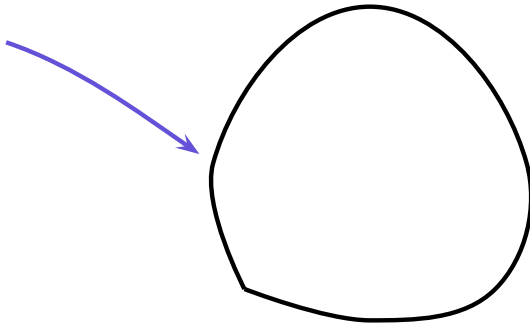

Pushing the limits of pushdown verification

Igor Walukiewicz

CNRS, Bordeaux

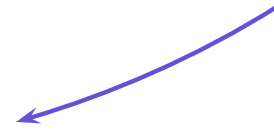
Model



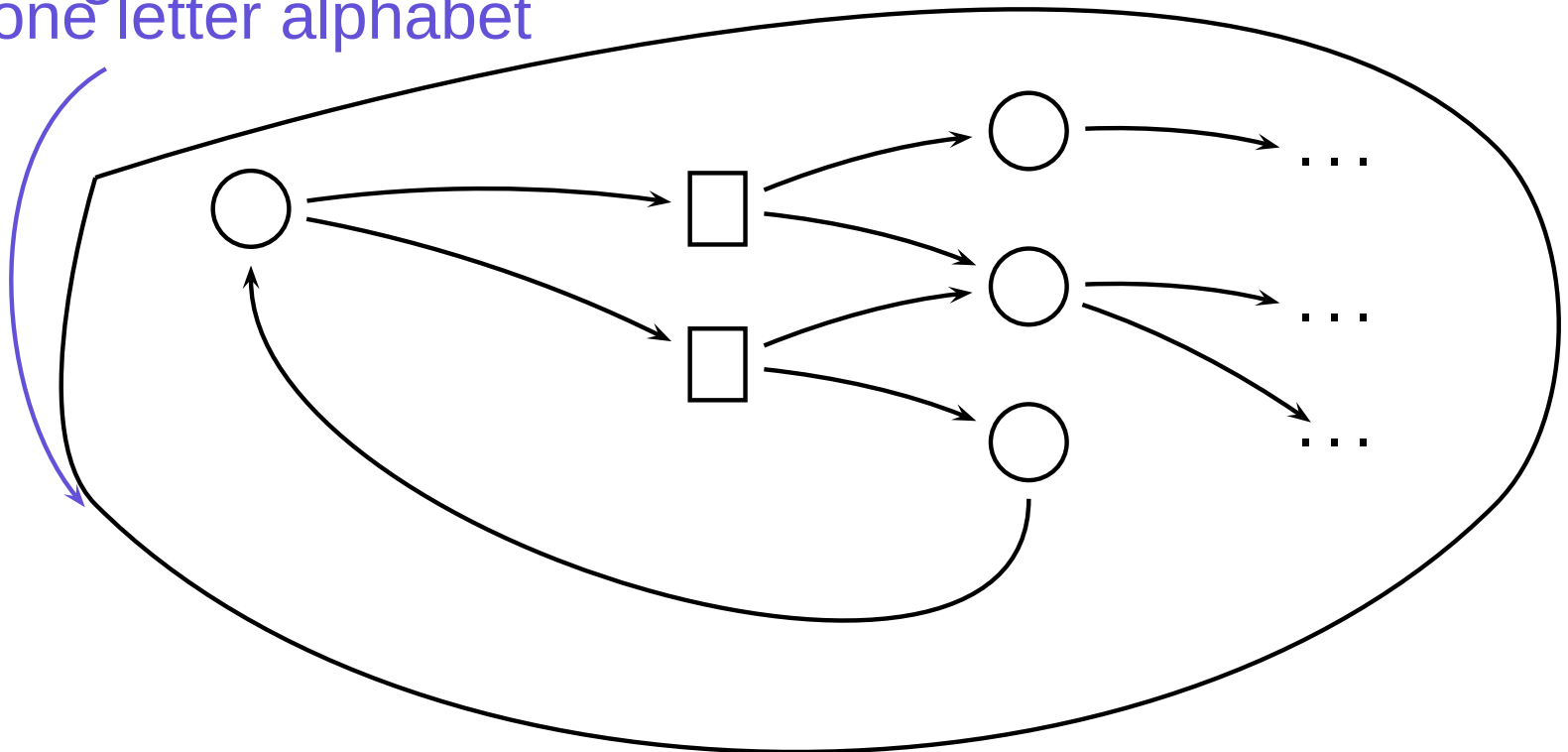
$\models$

α

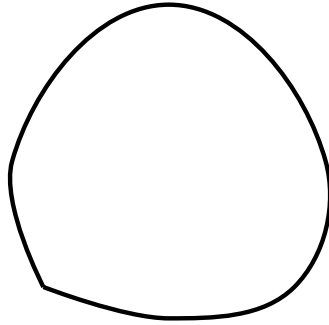
Formula



Alternating tree automaton
over one letter alphabet



Model



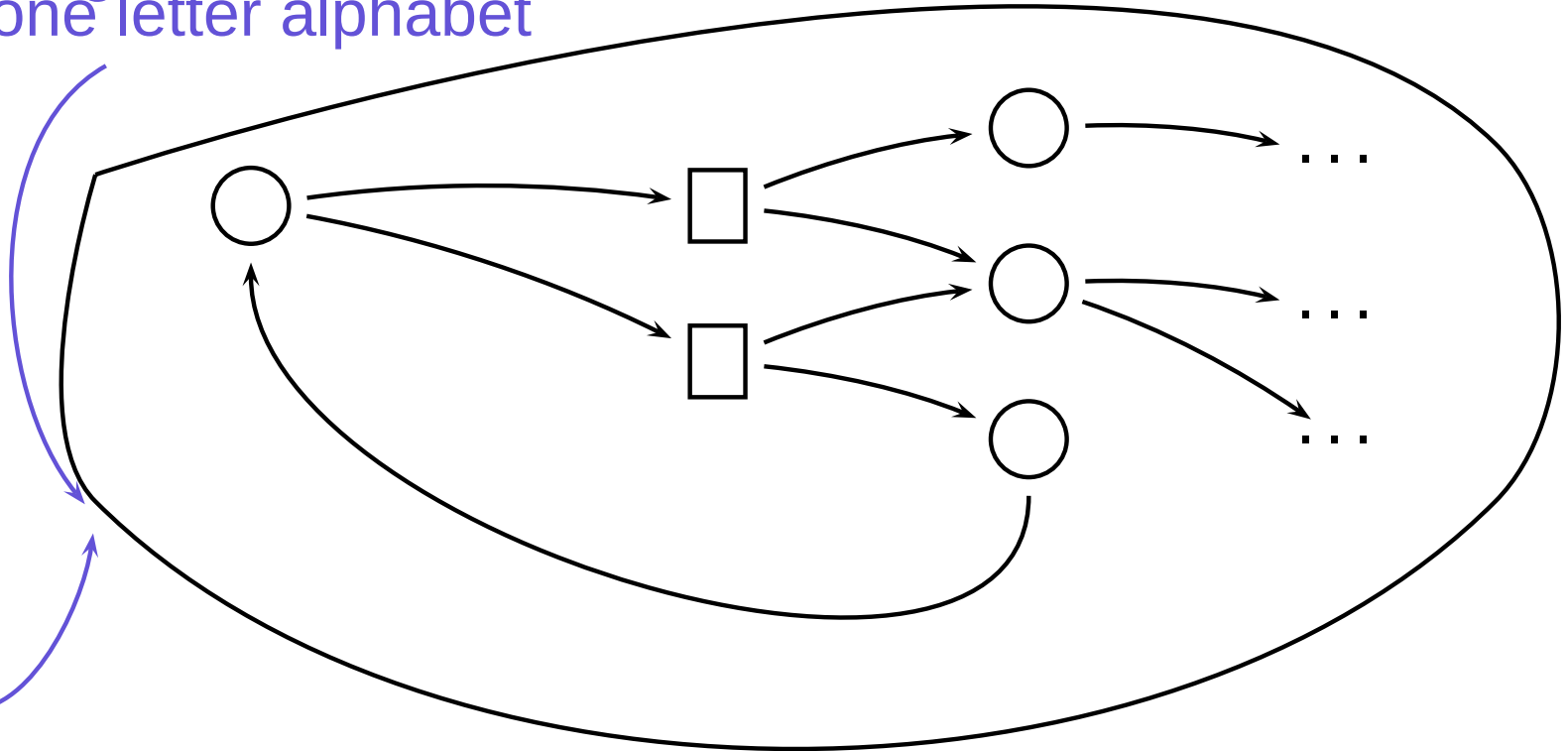
$\models \alpha$

Formula

α



Alternating tree automaton
over one letter alphabet



Game

- What are formulas?
 - Monadic Second Order Logic (MSOL)
 - Program logics like LTL, CTL, the μ -calculus.
- What are models? Transition systems.
 - Finite transition systems given explicitly.
 - Graph of configurations of some machine.
 - Graph given by rewriting rules.
 - Process rewriting graphs.
 - Automatic structures (over words and trees).
 - Regular graphs

⋮

- From pushdown to regular graphs.
- Verifying regular properties.
- Higher order pushdowns.
- Recursive program schemes.
- Non-regular properties of pushdowns.

- H.-D. Ebbinghaus, J. Flum, and W. Thomas. Mathematical Logic. Springer-Verlag, 1984.
- W. Thomas. Automata on infinite objects. In J. van Leeuwen, editor, Handbook of Theoretical Computer Science, volume B: Formal Models and Semantics, pages 133-192. Elsevier Science Publishers, Amsterdam, 1990.
- W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, Handbook of Formal Languages, volume III, pages 389-455. Springer, New York, 1997.

Part I

From pushdown to regular graphs

Graphs of pushdown machines

- Pushdown machine (deterministic):

$\langle Q, \Sigma, \Gamma, q_0 \in Q, \delta : Q \times \Sigma \times \Gamma \rightarrow Q \times \{pop, push(z) : z \in \Gamma\}, F \subseteq Q \rangle$.

- Configuration: $(q, w) \in Q \times \Gamma^*$.

- Configuration graph

- nodes: configurations

- transitions:

- $(q, zw) \rightarrow (q', w)$ if there is $a \in \Sigma$ and $\delta(q, a, z) = (q', pop)$

- $(q, zw) \rightarrow (q', z'zw)$ if there is $a \in \Sigma$ and $\delta(q, a, z) = (q', push(z'))$

- **Rem:** The input alphabet and accepting states do not play any role. Determinism is also not important.

Pushdown system: $P = (Q, \Gamma, \Delta)$

Rewrite rules: $\Delta \subseteq Q \times \Gamma \times Q \times (\{\varepsilon\} \cup \Gamma^2)$
 $qz \mapsto q' \quad qz \mapsto q'z'z$

Pushdown graph: $G(P)$

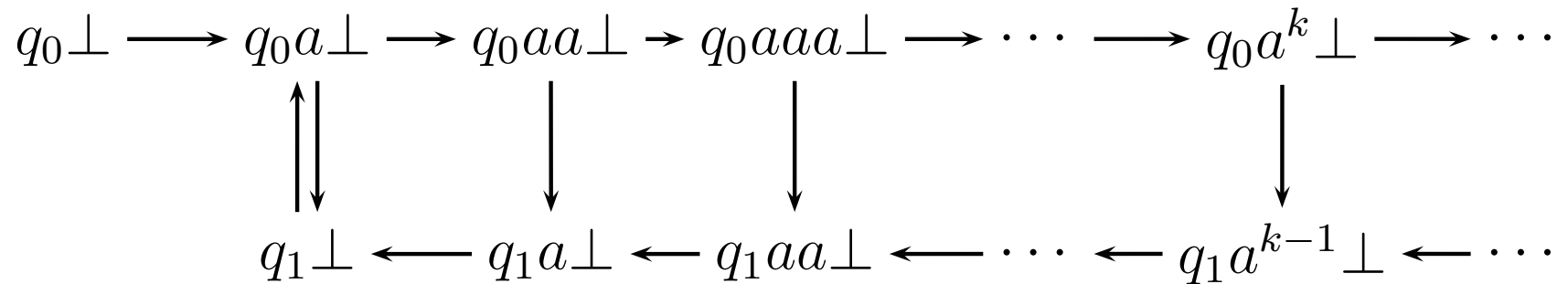
Vertices: $Q \times \Gamma^*$

Edges: $qw \rightarrow q'w'$ according to the rules **applied to prefixes**.

● q_0 is always the initial state and $\perp$ is the initial stack symbol.

TM graph: rules of the form $aqb \mapsto q'a'b$ or $aqb \mapsto ab'q'$ without restrictions on the place of application.

Pushdown graph: an example



● This is (a part of) the graph of the system:

$$q_0 \perp \succrightarrow q_0 a \perp$$

$$q_0 a \succrightarrow q_0 aa$$

$$q_1 a \succrightarrow q_1$$

$$q_1 \perp \succrightarrow q_0 a \perp$$

$$q_0 a \succrightarrow q_1$$

Pushdown system: $P = (\Gamma, \Delta)$

Rewrite rules: $\Delta \subseteq \mathcal{P}(Q^*) \times \mathcal{P}(Q^*)$

$L \rightharpoonup L'$ for L and L' regular languages.

Prefix-recognizable graph: $G(P)$

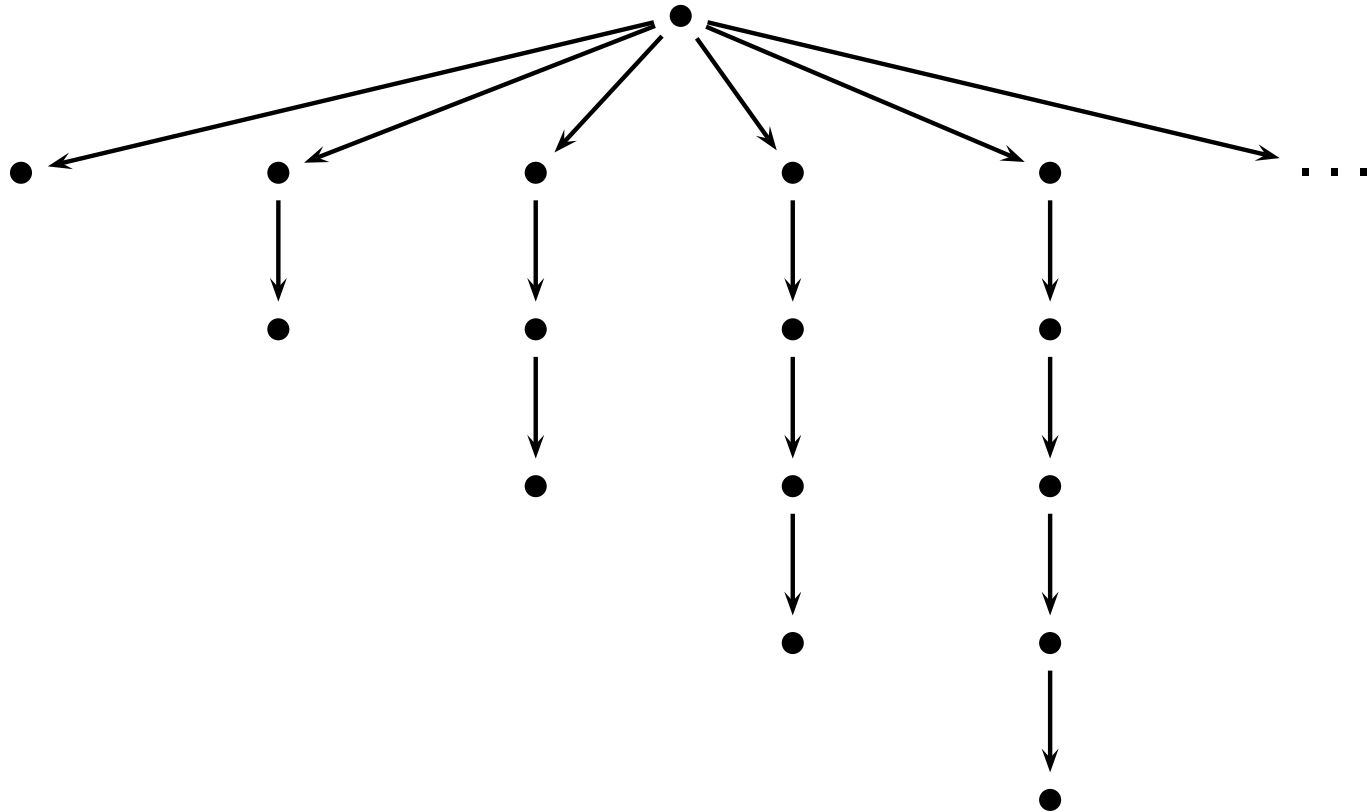
Vertices: Γ^*

Edges: $wu \rightarrow w'u$ if $w \in L$ and $w' \in L'$ for some $L \rightharpoonup L'$.

Rem: Prefix-recognizable graph of finite degree is a pushdown graph.

Thm [Carayol & Wöhrle]: Prefix-recognizable graphs are ε -closures of pushdown graphs.

Example of a prefix recognizable graph



Synchronized rational and rational graphs

- A relation $R \subseteq \Gamma^* \times \Gamma^*$ is **rational** if it is recognizable by a finite automaton with two heads moving asynchronously from left to right.
- A relation $R \subseteq \Gamma^* \times \Gamma^*$ is **synchronous rational** if the heads of the automaton always move together.
- A graph is rational if it is (Γ^*, R) where R is a rational relation.

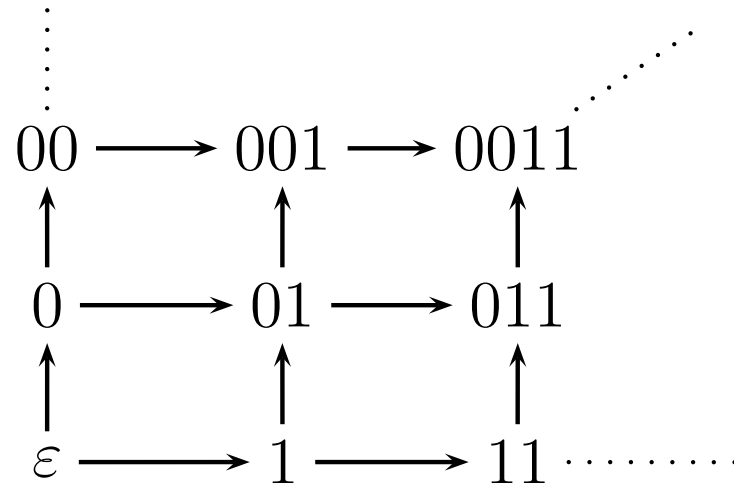
Rem: TM graphs are synchronous rational.

Rem: Synchronous rational are also called **automatic**.

Examples of automatic and rational graphs

- Synchronous rational graph: grid

$$R_u = \{(0^n 1^n, 0^{n+1} 1^n) : n \in \mathbb{N}\} \quad R_r = \{(0^n 1^n, 0^n 1^{n+1}) : n \in \mathbb{N}\}$$



- Rational graph: Given $(u_1, \dots, u_n), (v_1, \dots, v_n)$ of a Post correspondence problem define:

$$R = \{(u_{i_1} \dots u_{i_k}, v_{i_1} \dots v_{i_k}) : i_1, \dots, i_k \in \{1, \dots, n\}\}$$

- In general this is not a synchronous rational graph.

Part II

Verifying regular properties

- The Σ -tree is Σ^* with the root ε and wb the b son of w .

- MSOL over Σ -trees:

$$\text{succ}_b(x, y) \mid Z(x) \mid \neg\varphi \mid \varphi \wedge \psi \mid \exists x.\varphi \mid \exists Z.\varphi$$

- Semantics in the Σ -tree.

+ $G, V \models \text{succ}_b(x, y)$ iff $V(x)b = V(y)$

+ $G, V \models \exists Z.\alpha$ iff $G, V[A/Z] \models \alpha$ for some $A \subseteq E$.

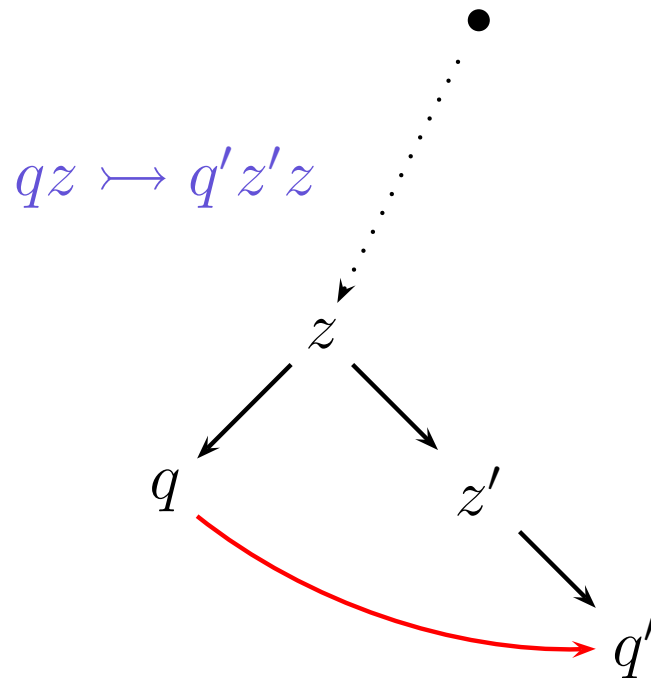
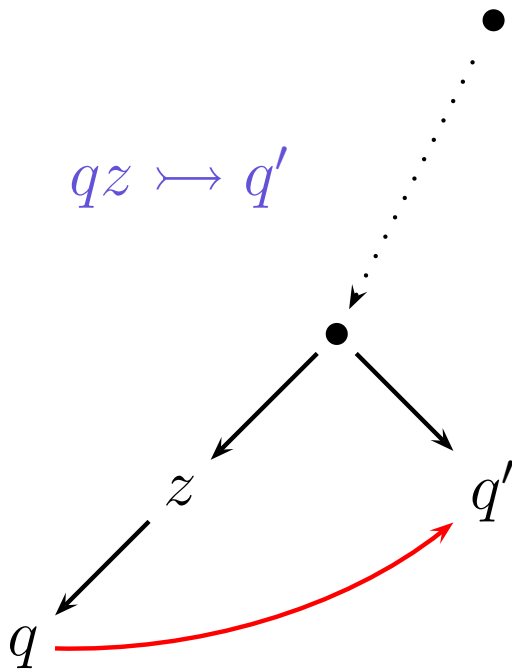
Thm[Rabin]: The MSO theory of the Σ -tree is decidable.

Thm[Caucal]: A pushdown tree can be defined inside the Σ -tree using MSOL formulas.

Cor[Muller, Schupp]: The MSO theory of a pushdown tree is decidable

Pushdown system inside a tree

- Pushdown rules: $qz \rightsquigarrow q'$ $qz \rightsquigarrow q'z'z$
- Configuration is represented by a word $qz_kz_{k-1} \dots z_1$.
- Hence we can identify configurations with some nodes of the tree $(Q \cup \Gamma)^*$.



● Let $\rho : Q \rightarrow \mathcal{P}(\text{Prop})$ be a valuation. This extends to

$$\rho : Q \times \Gamma^* \rightarrow \mathcal{P}(\text{Prop}) \quad \text{by} \quad \rho(qw) = \rho(q).$$

So we have a model $M(P, \rho)$.

Model checking problem: Given a pushdown system $\mathcal{P}$ with a valuation ρ and a formula α check if $M(P, \rho), q_0 \perp \models \alpha$.

● Example: Alternating reachability

Is there a choice of successors in E nodes such that every path from v passes through F .

● EF logic

$$p \mid \neg\alpha \mid \alpha \wedge \beta \mid \exists\langle\rangle a \mid \exists F\alpha$$

!No $\exists G\alpha$!

● CTL

$$\text{EF} \quad + \quad \left(\exists(\alpha_1 U \alpha_2) \mid \exists\neg(\alpha_1 U \alpha_2) \right)$$

● $G, v \models \exists F\alpha$ iff there is v' reachable from v with $G, v' \models \alpha$

● $G, v \models \exists G\alpha$ iff there is a path from v s.t. for every v' on it we have $G, v' \models \alpha$.

● μ -calculus

$$P \mid \neg P \mid \textcolor{red}{X} \mid \alpha \mid \alpha \vee \beta \mid \alpha \wedge \beta \mid \langle a \rangle \alpha \mid [a] \alpha \mid \textcolor{red}{\mu} X. \alpha \mid \textcolor{red}{\nu} X. \alpha$$

Thm: Model checking problem for the μ -calculus is EXPTIME-complete.

Thm: The model checking problem for EF-logic is in PSPACE. It is PSPACE-hard [Bouajjani, Esparza, Maler]

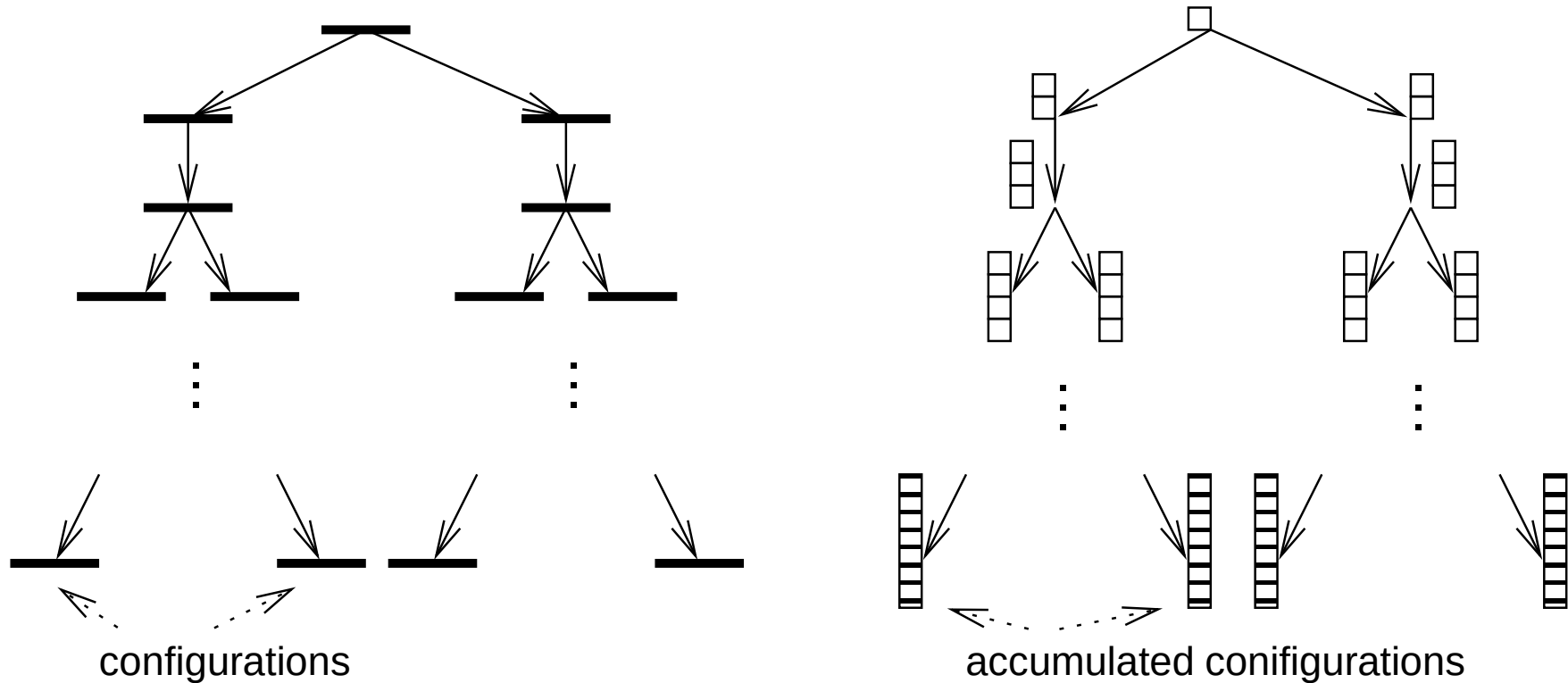
Thm: The same problem for CTL formulas is EXPTIME-complete.

Rem: The problem is with $\exists \alpha U \beta$.

Alt reachability: EXPTIME-hard

- Take $\text{ASPACE}(n)$ machine M and input w . Construct P_w :

P_w has alt. reach. prop. iff $w \in L(M)$.



- How to check that a sequence of accumulated configurations is correct?

Checking consecutive configurations

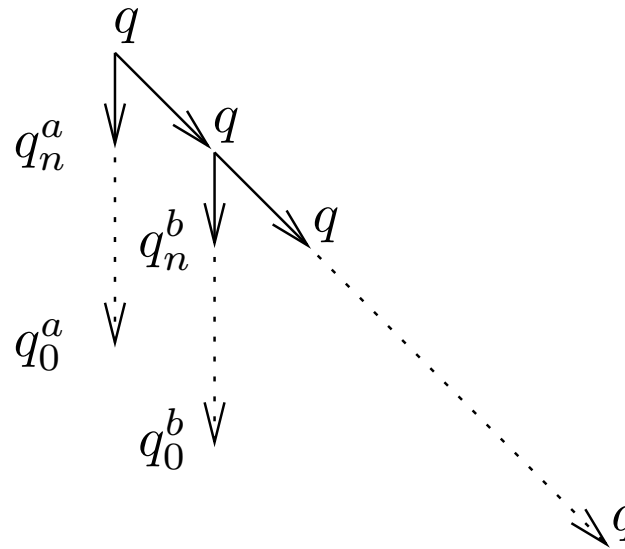
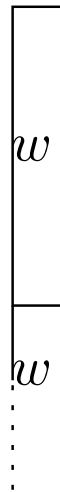
● Simpler problem: Given a word w of length n decide if the stack is of the form $w^k \perp$ for some k .

$$qa \rightarrow q, \quad q_n^a$$

$$q_n^a b \rightarrow q_{n-1}^a$$

$$q \perp \rightarrow q_F$$

$$q_0^a a \rightarrow q_F$$



Summary of pushdown model checking

μ -calc	EXPTIME-compl
Alt reach	EXPTIME-compl
CTL	EXPTIME-compl
LTL	EXPTIME-compl
EF	PSPACE-compl
reach	PTIME

Decidability: synchronized regular graphs

- TM graphs are synchronized regular.
- As reachability is expressible in MSOL, synchronized regular graphs may have undecidable MSO theory.

Thm: The FOL theory of a synchronized regular graph is decidable

- Every FOL definable relation in a synchronized regular graphs is synchronized regular. (Induction on the definition of the relation.)

Thm [Thomas]: There is a regular graph that has undecidable FO-theory.

● Consider a Post correspondence problem $(u_1, \dots, u_n), (v_1, \dots, v_n)$ and the associated graph:

$$R = \{(u_{i_1} \dots u_{i_k}, v_{i_1} \dots v_{i_k}) : i_1, \dots, i_k \in \{1, \dots, n\}\}$$

● [Morvan] The problem has a solution iff there is a vertex with a self-loop: $u_{i_1} \dots u_{i_k} = v_{i_1} \dots v_{i_k}$

● We want a fixed graph with undecidable FOL theory.

● $G(U)$ Post graph associated with an universal Turing machine.

● For every M and w : $w \in L(M)$ iff the Post correspondence problem has a solution starting with $w\#c(M)\# \dots$

● Equivalently: $w \in L(M)$ iff in $G(U)$ there is a vertex with a self loop and with prefix $w\#c(M)\#$.

- Push-down and prefix-recognizable graphs have decidable MSO theory.
- Push-down graphs have bounded out degree.
- Synchronous rational graphs have decidable FO theory. But may have undecidable MSO theory.
- Rational graphs may have undecidable FO theory.

Cor: The strict inclusion of the classes of graphs.

Part III

Higher-order pushdowns

- 2-stack is a stack of stacks. There are operations on top-most stack and of copying the top-most stack.

- A system where all paths are of the form $q_1^k q_2^k q_3^k$

$$q_1[a] \rightarrow q_1[aa] \rightarrow \dots \rightarrow q_1[a^k] \rightarrow$$

$$q_2[a^k] \rightarrow q_2[a^{k-1}][a^k] \rightarrow \dots \rightarrow q_2[a][aa] \dots [a^k] \rightarrow$$

$$q_3[a][aa] \dots [a^k] \rightarrow q_3[aa] \dots [a^k] \rightarrow \dots \rightarrow q_3[a^k] \rightarrow q_3 \perp$$

- 2-stack gives additional power. If considered as an accepting device 2-store automaton would recognize $\{a^k b^k c^k : k \in \mathbb{N}\}$.

2nd order pushdown system

- 2nd order pushdown system $\langle Q, \Gamma, Inst \rangle$
- A configuration is a sequence $[s_1][s_2] \dots [s_k] \in (\Gamma^*)^*$
- Instructions:

$$q'[w]v \xrightarrow{push_1(a)} q'[aw]v$$

$$q'[aw]v \xrightarrow{pop_1} q'[w]v$$

$$q'[w]v \xrightarrow{push_2} q'[w][w]v$$

$$q'[w]v \xrightarrow{pop_2} q'v$$

- Similarly for higher orders: stacks become of type $((\Gamma^*)^{* \dots *})$

● The n -th level of **Caucal hierarchy** consists of ε -closures of n -level pushdown graphs.

Rem: 1-st level graphs are prefix-recognizable graphs.

Thm [Carayol & Wöhrle]: n -th level Caucal graphs are precisely those that are MSOL interpretable in n -tree.

- A 2-tree over the set Γ is $\langle (\Gamma^*)^*, \{succ_z : z \in \Gamma\}, succ_2 \rangle$ where
 - $(v[w], v[wz]) \in succ_z$
 - $(v[w], v[w][w]) \in succ_2$

- A 2-tree over the set Γ is $\langle (\Gamma^*)^*, \{succ_z : z \in \Gamma\}, succ_2 \rangle$ where
 - $(v[w], v[wz]) \in succ_z$
 - $(v[w], v[w][w]) \in succ_2$

● A 2-tree is MSO-definable in the iteration of a normal tree. Iteration is an operation introduced by Muchnik that preserves MSOL decidability.

Cor: MSO-theory of n -tree is decidable.

Cor: Every graph in the Caucal hierarchy has decidable MSO-theory.

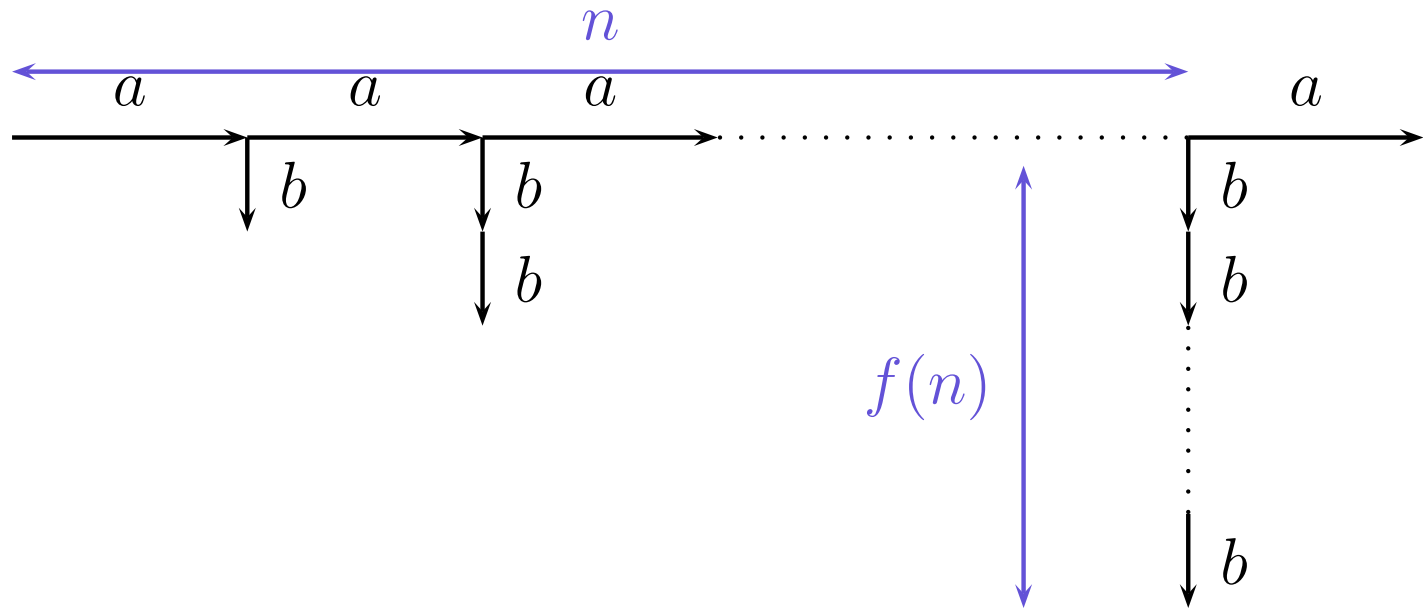
Thm[Engelfriet]: Alternating reachability problem for n -th level pushdown-graphs is n -EXPTIME-hard.

Thm[Cachat]: The μ -calculus model-checking problem for n -th level pushdown graphs is solvable in n -EXPTIME.

Rem: Not clear what is the complexity for other logics of programs.

Infiniteness of Caucal hierarchy

- For a function $f : \mathbb{N} \rightarrow \mathbb{N}$ we define a T_f tree:



- Define $Tower_0(n) = n$ and $Tower_{k+1}(n) = 2^{Tower_k(n)}$. Moreover $Tower_\omega(n) = Tower_n(n)$.

- T_{Tower_k} tree is an k -level graph that is not $(k - 1)$ -level.

- T_{Tower_ω} tree is not in the Caucal hierarchy. It has decidable MSO theory (follows from morphic predicates of Carton and Thomas).

Part IV

Recursive program schemes

- $\mathcal{F}x \Rightarrow x \mid a(\mathcal{F}(bx))$

$$\mathcal{F}(c) \Rightarrow a(\mathcal{F}(bc)) \Rightarrow a(a(\mathcal{F}(b(bc)))) \Rightarrow \dots$$

- In the rewriting the size grows “from the middle”. Pushdown?

Recursive program schemes

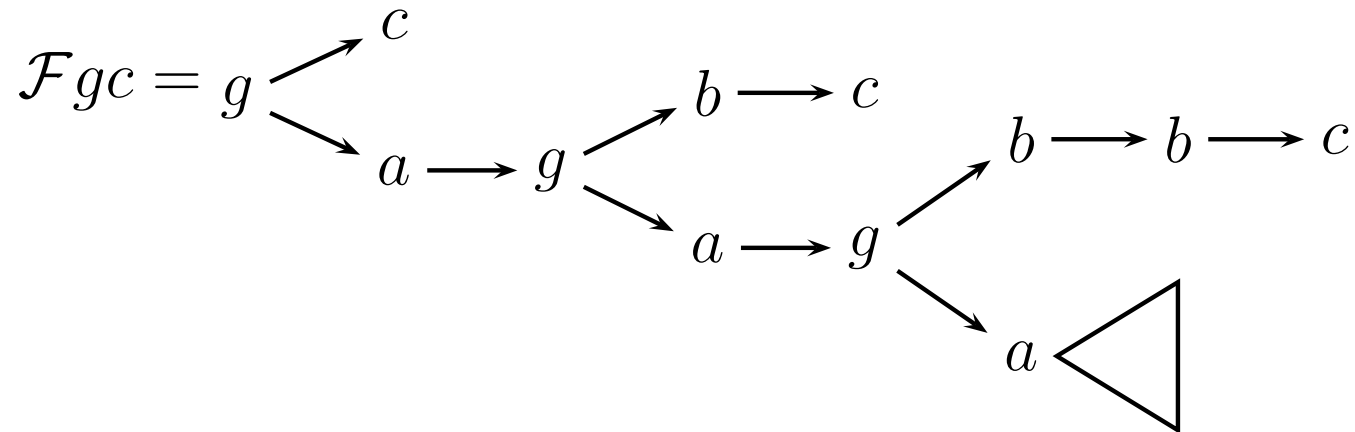
- $\mathcal{F}x \Rightarrow x \mid a(\mathcal{F}(bx))$

$$\mathcal{F}(c) \Rightarrow a(\mathcal{F}(bc)) \Rightarrow a(a(\mathcal{F}(b(bc)))) \Rightarrow \dots$$

- In the rewriting the size grows “from the middle”. Pushdown?

- $\mathcal{F}x \Rightarrow g(a(\mathcal{F}(bx)))x$

- Equations generate infinite trees (infinite terms).



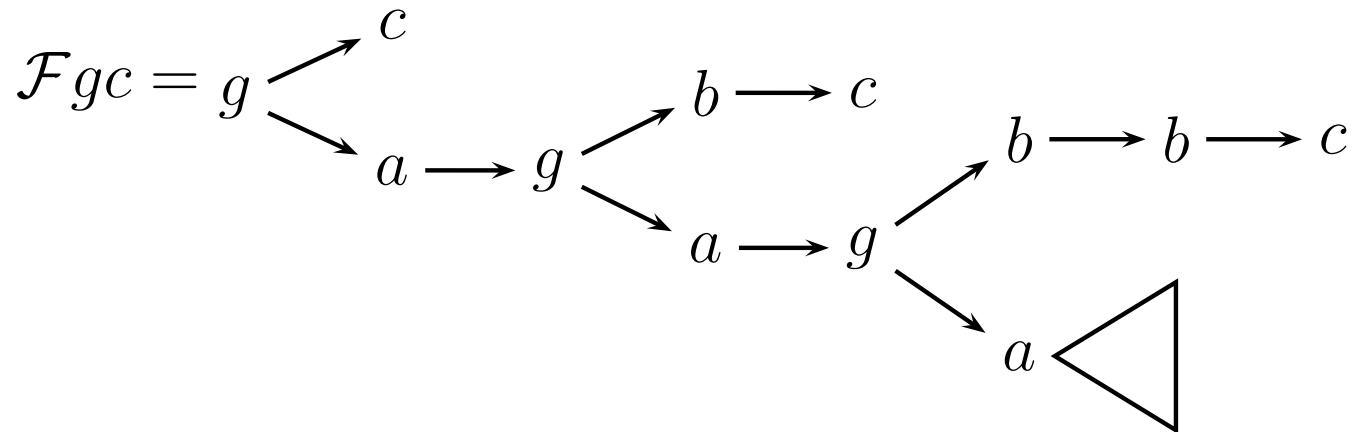
- $\mathcal{F}x \Rightarrow x \mid a(\mathcal{F}(bx))$

$$\mathcal{F}(c) \Rightarrow a(\mathcal{F}(bc)) \Rightarrow a(a(\mathcal{F}(b(bc)))) \Rightarrow \dots$$

- In the rewriting the size grows “from the middle”. Pushdown?

- $\mathcal{F}x \Rightarrow g(a(\mathcal{F}(bx)))x$

- Equations generate infinite trees (infinite terms).



Thm[Courcelle]: The meanings of (1st order) recursive schemes $\equiv$ pushdown graphs.

Recursive scheme: formally

- **Types:** $0, \quad 0 \rightarrow 0, \quad 0 \rightarrow 0 \rightarrow 0, \quad (0 \rightarrow 0) \rightarrow 0$
- **Σ constants:** $c : 0, \quad f : 0 \rightarrow 0.$
- **Var variables:** $x : 0, \quad z : 0 \rightarrow 0.$
- **V grammar nonterminals:** $\mathcal{F} : 0 \rightarrow 0, \quad \mathcal{G} : (0 \rightarrow 0) \rightarrow 0 \rightarrow 0.$
- **Applicative terms over $\Sigma \cup Var \cup V$.**

Recursive scheme: formally

- **Types**: $0, \quad 0 \rightarrow 0, \quad 0 \rightarrow 0 \rightarrow 0, \quad (0 \rightarrow 0) \rightarrow 0$
- Σ **constants**: $c : 0, \quad f : 0 \rightarrow 0.$
- Var **variables**: $x : 0, \quad z : 0 \rightarrow 0.$
- V **grammar nonterminals**: $\mathcal{F} : 0 \rightarrow 0, \quad \mathcal{G} : (0 \rightarrow 0) \rightarrow 0 \rightarrow 0.$
- **Applicative terms over** $\Sigma \cup Var \cup V.$

$$\mathcal{G} = \langle \Sigma, V, S, Prod \rangle$$

$$\mathcal{F} z_1 \dots z_m \Rightarrow r$$

- $\mathcal{F} z_1 \dots z_m : 0$
- $r : 0$ an applicative term over $\Sigma \cup V \cup \{z_1, \dots, z_m\}$
(no free variables)
- One rule per nonterminal.

Semantics of a recursive scheme/grammar

$$\mathcal{G} = \langle \Sigma, V, S, Prod \rangle$$

- $\mathcal{F}t_1 \dots t_n \rightarrow_{\mathcal{G}} r[z_1 := t_1, \dots, z_k := t_k]$ if there is a production:
 $\mathcal{F}z_1 \dots z_k \Rightarrow r$
- If $t \rightarrow_{\mathcal{G}} t'$ then $(st) \rightarrow_{\mathcal{G}} (st')$ and $(tr) \rightarrow_{\mathcal{G}} (t'r)$
- This means “reduce where you want”, i.e., call by name.
- This rewriting system is confluent and the result is a term (in general infinite).
- The **meaning** of the grammar is the result of rewriting of S .

Thm[Knapik, Niwiński, Urzyczyn]: Meanings of 2nd order **safe** schemes $\equiv$ 2nd order pushdown systems.
Similarly for higher orders.

- Example of a second order grammar:

$$\mathcal{F}\psi x \Rightarrow f(\mathcal{F}(\mathcal{D}\psi)x)(\psi x) \qquad \mathcal{D}\psi y \Rightarrow \psi(\psi y)$$

Thm[Knapik, Niwiński, Urzyczyn]: Meanings of 2nd order **safe** schemes $\equiv$ 2nd order pushdown systems.
Similarly for higher orders.

- Example of a second order grammar:

$$\mathcal{F}\psi x \Rightarrow f(\mathcal{F}(\mathcal{D}\psi)x)(\psi x) \qquad \mathcal{D}\psi y \Rightarrow \psi(\psi y)$$

- Safety allows to destroy the callers environment.

- $\mathcal{F}\psi x \Rightarrow f(\mathcal{F}(f(\psi x))x)x$

$$\mathcal{F}gc \mid \mathcal{F}(f(\psi_1 x_1))x_1 \mid \mathcal{F}(f(\psi_2 x_2))x_2 \mid \mathcal{F}(f(\psi_3 x_3))x_3$$

$$f(\psi_3 x_3) \rightarrow f(f(\psi_2 x_2)x_3) \rightarrow f(f(f(\psi_1 x_1)x_2)x_3)$$

2nd order pushdown systems with panic

- Stack $[s_k][s_{k-1}] \dots [s_1] \in ((\Gamma \times \mathbb{N})^*)^*$.
- 2nd order pushdown system with panic $\langle Q, \Gamma, Inst \rangle$
- A configuration is a sequence $q[s_k][s_{k-1}] \dots [s_1] \in Q \times ((\Gamma \times \mathbb{N})^*)^*$
- Instructions:

$$q[w]v \xrightarrow{push_1(a)} q'[(a, k-1), w]v$$

$$q[aw]v \xrightarrow{pop_1} q'[w]v$$

$$q[w]v \xrightarrow{push_2} q'[w][w]v$$

$$q[w]v \xrightarrow{pop_2} q'v$$

$$q[(a, l)w]v \xrightarrow{panic} q'[v_l] \dots [v_1]$$

Example how panic works

$\perp$

$push_1(a)$

$\perp a$

$push_2$

$\perp a$ $\perp a$

$push_1(b)$

$\perp a$ $\perp ab$

$push_2$

$\perp a$ $\perp ab$ $\perp ab$

$push_2$

$\perp a$ $\perp ab$ $\perp ab$ $\perp ab$

$push_2$

$\perp a$ $\perp ab$ $\perp ab$ $\perp ab$

$push_1(a)$

$\perp a$ $\perp ab$ $\perp ab$ $\perp aba$

pop_1

$\perp a$ $\perp ab$ $\perp ab$ $\perp ab$

$panic$

$\perp a$

Example: how to evaluate expressions

$$F\varphi x \Rightarrow \varphi(F\varphi x)$$

... $F(Am)n$

... $F(Am)n, \varphi(F\varphi x)$

... $F(Am)n, \varphi(F\varphi x)$... $F(Am)n, \varphi(F\varphi x)$

... $F(Am)n, \varphi(F\varphi x)$... $F(Am)n, Am \circ$

... $F(Am)n, \varphi(F\varphi x)$... $F(Am)n, Am \circ$... $F(Am)n, Am \circ \dots \circ$

... $F(Am)n, \varphi(F\varphi x)$... $F(Am)n, Am \circ$... $F(Am)n, Am \circ \dots \circ$

panic

... $F(Am)n, F\varphi x$

Thm[Knapik, Niwiński, Urzyczyn, W.]: Meanings of 2nd order schemes $\equiv$ trees of 2nd order pushdown systems with panic.

Thm[Knapik, Niwiński, Urzyczyn, W.]: For every 2nd order scheme, the tree defined by this scheme has decidable MSO theory.

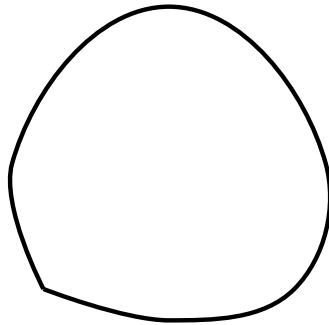
Question: Do unsafe schemes express more than safe ones?

Question: Decidability of equivalence between schemes.

Part V

Verifying non-regular properties

Model



$\models$

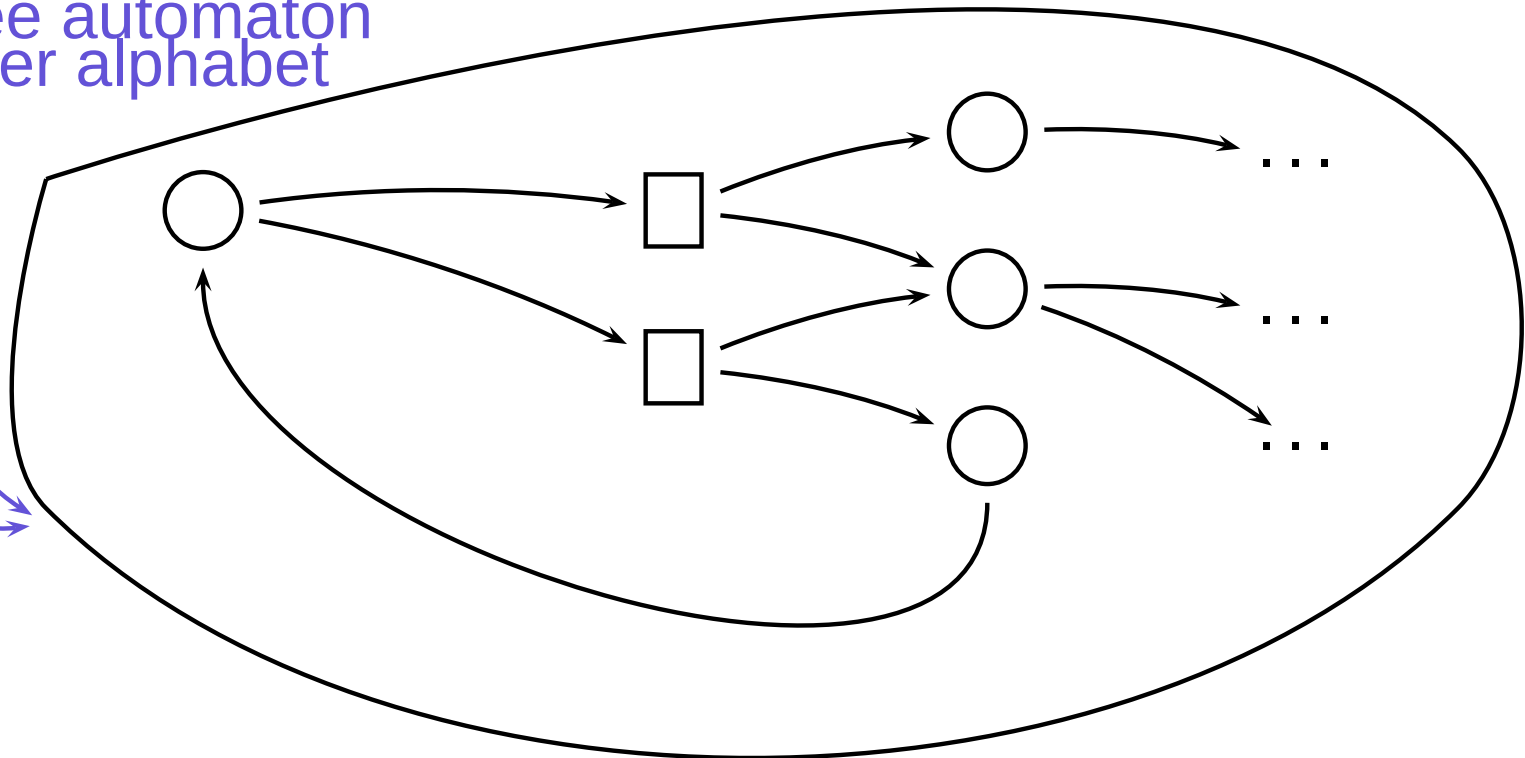
α

Formula



Alternating tree automaton
over one letter alphabet

Game



Model checking pushdown systems

- Given P and α decide if α holds in the initial vertex of $G(P)$.

- Construct an infinite **pushdown game** $G(P, \alpha)$:

$$G(P, \alpha) = \langle Q, \Gamma, \Delta, Q_E, Q_A, Acc \rangle$$

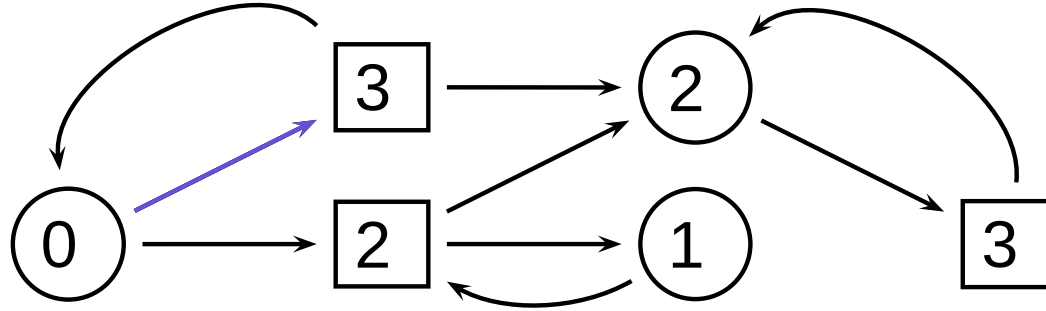
- pushdown system with states partitioned between Eve and Adam

- Property:

α holds in the initial vertex of $G(P)$

iff

Eve has a winning strategy from the initial vertex of $G(P, \alpha)$.



$$\mathcal{G} = \langle V_E, V_A, R, \lambda : V \rightarrow C, Acc \subseteq C^\omega \rangle$$

- $\text{Inf}_\lambda(\vec{v})$: the set of colours appearing infinitely often on a path $\vec{v}$.
- **Parity condition** colours are numbers $\{0, \dots, d\}$ and:

$$\vec{v} \in Acc \quad \text{iff} \quad \min(\text{Inf}_\lambda(\vec{v})) \text{ is even.}$$

Thm: Every game with a Muller winning condition is determined, i.e., from every vertex one of the players has a winning strategy.

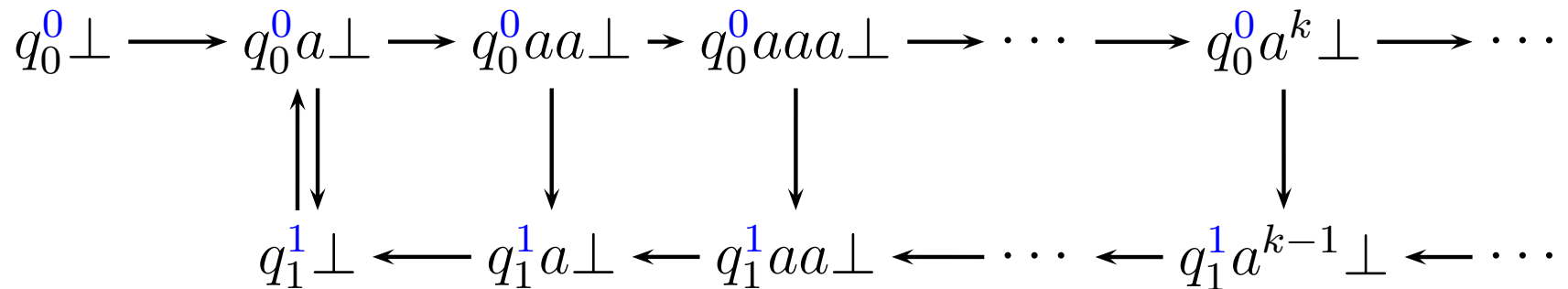
Thm: In a parity game a player has a memoryless winning strategy from each of his winning vertices.

Def: To **solve a game** is to determine for each position who has a winning strategy from this position.

Thm: There is an algorithm for solving finite Muller games.

[Martin, Emerson & Jutla, Mostowski]

Pushdown game: an example



● We have that:

q_0 is a vertex of Adam and q_1 of Eve;

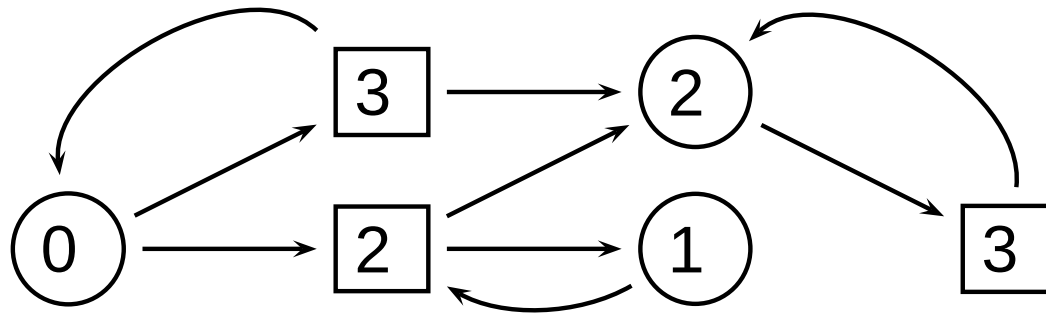
$\Omega(q_0) = 0$ and $\Omega(q_1) = 1$.

● Eve has a winning strategy in this game.

● The game solving problem: Given P with a partition (Q_E, Q_A) of states, and a function $\Omega : Q \rightarrow \mathbb{N}$ decide who has a winning strategy from the initial vertex of $G(P)$.

Thm: The problem of solving parity pushdown games is EXPTIME-complete. The MC problem is PTIME equivalent.

In contrast to the finite-state case, there are now natural winning conditions which are no more monadic second-order definable and occur at higher Borel levels than $\mathcal{B}(\Sigma_2^0)$. [Thomas, STACS'95]



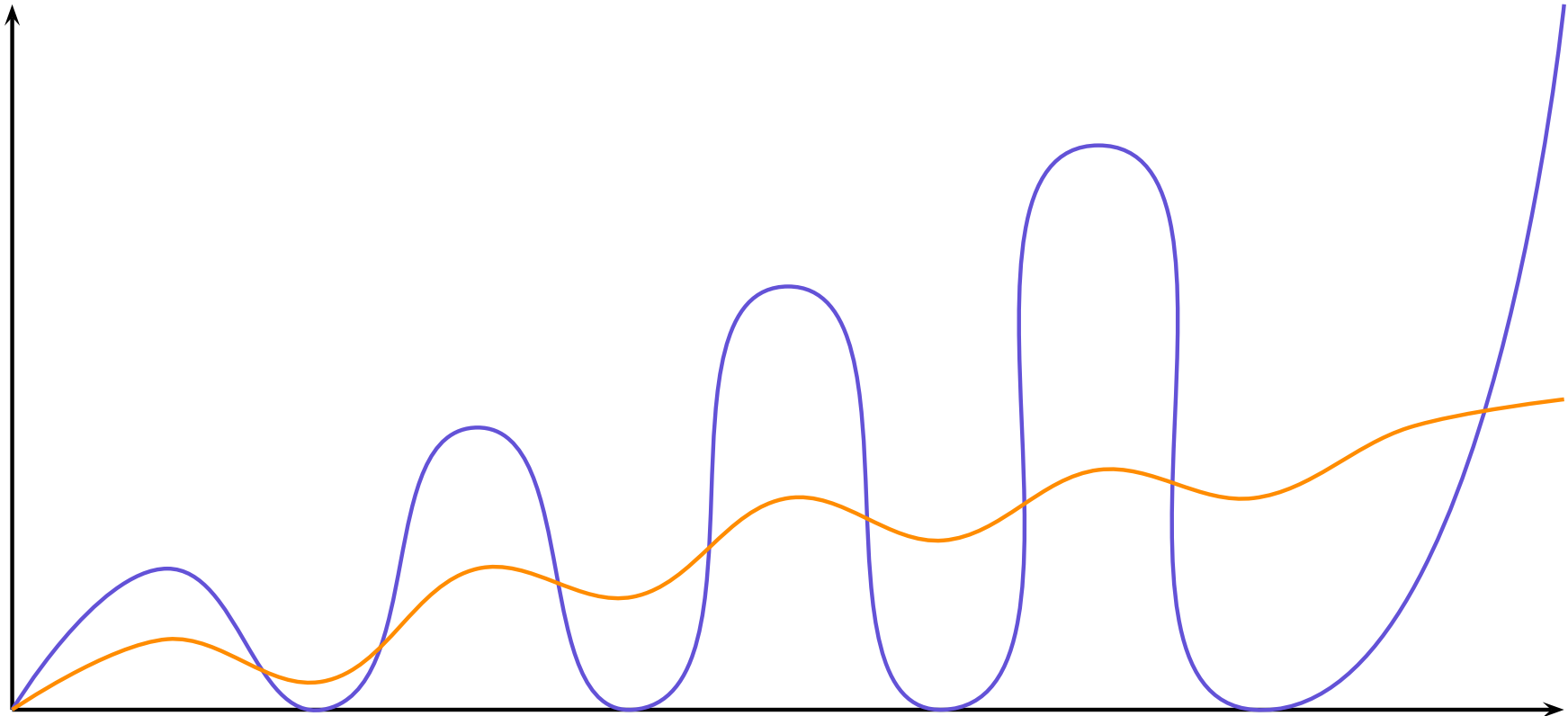
$$\mathcal{G} = \langle V_E, V_A, R, \lambda : V \rightarrow C, Acc \subseteq C^\omega \rangle$$

● $ht(v)$ the height of the stack at position v

Unboundedness conditions

● **Unboundedness condition** The height of the stack is unbounded: $\vec{v} \in Acc$ iff $\forall n. \exists i. ht(v_i) > n$.

● **Strict unboundedness condition** The liminf of the stack height is infinity: $\vec{v} \in Acc$ iff $\forall n. \exists i. \forall j > i. ht(v_j) > n$.

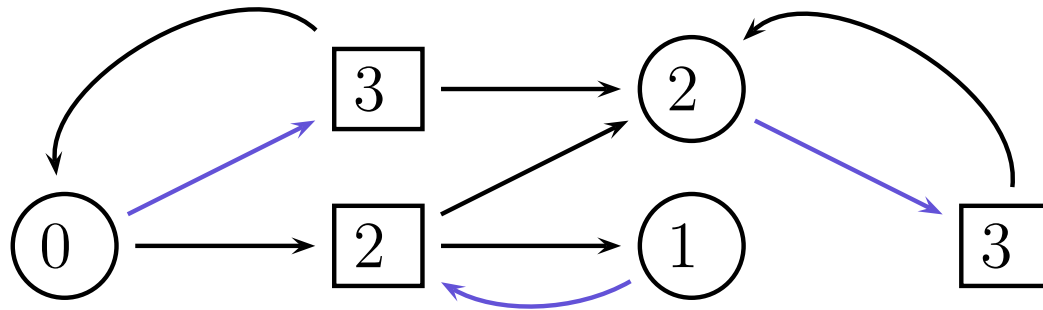


Memoryless strategies for unboundedness

$$\mathcal{G} = \langle V_E, V_A, R, \lambda : V \rightarrow C, Acc \subseteq C^\omega \rangle$$

- **Strategy** for player 0 is $\sigma : V^* \times V_0 \rightarrow V$ such that $\sigma(\vec{v}v_0) \in R(v_0)$

- A strategy σ for Eve is **winning from** v if all plays from v respecting the strategy are winning for Eve.



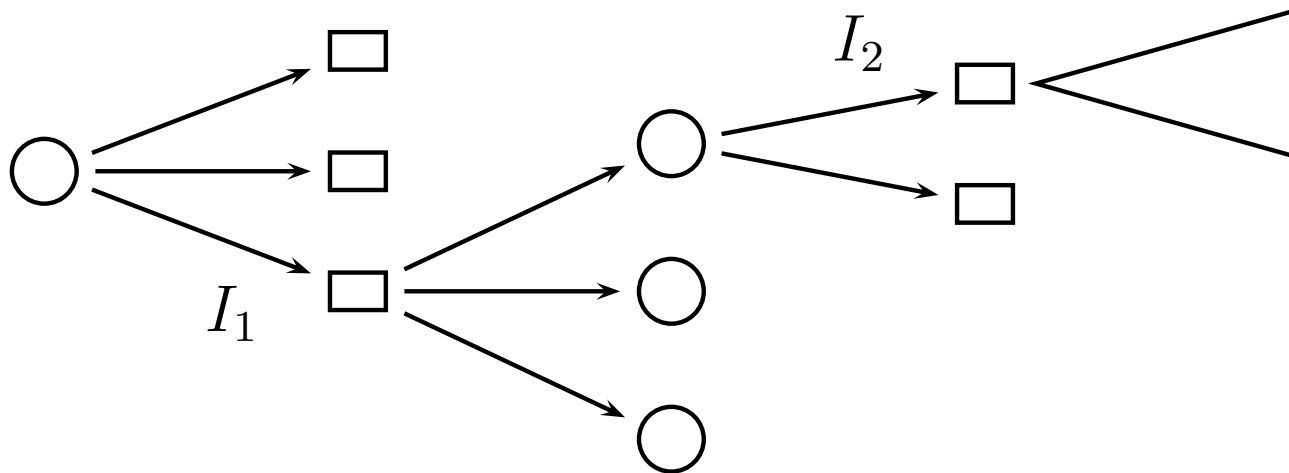
- **Positional/memoryless strategy** for Eve is a function $\sigma : V_0 \rightarrow V$ such that $\sigma(v) \in R(v)$.

Thm: In the game with unboundedness conditions Eve has a memoryless winning strategy.

Memoryless unboundedness: proof

Thm: In the game with unboundedness conditions Eve has a memoryless winning strategy.

- $\mathcal{G}_k$: game with the objective to reach the stack of height k .
- If v is losing in some $\mathcal{G}_k$ then it is losing in $\mathcal{G}$.
- Suppose v is winning in all $\mathcal{G}_k$. Let σ_k be a strategy in σ_k . We construct σ_∞ winning in $\mathcal{G}$.



- Take a successor that is chosen by infinitely many strategies.
- σ_∞ is memoryless and winning.

Unboundedness $\equiv$ strict unboundedness

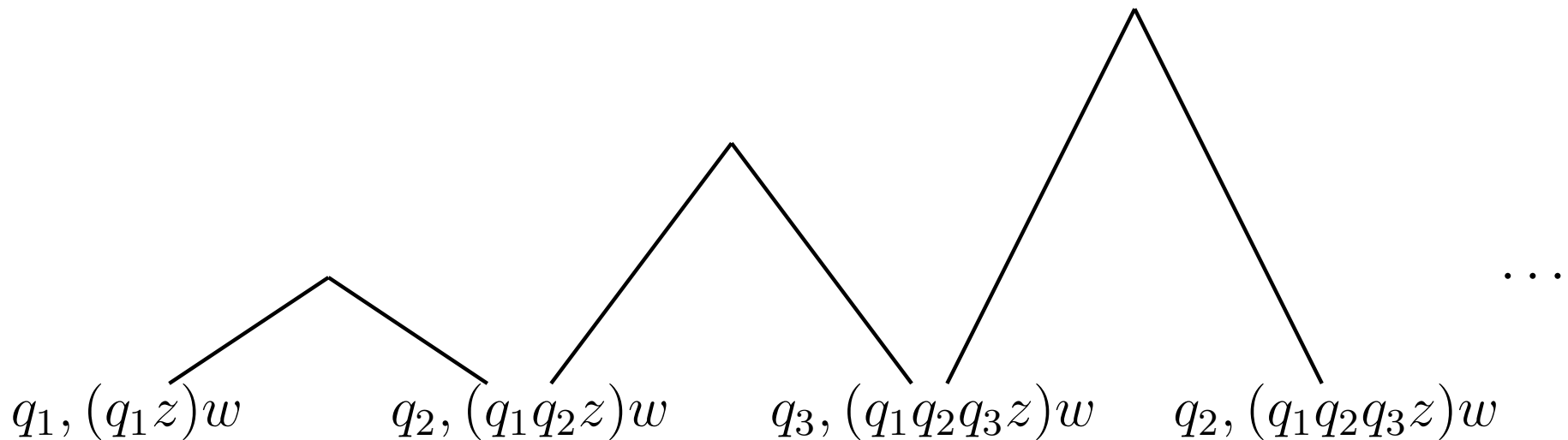
- The strategy σ_∞ never visits a vertex twice.
- If σ_∞ is winning for unboundedness then it is also winning for strict unboundedness.

Cor: The game is winning for unboundedness iff it is winning for strict unboundedness.

Cor: The same for disjunction of unboundedness and a parity condition.

Reduction: unboundedness to safety

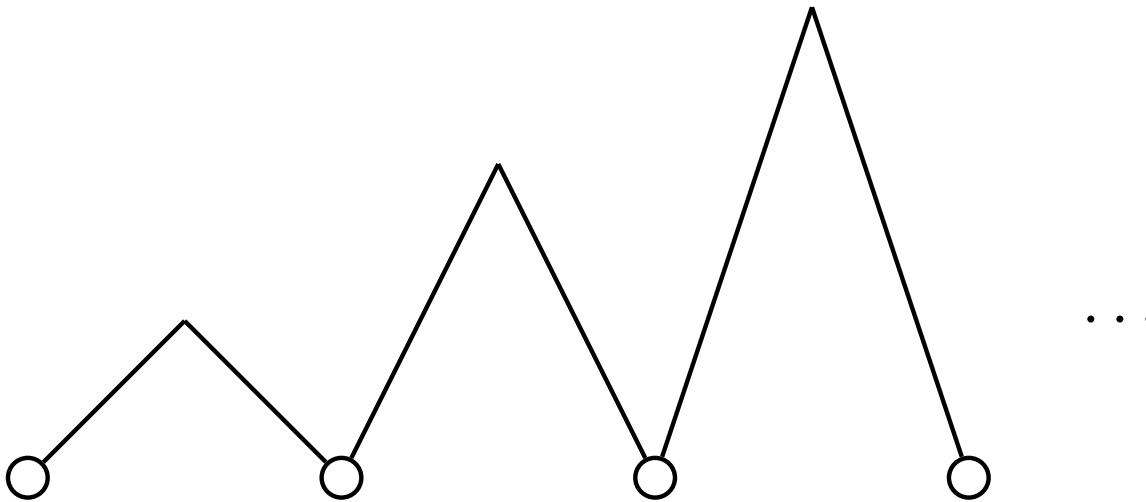
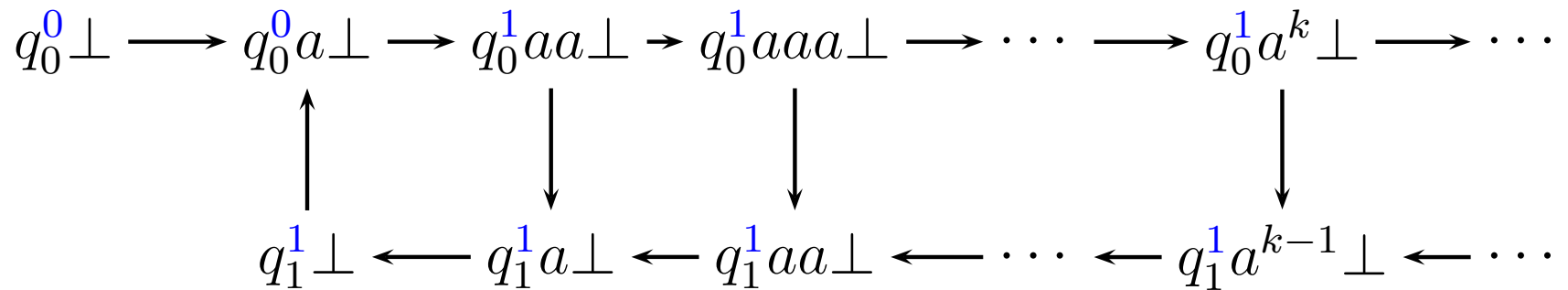
- There is a winning strategy for Eve in unboundedness game iff there is a memoryless winning strategy.
- Modify a pushdown system P in such a way that states seen with the current stack contents are recorded in the top of the stack.



- Make a position losing if the current state is the same as recorded on in the top of the stack

$$\text{unboundedness in } P \equiv \text{safety in } P'$$

Unboundedness and parity



- Infinite memory is need to win.

Thm: Winning conditions that are unions of explosion and parity conditions admit memoryless strategies. Intersection of Büchi and explosion conditions may need infinite memory.

Thm[Cachat & Duparc & Thomas]: Strict unboundedness condition is Σ_3 -complete in the Borel hierarchy.

Thm[Cachat & Duparc & Thomas, Bouquet & Serre & W., Gimbert]: Games with winning conditions that are combinations of parity and explosion conditions can be solved in EXPTIME.

Thm[Serre]: For every finite level of the Borel hierarchy there is a winning condition complete for this level and such that games with this conditions are decidable.

- Deciding levels in Caucal hierarchy.
- Complexity of model checking in Caucal hierarchy of weaker logics (that cannot express alternating reachability).
- The power of panic give more than Caucal hierarchy?
- Good class of conditions for pushdown games.
- Quality of strategies in pushdown games.