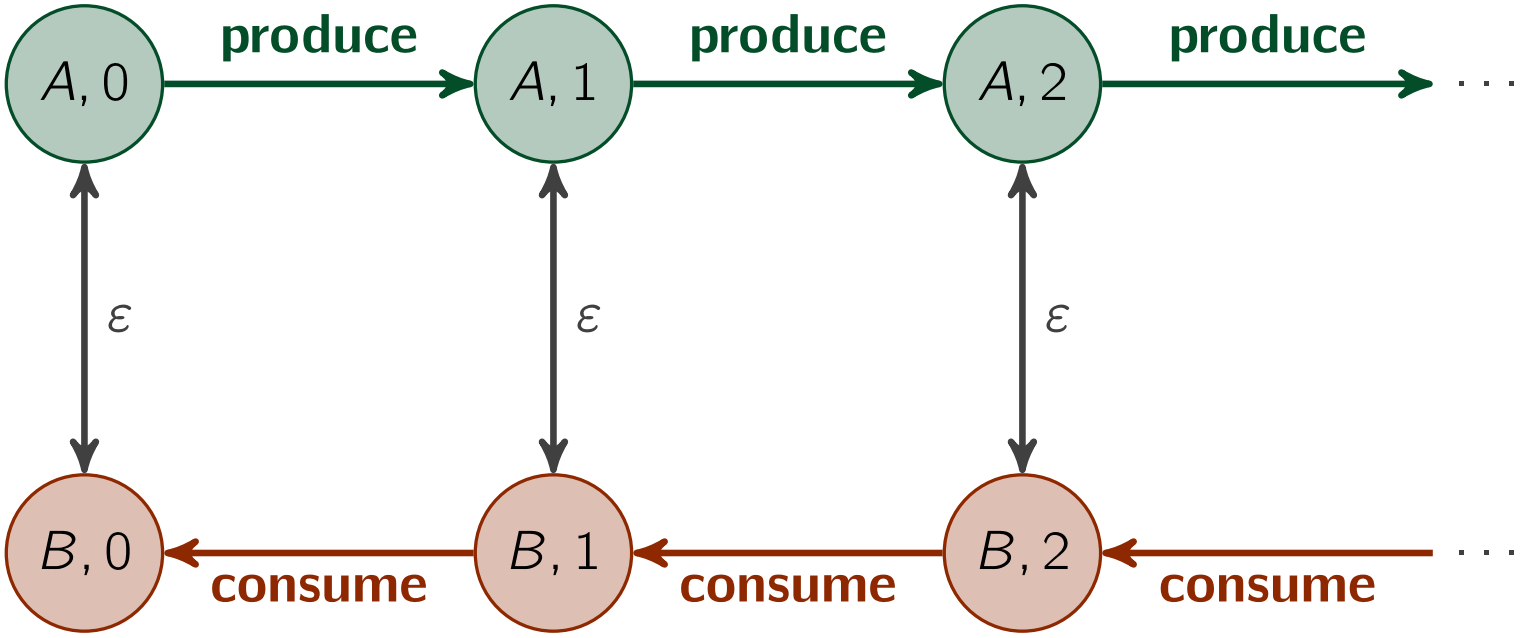
B : repeat forever
do atomically
if $count > 0$ then
consume
$count := count - 1$

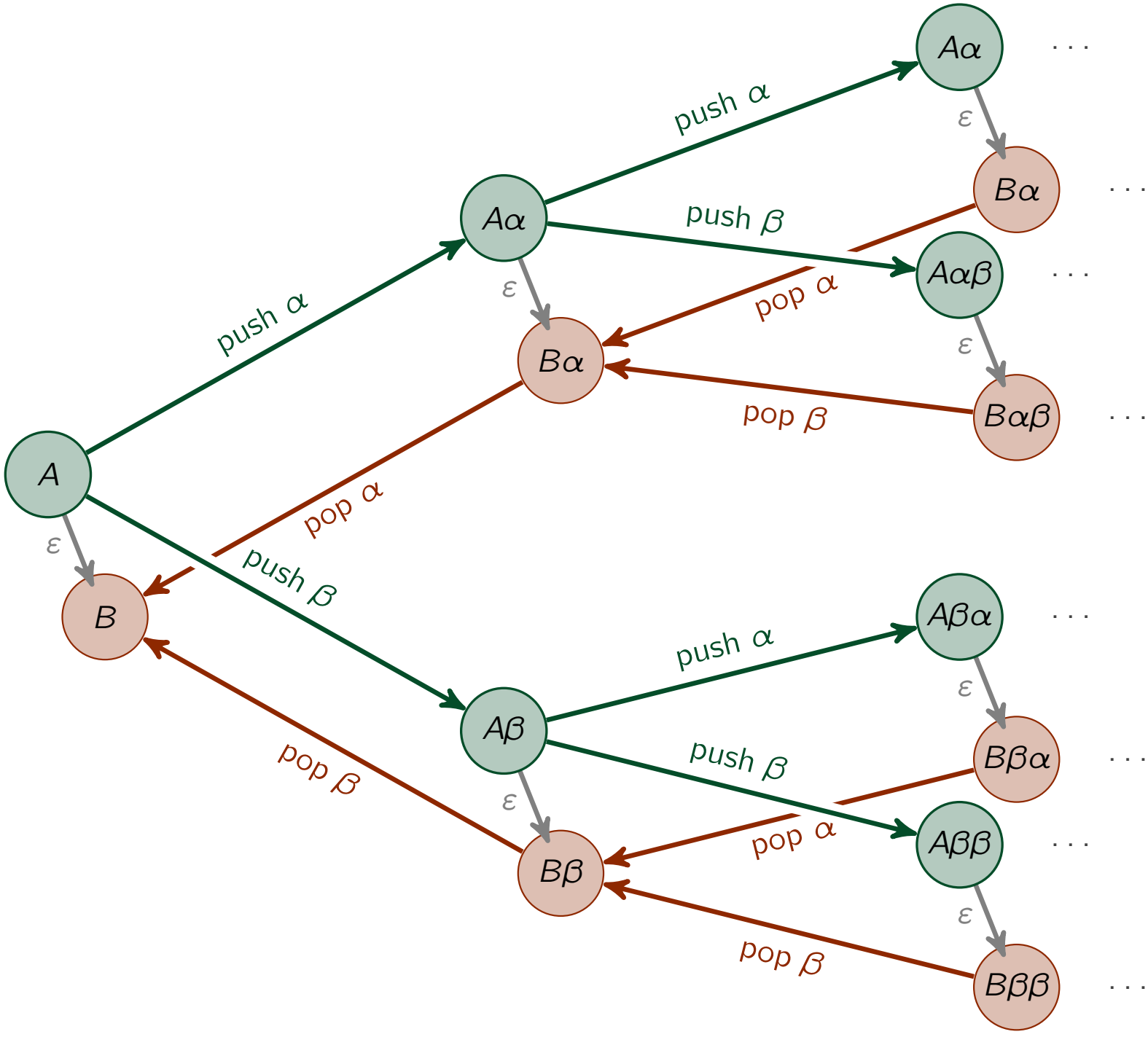


Definition

A **transition system** is a graph $G = \left((V_a)_{a \in \Sigma}, (E_b)_{b \in \Delta}\right)$ where

- vertices are associated with labels from a finite alphabet Σ
- edges are associated with labels from a finite alphabet Δ




A
B
Aα
Bα
Aαβ
Bαβ
Aβ
Bβ
Aβα
Bβα
Aββ
Bββ
push α
pop α
push β
pop β
ε
...

Automata = finite transition systems

 but mostly used as representations of **languages**

Automata = finite transition systems

 but mostly used as representations of **languages**

Definition

A **finite state automaton** is a tuple $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$, where

- Q is a finite set of control states
- Σ is a finite alphabet for transition labels
- $\Delta \subseteq Q \times \Sigma \times Q$ is a finite set of transition rules
- $I \subseteq Q$ is a set of initial states
- $F \subseteq Q$ is a set of final states

Automata = finite transition systems

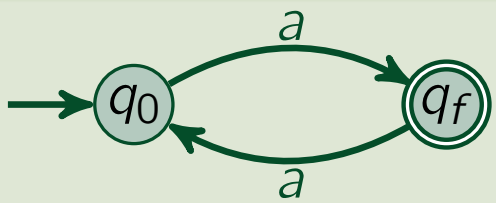
☞ but mostly used as representations of **languages**

Definition

A **finite state automaton** is a tuple $\mathcal{A} = (Q, \Sigma, \Delta, I, F)$, where

- Q is a finite set of control states
- Σ is a finite alphabet for transition labels
- $\Delta \subseteq Q \times \Sigma \times Q$ is a finite set of transition rules
- $I \subseteq Q$ is a set of initial states
- $F \subseteq Q$ is a set of final states

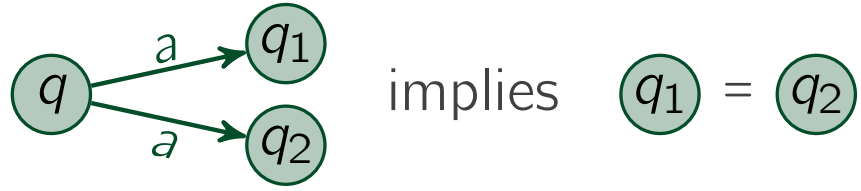
Example



$$\mathscr{L}(\mathcal{A}) = a\,(a\,a)^\star$$

Different types of automata:

- **deterministic**

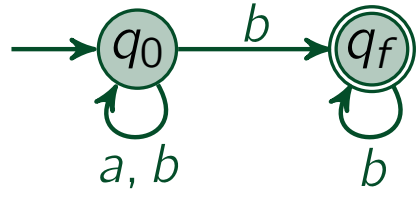


- **with ε-transitions**

- **complete**

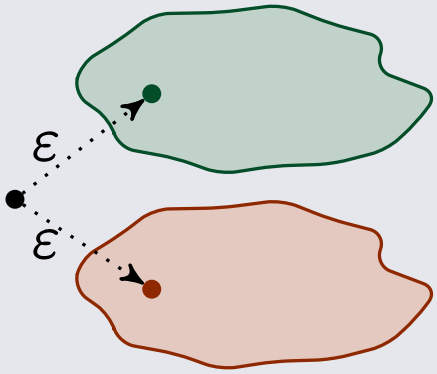
$\forall a.\ \forall q.\ \exists q'.$

- **Büchi/parity conditions**

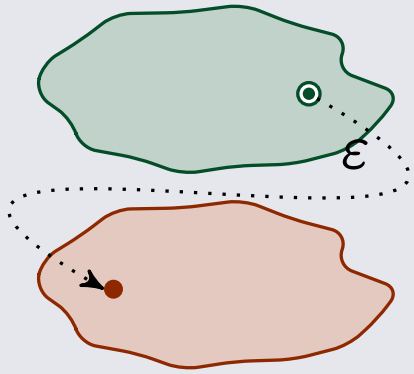


Language theoretic operations on automata

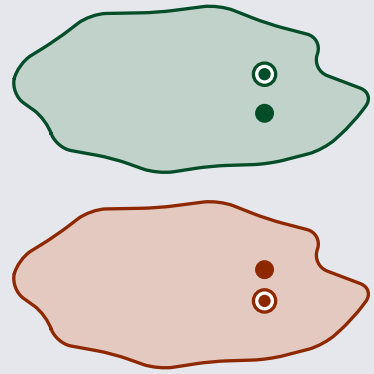
union



concatenation

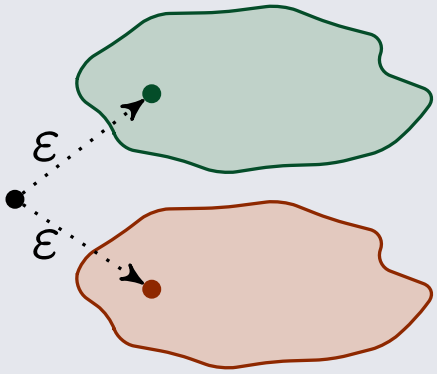


complementation

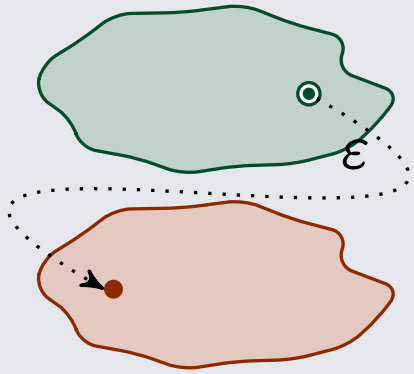


Language theoretic operations on automata

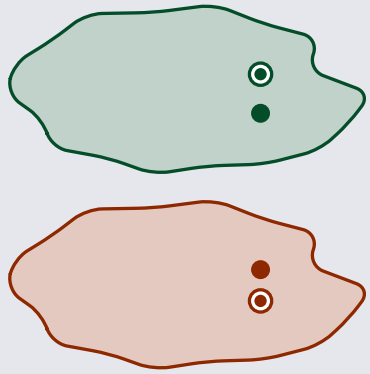
union



concatenation

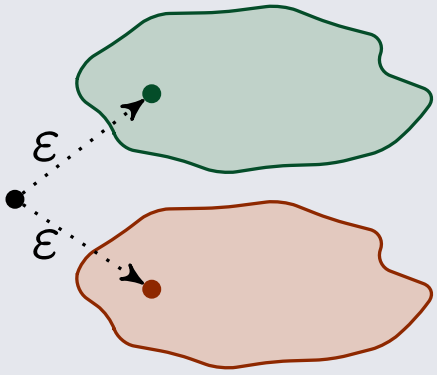


complementation

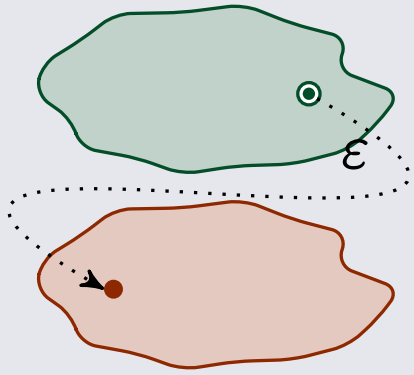


Language theoretic operations on automata

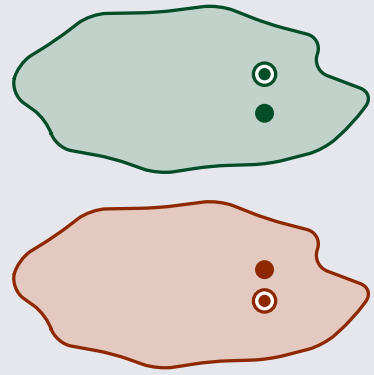
union



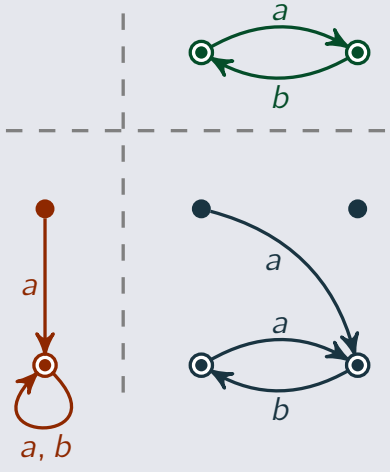
concatenation



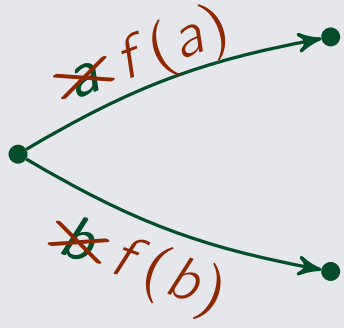
complementation



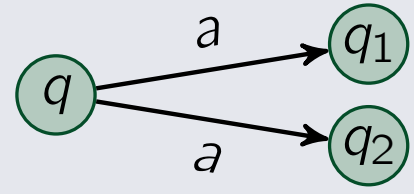
intersection

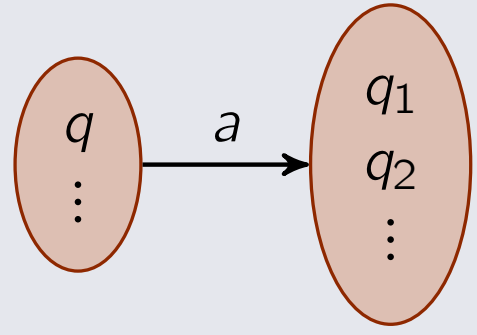


projection



subset construction

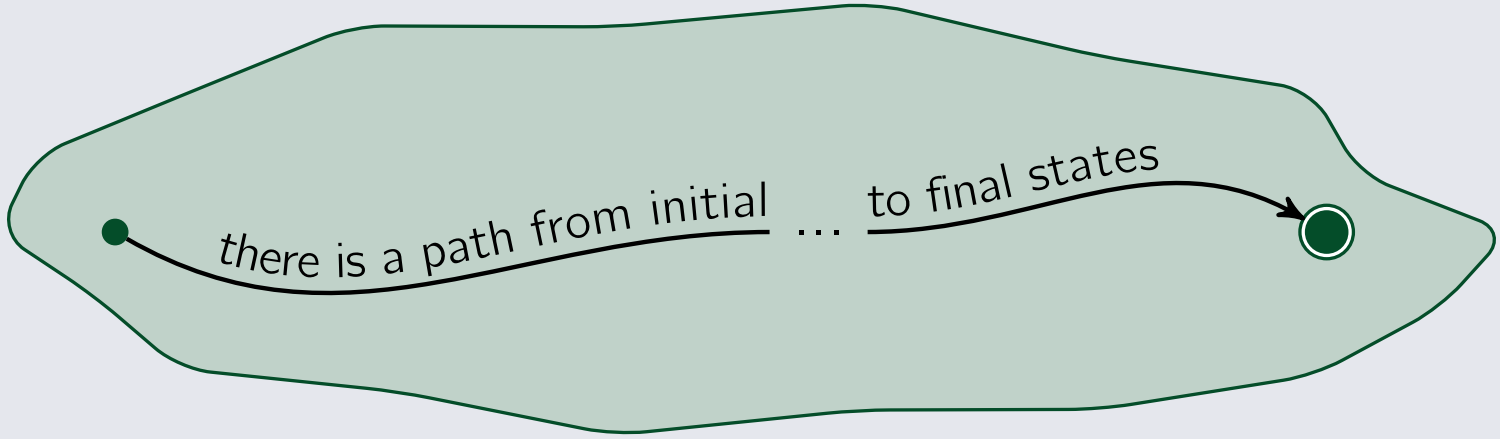




Problems on automata

- **Non-emptiness**

  $\mathscr{L}(\mathcal{A}) \neq \varnothing$ iff



- **Universality**

  $\mathscr{L}(\mathcal{A}) = \Sigma^\star$ iff $\mathscr{L}(\mathcal{A}^\mathsf{C}) = \varnothing$

- **Containment**

  $\mathscr{L}(\mathcal{A}) \subseteq \mathscr{L}(\mathcal{B})$ iff $\mathscr{L}(\mathcal{A}) \cap \mathscr{L}(\mathcal{B}^\mathsf{C}) = \varnothing$

☞ These are simple graph search problems!

Logics for specification of properties

- **Propositional logic**

$b \vee \neg b$

Logics for specification of properties

- **Propositional logic**

$$b \lor \neg b$$

- **First-order logic**

$$a(x_0) \land \forall x.\, \big(a(x) \to b(x)\big) \to b(x_0)$$

Logics for specification of properties

- **Propositional logic**

$$b \lor \neg b$$

- **First-order logic**

$$a(x_0) \land \forall x.\, \big(a(x) \to b(x)\big) \to b(x_0)$$

- **Monadic second-order logic**

$$\exists Z.\ \forall x.\ \exists y.\ (Z(y) \land y = x + 1)$$

Examples of sentences and formulas

$$\psi_{\text{dense}} \quad = \quad \forall x, y.\ \exists z.\ (x < y \rightarrow x < z \ \wedge \ z < y)$$

$$\psi_{\text{connected}} \quad = \quad \forall Z.\ \big(\exists x, y.\ Z(x) \ \wedge \ \neg Z(y)\big) \ \rightarrow \ \big(\exists x, y.\ Z(x) \ \wedge \ \neg Z(y) \ \wedge \ E(x, y)\big)$$

$$\psi_{\text{path}}(x, y) \quad = \quad \forall Z.\ Z(x) \ \wedge \ \forall x', y'.\ \big(Z(x') \wedge E(x', y') \rightarrow Z(y')\big) \ \rightarrow \ Z(y)$$

Examples of sentences and formulas

$$\psi_{\text{dense}} = \underbrace{\forall x, y.\ \exists z.\ (x < y \to x < z \ \wedge\ z < y)}_{\text{FO}[<] \text{ over } \mathbb{Q}}$$

$$\psi_{\text{connected}} = \underbrace{\forall Z.\ \big(\exists x, y.\ Z(x) \ \wedge\ \neg Z(y)\big) \ \to\ \big(\exists x, y.\ Z(x) \ \wedge\ \neg Z(y) \ \wedge\ E(x, y)\big)}_{\text{MSO}[E] \text{ over } G=(V,E)}$$

$$\psi_{\text{path}}(x, y) = \underbrace{\forall Z.\ Z(x) \ \wedge\ \forall x', y'.\ \big(Z(x') \wedge E(x', y') \to Z(y')\big) \ \to\ Z(y)}_{\text{MSO}[E] \text{ over } G=(V,E)}$$

☞ The underlying **signature** and **domain** are important!

- MSO[+1] over $\mathbb{N}$ = MSO[<] over $\mathbb{N}$
- FO[0, 1, +] = Presburger arithmetic
- MSO over $\mathbb{N}$ = FO[$\subseteq$] over $2^{\mathbb{N}}$
- FO[$\in$] over models of Zermelo–Fraenkel set theory...

Other examples of MSO properties

$$\begin{aligned}\psi_{\text{3-colorability}} \;=\; & \exists X, Y, Z.\ \forall v.\ \big(X(v) \vee Y(v) \vee Z(v)\big) \\ & \forall u, v.\ E(u,v) \;\rightarrow\; \neg\big(X(u) \wedge X(v)\big) \wedge \\ & \qquad\qquad \neg\big(Y(u) \wedge Y(v)\big) \wedge \\ & \qquad\qquad \neg\big(Z(u) \wedge Z(v)\big)\end{aligned}$$

$$\psi_{\mathrm{K5}}(x_1, \ldots, x_5) \;=\; \bigwedge_{i \neq j} \big(x_i \neq x_j \;\wedge\; \psi_{\text{path}}(x_i, x_j)\big)$$

$$\psi_{\mathrm{K3,3}}(x_1, \ldots, x_3, y_1, \ldots, y_3) \;=\; \bigwedge_{i \neq j} (x_i \neq x_j \wedge y_i \neq y_j) \;\wedge\; \bigwedge_{i,j} \psi_{\text{path}}(x_i, y_j)$$

$$\begin{aligned}\psi_{\text{planar}} \;=\; & \neg\exists x_1, \ldots, x_5.\ \psi_{\mathrm{K5}}(x_1, \ldots, x_5) \;\wedge \\ & \neg\exists x_1, \ldots, x_3, y_1, \ldots, y_3.\ \psi_{\mathrm{K33}}(x_1, \ldots, x_3, y_1, \ldots, y_3)\end{aligned}$$

A real example

let $Foo(g, h)$ =

$g(h)$

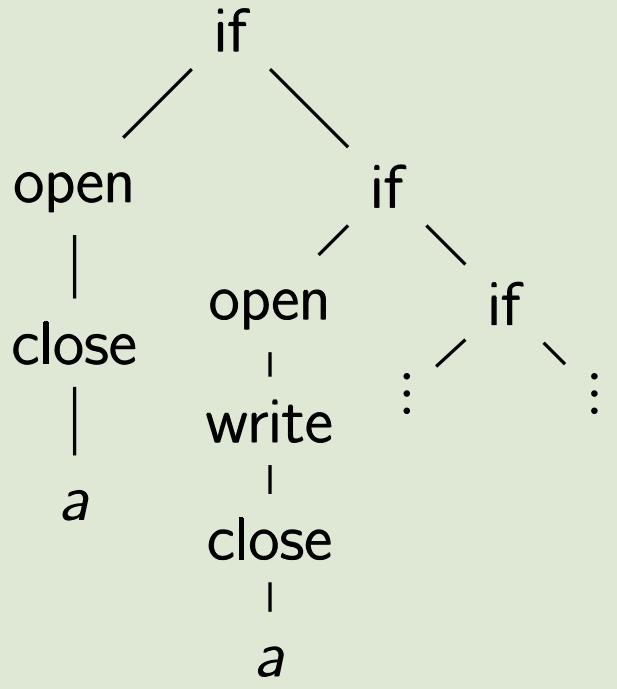
if [*user hits key*] then

$g \cdot \mathsf{close}(h)$

else

$Foo(g \cdot \mathsf{write}, h)$

$Foo(\mathsf{open}, a)$



A real example

let $Foo(g, h)$ =

$g(h)$

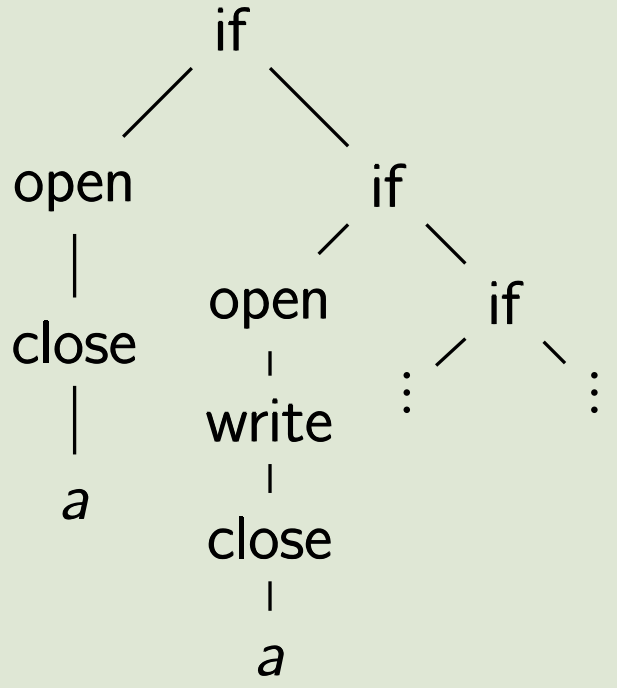
if [*user hits key*] then

$g \cdot \mathsf{close}(h)$

else

$Foo(g \cdot \mathsf{write}, h)$

$Foo(\mathsf{open}, a)$



One may want to verify that all sequences of write operations occur between open and close operations:

$$\forall Z \text{ path}.\ \forall z \in Z.\ \ \mathsf{write}(z) \ \rightarrow\ \exists x, y \in Z.\ \ x < z < y \ \wedge\ \mathsf{open}(x) \ \wedge\ \mathsf{close}(y)$$

Next