Unsafe grammars and panic automata

Teodor Knapik

Université de la Nouvelle Calédonie

Damian Niwiński and Paweł Urzyczyn

Warsaw University

Igor Walukiewicz

Université Bordeaux I

ICALP 2005 — GAMES 2005

Recursive program schemes

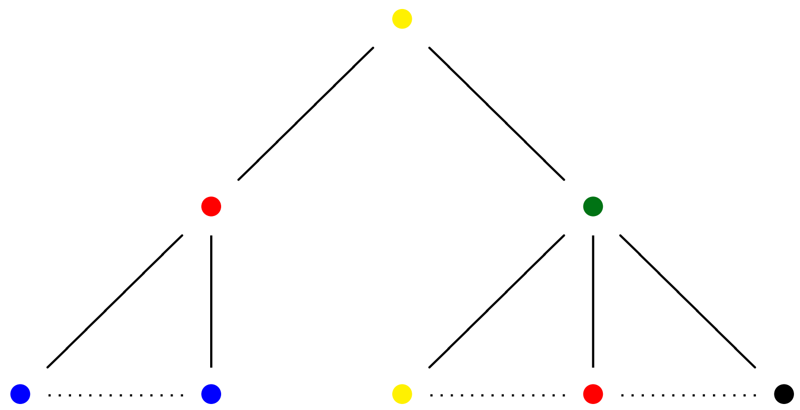
$x + y$

$x \cdot y = x + \underbrace{x + x + \ldots + x}_{y-1}$

$x^y = x \cdot \underbrace{x \cdot x \cdot \ldots \cdot x}_{y-1}$

$\ldots\ldots\ldots\ldots\ldots$

$$
\begin{aligned}
A(n,x,y) &= \textbf{if } n = 0 \textbf{ then } x + y \textbf{ else } Iter(A(n-1,x,?),y,x) \\
Iter(\varphi,m,z) &= \textbf{if } m = 1 \textbf{ then } z \textbf{ else } \varphi(Iter((\varphi,m-1,z)))
\end{aligned}
$$



Initial semantics of recursive schemes is given by infinite terms.

Questions about expressiveness.

Engelfriet, Schmidt, Damm, Arnold, Nivat, . . . 1970–1980.

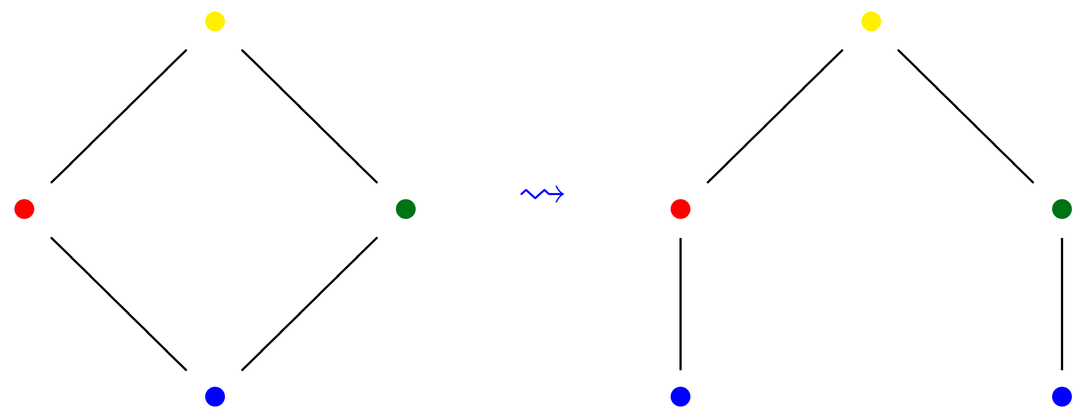
Revival of interest in context of model checking of infinite-state systems.

Questions about complexity.

Courcelle, Hungar, Caucal, . . . 1990–2000.

$\mathcal{M} \models \varphi$? Graphs *vs* trees, μ-calculus *vs* monadic logic

- The $L\mu$ theories of a graph and its unfolding are the same.
- $L\mu$ captures the bisimulation–invariant fragment of MSO logic.
- Over trees, $L\mu$ is equivalent to MSO in the expressive power, but better tractable algorithmically.



Trees with decidable (MSO, $L\mu$) theories

Rabin 1969 : regular trees.

Courcelle 1995 : algebraic trees.

KNU 2001 : trees generated by save grammars of level 2.

KNU 2002 : trees generated by save grammars of level n,
or, equivalently, trees recognized by higher-order pushdown automata of level n.

Aehlig, de Miranda and Ong 2005 : trees generated by all grammars of level 2.

Independently: this paper.

Additionally, the 2-EXPTIME-completeness of the μ-calculus model checking.

Trees as transition systems (Kripke structures)

A tree (term) over signature Σ is $t : Dom\, t \to \Sigma$,

with $Dom\, t \subseteq \omega^*$.

$$\mathbf{t} = \langle Dom\, t,\ \{p_f^{\mathbf{t}} : f \in \Sigma\} \cup \{succ_i^{\mathbf{t}} : 1 \leq i \leq m_\Sigma\}\rangle$$

with $p_f^{\mathbf{t}} = \{w \in Dom\, t :\ t(w) = f\}$, for $f \in \Sigma$, and
$succ_i^{\mathbf{t}} = \{(w, wi) : wi \in Dom\, t\}$, for $1 \leq i \leq m_\Sigma$.

Monadic second-order formulas:

$p_f(x),\ succ_i(x, y),\ x = y,\ x \in X$

$\varphi \vee \psi,\ \neg\varphi, \exists x\varphi,\ \exists X\varphi$.

$L\mu$ formulas:

$p_f,\ X,\ \langle succ_i\rangle\, \alpha,\ [succ_i]\alpha, \alpha \wedge \beta,\ \alpha \vee \beta,\ \mu X.\alpha,\ \nu X.\alpha$.

Tree grammars

Types $\mathcal{T}$ $\quad \tau ::= \mathbf{0} \mid \tau \to \tau$

Nonterminals $N = \{N_\tau\}_{\tau \in \mathcal{T}}$

Variables $\mathcal{X} = \{\mathcal{X}_\tau\}_{\tau \in \mathcal{T}}$

Signature constants $f, g, c, \ldots : \mathbf{0}^k \to \mathbf{0}$

Grammar $\mathcal{G} = (\Sigma, V, S, E)$

with Σ a *signature*, $V \subseteq \bigcup_{\tau \in \mathcal{T}} N_\tau$, $\quad V \ni S : \mathbf{0}$,

and E a finite set of *productions* of the form

$$\mathcal{F} z_1 \ldots z_m \quad \Rightarrow \quad w$$

with $V \ni \mathcal{F} : \tau_1 \to \tau_2 \cdots \to \tau_m \to \mathbf{0}$, $\quad z_i \in \mathcal{X}_{\tau_i}$,

and w an applicative term over $\Sigma \cup V \cup \{z_1 \ldots z_m\}$ of type $\mathbf{0}$.

Reductions

We assume that a grammar $\mathcal{G}$ is deterministic,
i.e., one production per nonterminal.

Hence there is a unique outermost reduction

$$S = t_0 \rightarrow_{\mathcal{G}} t_1 \rightarrow_{\mathcal{G}} t_2 \rightarrow_{\mathcal{G}} \ldots$$

producing the tree $[\![\mathcal{G}]\!]$ generated by $\mathcal{G}$.

Levels

$\ell(\mathbf{0}) = 0$, $\qquad \ell(\tau_1 \rightarrow \tau_2) = \max(1 + \ell(\tau_1), \ell(\tau_2))$

We consider grammars with nonterminals of level at most 2.

Example: Ackermann revisited

$A : \mathbf{0}^3 \to \mathbf{0}$

$Iter : (\mathbf{0} \to \mathbf{0}) \to \mathbf{0} \to \mathbf{0} \to \mathbf{0}$

$S : \mathbf{0}$

$$
\begin{aligned}
A\,nxy &\Rightarrow Cond(Zero\,n)(Plus\,xy)(Iter\,(A\,(Pred\,n)x)yx) \\
Iter\,\varphi mz &\Rightarrow Cond\,(One\,m)z(\varphi(Iter\,\varphi(Pred\,m)z) \\
S &\Rightarrow A\,bcd
\end{aligned}
$$

Model checking

Given a grammar $\mathcal{G}$ and a property φ. Does $[\![\mathcal{G}]\!] \models \varphi$?

For MSO (even for FSO), the problem is non-elementary already for regular tree grammars.

For the μ-calculus, it is EXPTIME-complete for algebraic grammars (W. 1996), and n-EXPTIME-complete for *safe* grammars of level n (Cachat and W. 2004).

For regular grammars, the complexity is still open!

Here we will remove the safety assumption for level 2.

We use an equivalent formulation *via* *alternating automata* and *parity games*.

Parity games

$V_\exists$	positions of Eve
$V_\forall$	positions of Adam (disjoint)
$\longrightarrow \subseteq V \times V$	possible moves (with $V = V_\exists \cup V_\forall$)
$p_1 \in V$	initial position
$\Omega : Q \to \omega$	the ranking function.

An infinite play $v_0 \longrightarrow v_1 \longrightarrow v_2 \longrightarrow \ldots$ is won by Eve iff $\limsup_{n\to\infty} \Omega(v_n)$ is even.

Parity games enjoy *positional determinacy*

(Emerson and Jutla 1991, Mostowski 1991).

Alternating automata

$$\mathcal{B} = \langle \Sigma, Q_\exists, Q_\forall, q_1, \delta, \Omega \rangle$$

where $Q_\exists \cup Q_\forall = Q$ is a set of states, and δ is a set of transitions of the form $q \longrightarrow f(q_1, \ldots, q_k)$, with $\Sigma \ni f : \mathbf{0}^k \rightarrow \mathbf{0}$.

For a tree t, Eve and Adam play a suitable parity game.

$\mathcal{B}$ accepts t iff Eve wins the game.

Problem 1. Given a 2nd order grammar $\mathcal{G}$ and an alternating parity tree automaton $\mathcal{B}$. Does $\mathcal{B}$ accept $[\![\mathcal{G}]\!]$?

When the grammar is safe (KNU 2001, 2002)

A term of level $k > 0$ is *unsafe* if it contains an occurrence of a parameter of level strictly less than k.

An *occurrence* of an unsafe term t is *unsafe*, unless it is in the context $\ldots (ts) \ldots$

A grammar is *safe* if no unsafe occurrence of an unsafe term appears.

Example:

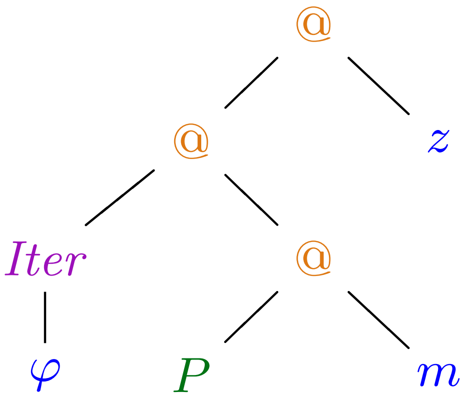
$$A\, n x y \quad \Rightarrow \quad Cond(Zero\, n)(Plus\, xy)(Iter\, (A\, (Pred\, n) x) y x)$$

The method for safe grammars: reduction of $\mathcal{G}$ of level n to $\mathcal{G}^{\alpha}$ of level $n-1$.

Why is safety a problem ?

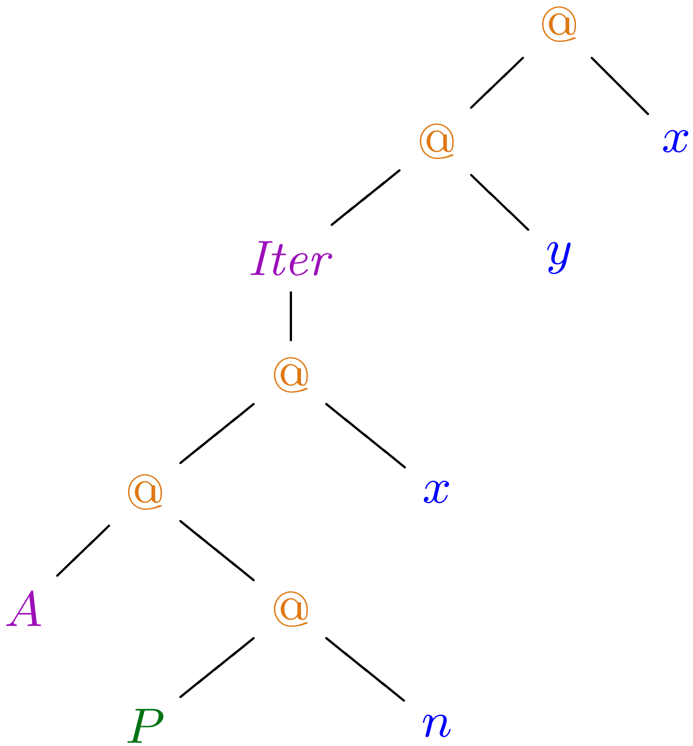
Safe

$Iter\,\varphi(Pm)z$



Unsafe

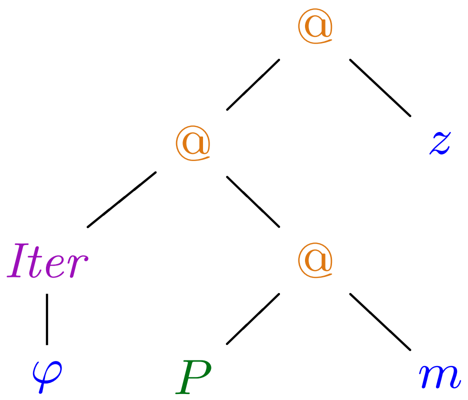
$Iter(A(Pn)x)yx$



Why is safety a problem ?

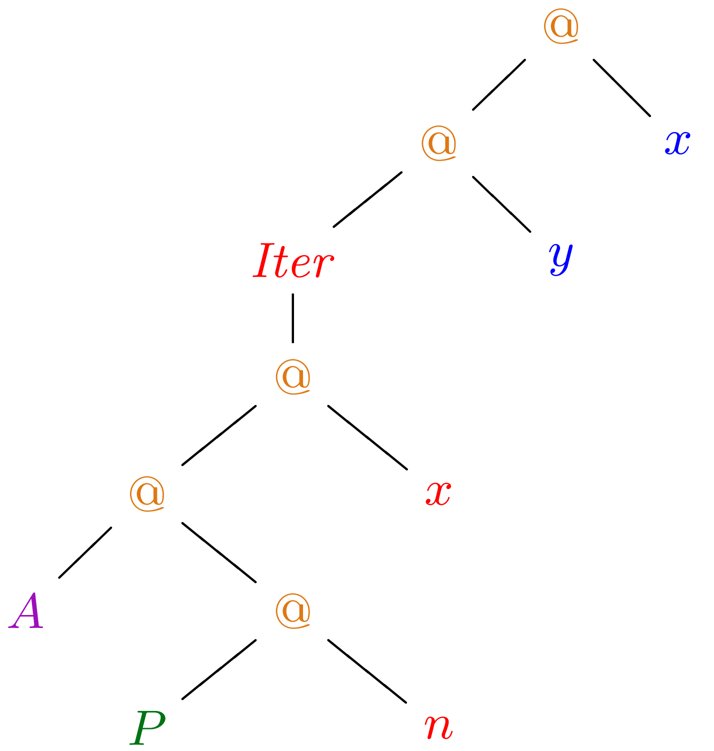
Safe

$Iter\,\varphi(Pm)z$



Unsafe

$Iter(A(Pn)x)yx$



When the grammar is unsafe. . .

. . . we use panic automata

Second-order pushdown stores

A *level* 1 *pushdown store* is a non-empty word $a_1 \ldots a_k$ over Γ.

A *level* 2 *pds* is a non-empty sequence of 1-pds' $[s_1][s_2]\ldots[s_l]$.

Operations :

$$\begin{aligned} push_1\langle a\rangle([s_1][s_2]\ldots[s_l][w]) &= [s_1][s_2]\ldots[s_l][wa] \\ pop_1(\alpha[w\xi]) &= \alpha[w] \\ push_2(\alpha[w]) &= \alpha[w][w] \\ pop_2(\alpha[v][w]) &= \alpha[v] \end{aligned}$$

Second-order pushdown stores with time stamps

A *level* 1 *pushdown store* is a non-empty word $a_1 \dots a_k$ over $\Gamma \times \omega$.

A *level* 2 *pds* is a non-empty sequence of 1-pds' $[s_1][s_2]\dots[s_l]$.

Operations (Op_2) :

$$\begin{aligned}
push_1\langle a\rangle([s_1][s_2]\dots[s_l][w]) &= [s_1][s_2]\dots[s_l][w(a,l)] \\
pop_1(\alpha[w\xi]) &= \alpha[w] \\
push_2(\alpha[w]) &= \alpha[w][w] \\
pop_2(\alpha[v][w]) &= \alpha[v] \\
panic([s_1][s_2]\dots[s_m]\dots[s_l][w(a,m)]) &= [s_1][s_2]\dots[s_m]
\end{aligned}$$

[⊥] $push_1\langle a\rangle$

[⊥ a]

[⊥ a b] $push_2$

[⊥ a b] [⊥ a b] pop_1

[⊥ a b] [⊥ a]

[⊥ a b] [⊥ a a]

[⊥ a b] [⊥ a a] [⊥ a a]

[⊥ a b] [⊥ a a] [⊥ a a] [⊥ a a]

[⊥ a b] [⊥ a a] [⊥ a a] [⊥ a a b]

[⊥ a b] [⊥ a a] [⊥ a a] [⊥ a a] **panic!**

[⊥ a b]

($\perp$,0)			
($\perp$,0) (a,0)			
($\perp$,0) (a,0)			
($\perp$,0) (a,0) (b,0)			
($\perp$,0) (a,0) (b,0)	($\perp$,0) (a,0) (b,0)		
($\perp$,0) (a,0) (b,0)	($\perp$,0) (a,0)		
($\perp$,0) (a,0) (b,0)	($\perp$,0) (a,0) (a,1)		
($\perp$,0) (a,0) (b,0)	($\perp$,0) (a,0) (a,1)	($\perp$,0) (a,0) (a,1)	
($\perp$,0) (a,0) (b,0)	($\perp$,0) (a,0) (a,1)	($\perp$,0) (a,0) (a,1)	($\perp$,0) (a,0) (a,1)
($\perp$,0) (a,0) (b,0)	($\perp$,0) (a,0) (a,1)	($\perp$,0) (a,0) (a,1)	($\perp$,0) (a,0) (a,1) (b,3)
($\perp$,0) (a,0) (b,0)	($\perp$,0) (a,0) (a,1)	($\perp$,0) (a,0) (a,1)	($\perp$,0) (a,0) (a,1) !
($\perp$,0) (a,0) (b,0)			

Panic automata *vs* hyperalgebraic grammars

There are polynomial-time translations:

grammar $\mathcal{G}$	$\mapsto$	automaton $\mathcal{A}_{\mathcal{G}}$
automaton $\mathcal{A}$	$\mapsto$	grammar $\mathcal{G}_{\mathcal{A}}$

such that

- $\mathcal{A}_{\mathcal{G}}$ recognizes the tree generated by $\mathcal{G}$,
- $\mathcal{G}_{\mathcal{A}}$ generates the tree recognized by $\mathcal{A}$.

Simulation of grammar by automaton

Pushdown symbols: subterms of the grammar.

The top symbol of 2-pds "approximates" an expression in the derivation.

The whole content of 2-pds represents the environment, where this approximation is evaluated.

. . . .

| ......A b c |

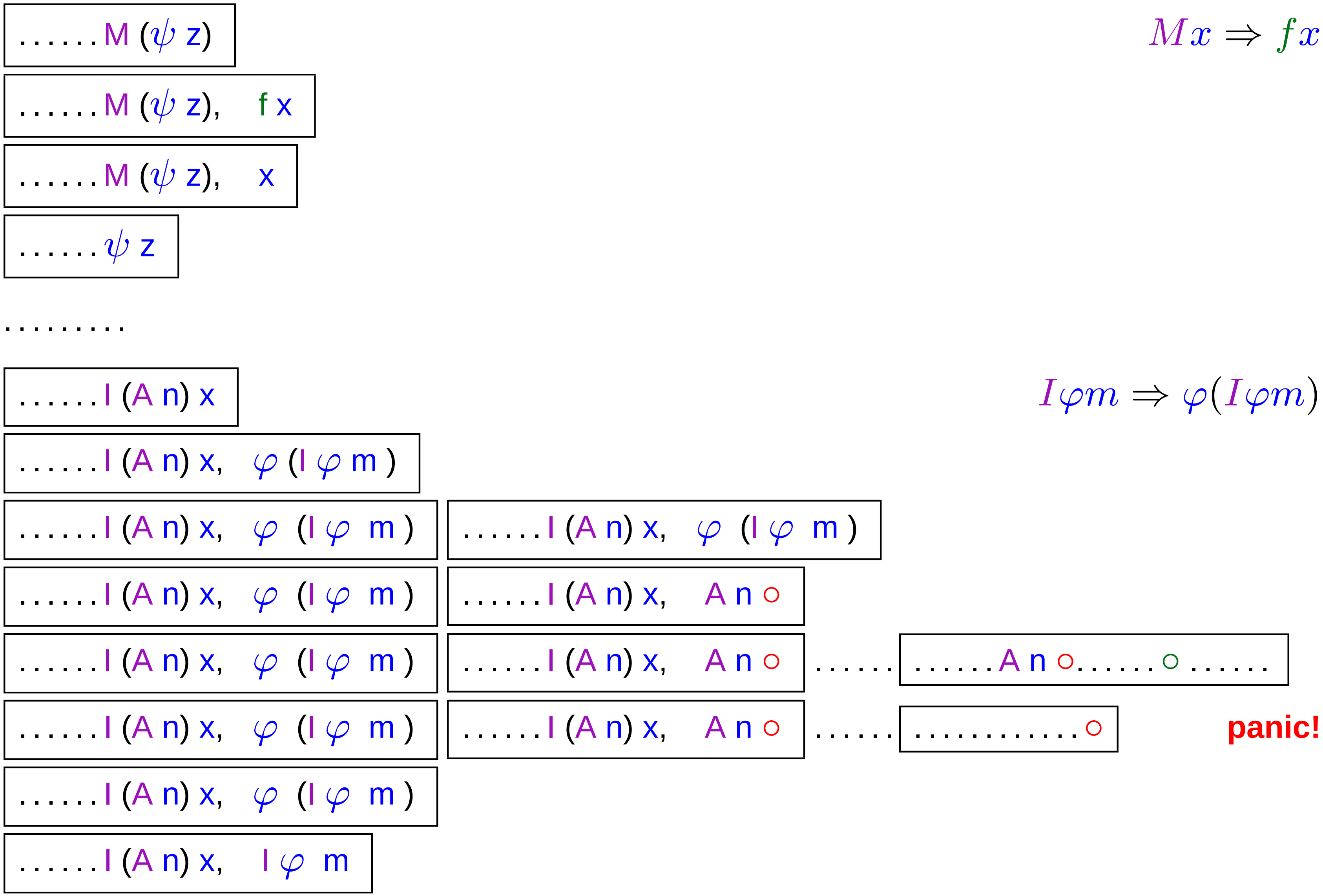
$$A n x \Rightarrow I(A n) x \quad \rightsquigarrow \quad push_1 \langle I(A n) x \rangle$$

| ......A b c, I (A n) x |

. . . .

$$Mx \Rightarrow fx$$

$$I\varphi m \Rightarrow \varphi(I\varphi m)$$



Let's play a parity game on the configuration graph of a panic automaton.

Second-order pushdown systems with panic

A system $\mathcal{C} = (P, P_\exists, P_\forall, \Gamma, p_1, \Delta, \Omega)$ consists of

$P = P_\exists \dot\cup P_\forall$	finite set of control locations
$p_1 \in P$	initial location
$\Delta \subseteq P \times \Gamma \times P \times Op_2$	transition rules
$\Omega : P \to \omega$	rank function

A *configuration* is (p, s), where $p \in P$ and s a 2-pds.

We define a **parity game** $Game(\mathcal{C})$, with initial position $(p_1, [(\bot, 0)])$, by

$$(p, s) \longrightarrow (p', I(s))$$

whenever $p, top(s) \to_\Delta p', I$.

Problem 1. Given a 2nd order grammar $\mathcal{G}$, and an alternating parity tree automaton $\mathcal{B}$, decide if $\mathcal{B}$ accepts $[\![\mathcal{G}]\!]$.

$$\Updownarrow$$

Problem 2. Given a second-order pushdown systems with panic $\mathcal{C}$, decide if Eve wins $Game(\mathcal{C})$.

$$\mathcal{C} \approx \mathcal{G} \times \mathcal{B}$$

Making pushdown systems rank-aware

$$(p, [s_1] \ldots [s_m]) \longrightarrow \cdots\cdots\cdots \longrightarrow (q, [s_1] \ldots [s_m] \ldots [w(a, m)])$$

$panic$

We can force a to "remember" the highest rank on the path, say $Rank(a)$.

Deciding the winner in $Game(\mathcal{C})$.

W transform $\mathcal{C}$ to a second-order pushdown systems without panic $\mathcal{C}'$, i.e.,

$$\Delta' \subseteq P' \times \Gamma' \times P' \times (Op_2 - \{panic\}),$$

such that Eve wins $Game(\mathcal{C}) \Longleftrightarrow$ Eve wins $Game(\mathcal{C}')$.

For the latter games, the problem is 2-EXPTIME-complete (Cachat and W. 2004).

Hint : *Panic in advance!*

Construction of $\mathcal{C}'$

The set of "happy returns" is defined by

$$Ret = P \stackrel{.}{\to} \{0, 1, \ldots, d\}$$

We let

$$\Gamma' = Ret \cup (\Gamma \times Ret)$$

If $\mathcal{C}: \quad q, a \quad \to_\Delta \quad q', panic,$ we let

$$\begin{array}{llll} \mathcal{C}': & q, (a, R) & \to_{\Delta'} & \top \quad \text{if } Rank(a) \succcurlyeq R(q') \\ & q, (a, R) & \to_{\Delta'} & \bot \quad \text{otherwise.} \end{array}$$

where $\ldots 5 \preccurlyeq 3 \preccurlyeq 1 \preccurlyeq 0 \preccurlyeq 2 \preccurlyeq 4 \preccurlyeq \ldots$

Simulation of $\mathcal{C}$ by $\mathcal{C}'$

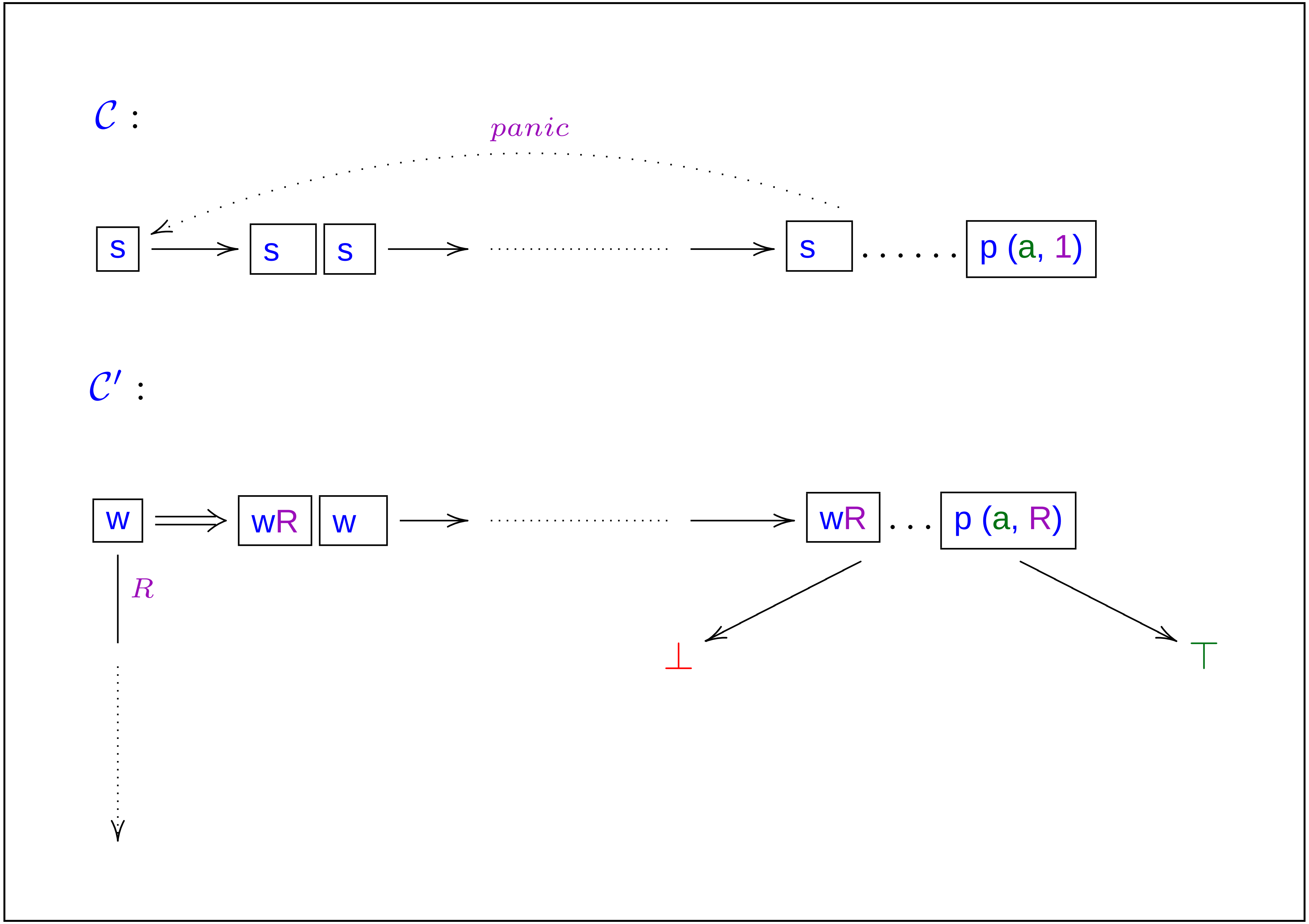
A pds	$\boxed{w_1 R_1} \ldots \boxed{w_k R_k} \boxed{w_{k+1}}$	of $\mathcal{C}'$
represents	$\boxed{s_1} \ldots\ldots\ldots\ldots \boxed{s_k} \boxed{s_{k+1}}$	of $\mathcal{C}$.

For example, $w_5 R_5$

$$\boxed{(a, R_1)\ (b, R_2)\ (c, R_2)\ (b, R_3)\ R_5}$$

represents s_5

$$\boxed{(a,\ 1)\ (b,\ 2)\ (c,\ 2)\ (b,\ 3)}$$


C :
panic
s
s
s
s
p (a, 1)
C′ :
w
wR
w
wR
p (a, R)
R
⊥
⊤

Conclusion

The problem $[\![\mathcal{G}]\!] \models \varphi$?, where $[\![\mathcal{G}]\!]$ is a grammar of level 2 (possibly unsafe), and φ a formula of the μ-calculus, is 2-EXPTIME-complete.

Open problems

Does the result generalize to level n ?

Are there grammars that are intrinsically unsafe ?

In other words, is panic inevitable ?