On the Complexity of VAS Reachability

Sylvain Schmitz

LSV, ENS Paris-Saclay & CNRS, Université Paris-Saclay

INFINITY 2018

Outline

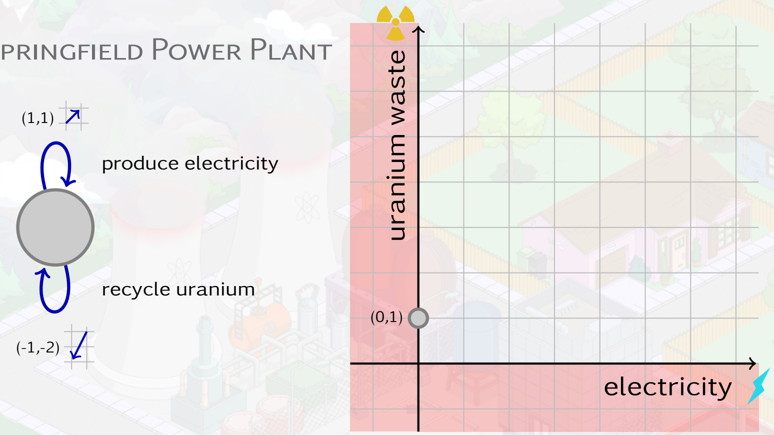
- VASS Reachability
- Decomposition Algorithm
- Upper Bounds
- Complexity

Vector Addition Systems

Vector Addition Systems

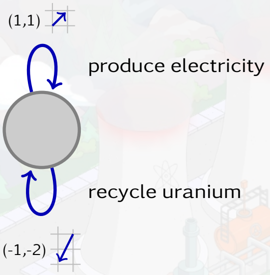
Vector Addition Systems

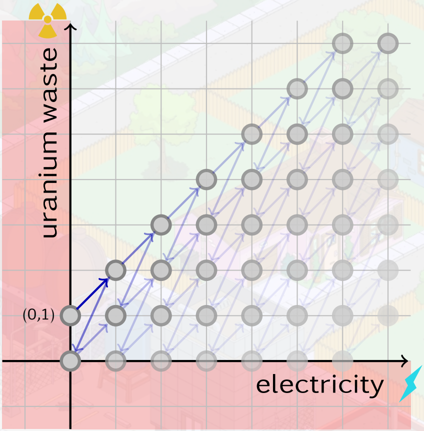
Springfield Power Plant



Vector Addition Systems

Springfield Power Plant

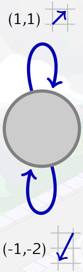


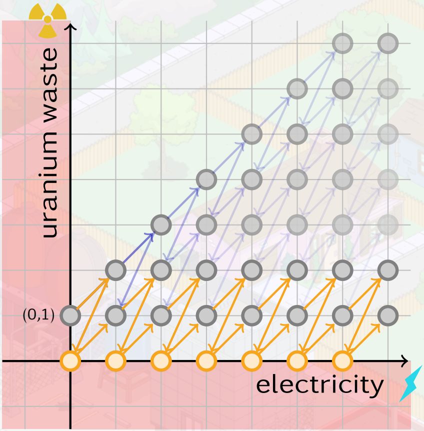


Can we produce unbounded electricity with no left-over uranium waste?

Vector Addition Systems

Springfield Power Plant





Can we produce unbounded electricity with no left-over uranium waste? Yes, $(\infty,0)$ is reachable

Importance of the Problem

Reachability Problem

input: *a vector addition system and two configurations* **source** *and* **target**

question: $\mathbf{source} \to^* \mathbf{target}$?

Discrete Resources

- modelling: items, money, energy, molecules, ...
- distributed computing: active threads in thread pool
- data: isomorphism types in data logics and data-centric systems

Importance of the Problem

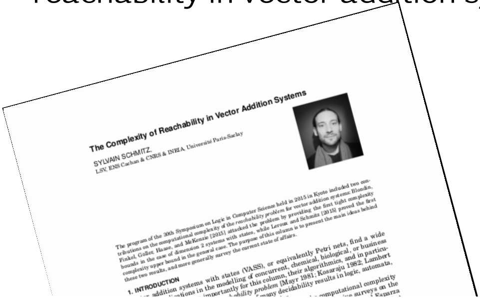
Reachability Problem

input: *a vector addition system and two configurations* **source** *and* **target**

question: **source** $\to^*$ **target***?*

Central Decision Problem [invited survey S., SIGLOG'16]

Large number of problems interreducible with reachability in vector addition systems



The Complexity of Reachability in Vector Addition Systems

SYLVAIN SCHMITZ,
LSV, ENS Cachan & CNRS & INRIA, Université Paris-Saclay

The program of the 30th Symposium on Logic in Computer Science held in 2015 in Kyoto included two contributions on the computational complexity of the *reachability problem* for vector addition systems: Blondin, Finkel, Goller, Haase, and McKenzie [2015] attacked the problem by providing the first tight complexity bounds in the case of dimension 2 systems with states, while Leroux and Schmitz [2015] proved the first complexity upper bound in the general case. The purpose of this column is to present the main ideas behind these two results, and more generally survey the current state of affairs.

1. INTRODUCTION

... addition systems with states (VASS), or equivalently Petri nets, find a wide ... in the modelling of concurrent, chemical, biological, or business ... importantly for this column, their algorithmics, and in particu... *reachability problem* [Mayr 1981; Kosaraju 1982; Lambert ... many decidability results in logic, automata, ... computational complexity ... surveys on the ...

Importance of the Problem

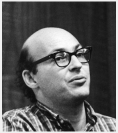
Reachability Problem

input: *a vector addition system and two configurations* **source** *and* **target**

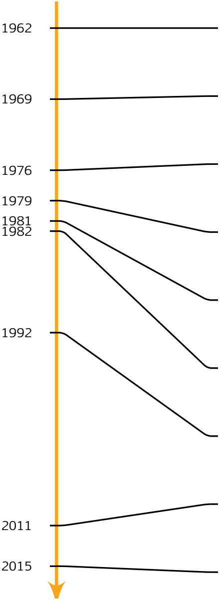
question: **source** $\rightarrow^*$ **target**?

Theorem (Minsky'67)

Reachability is undecidable in 2-dimensional Minsky machines (vector addition systems with zero tests).



Importance of the Problem




C. A. Petri: Petri nets

 
R. M. Karp & R. E. Miller: coverability trees


R. J. Lipton: EXPSPACE lower bound

 
J. E. Hopcroft & J.-J. Pansiot: dim. $\geqslant 3$ not definable in Presburger arithmetic


E. W. Mayr: decidability by decomposition


S. R. Kosaraju: decidability by decomposition


J.-L. Lambert: decidability by decomposition


J. Leroux: decidability by Presburger inductive invariants

this talk: Leroux & S.'15

Demystifying Reachability in Vector Addition Systems

[Leroux & S.'15]

Ideal Decomposition Theorem

The Decomposition Algorithm computes the ideal decomposition of the set of runs from source to target.

Upper Bound Theorem

Reachability in vector addition systems is in cubic Ackermann.

Demystifying Reachability in Vector Addition Systems

[Leroux & S.'15]

Ideal Decomposition Theorem

The Decomposition Algorithm computes the ideal decomposition of the set of runs from source to target.

Upper Bound Theorem

Reachability in vector addition systems is in cubic Ackermann.

Demystifying Reachability in Vector Addition Systems

[Leroux & S.'15; S.'17]

Ideal Decomposition Theorem

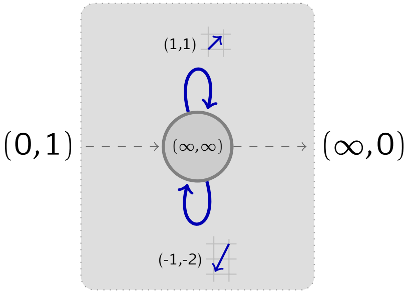
The Decomposition Algorithm computes the ideal decomposition of the set of runs from source to target.

Upper Bound Theorem

Reachability in vector addition systems is in quadratic Ackermann.

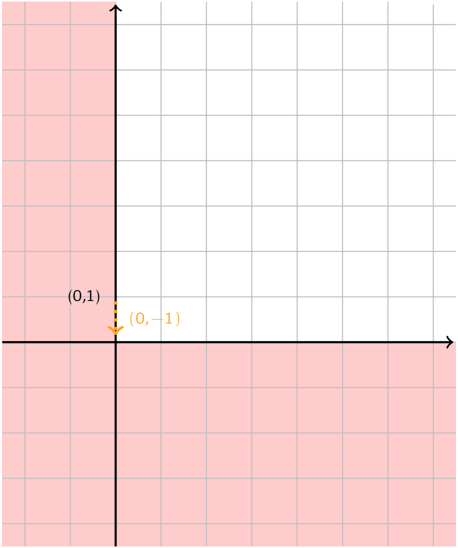
"Simple Runs" (Θ Condition)

[Mayr'81, Kosaraju'82, Lambert'92]



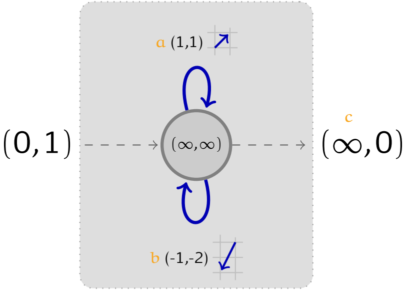
"Simple Runs" (Θ Condition)

[Mayr'81, Kosaraju'82, Lambert'92]



"Simple Runs" (Θ Condition)

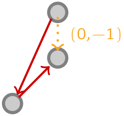
[Mayr'81, Kosaraju'82, Lambert'92]



Characteristic System

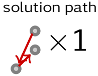
$$0 + 1 \cdot a - 1 \cdot b = c$$
$$1 + 1 \cdot a - 2 \cdot b = 0$$

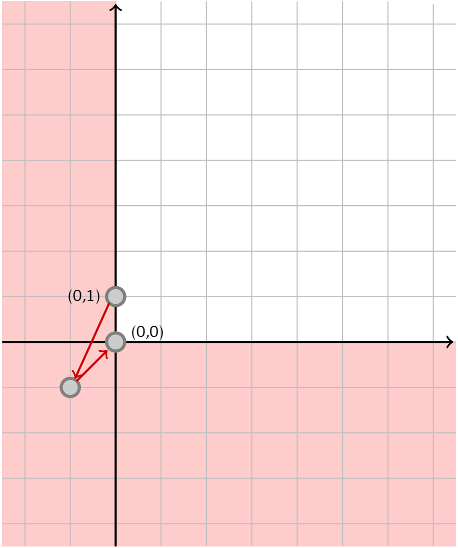
Solution Path



"Simple Runs" (Θ Condition)

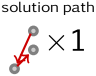
[Mayr'81, Kosaraju'82, Lambert'92]

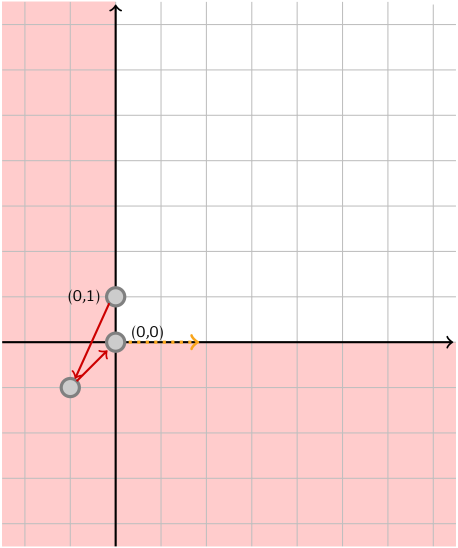




"Simple Runs" (Θ Condition)

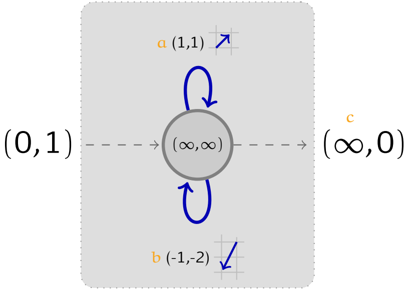
[Mayr'81, Kosaraju'82, Lambert'92]





"Simple Runs" (Θ Condition)

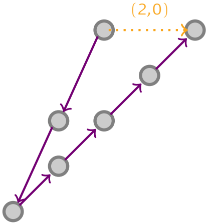
[Mayr'81, Kosaraju'82, Lambert'92]



Homogeneous System

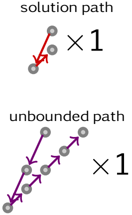
$$1 \cdot a - 1 \cdot b = c$$
$$1 \cdot a - 2 \cdot b = 0$$
$$a, b, c > 0$$

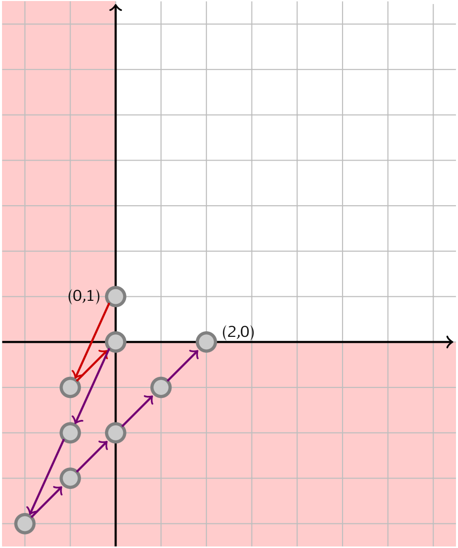
Unbounded Path



"Simple Runs" (Θ Condition)

[Mayr'81, Kosaraju'82, Lambert'92]





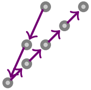
"Simple Runs" (Θ Condition)

[Mayr'81, Kosaraju'82, Lambert'92]

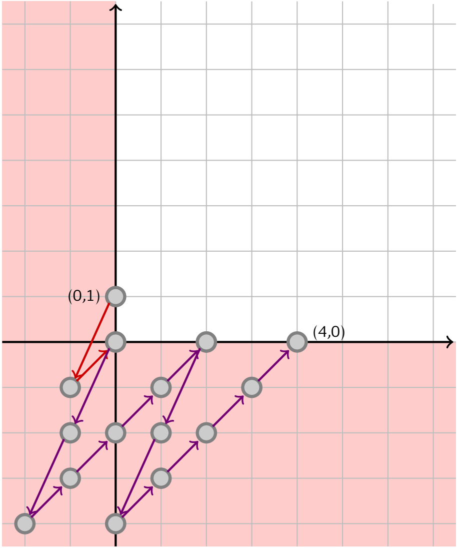
solution path

×1

unbounded path



×2



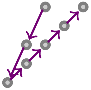
“Simple Runs” (Θ Condition)

[Mayr’81, Kosaraju’82, Lambert’92]

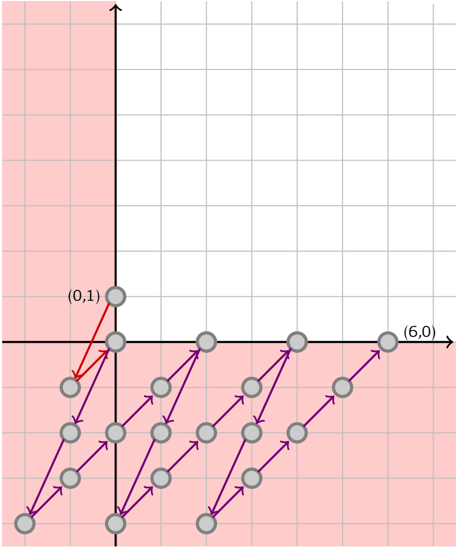
solution path

$\times 1$

unbounded path



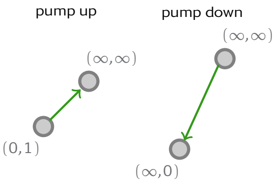
$\times 3$



"Simple Runs" (Θ Condition)

[Mayr'81, Kosaraju'82, Lambert'92]

Pumpable Paths

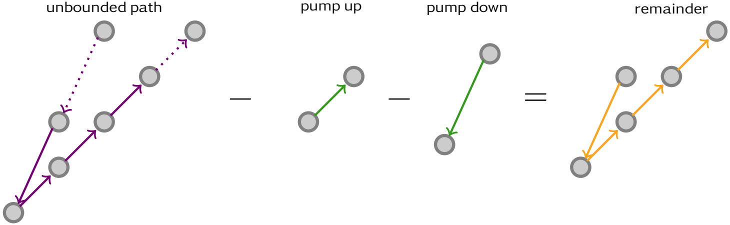


uses *coverability trees* [Karp & Miller'69] (Jérôme's talk)
which relies on *Dickson's Lemma* [Dickson, 1913]

"Simple Runs" (Θ Condition)

[Mayr'81, Kosaraju'82, Lambert'92]

Pumpable Paths

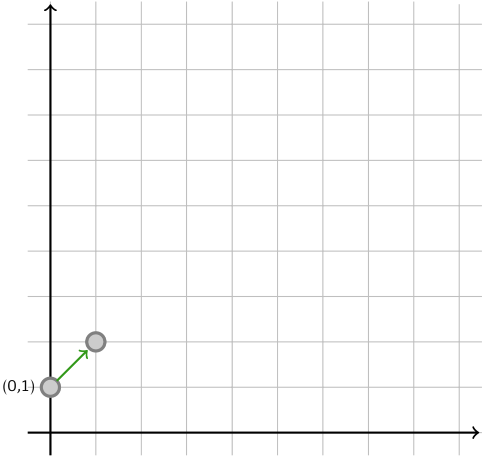


"Simple Runs" (Θ Condition)

[Mayr'81, Kosaraju'82, Lambert'92]

pump up

$\times 1$

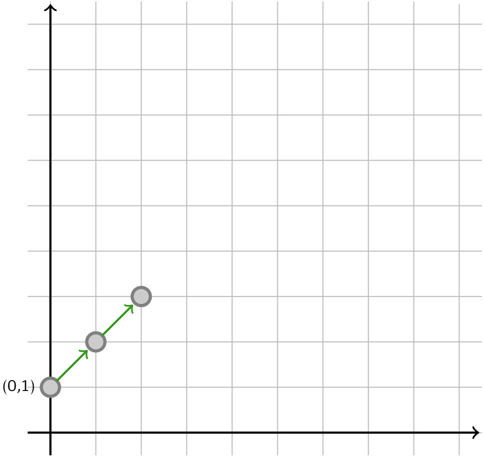


"Simple Runs" (Θ Condition)

[Mayr'81, Kosaraju'82, Lambert'92]

pump up

$\times 2$



"Simple Runs" (Θ Condition)

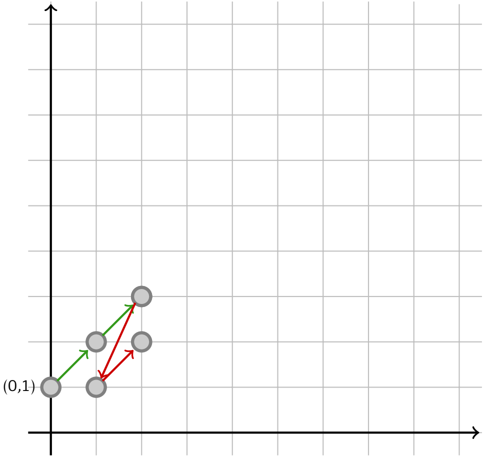
[Mayr'81, Kosaraju'82, Lambert'92]

pump up

$\times 2$

solution path

$\times 1$



"Simple Runs" (Θ Condition)

[Mayr'81, Kosaraju'82, Lambert'92]

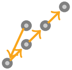
pump up

$\times 2$

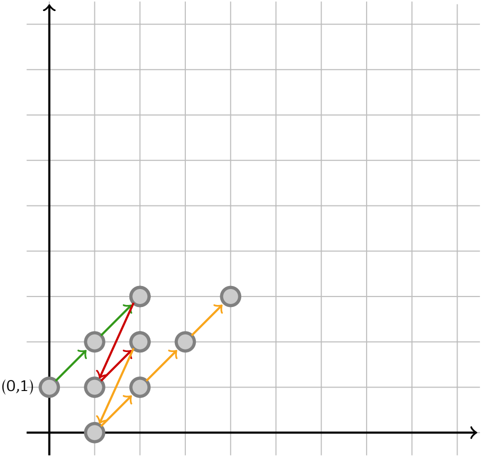
solution path

$\times 1$

remainder



$\times 1$



"Simple Runs" (Θ Condition)

[Mayr'81, Kosaraju'82, Lambert'92]

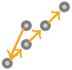
pump up

$\times 2$

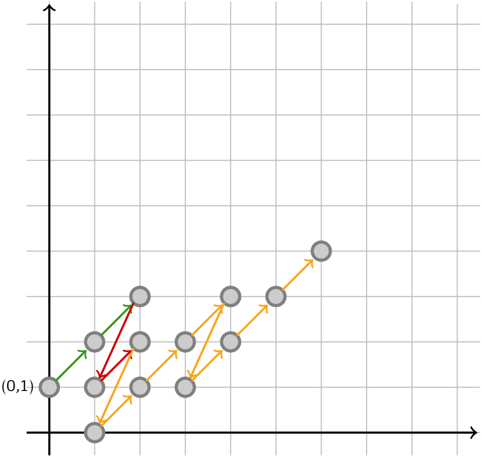
solution path

$\times 1$

remainder

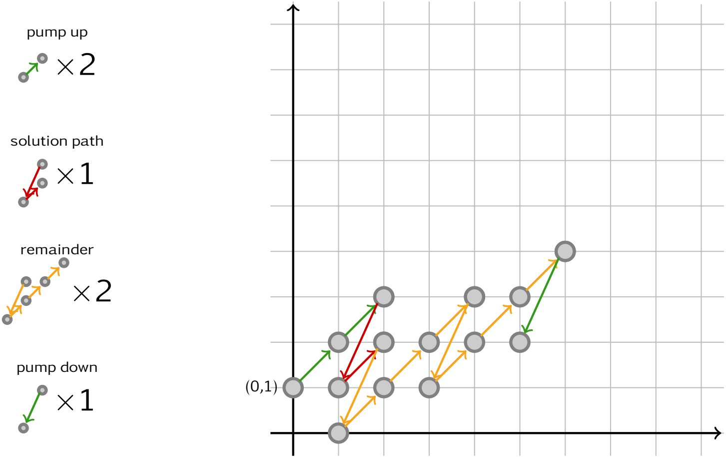


$\times 2$



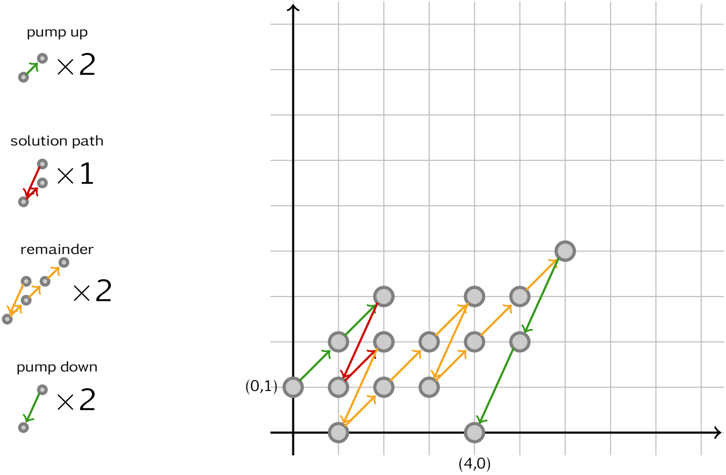
“Simple Runs” (Θ Condition)

[Mayr’81, Kosaraju’82, Lambert’92]



"Simple Runs" (Θ Condition)

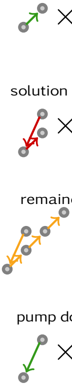
[Mayr'81, Kosaraju'82, Lambert'92]



“Simple Runs” (Θ Condition)

[Mayr’81, Kosaraju’82, Lambert’92]

pump up



×3

solution path

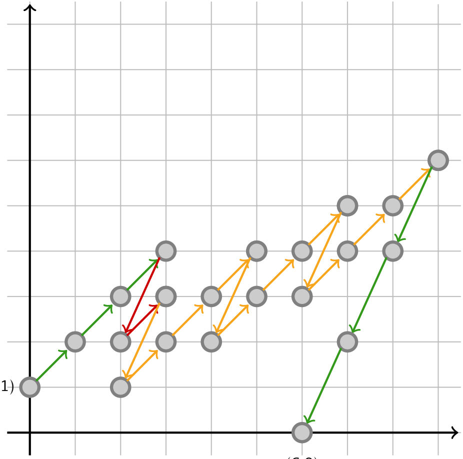
×1

remainder

×3

pump down

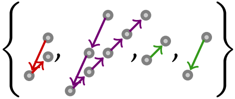
×3



Decomposition Algorithm

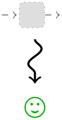
[Mayr'81, Kosaraju'82, Lambert'92]

can we build a "simple run"?

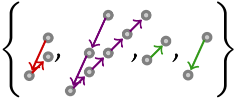


Decomposition Algorithm

[Mayr'81, Kosaraju'82, Lambert'92]



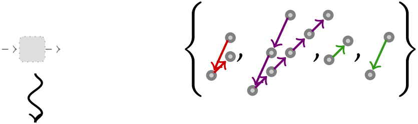
can we build a "simple run"? yes



Decomposition Algorithm

[Mayr'81, Kosaraju'82, Lambert'92]

can we build a "simple run"? no



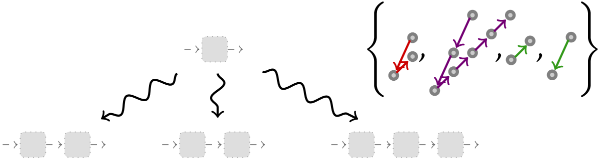
☹

decompose

Decomposition Algorithm

[Mayr'81, Kosaraju'82, Lambert'92]

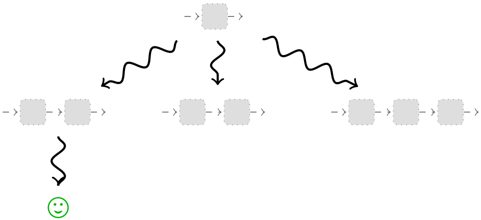
can we build a "simple run"? no



decompose

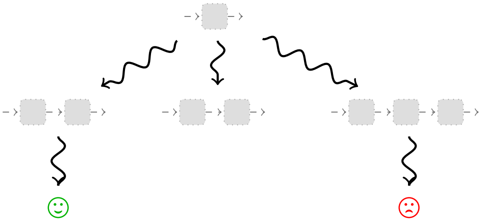
Decomposition Algorithm

[Mayr'81, Kosaraju'82, Lambert'92]



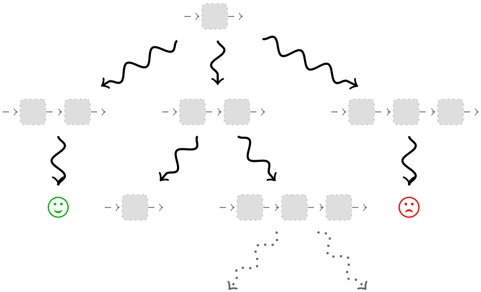
Decomposition Algorithm

[Mayr'81, Kosaraju'82, Lambert'92]



Decomposition Algorithm

[Mayr'81, Kosaraju'82, Lambert'92]



Termination

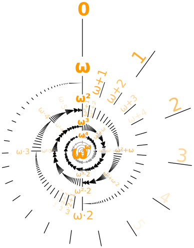
"Finally the checker has to verify that the process comes to an end. Here again he should be assisted by the programmer giving a further definite assertion to be verified. This may take the form of a quantity which is asserted to decrease continually and vanish when the machine stops."

[Turing'49]

Termination

"Finally the checker has to verify that the process comes to an end. Here again he should be assisted by the programmer giving a further definite assertion to be verified. This may take the form of a quantity which is asserted to decrease continually and vanish when the machine stops. To the pure mathematician it is natural to give an ordinal number."

[Turing'49]



Termination of the Decomposition Algorithm

[Mayr'81, Kosaraju'82, Lambert'92]

Ranking Function

$$\omega^{\omega^2}$$

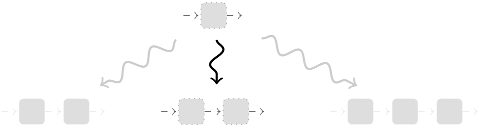
$$\vee$$

$$\alpha_0$$

Termination of the Decomposition Algorithm

[Mayr'81, Kosaraju'82, Lambert'92]

Ranking Function



ω^{ω^2}

$\vee$

α_0

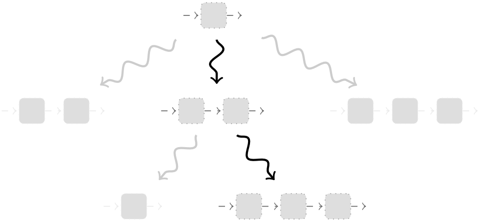
$\vee$

α_1

Termination of the Decomposition Algorithm

[Mayr'81, Kosaraju'82, Lambert'92]

Ranking Function

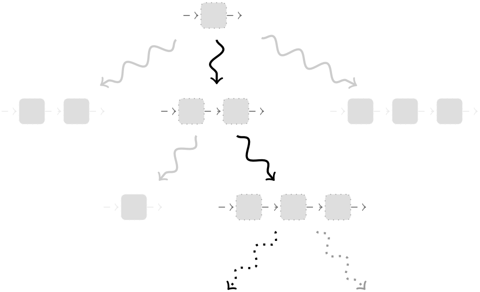


$$\omega^{\omega^2} > \alpha_0 > \alpha_1 > \alpha_2$$

Termination of the Decomposition Algorithm

[Mayr'81, Kosaraju'82, Lambert'92]

Ranking Function



$$\omega^{\omega^2} > \alpha_0 > \alpha_1 > \alpha_2 > \vdots$$

Demystifying Reachability in Vector Addition Systems

[Leroux & S.'15; S.'17]

Ideal Decomposition Theorem

The Decomposition Algorithm computes the ideal decomposition of the set of runs from source to target.

Upper Bound Theorem

Reachability in vector addition systems is in quadratic Ackermann.

Upper Bounds

How to bound the running time of algorithms with ordinal-based termination proofs?

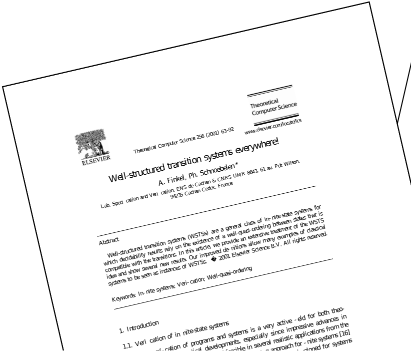
Upper Bounds

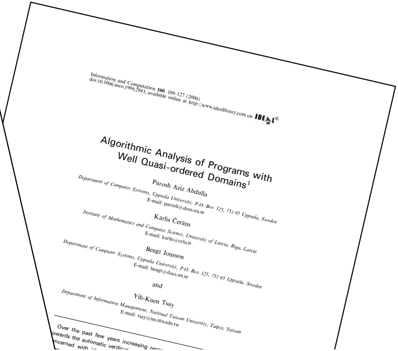
How to bound the running time of algorithms with wqo-based termination proofs?

Upper Bounds

How to bound the running time of algorithms with wqo-based termination proofs?

wqos ubiquitous in infinite-state verification

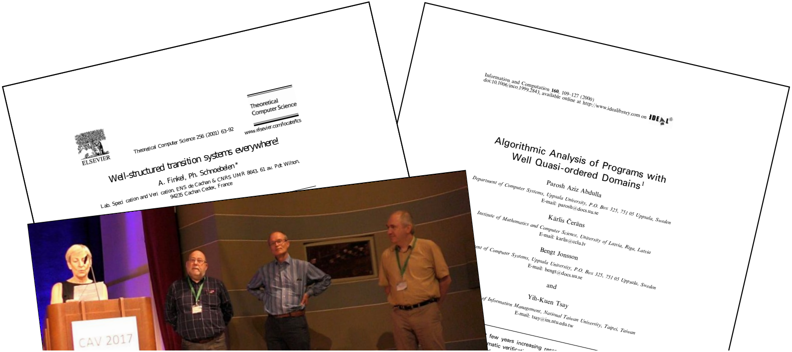




Upper Bounds

How to bound the running time of algorithms with wqo-based termination proofs?

wqos ubiquitous in infinite-state verification



Bad Sequences

Over a qo $(X, \leqslant)$

- $x_0, x_1, \ldots$ is bad if $\forall i < j \,.\, x_i \not\leqslant x_j$
- $(X, \leqslant)$ wqo iff all bad sequences are finite
- but can be of arbitrary length

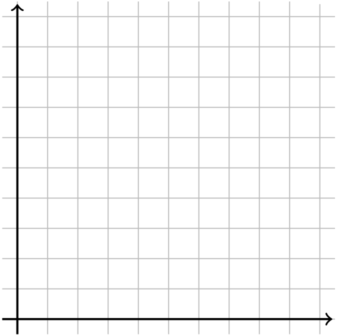
Bad Sequences

Over a qo $(X, \leqslant)$

$x_0, x_1, \ldots$ is bad if $\forall i < j \,.\, x_i \not\leqslant x_j$

$(X, \leqslant)$ wqo iff all bad sequences are finite

but can be of arbitrary length



Example (over $\mathbb{N}^2$)

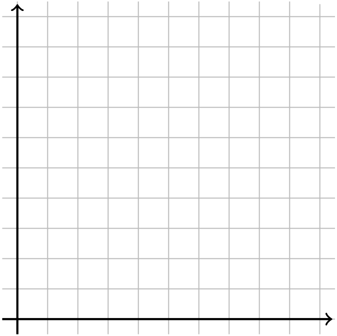
Bad Sequences

Over a qo $(X, \leqslant)$

$x_0, x_1, \ldots$ is bad if $\forall i < j \,.\, x_i \not\leqslant x_j$

$(X, \leqslant)$ wqo iff all bad sequences are finite

but can be of arbitrary length



Example (over $\mathbb{N}^2$)

Controlled Bad Sequences

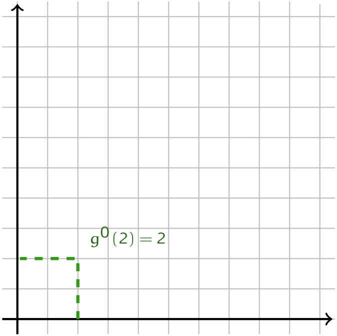
Over a qo $(X, \leqslant)$ with norm $\|\cdot\|$

$x_0, x_1, \ldots$ is bad if $\forall i < j \,.\, x_i \not\leqslant x_j$

$(X, \leqslant)$ wqo iff all bad sequences are finite

controlled by $g : \mathbb{N} \to \mathbb{N}$ monotone and inflationary and $n_0 \in \mathbb{N}$ if $\forall i \,.\, \|x_i\| \leqslant g^i(n_0)$

[Cichoń & Tahhan Bittar'98]



Example (over $\mathbb{N}^2$ with $n_0 = 2$ and $g(n) = n + 1$)

Controlled Bad Sequences

Over a qo $(X,\leqslant)$ with norm $\|\cdot\|$

- $x_0, x_1, \ldots$ is bad if $\forall i < j \,.\, x_i \not\leqslant x_j$
- $(X,\leqslant)$ wqo iff all bad sequences are finite
- controlled by $g\colon \mathbb{N} \to \mathbb{N}$ monotone and inflationary and $n_0 \in \mathbb{N}$ if $\forall i \,.\, \|x_i\| \leqslant g^i(n_0)$

[Cichoń & Tahhan Bittar'98]

Proposition

Over $(X,\leqslant)$, assuming $\forall n\, \{x \in X \mid \|x\| \leqslant n\}$ finite, (g,n_0)-controlled bad sequences have a maximal length, noted $L_{g,X}(n_0)$.

Controlled Bad Sequences

Proposition

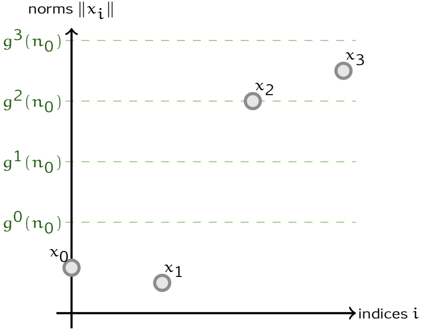
Over a wqo $(X, \leqslant)$, *assuming* $\{x \in X \mid \|x\| \leqslant n\}$ *to be finite* $\forall n$, (g, n_0)*-controlled bad sequences have a maximal length, noted* $L_{g,X}(n_0)$.

Objective

Provide upper bounds for $L_{g,X}(n_0)$.

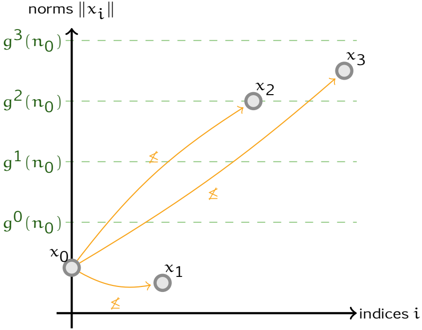
Descent Equation

(g, n_0)-controlled bad sequence $x_0, x_1, x_2, x_3, \ldots$ over a wqo $(X, \leqslant)$:



Descent Equation

(g, n_0)-controlled bad sequence $x_0, x_1, x_2, x_3, \ldots$ over a wqo $(X, \leqslant)$:

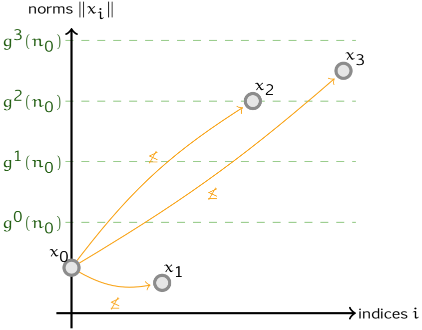


over the suffix $x_1, x_2, x_3, \ldots, \forall i > 0$,

$x_0 \not\leq x_i$

Descent Equation

(g, n_0)-controlled bad sequence $x_0, x_1, x_2, x_3, \ldots$ over a wqo $(X, \leqslant)$:

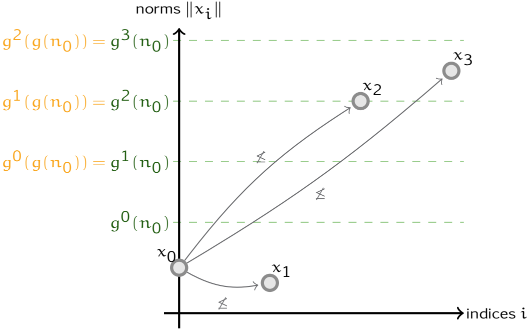


over the suffix $x_1, x_2, x_3, \ldots, \forall i > 0$,

$$x_i \in X \setminus \uparrow x_0 \stackrel{\text{def}}{=} \{x \in X \mid x_0 \not\leqslant x\}$$

Descent Equation

(g, n_0)-controlled bad sequence $x_0, x_1, x_2, x_3, \ldots$ over a wqo $(X, \leqslant)$:



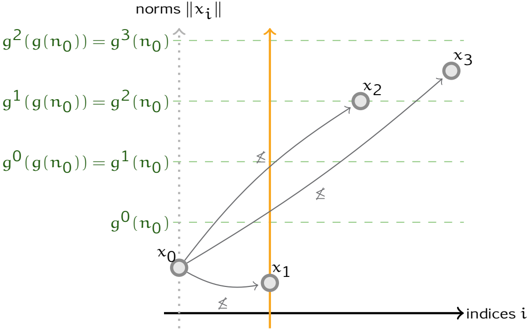
over the suffix $x_1, x_2, x_3, \ldots, \forall i > 0$,

$$x_i \in X \setminus \uparrow x_0 \stackrel{\text{def}}{=} \{x \in X \mid x_0 \not\leq x\}$$

$$\|x_i\| \leqslant g^{i-1}(g(n_0))$$

Descent Equation

(g, n_0)-controlled bad sequence $x_0, x_1, x_2, x_3, \ldots$ over a wqo $(X, \leqslant)$:



over the suffix $x_1, x_2, x_3, \ldots, \forall i > 0$,

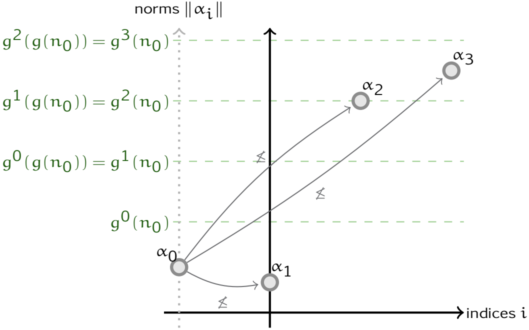
$$x_i \in X \backslash \uparrow x_0 \stackrel{\text{def}}{=} \{x \in X \mid x_0 \not\leqslant x\}$$

$$\|x_i\| \leqslant g^{i-1}(g(n_0))$$

$$L_{g,X}(n_0) = \max_{x_0 \in X, \|x_0\| \leqslant n_0} 1 + L_{g, X \backslash \uparrow x_0}(g(n_0))$$

Descent Equation

(g, n_0)-controlled bad sequence $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \ldots$ over an ordinal α:



over the suffix $\alpha_1, \alpha_2, \alpha_3, \ldots, \forall i > 0$,

$$\alpha_i \in \alpha_0 \stackrel{\text{def}}{=} \{\beta \in \alpha \mid \beta \not\leqslant \alpha_0\}$$

$$\|\alpha_i\| \leqslant g^{i-1}(g(n_0))$$

$$L_{g,\alpha}(n_0) = \max_{\alpha_0 \in \alpha, \|\alpha_0\| \leqslant n_0} 1 + L_{g,\alpha_0}(g(n_0))$$

The Case of Ordinals

[S.'14]

- Cantor Normal Form for ordinals $\alpha < \varepsilon_0$:

$$\alpha = \omega^{\alpha_1} \cdot c_1 + \cdots + \omega^{\alpha_k} \cdot c_k$$
$$\alpha > \alpha_1 > \cdots > \alpha_k \qquad 0 < c_1, \ldots, c_k < \omega$$

- Norm of ordinals $\alpha < \varepsilon_0$: "maximal constant"

$$\|\alpha\| \stackrel{\text{def}}{=} \max_{1 \leqslant i \leqslant k} (\max(\|\alpha_i\|, c_i))$$

e.g. $\|\omega^{\omega^2}\| = 2$, $\|\omega^{\omega \cdot 5} + \omega^2 \cdot 3\| = 5$

The Case of Ordinals

[S.'14]

- Cantor Normal Form for ordinals $\alpha < \varepsilon_0$:

$$\alpha = \omega^{\alpha_1} \cdot c_1 + \cdots + \omega^{\alpha_k} \cdot c_k$$
$$\alpha > \alpha_1 > \cdots > \alpha_k \qquad 0 < c_1, \ldots, c_k < \omega$$

- Norm of ordinals $\alpha < \varepsilon_0$: "maximal constant"

$$\|\alpha\| \stackrel{\text{def}}{=} \max_{1 \leqslant i \leqslant k} \left(\max(\|\alpha_i\|, c_i) \right)$$

e.g. $\|\omega^{\omega^2}\| = 2$, $\|\omega^{\omega \cdot 5} + \omega^2 \cdot 3\| = 5$

The Case of Ordinals

[S.'14]

Recall the descent equation:

$$L_{g,\alpha}(n_0) = \max_{\alpha_0 \in \alpha, \|\alpha_0\| \leqslant n_0} 1 + L_{g,\alpha_0}(g(n_0))$$

Proposition (variant of [Buchholtz, Cichoń & Weiermann'94])

Let $0 < \alpha < \varepsilon_0$ *and* $\|\alpha\| \leqslant n_0$. *Then*

$$L_{g,0}(n_0) = 0 \qquad L_{g,\alpha}(n_0) = 1 + L_{g,P_{n_0}(\alpha)}(g(n_0))$$

$P_x(\alpha)$ denotes the predecessor at x of $\alpha > 0$: "maximal ordinal $\beta < \alpha$ s.t. $\|\beta\| \leqslant x$"

The Case of Ordinals

[S.'14]

Recall the descent equation:

$$L_{g,\alpha}(n_0) = \max_{\alpha_0 \in \alpha, \|\alpha_0\| \leqslant n_0} 1 + L_{g,\alpha_0}(g(n_0))$$

Proposition (variant of [Buchholtz, Cichoń & Weiermann'94])

Let $0 < \alpha < \varepsilon_0$ *and* $\|\alpha\| \leqslant n_0$. *Then*

$$L_{g,0}(n_0) = 0 \qquad L_{g,\alpha}(n_0) = 1 + L_{g,P_{n_0}(\alpha)}(g(n_0))$$

$P_x(\alpha)$ denotes the predecessor at x of $\alpha > 0$: "maximal ordinal $\beta < \alpha$ s.t. $\|\beta\| \leqslant x$"

The Case of Ordinals

[S.'14]

Proposition (variant of [Buchholtz, Cichoń & Weiermann'94])

Let $0 < \alpha < \varepsilon_0$ *and* $\|\alpha\| \leqslant n_0$. *Then*

$$L_{g,0}(n_0) = 0 \qquad L_{g,\alpha}(n_0) = 1 + L_{g,P_{n_0}(\alpha)}(g(n_0))$$

$P_x(\alpha)$ denotes the predecessor at x of $\alpha > 0$: "maximal ordinal $\beta < \alpha$ s.t. $\|\beta\| \leqslant x$"

Example

$$P_3(\omega^2) = \omega \cdot 3 + 3$$

$$\begin{aligned}P_3(\omega^{\omega^2}) = {} & \omega^{\omega\cdot 3+3}\cdot 3 + \omega^{\omega\cdot 3+2}\cdot 3 + \omega^{\omega\cdot 3+1}\cdot 3 + \omega^{\omega\cdot 3}\cdot 3\\ & + \omega^{\omega\cdot 2+3}\cdot 3 + \omega^{\omega\cdot 2+2}\cdot 3 + \omega^{\omega\cdot 2+1}\cdot 3 + \omega^{\omega\cdot 2}\cdot 3\\ & + \omega^{\omega+3}\cdot 3 + \omega^{\omega+2}\cdot 3 + \omega^{\omega+1}\cdot 3 + \omega^{\omega}\cdot 3\\ & + \omega^3\cdot 3 + \omega^2\cdot 3 + \omega\cdot 3 + 3\end{aligned}$$

The Case of Ordinals

[S.'14]

Proposition (variant of [Buchholtz, Cichoń & Weiermann'94])

Let $0 < \alpha < \varepsilon_0$ *and* $\|\alpha\| \leqslant n_0$. *Then*

$$L_{g,0}(n_0) = 0 \qquad L_{g,\alpha}(n_0) = 1 + L_{g,P_{n_0}(\alpha)}(g(n_0))$$

$P_x(\alpha)$ denotes the predecessor at x of $\alpha > 0$: "maximal ordinal $\beta < \alpha$ s.t. $\|\beta\| \leqslant x$"

Example

$$\begin{aligned} P_3(\omega^2) &= \omega \cdot 3 + 3 \\ P_3(\omega^{\omega^2}) &= \omega^{\omega\cdot 3+3} \cdot 3 + \omega^{\omega\cdot 3+2} \cdot 3 + \omega^{\omega\cdot 3+1} \cdot 3 + \omega^{\omega\cdot 3} \cdot 3 \\ &\quad + \omega^{\omega\cdot 2+3} \cdot 3 + \omega^{\omega\cdot 2+2} \cdot 3 + \omega^{\omega\cdot 2+1} \cdot 3 + \omega^{\omega\cdot 2} \cdot 3 \\ &\quad + \omega^{\omega+3} \cdot 3 + \omega^{\omega+2} \cdot 3 + \omega^{\omega+1} \cdot 3 + \omega^{\omega} \cdot 3 \\ &\quad + \omega^3 \cdot 3 + \omega^2 \cdot 3 + \omega \cdot 3 + 3 \end{aligned}$$

The Case of Ordinals

[S.'14]

Proposition (variant of [Buchholtz, Cichoń & Weiermann'94])

Let $0 < \alpha < \varepsilon_0$ *and* $\|\alpha\| \leqslant n_0$*. Then*

$$L_{g,0}(n_0) = 0 \qquad L_{g,\alpha}(n_0) = 1 + L_{g,P_{n_0}(\alpha)}(g(n_0))$$

This function was already known in the literature!

Definition (Cichoń Hierarchy [Cichoń & Tahhan Bittar'98])

For $g : \mathbb{N} \to \mathbb{N}$, define $(g_\alpha : \mathbb{N} \to \mathbb{N})_\alpha$ by

$$g_0(x) \stackrel{\text{def}}{=} 0 \qquad g_\alpha(x) \stackrel{\text{def}}{=} 1 + g_{P_x(\alpha)}(g(x)) \text{ for } \alpha > 0$$

The Case of Ordinals

[S.'14]

Length Function Theorem (for Ordinals)

Let $\alpha < \varepsilon_0$ *and* $n_0 \geqslant \|\alpha\|$. *Then the longest* (g, n_0)*-controlled descending sequence over* α *is of length* $L_{g,\alpha}(n_0) = g_\alpha(n_0)$

Relating Norm and Length

[Cichoń & Tahhan Bittar'98]

Recall the definition of the Cichoń Hierarchy:

$$g_0(x) \stackrel{\text{def}}{=} 0 \qquad g_\alpha(x) \stackrel{\text{def}}{=} 1 + g_{P_x(\alpha)}(g(x)) \text{ for } \alpha > 0$$

Definition (Hardy Hierarchy)

For $g : \mathbb{N} \to \mathbb{N}$, define $(g^\alpha : \mathbb{N} \to \mathbb{N})_\alpha$ by

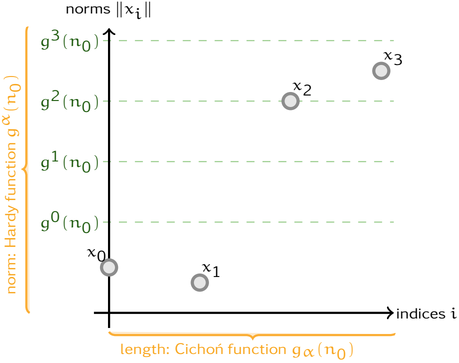
$$g^0(x) \stackrel{\text{def}}{=} x \qquad g^\alpha(x) \stackrel{\text{def}}{=} g^{P_x(\alpha)}(g(x)) \text{ for } \alpha > 0$$

Relating Norm and Length

[Cichoń & Tahhan Bittar'98]

$$g_0(x) \stackrel{\text{def}}{=} 0 \qquad g_\alpha(x) \stackrel{\text{def}}{=} 1 + g_{P_x(\alpha)}(g(x)) \qquad \text{for } \alpha > 0$$

$$g^0(x) \stackrel{\text{def}}{=} x \qquad g^\alpha(x) \stackrel{\text{def}}{=} g^{P_x(\alpha)}(g(x)) \qquad \text{for } \alpha > 0$$



$$g^\alpha(x) = g^{g_\alpha(x)}(x)$$

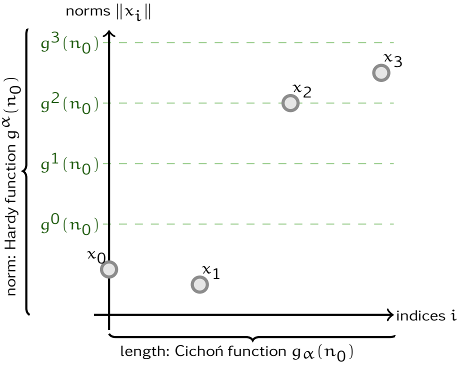
$$g^\alpha(x) \geqslant g_\alpha(x) + x$$

Relating Norm and Length

[Cichoń & Tahhan Bittar'98]

$$g_0(x) \stackrel{\text{def}}{=} 0 \qquad g_\alpha(x) \stackrel{\text{def}}{=} 1 + g_{P_x(\alpha)}(g(x)) \qquad \text{for } \alpha > 0$$

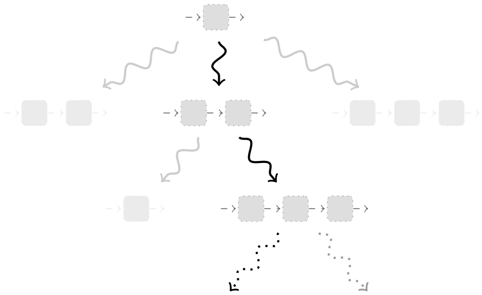
$$g^0(x) \stackrel{\text{def}}{=} x \qquad g^\alpha(x) \stackrel{\text{def}}{=} g^{P_x(\alpha)}(g(x)) \qquad \text{for } \alpha > 0$$



$$g^\alpha(x) = g^{g_\alpha(x)}(x)$$

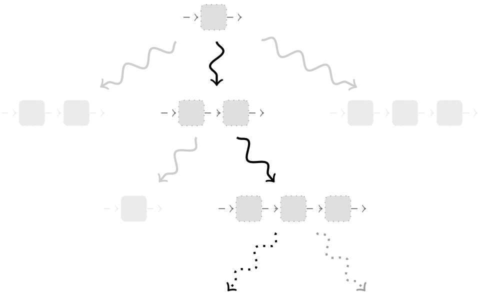
$$g^\alpha(x) \geqslant g_\alpha(x) + x$$

The Length of Decomposition Branches



$$\alpha_0 > \alpha_1 > \alpha_2 > \vdots$$

The Length of Decomposition Branches

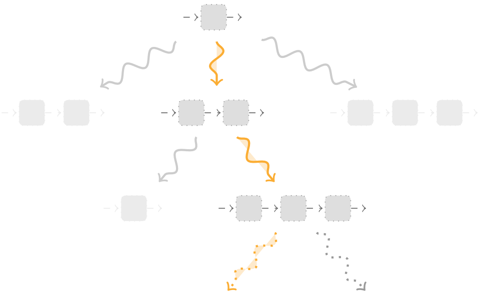


$$\alpha_0 > \alpha_1 > \alpha_2 > \cdots$$

Corollary

Let $n_0 \geqslant 2$ and $g : \mathbb{N} \to \mathbb{N}$ be such that the sequence of ordinal ranks computed by the decomposition algorithm is (g, n_0)-controlled. The algorithm runs in $\mathit{SPACE}(g^{\omega^{\omega^2}}(n_0))$.

The Length of Decomposition Branches

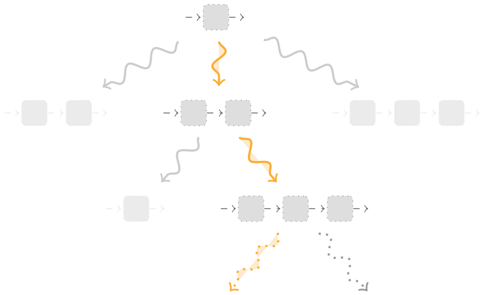


$$\alpha_0 > \alpha_1 > \alpha_2 > \vdots$$

Corollary

Let $n_0 \geqslant 2$ and $g : \mathbb{N} \to \mathbb{N}$ be such that the sequence of ordinal ranks computed by the decomposition algorithm is (g, n_0)-controlled. The algorithm runs in $\mathit{SPACE}(g^{\omega^{\omega^2}}(n_0))$.

The Length of Decomposition Branches



$$\alpha_0 > \alpha_1 > \alpha_2 > \cdots$$

Consequence of (Figueira, Figueira, S. & Schnoebelen'11)

The control $g(x) \stackrel{def}{=} H^{\omega^\omega}(e(x))$ for $H(x) \stackrel{def}{=} x+1$ and an elementary function e, and n_0 the size of the reachability instance fit. Thus the decomposition algorithm runs in $\mathit{SPACE}\big((H^{\omega^\omega} \circ e)^{\omega^{\omega^2}}(n)\big)$.

Restating the Result

"$\mathrm{SPACE}((H^{\omega^\omega} \circ e)^{\omega^{\omega^2}}(n))$" is unreadable!

1. give names
   - H^{ω^ω} is the Ackermann function
   - $H^{\omega^{\omega^2}}$ is the "quadratic Ackermann" function
2. define coarse-grained complexity classes

$$\mathscr{F}_{<\alpha} \stackrel{\text{def}}{=} \bigcup_{\gamma<\omega^\alpha} \mathrm{FDTIME}(H^\gamma(n)) \quad \mathbf{F}_\alpha \stackrel{\text{def}}{=} \bigcup_{f\in\mathscr{F}_{<\alpha}} \mathrm{DTIME}(H^{\omega^\alpha}(f(n)))$$

Consequence of (S.'16, Thm. 4.4)
VAS Reachability is in $\mathbf{F}_{\omega^2}$.

Restating the Result

"$\mathrm{SPACE}((H^{\omega^\omega} \circ e)^{\omega^{\omega^2}}(n))$" is unreadable!

1. give names
   - H^{ω^ω} is the Ackermann function
   - $H^{\omega^{\omega^2}}$ is the "quadratic Ackermann" function
2. define coarse-grained complexity classes

$$\mathscr{F}_{<\alpha} \stackrel{\text{def}}{=} \bigcup_{\gamma<\omega^\alpha} \mathrm{FDTIME}(H^\gamma(n)) \quad \mathbf{F}_\alpha \stackrel{\text{def}}{=} \bigcup_{f\in\mathscr{F}_{<\alpha}} \mathrm{DTIME}(H^{\omega^\alpha}(f(n)))$$

Consequence of (S.'16, Thm. 4.4)

VAS Reachability is in $\mathbf{F}_{\omega^2}$.

Restating the Result

"$\mathrm{SPACE}((H^{\omega^\omega} \circ e)^{\omega^{\omega^2}}(n))$" is unreadable!

1. give names
   - H^{ω^ω} is the Ackermann function
   - $H^{\omega^{\omega^2}}$ is the "quadratic Ackermann" function
2. define coarse-grained complexity classes

$$\mathscr{F}_{<\alpha} \stackrel{\text{def}}{=} \bigcup_{\gamma<\omega^\alpha} \mathrm{FDTIME}(H^\gamma(n)) \quad \mathbf{F}_\alpha \stackrel{\text{def}}{=} \bigcup_{f\in\mathscr{F}_{<\alpha}} \mathrm{DTIME}(H^{\omega^\alpha}(f(n)))$$

Consequence of (S.'16, Thm. 4.4)

VAS Reachability is in $\mathbf{F}_{\omega^2}$.

Restating the Result

"$\mathrm{SPACE}((H^{\omega^\omega} \circ e)^{\omega^{\omega^2}}(n))$" is unreadable!

1. give names
   - H^{ω^ω} is the Ackermann function
   - $H^{\omega^{\omega^2}}$ is the "quadratic Ackermann" function
2. define coarse-grained complexity classes

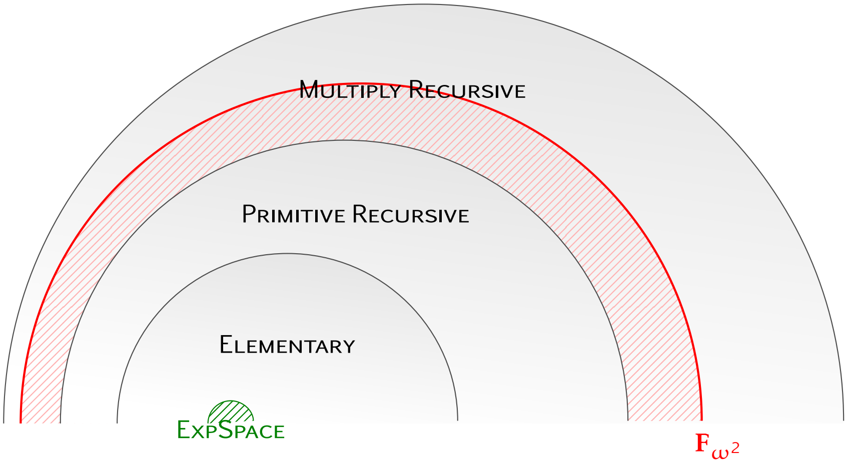
$$\mathscr{F}_{<\alpha} \stackrel{\text{def}}{=} \bigcup_{\gamma<\omega^\alpha} \mathrm{FDTIME}(H^\gamma(n)) \quad \mathbf{F}_\alpha \stackrel{\text{def}}{=} \bigcup_{f\in\mathscr{F}_{<\alpha}} \mathrm{DTIME}(H^{\omega^\alpha}(f(n)))$$

Consequence of (S.'16, Thm. 4.4)

VAS Reachability is in $\mathbf{F}_{\omega^2}$.

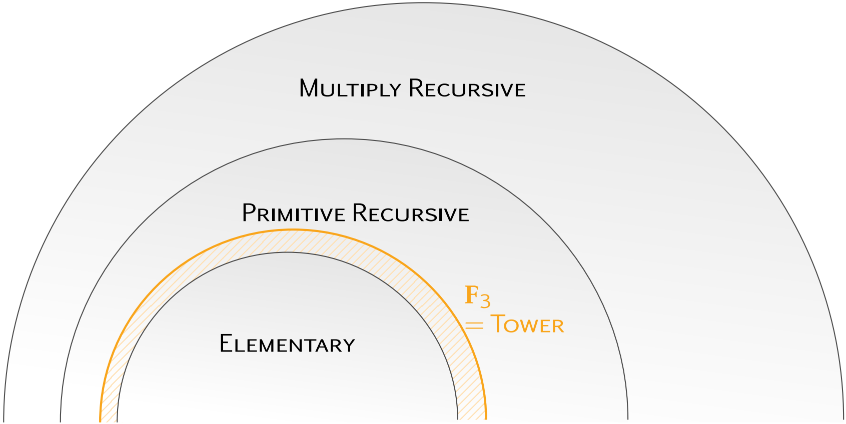
Complexity Classes Beyond Elementary

[S.'16]



Complexity Classes Beyond Elementary

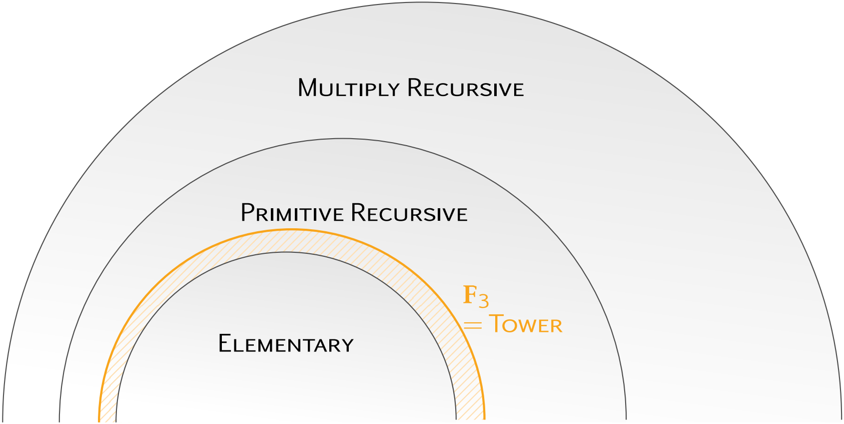
[S.'16]



$$\mathbf{F}_3 \stackrel{\text{def}}{=} \bigcup_{e \text{ elementary}} \mathrm{DTime}(\mathrm{tower}(e(n)))$$

Complexity Classes Beyond Elementary

[S.'16]

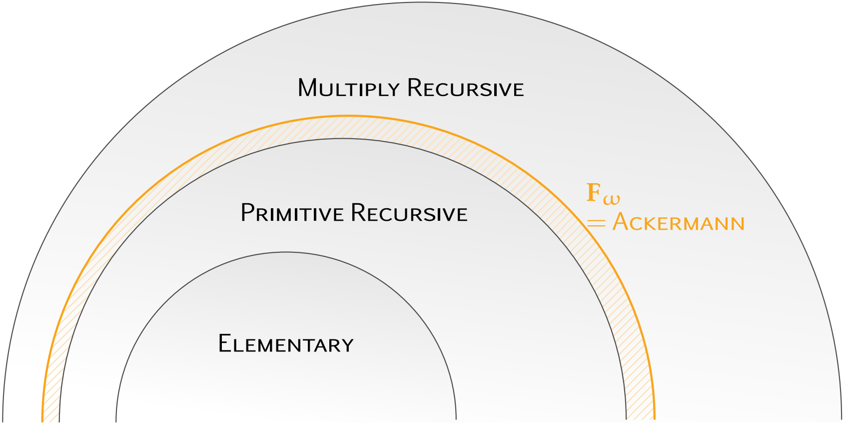


Examples of Tower-Complete Problems:

- satisfiability of first-order logic on words [Meyer'75]
- β-equivalence of simply typed λ terms [Statman'79]
- model-checking higher-order recursion schemes [Ong'06]

Complexity Classes Beyond Elementary

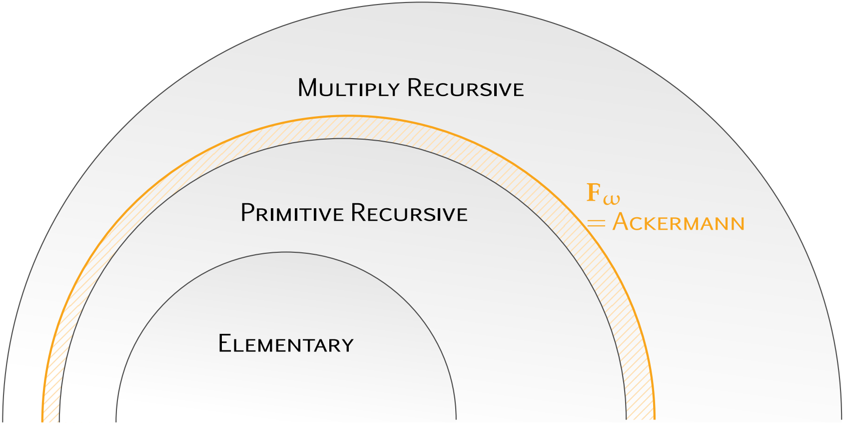
[S.'16]



$$\mathbf{F}_\omega \stackrel{\text{def}}{=} \bigcup_{p \text{ primitive recursive}} \mathrm{DTime}\big(\mathrm{ackermann}(p(n))\big)$$

Complexity Classes Beyond Elementary

[S.'16]

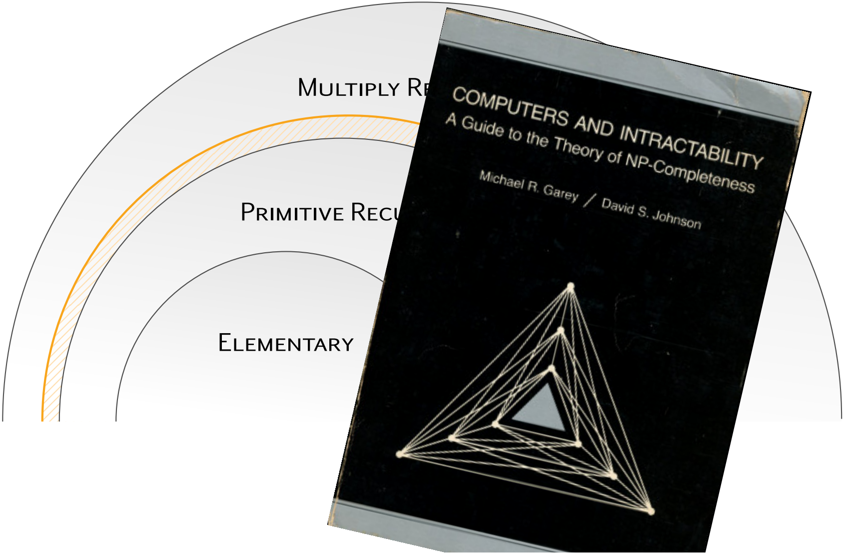


Examples of Ackermann-Complete Problems:

- reachability in lossy Minsky machines [Urquhart'98, Schnoebelen'02]
- satisfiability of safety Metric Temporal Logic [Lazić et al.'16]
- satisfiability of Vertical XPath [Figueira and Segoufin'17]

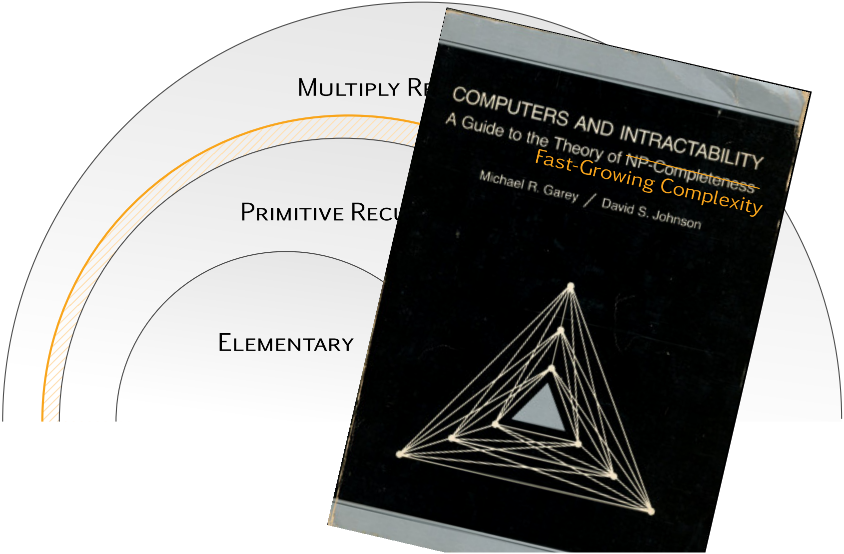
Complexity Classes Beyond Elementary

[S.'16]



Complexity Classes Beyond Elementary

[S.'16]



Summary

well-quasi-orders (wqo):

- proving algorithm termination

a toolbox for wqo-based complexity

- upper bounds: length function theorems (for ordinals, Dickson's Lemma, Higman's Lemma, and combinations)
- lower bounds
- complexity classes: $(\mathbf{F}_\alpha)_\alpha$

this talk: focus on one problem

- reachability in vector addition systems in $\mathbf{F}_{\omega^2}$

Perspectives

1. complexity gap for VAS reachability
   - ExpSpace-hard [Lipton'76]
     better lower bounds? (Wojciech's talk)
   - decomposition algorithm: at least $\mathbf{F}_\omega$ (Ackermannian) time [Zetzsche'16]
2. reachability in VAS extensions
   - decidable in VAS with hierarchical zero tests [Reinhardt'08]
   - what about (Jérôme's talk)
     - branching VAS
     - unordered data Petri nets
     - pushdown VAS

Perspectives

1. complexity gap for VAS reachability
    - ExpSpace-hard [Lipton'76]
      better lower bounds? (Wojciech's talk)
    - decomposition algorithm: at least $\mathbf{F}_\omega$ (Ackermannian) time [Zetzsche'16]
2. reachability in VAS extensions
    - decidable in VAS with hierarchical zero tests [Reinhardt'08]
    - what about (Jérôme's talk)
        - branching VAS
        - unordered data Petri nets
        - pushdown VAS

Demystifying Reachability in Vector Addition Systems

[Leroux & S.'15]

Ideal Decomposition Theorem

The Decomposition Algorithm computes the ideal decomposition of the set of runs from source to target.

Upper Bound Theorem

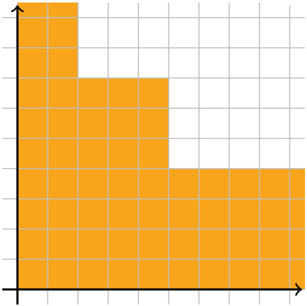
Reachability in vector addition systems is in cubic Ackermann.

Ideals of Well-Quasi-Orders $(X, \leqslant)$

- Canonical decompositions
  [Bonnet'75]
  if $D \subseteq X$ is $\downarrow$-closed, then

  $$D = I_1 \cup \cdots \cup I_n$$

  for (maximal) ideals $I_1, \ldots, I_n$



Example (over $\mathbb{N}^2$)

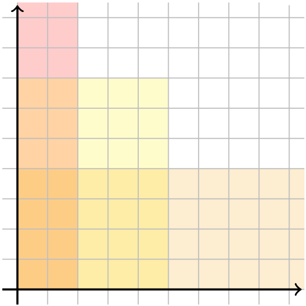
$$D = (\{0,\ldots,2\} \times \mathbb{N}) \cup (\{0,\ldots,5\} \times \{0,\ldots,7\}) \cup (\mathbb{N} \times \{0,\ldots,4\})$$

Ideals of Well-Quasi-Orders $(X, \leqslant)$

- Canonical decompositions [Bonnet'75]
  if $D \subseteq X$ is $\downarrow$-closed, then

  $$D = I_1 \cup \cdots \cup I_n$$

  for (maximal) ideals $I_1, \ldots, I_n$



Example (over $\mathbb{N}^2$)

$$D = (\{0,\ldots,2\} \times \mathbb{N}) \cup (\{0,\ldots,5\} \times \{0,\ldots,7\}) \cup (\mathbb{N} \times \{0,\ldots,4\})$$

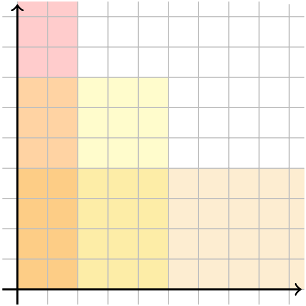
IDEALS OF WELL-QUASI-ORDERS $(X, \leqslant)$

- Canonical decompositions
  [Bonnet'75]
  if $D \subseteq X$ is $\downarrow$-closed, then

$$D = I_1 \cup \cdots \cup I_n$$

  for (maximal) ideals $I_1, \ldots, I_n$

- Effective representations
  [Goubault-Larrecq et al.'17]



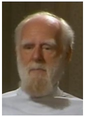
EXAMPLE (OVER $\mathbb{N}^2$)

$$D = [\![(2,\infty)]\!] \cup [\![(5,7)]\!] \cup [\![(\infty,4)]\!]$$

Decomposition Theorem

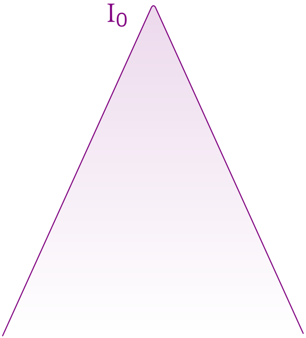
Well-Quasi-Order on Runs

combination of Dickson's and Higman's lemmata



Syntax

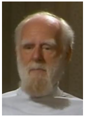
Semantics



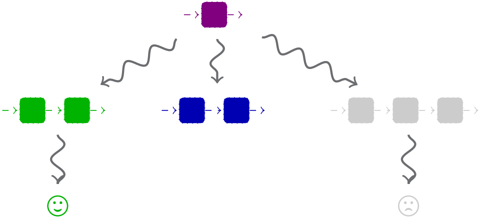
Decomposition Theorem

Well-Quasi-Order on Runs

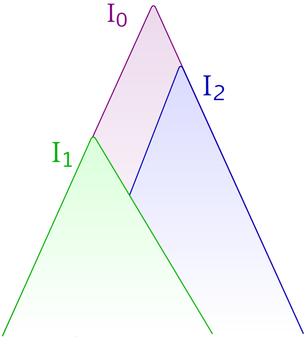
combination of Dickson's and Higman's lemmata



Syntax



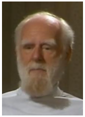
Semantics



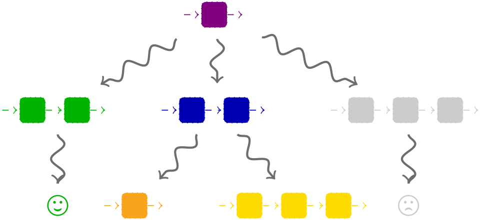
Decomposition Theorem

Well-Quasi-Order on Runs

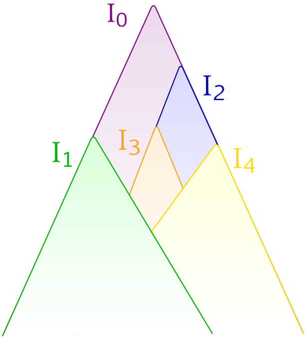
combination of Dickson's and Higman's lemmata



Syntax



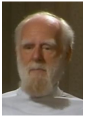
Semantics



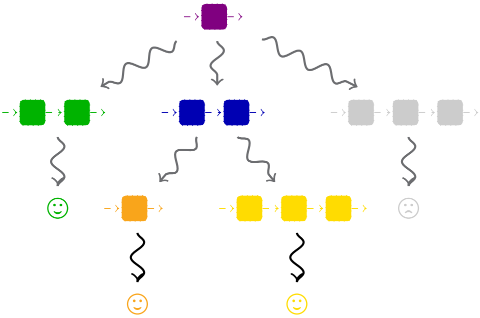
Decomposition Theorem

Well-Quasi-Order on Runs

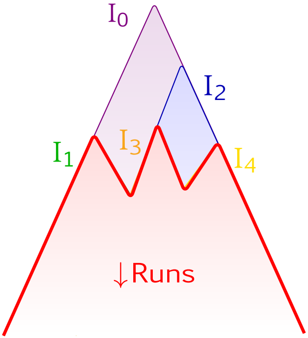
combination of Dickson's and Higman's lemmata



Syntax

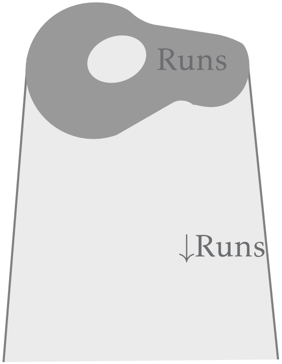


Semantics



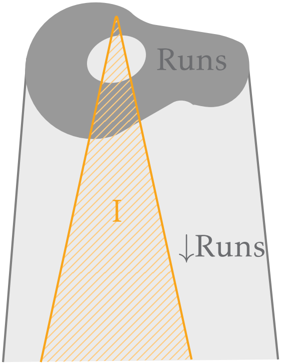
Adherence Membership

- I is adherent to Runs if $\mathrm{I} \subseteq {\downarrow}(\mathrm{I} \cap \mathrm{Runs})$
- semantic equivalent to Θ condition
- undecidable for arbitrary ideals
- decidable for the ideals arising in the decomposition algorithm



Adherence Membership

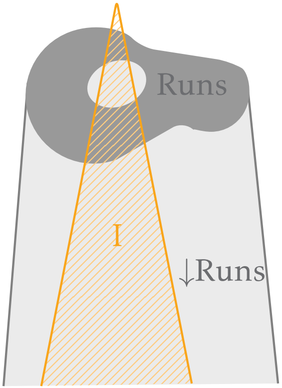
- I is adherent to Runs if $I \subseteq {\downarrow}(I \cap Runs)$
- semantic equivalent to Θ condition
- undecidable for arbitrary ideals
- decidable for the ideals arising in the decomposition algorithm



I adherent

Adherence Membership

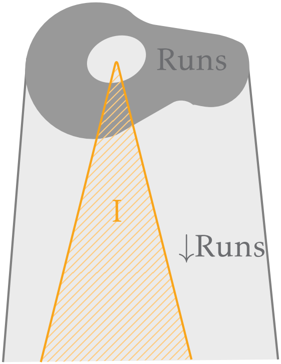
- I is adherent to Runs if $I \subseteq {\downarrow}(I \cap \mathrm{Runs})$
- semantic equivalent to Θ condition
- undecidable for arbitrary ideals
- decidable for the ideals arising in the decomposition algorithm



I not adherent

Adherence Membership

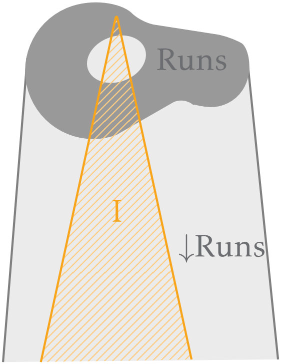
- I is adherent to Runs if $\mathrm{I} \subseteq {\downarrow}(\mathrm{I} \cap \mathrm{Runs})$
- semantic equivalent to Θ condition
- undecidable for arbitrary ideals
- decidable for the ideals arising in the decomposition algorithm



I not adherent

Adherence Membership

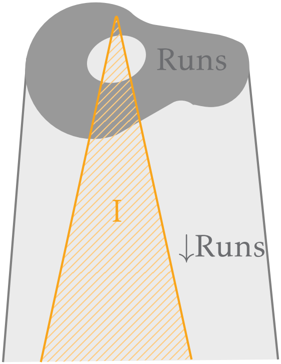
- I is adherent to Runs if $\mathrm{I} \subseteq {\downarrow}(\mathrm{I} \cap \mathrm{Runs})$
- semantic equivalent to Θ condition
- undecidable for arbitrary ideals
- decidable for the ideals arising in the decomposition algorithm



I adherent

Adherence Membership

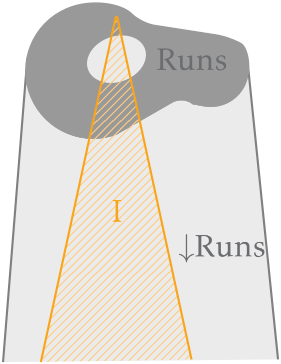
- I is adherent to Runs if $I \subseteq \downarrow(I \cap \text{Runs})$
- semantic equivalent to Θ condition
- undecidable for arbitrary ideals
- decidable for the ideals arising in the decomposition algorithm



I adherent

Adherence Membership

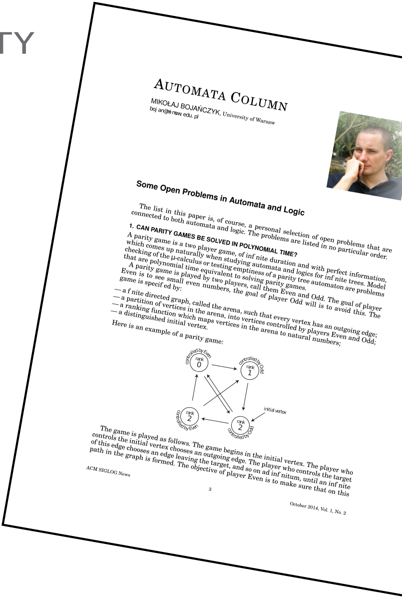
- I is adherent to Runs if $I \subseteq \downarrow(I \cap \mathrm{Runs})$
- semantic equivalent to Θ condition
- undecidable for arbitrary ideals
- decidable for the ideals arising in the decomposition algorithm



I adherent

Branching VAS Reachability

- important open problem [Bojańczyk'14]
- incorrect decidability proof in [Bimbó'15]
- application domains:



Automata Column

MIKOŁAJ BOJAŃCZYK, University of Warsaw
boj an@mimuw.edu.pl

Some Open Problems in Automata and Logic

The list in this paper is, of course, a personal selection of open problems that are connected to both automata and logic. The problems are listed in no particular order.

1. CAN PARITY GAMES BE SOLVED IN POLYNOMIAL TIME?

A parity game is a two player game, of inf nite duration and with perfect information, which comes up naturally when studying automata and logics for inf nite trees. Model checking of the μ-calculus or testing emptiness of a parity tree automaton are problems that are polynomial time equivalent to solving parity games.

A parity game is played by two players, call them Even and Odd. The goal of player Even is to see small even numbers, the goal of player Odd will is to avoid this. The game is specif ed by:

— a f nite directed graph, called the arena, such that every vertex has an outgoing edge;
— a partition of vertices in the arena, into vertices controlled by players Even and Odd;
— a ranking function which maps vertices in the arena to natural numbers;
— a distinguished initial vertex.

Here is an example of a parity game:

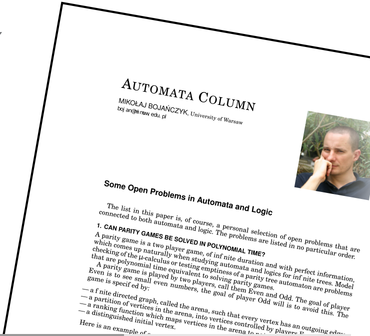
controlled by Even, rank 0; controlled by Odd, rank 1; controlled by Even, rank 2; controlled by Odd, rank 2; initial vertex

The game is played as follows. The game begins in the initial vertex. The player who controls the initial vertex chooses an outgoing edge. The player who controls the target of this edge chooses an edge leaving the target, and so on ad inf nitum, until an inf nite path in the graph is formed. The objective of player Even is to make sure that on this

ACM SIGLOG News 3 October 2014, Vol. 1, No. 2

Branching VAS Reachability

- important open problem [Bojańczyk'14]
- incorrect decidability proof in [Bimbó'15]
- application domains:



Automata Column

Mikołaj Bojańczyk, University of Warsaw

Some Open Problems in Automata and Logic

The list in this paper is, of course, a personal selection of open problems that are connected to both automata and logic. The problems are listed in no particular order.

1. CAN PARITY GAMES BE SOLVED IN POLYNOMIAL TIME?

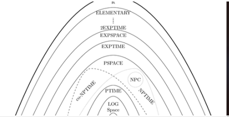
A parity game is a two player game, of infinite duration and with perfect information, which comes up naturally when studying automata and logics for infinite trees. Model checking of the μ-calculus or testing emptiness of a parity tree automaton are problems that are polynomial time equivalent to solving parity games.

A parity game is played by two players, call them Even and Odd. The goal of player Even is to see small even numbers, the goal of player Odd will is to avoid this. The game is specified by:

— a finite directed graph, called the arena, such that every vertex has an outgoing edge
— a partition of vertices in the arena, into vertices controlled by players
— a ranking function which maps vertices in the arena
— a distinguished initial vertex.

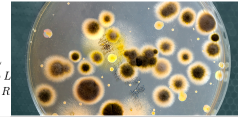
Here is an example of

Complexity Theory	Distributed Computing	Proof Theory	Computational Biology



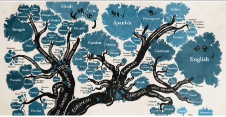
$$\dfrac{\dfrac{\dfrac{}{X \vdash X}\,\mathrm{ax} \qquad \dfrac{\dfrac{}{0 \vdash Y, Z}\,0L}{\vdash Y, 0 \multimap Z}\multimap R}{X \vdash Y, X \otimes (0 \multimap Z)}\otimes R}{\vdash X \multimap Y, X \otimes (0 \multimap Z)}\multimap R \qquad \dfrac{}{0 \vdash}\,0L$$

$$\dfrac{\dfrac{\vdash X \multimap Y, X \otimes (0 \multimap Z) \qquad 0 \vdash}{(X \multimap Y) \multimap 0 \vdash X \otimes (0 \multimap Z)}\multimap L}{\vdash ((X \multimap Y) \multimap 0) \multimap (X \otimes (0 \multimap Z))}\multimap R$$



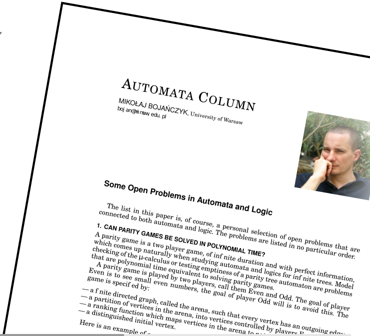
Programming Languages	Database Theory	Security	Computational Linguistics

```
fun append (xs, ys) =
    if null xs
    then ys
    else (hd xs):: append (tl xs, ys)

fun map (f, xs) =
    case xs of
        [] => []
      | x :: xs' => (f x)::(map (f, xs'))

val a = map (increment, [4,8,12,16])
val b = map (hd, [[8,6],[7,5],[3,0,9]])
```



Branching VAS Reachability

- important open problem [Bojańczyk'14]
- incorrect decidability proof in [Bimbó'15]
- application domains:


Automata Column
Mikołaj Bojańczyk, University of Warsaw
bojan@mimuw.edu.pl

Some Open Problems in Automata and Logic

The list in this paper is, of course, a personal selection of open problems that are connected to both automata and logic. The problems are listed in no particular order.

1. CAN PARITY GAMES BE SOLVED IN POLYNOMIAL TIME?

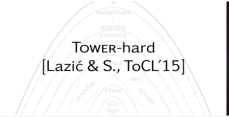
A parity game is a two player game, of infinite duration and with perfect information, which comes up naturally when studying automata and logics for infinite trees. Model checking of the μ-calculus or testing emptiness of a parity tree automaton are problems that are polynomial time equivalent to solving parity games.

A parity game is played by two players, call them Even and Odd. The goal of player Even is to see small even numbers, the goal of player Odd will is to avoid this. The game is specified by:

— a finite directed graph, called the arena, such that every vertex has an outgoing edge
— a partition of vertices in the arena, into vertices controlled by players E...
— a ranking function which maps vertices in the arena to n...
— a distinguished initial vertex.

Here is an example of ...

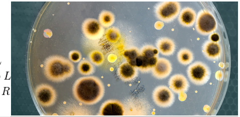
Complexity Theory	Distributed Computing	Proof Theory	Computational Biology



Tower-hard
[Lazić & S., ToCL'15]

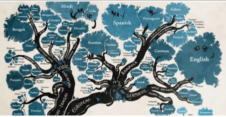
$$\dfrac{\dfrac{}{X \vdash X}\,\mathrm{ax} \quad \dfrac{\dfrac{}{0 \vdash Y, Z}\,0L}{\vdash Y, 0 \multimap Z}\multimap R}{\dfrac{X \vdash Y, X \otimes (0 \multimap Z)}{\dfrac{\vdash X \multimap Y, X \otimes (0 \multimap Z)}{}\multimap R}}\otimes R \quad \dfrac{}{0 \vdash}\,0L$$

$$\dfrac{\dfrac{\vdash X \multimap Y, X \otimes (0 \multimap Z) \quad 0 \vdash}{(X \multimap Y) \multimap 0 \vdash X \otimes (0 \multimap Z)}\multimap L}{\vdash ((X \multimap Y) \multimap 0) \multimap (X \otimes (0 \multimap Z))}\multimap R$$



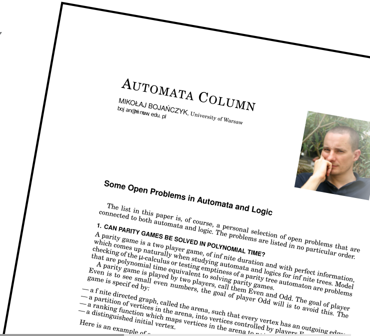
Programming Languages	Database Theory	Security	Computational Linguistics

```
fun append (xs, ys) =
    if null xs
    then ys
    else (hd xs):: append (tl xs, ys)

fun map (f, xs) =
    case xs of
        [] => []
      | x :: xs' => (f x)::(map (f, xs'))

val a = map (increment, [4,8,12,16])
val b = map (hd, [[8,6],[7,5],[3,0,9]])
```



Branching VAS Reachability

- important open problem [Bojańczyk'14]
- incorrect decidability proof in [Bimbó'15]
- application domains:


Automata Column

Mikołaj Bojańczyk, University of Warsaw

bojan@mimuw.edu.pl

Some Open Problems in Automata and Logic

The list in this paper is, of course, a personal selection of open problems that are connected to both automata and logic. The problems are listed in no particular order.

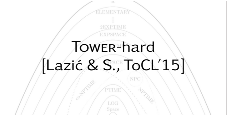
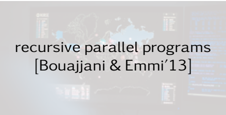
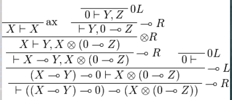
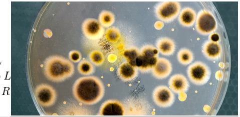
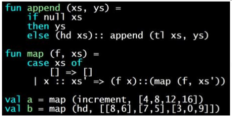
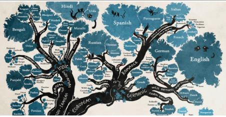
1. CAN PARITY GAMES BE SOLVED IN POLYNOMIAL TIME?

A parity game is a two player game, of infinite duration and with perfect information, which comes up naturally when studying automata and logics for infinite trees. Model checking of the μ-calculus or testing emptiness of a parity tree automaton are problems that are polynomial time equivalent to solving parity games.

A parity game is played by two players, call them Even and Odd. The goal of player Even is to see small even numbers, the goal of player Odd will is to avoid this. The game is specified by:

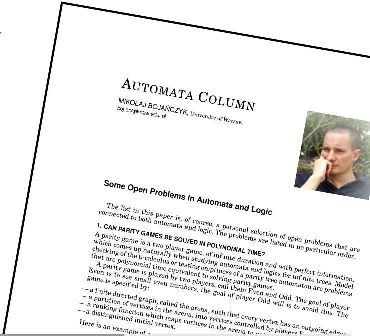
— a finite directed graph, called the arena, such that every vertex has an outgoing edge
— a partition of vertices in the arena, into vertices controlled by players E…
— a ranking function which maps vertices in the arena to n…
— a distinguished initial vertex.

Here is an example of …

Complexity Theory	Distributed Computing	Proof Theory	Computational Biology
 Tower-hard [Lazić & S., ToCL'15]	 recursive parallel programs [Bouajjani & Emmi'13]		
Programming Languages	**Database Theory**	**Security**	**Computational Linguistics**
			

Branching VAS Reachability

- important open problem [Bojańczyk'14]
- incorrect decidability proof in [Bimbó'15]
- application domains:



Automata Column

MIKOŁAJ BOJAŃCZYK, University of Warsaw
bojan@mimuw.edu.pl

Some Open Problems in Automata and Logic

The list in this paper is, of course, a personal selection of open problems that are connected to both automata and logic. The problems are listed in no particular order.

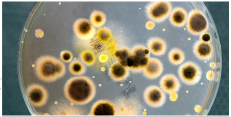
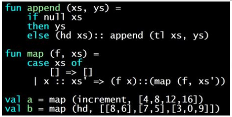
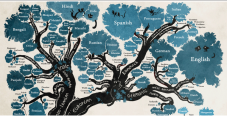
1. CAN PARITY GAMES BE SOLVED IN POLYNOMIAL TIME?

A parity game is a two player game, of inf nite duration and with perfect information, which comes up naturally when studying automata and logics for inf nite trees. Model checking of the μ-calculus or testing emptiness of a parity tree automaton are problems that are polynomial time equivalent to solving parity games.

A parity game is played by two players, call them Even and Odd. The goal of player Even is to see small even numbers, the goal of player Odd will is to avoid this. The game is specif ed by:

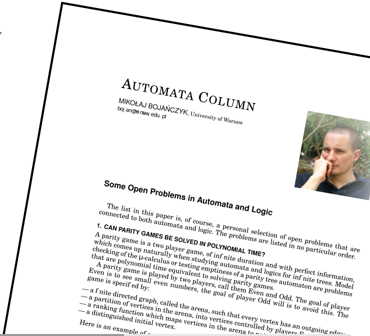
— a f nite directed graph, called the arena, such that every vertex has an outgoing edge;
— a partition of vertices in the arena, into vertices controlled by players E...
— a ranking function which maps vertices in the arena to n...
— a distinguished initial vertex.

Here is an example of a...

Complexity Theory	Distributed Computing	Proof Theory	Computational Biology
Tower-hard [Lazić & S., ToCL'15]	recursive parallel programs [Bouajjani & Emmi'13]	linear and relevance logics [de Groote et al.'04 Lazić & S., ToCL'15 S., JSL'16]	
Programming Languages	**Database Theory**	**Security**	**Computational Linguistics**
			

Branching VAS Reachability

- important open problem [Bojańczyk'14]
- incorrect decidability proof in [Bimbó'15]
- application domains:



Automata Column

MIKOŁAJ BOJAŃCZYK, University of Warsaw
boj an@mimuw.edu.pl

Some Open Problems in Automata and Logic

The list in this paper is, of course, a personal selection of open problems that are connected to both automata and logic. The problems are listed in no particular order.

1. CAN PARITY GAMES BE SOLVED IN POLYNOMIAL TIME?

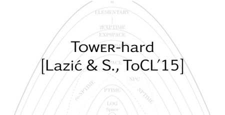
A parity game is a two player game, of infinite duration and with perfect information, which comes up naturally when studying automata and logics for infinite trees. Model checking of the μ-calculus or testing emptiness of a parity tree automaton are problems that are polynomial time equivalent to solving parity games.

A parity game is played by two players, call them Even and Odd. The goal of player Even is to see small even numbers, the goal of player Odd will is to avoid this. The game is specified by:

— a finite directed graph, called the arena, such that every vertex has an outgoing edge;
— a partition of vertices in the arena, into vertices controlled by players E…
— a ranking function which maps vertices in the arena to n…
— a distinguished initial vertex.

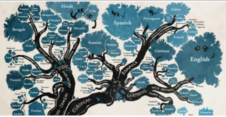
Here is an example of …

Complexity Theory	Distributed Computing	Proof Theory	Computational Biology
Tower-hard [Lazić & S., ToCL'15]	recursive parallel programs [Bouajjani & Emmi'13]	linear and relevance logics [de Groote et al.'04 Lazić & S., ToCL'15 S., JSL'16]	population protocols [Bertrand et al.'17]



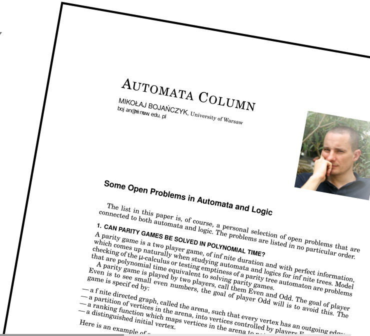
Programming Languages	Database Theory	Security	Computational Linguistics

```
fun append (xs, ys) =
    if null xs
    then ys
    else (hd xs):: append (tl xs, ys)

fun map (f, xs) =
    case xs of
       [] => []
     | x :: xs' => (f x)::(map (f, xs'))

val a = map (increment, [4,8,12,16])
val b = map (hd, [[8,6],[7,5],[3,0,9]])
```



Branching VAS Reachability

- important open problem [Bojańczyk'14]
- incorrect decidability proof in [Bimbó'15]
- application domains:


Automata Column

Mikołaj Bojańczyk, University of Warsaw

Some Open Problems in Automata and Logic

The list in this paper is, of course, a personal selection of open problems that are connected to both automata and logic. The problems are listed in no particular order.

1. CAN PARITY GAMES BE SOLVED IN POLYNOMIAL TIME?

A parity game is a two player game, of infinite duration and with perfect information, which comes up naturally when studying automata and logics for infinite trees. Model checking of the μ-calculus or testing emptiness of a parity tree automaton are problems that are polynomial time equivalent to solving parity games.

A parity game is played by two players, call them Even and Odd. The goal of player Even is to see small even numbers, the goal of player Odd will is to avoid this. The game is specified by:

— a finite directed graph, called the arena, such that every vertex has an outgoing edge
— a partition of vertices in the arena, into vertices controlled by players
— a ranking function which maps vertices in the arena to
— a distinguished initial vertex.

Here is an example of

Complexity Theory

Tower-hard [Lazić & S., ToCL'15]

Distributed Computing

recursive parallel programs [Bouajjani & Emmi'13]

Proof Theory

linear and relevance logics [de Groote et al.'04 Lazić & S., ToCL'15 S., JSL'16]

Computational Biology

population protocols [Bertrand et al.'17]

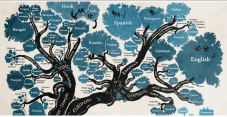
Programming Languages

observational equivalence [Cotton-Barratt et al.'17]

Database Theory

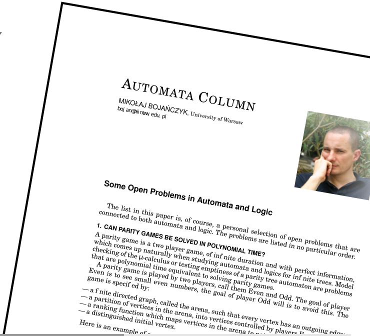
Security

Computational Linguistics



Branching VAS Reachability

- important open problem [Bojańczyk'14]
- incorrect decidability proof in [Bimbó'15]
- application domains:



Automata Column

MIKOŁAJ BOJAŃCZYK, University of Warsaw

Some Open Problems in Automata and Logic

The list in this paper is, of course, a personal selection of open problems that are connected to both automata and logic. The problems are listed in no particular order.

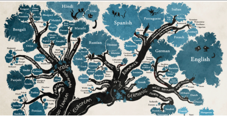
1. CAN PARITY GAMES BE SOLVED IN POLYNOMIAL TIME?

A parity game is a two player game, of infinite duration and with perfect information, which comes up naturally when studying automata and logics for infinite trees. Model checking of the μ-calculus or testing emptiness of a parity tree automaton are problems that are polynomial time equivalent to solving parity games.

A parity game is played by two players, call them Even and Odd. The goal of player Even is to see small even numbers, the goal of player Odd will is to avoid this. The game is specified by:

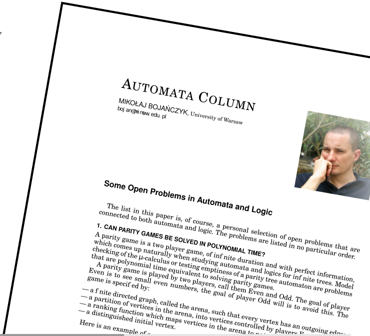
— a finite directed graph, called the arena, such that every vertex has an outgoing edge
— a partition of vertices in the arena, into vertices controlled by players
— a ranking function which maps vertices in the arena to
— a distinguished initial vertex.

Here is an example of

Complexity Theory	Distributed Computing	Proof Theory	Computational Biology
Tower-hard [Lazić & S., ToCL'15]	recursive parallel programs [Bouajjani & Emmi'13]	linear and relevance logics [de Groote et al.'04 Lazić & S., ToCL'15 S., JSL'16]	population protocols [Bertrand et al.'17]
Programming Languages	**Database Theory**	**Security**	**Computational Linguistics**
observational equivalence [Cotton-Barratt et al.'17]	data logics [Bojańczyk et al.'09, Abriola et al.'17]		

Branching VAS Reachability

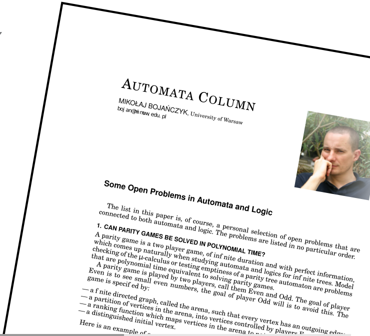
- important open problem [Bojańczyk'14]
- incorrect decidability proof in [Bimbó'15]
- application domains:



Complexity Theory	Distributed Computing	Proof Theory	Computational Biology
Tower-hard [Lazić & S., ToCL'15]	recursive parallel programs [Bouajjani & Emmi'13]	linear and relevance logics [de Groote et al.'04 Lazić & S., ToCL'15 S., JSL'16]	population protocols [Bertrand et al.'17]
Programming Languages	**Database Theory**	**Security**	**Computational Linguistics**
observational equivalence [Cotton-Barratt et al.'17]	data logics [Bojańczyk et al.'09, Abriola et al.'17]	security protocols [Verma & Goubault-Larrecq'05]	

Branching VAS Reachability

- important open problem [Bojańczyk'14]
- incorrect decidability proof in [Bimbó'15]
- application domains:



Automata Column

Mikołaj Bojańczyk, University of Warsaw

Some Open Problems in Automata and Logic

The list in this paper is, of course, a personal selection of open problems that are connected to both automata and logic. The problems are listed in no particular order.

1. CAN PARITY GAMES BE SOLVED IN POLYNOMIAL TIME?

A parity game is a two player game, of infinite duration and with perfect information, which comes up naturally when studying automata and logics for infinite trees. Model checking of the μ-calculus or testing emptiness of a parity tree automaton are problems that are polynomial time equivalent to solving parity games.

A parity game is played by two players, call them Even and Odd. The goal of player Even is to see small even numbers, the goal of player Odd will is to avoid this. The game is specified by:

Complexity Theory	Distributed Computing	Proof Theory	Computational Biology
Tower-hard [Lazić & S., ToCL'15]	recursive parallel programs [Bouajjani & Emmi'13]	linear and relevance logics [de Groote et al.'04 Lazić & S., ToCL'15 S., JSL'16]	population protocols [Bertrand et al.'17]
Programming Languages	**Database Theory**	**Security**	**Computational Linguistics**
observational equivalence [Cotton-Barratt et al.'17]	data logics [Bojańczyk et al.'09, Abriola et al.'17]	security protocols [Verma & Goubault-Larrecq'05]	dominance grammars [Rambow'94; S., ACL'10] minimalist syntax [Salvati'10]