Model-checking Vector Addition Systems with one zero-test

Rémi Bonnet, Alain Finkel, Jerôme Leroux, Marc Zeitoun

LSV, ENS Cachan — LaBRI, U. Bordeaux — CNRS — INRIA

LaBRI, September 2011

Motivation

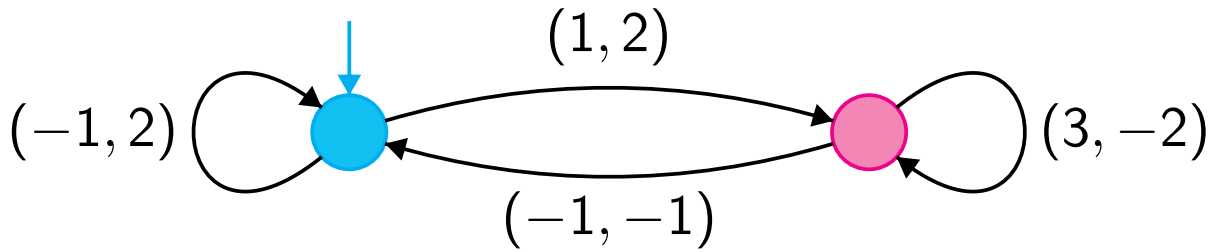
- Verification of vector addition systems against temporal logics.
- What can we add to VASS without losing the decidability properties?
  - One stack? Motivation for looking at one zero-test.
  - One zero-test? Two = Minsky machines ☹.
- Hard: Reachability (Reinhardt '04–08/ Bonnet '11).
- What about "easy" problems for VASS, if one adds a zero-test?
  - Coverability ? Computation of the cover ?
  - LTL model-checking ?

Main Tools

1. Refinement of decidability of reachability set of a VASS.
2. Well-quasi orders over strings, Higman's Lemma.
3. Valk & Jentzen's Lemma.
4. New sets hybrid between the cover and the reachability set.
5. Karp & Miller's trees.

VAS(S): Vector Addition Systems (with States)

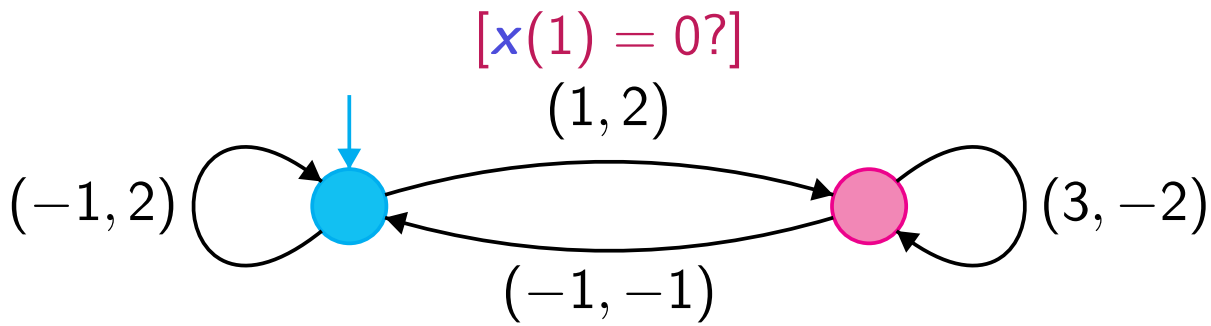
- Automaton with states in finite set Q, and with d counters in $\mathbb{N}$.
- Counters can be incremented or decremented, but remain nonnegative.



$(\bullet, 0, 0) \to (\bullet, 1, 2) \to (\bullet, 0, 1) \to (\bullet, 1, 3) \to (\bullet, 4, 1) \to (\bullet, 3, 0) \to (\bullet, 4, 2)$

VAS(S): Vector Addition Systems (with States)

- Automaton with states in finite set Q, and with d counters in $\mathbb{N}$.
- Counters can be incremented or decremented, but remain nonnegative.

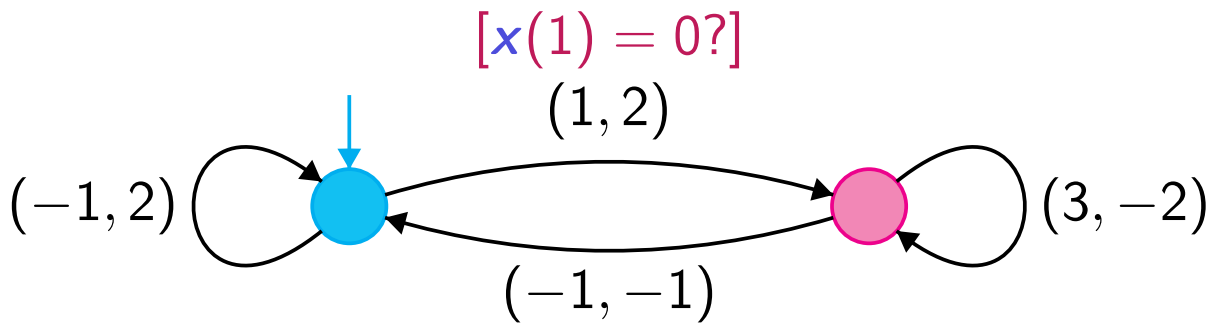


$(\bullet, 0, 0) \to (\bullet, 1, 2) \to (\bullet, 0, 1) \to (\bullet, 1, 3) \to (\bullet, 4, 1) \to (\bullet, 3, 0) \to (\bullet, 4, 2)$

- In a VASS_0, special zero-test transition a_z, fireable only if $x(1) = 0$.

VAS(S): Vector Addition Systems (with States)

- Automaton with states in finite set Q, and with d counters in $\mathbb{N}$.
- Counters can be incremented or decremented, but remain nonnegative.



$(\bullet, 0, 0) \to (\bullet, 1, 2) \to (\bullet, 0, 1) \to (\bullet, 1, 3) \to (\bullet, 4, 1) \to (\bullet, 3, 0) \to (\bullet, 4, 2)$

- In a VASS_0, special zero-test transition a_z, fireable only if $x(1) = 0$.
- Terminology. $\bullet$ is a control state / $(\bullet, 1, 2)$ is a state.
- Notation. VASS_0 $\mathcal{V} = \langle Q, A, a_z, \delta, (q_{in}, x_{in}) \rangle$. VAS $\mathcal{V} = \langle A, \delta, x_{in} \rangle$

Model-checking VASS

- Reachability — Is (q, x) reachable from initial (q_{in}, x_{in})?
- Finiteness — Is the state space finite?
- Termination — Are all runs finite?
- Place boundedness — Does a given counter remain bounded?
- Coverability — Given p, x, is there $y \geqslant x$ with $(q_{in}, x_{in}) \xrightarrow{*} (p, y)$?
- LTL model-checking — Does an LTL formula over action sequences hold?

Except for reachability, these problems can be solved using Karp-Miller trees.

Outline

1. Adapting the Karp & Miller's algorithm.
2. Outline of the proof.
3. Proof details: limits of reachable states.
4. Application: LTL model-checking.

Karp-Miller's algorithm for VAS

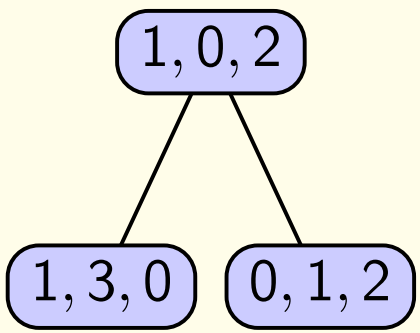
- Computes the downward closure of the reachability set.

$$\mathrm{Reach}(\mathcal{V}) = \left\{ y \in \mathbb{N}^d \mid x_{in} \xrightarrow{*} y \right\},$$
$$\mathrm{Cover}(\mathcal{V}) = \downarrow \mathrm{Reach}(\mathcal{V}).$$

Karp-Miller's algorithm for VAS

- Computes the downward closure of the reachability set.

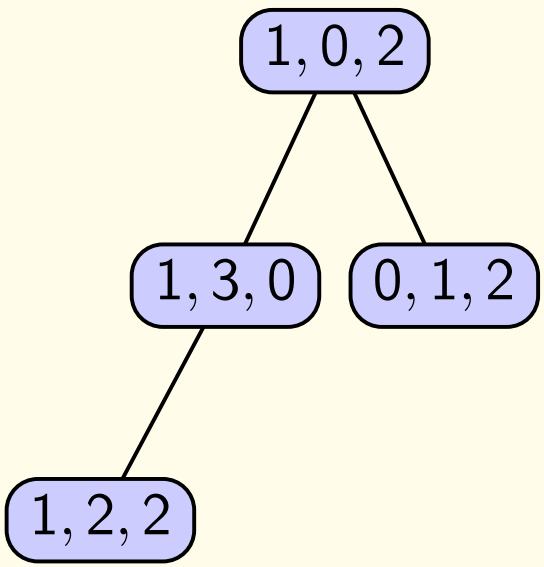
$$\mathrm{Reach}(\mathcal{V}) = \left\{ y \in \mathbb{N}^d \mid x_{in} \xrightarrow{*} y \right\},$$
$$\mathrm{Cover}(\mathcal{V}) = \downarrow \mathrm{Reach}(\mathcal{V}).$$



Karp-Miller's algorithm for VAS

- Computes the downward closure of the reachability set.

$$\mathrm{Reach}(\mathcal{V}) = \{ y \in \mathbb{N}^d \mid x_{in} \xrightarrow{*} y \},$$
$$\mathrm{Cover}(\mathcal{V}) = \downarrow \mathrm{Reach}(\mathcal{V}).$$

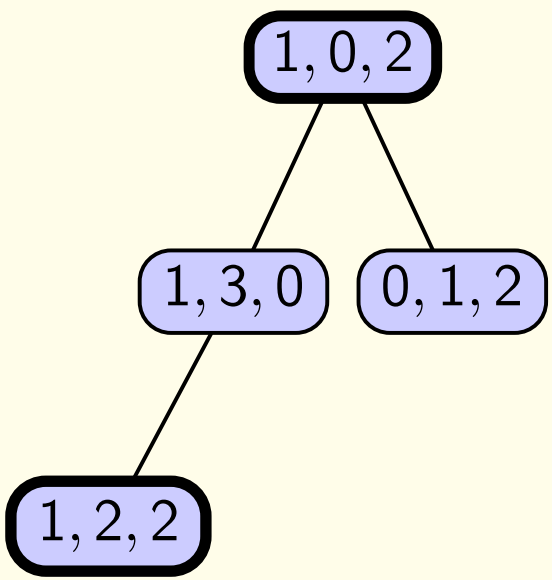


- Start from initial state.
- Unfold the VAS.
- Accelerate leaves with smaller ancestor
- Stop branch if ancestor with same label
- Acceleration guarantees termination.

Karp-Miller's algorithm for VAS

- Computes the downward closure of the reachability set.

$$\mathrm{Reach}(\mathcal{V}) = \{ y \in \mathbb{N}^d \mid x_{in} \xrightarrow{*} y \},$$
$$\mathrm{Cover}(\mathcal{V}) = \downarrow \mathrm{Reach}(\mathcal{V}).$$

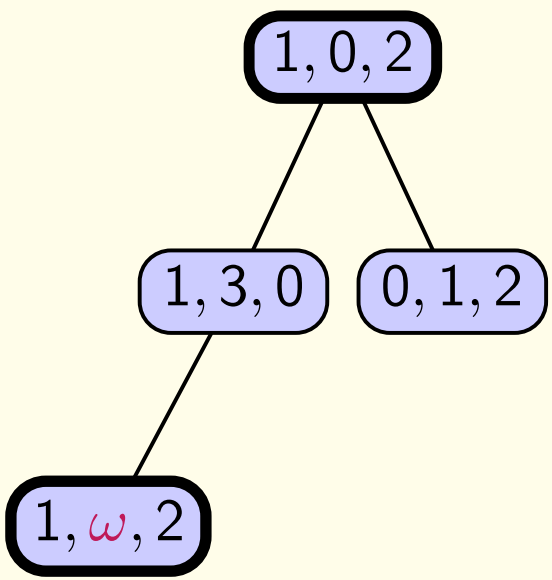


- Start from initial state.
- Unfold the VAS.
- Accelerate leaves with smaller ancestor
- Stop branch if ancestor with same label
- Acceleration guarantees termination.

Karp-Miller's algorithm for VAS

- Computes the downward closure of the reachability set.

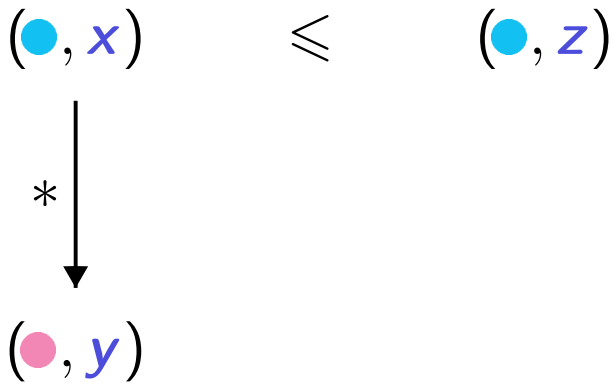
$$\mathrm{Reach}(\mathcal{V}) = \{ y \in \mathbb{N}^d \mid x_{in} \xrightarrow{*} y \},$$
$$\mathrm{Cover}(\mathcal{V}) = \downarrow \mathrm{Reach}(\mathcal{V}).$$



- Start from initial state.
- Unfold the VAS.
- Accelerate leaves with smaller ancestor
- Stop branch if ancestor with same label
- Acceleration guarantees termination.

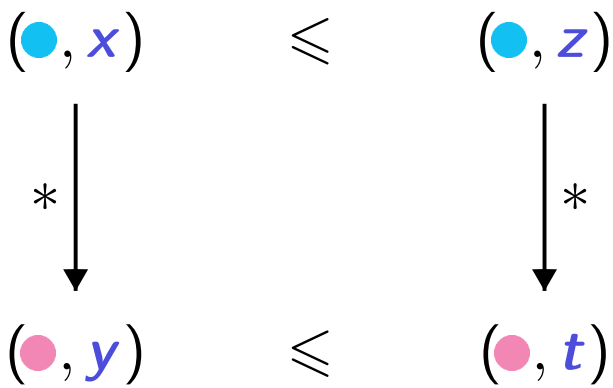
The monotonicity property

- Karp-Miller's algorithm and variants work for well-structured systems.
- The key property is monotonicity of VAS:



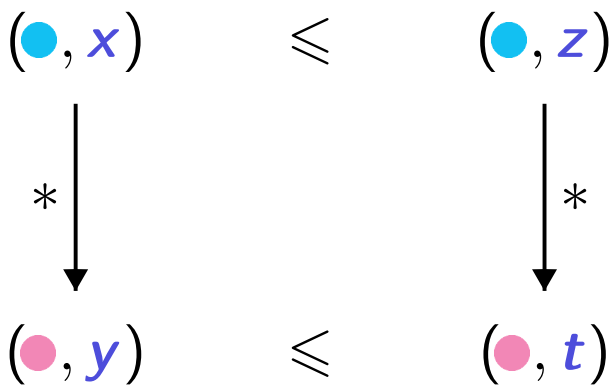
The monotonicity property

- Karp-Miller's algorithm and variants work for well-structured systems.
- The key property is monotonicity of VAS:



The monotonicity property

- Karp-Miller's algorithm and variants work for well-structured systems.
- The key property is monotonicity of VAS:



☹ Monotonicity fails with one zero-test

Ideas for adapting the Karp-Miller construction

- First build the Karp-Miller tree, without firing zero test.
- To proceed, need to fire the zero-test from the leaves of KM tree.

- Problem: accelerations may have produced on the first component an ω
  - represents arbitrarily large values and abstracts actual values.
  - Hides if possible actual values can be 0.

- $\Longrightarrow$ One may not be able to determine if the zero test succeeds or not.
- We need accurate information in labeling for the first component.

Two new cover sets

- To compute accurately on first component, we introduce
  - Refined cover, parameterized by a set $P \subseteq \{1, \ldots, d\}$ of positions.
  - Filtered covers, parameterized by a filter vector $\boldsymbol{f} \in \mathbb{N}_\omega^d$.

$$\mathrm{Reach}(\mathcal{V}) \subseteq \mathrm{RefinedCover}(\mathcal{V}) \subseteq \mathrm{Cover}(\mathcal{V}),$$

$$\mathrm{FilteredCover}(\mathcal{V}) \subseteq \mathrm{Cover}(\mathcal{V})$$

Refined covers

For a set $P \subseteq \{1, \ldots, d\}$ of positions, let

$$x \leqslant_P y \quad \text{if} \quad \begin{cases} x(i) = y(i) & \text{for } i \in P, \qquad \text{(accuracy on } P\text{)} \\ x(i) \leqslant y(i) & \text{for } i \notin P. \qquad \text{(lossiness elsewhere)} \end{cases}$$

The refined cover is

$$\mathrm{Cover}_{\leqslant_P}(\mathcal{V}) = {\downarrow_{\leqslant_P}}\mathrm{Reach}(\mathcal{V}),$$

Refined covers

For a set $P \subseteq \{1, \ldots, d\}$ of positions, let

$$x \leqslant_P y \quad \text{if} \quad \begin{cases} x(i) = y(i) & \text{for } i \in P, \quad \text{(accuracy on } P\text{)} \\ x(i) \leqslant y(i) & \text{for } i \notin P. \quad \text{(lossiness elsewhere)} \end{cases}$$

The refined cover is

$$\mathrm{Cover}_{\leqslant_P}(\mathcal{V}) = {\downarrow_{\leqslant_P}}\mathrm{Reach}(\mathcal{V}),$$

Proposition (BFLZ)

Given two VAS $\mathcal{V}_1$, $\mathcal{V}_2$, it is undecidable whether $\mathrm{Cover}_{\leqslant_1}(\mathcal{V}_1) = \mathrm{Cover}_{\leqslant_1}(\mathcal{V}_2)$.

Proof.

Reduction from $\mathrm{Reach}(\mathcal{V}_1) = \mathrm{Reach}(\mathcal{V}_2)$, with extra "total sum" first component

Filtered covers

The f-filtered cover of a VAS$_0$/VAS $\mathcal{V}$ is $\Downarrow_f \mathrm{Reach}(\mathcal{V})$:

$$\mathrm{Filter}(M, f) = \left\{ x \in M \mid \bigwedge_{i=1}^{d} \left[f(i) < \omega \Longrightarrow x(i) = f(i) \right] \right\},$$

$$\Downarrow_f M = \downarrow \mathrm{Filter}(M, f).$$

- For $f = (\omega, \omega, \ldots, \omega)$, we have $\Downarrow_f M = \downarrow M$.
- We are interested in $\Downarrow_f M$ for $M = \mathrm{Reach}(\mathcal{V})$ and $f = (0, \omega, \omega, \ldots, \omega)$.

Filtered covers

The f-filtered cover of a VAS_0/VAS $\mathcal{V}$ is $\Downarrow_f \mathrm{Reach}(\mathcal{V})$:

$$\mathrm{Filter}(M, f) = \left\{ x \in M \mid \bigwedge_{i=1}^{d} \left[f(i) < \omega \Longrightarrow x(i) = f(i) \right] \right\},$$

$$\Downarrow_f M = \downarrow \mathrm{Filter}(M, f).$$

- For $f = (\omega, \omega, \ldots, \omega)$, we have $\Downarrow_f M = \downarrow M$.
- We are interested in $\Downarrow_f M$ for $M = \mathrm{Reach}(\mathcal{V})$ and $f = (0, \omega, \omega, \ldots, \omega)$.

Theorem (BFLZ, "KM progress theorem")

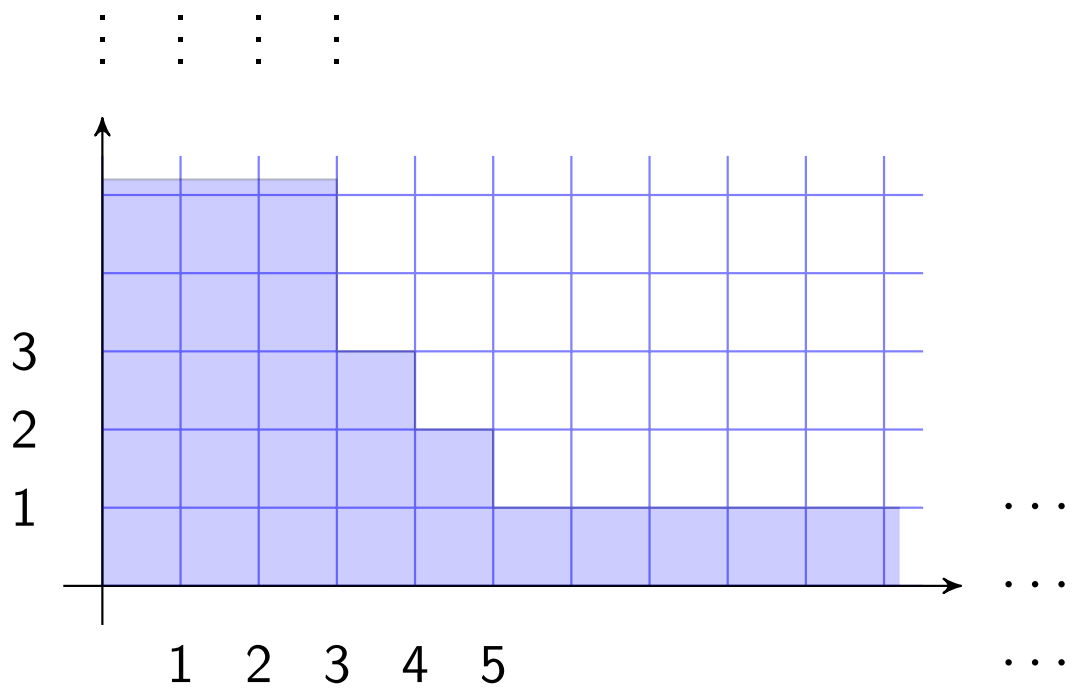
Given a VAS $\mathcal{V}$ and $f \in \mathbb{N}_\omega^d$, we can compute a representation of $\Downarrow_f \mathrm{Reach}(\mathcal{V})$

What means "compute a representation" here?

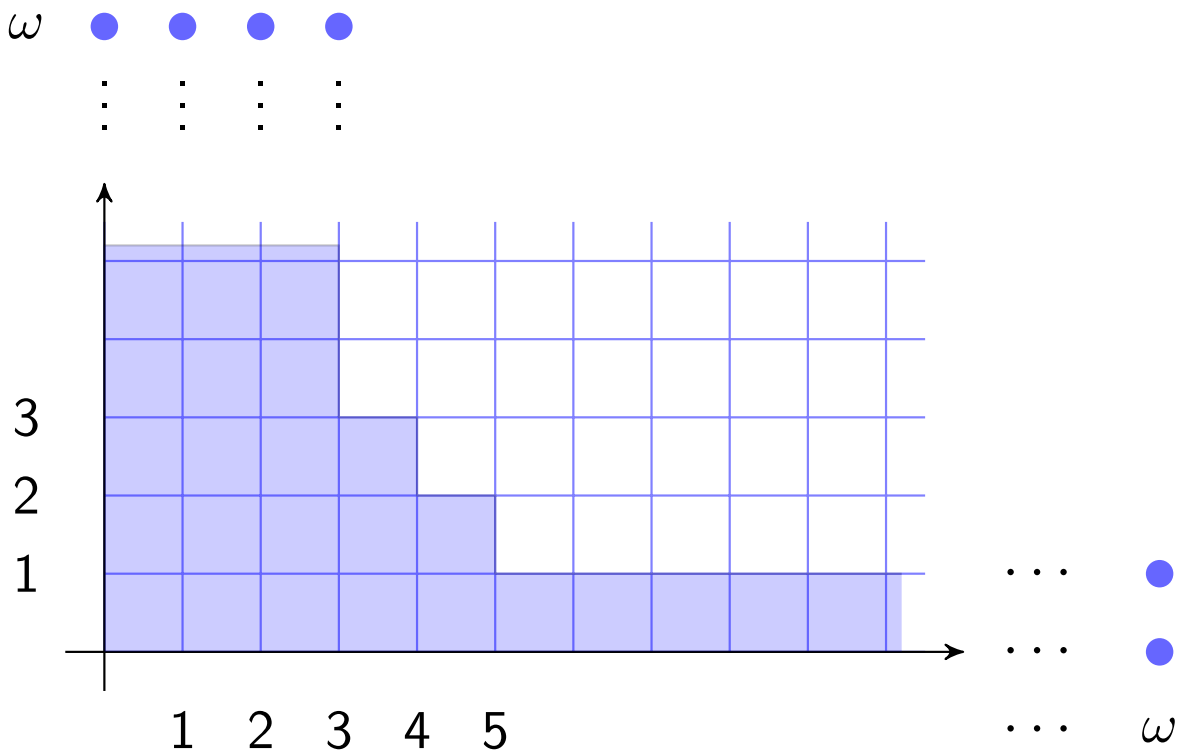
Limits in N_ω^d

- Let $\omega \notin \mathbb{N}$ and let $\mathbb{N}_\omega = \mathbb{N} \cup \{\omega\}$.
- Notion of converging sequence $(\ell_n)_n$ of elements of $\mathbb{N}_\omega$.
- Generalized componentwise to sequences $(\boldsymbol{x}_n)_n$ of vectors of $\mathbb{N}_\omega^d$.
- Given $\boldsymbol{M} \subseteq \mathbb{N}_\omega^d$, we write Lim $\boldsymbol{M}$ for the set of limits of $\boldsymbol{M}$.

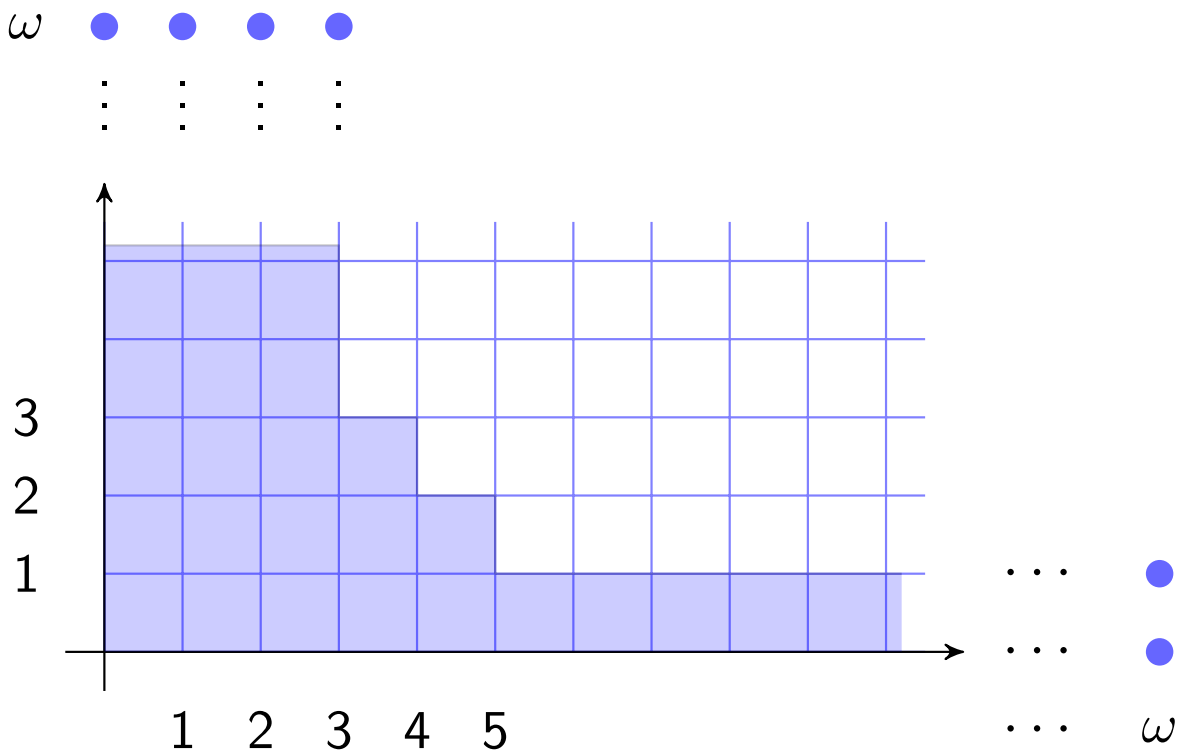
Limits: example



Limits: example



Limits: example



- Remark: $M \subseteq Lim M$.
- If D is downward closed: $D = Lim D \cap \mathbb{N}^d$.

Bases for downward closed sets

- Basis of $D \subseteq \mathbb{N}_\omega^d$: finite set $B \subseteq \mathbb{N}_\omega^d$ such that $\operatorname{Lim} D = {\downarrow}B$.
- [FG-L09]: any downward closed set $D \subseteq \mathbb{N}^d$ admits a basis.
- Canonical basis: minimal for $\subseteq$.
- Extends to any downward closed set $D \subseteq \mathbb{N}_\omega^d$.
  [Just because by def., D and $D \cap \mathbb{N}^d$ have same basis]

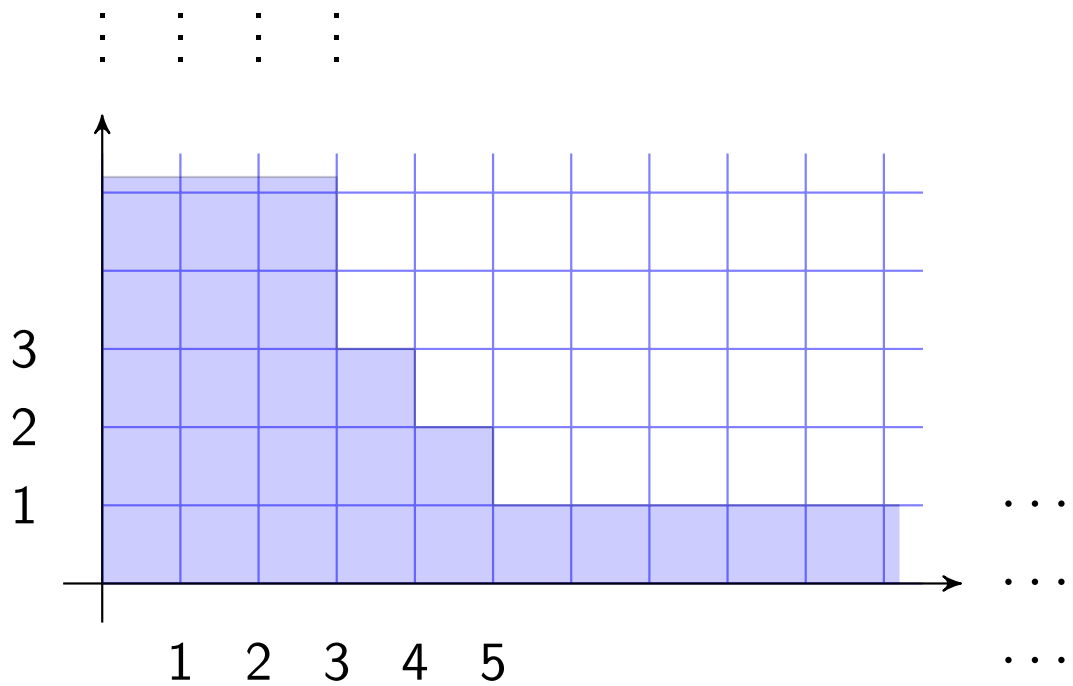
Bases for downward closed sets

- Basis of $D \subseteq \mathbb{N}_\omega^d$: finite set $B \subseteq \mathbb{N}_\omega^d$ such that $\mathrm{Lim}\, D = {\downarrow}B$.
- [FG-L09]: any downward closed set $D \subseteq \mathbb{N}^d$ admits a basis.
- Canonical basis: minimal for $\subseteq$.
- Extends to any downward closed set $D \subseteq \mathbb{N}_\omega^d$.
  [Just because by def., D and $D \cap \mathbb{N}^d$ have same basis]

Theorem (KM progress theorem)

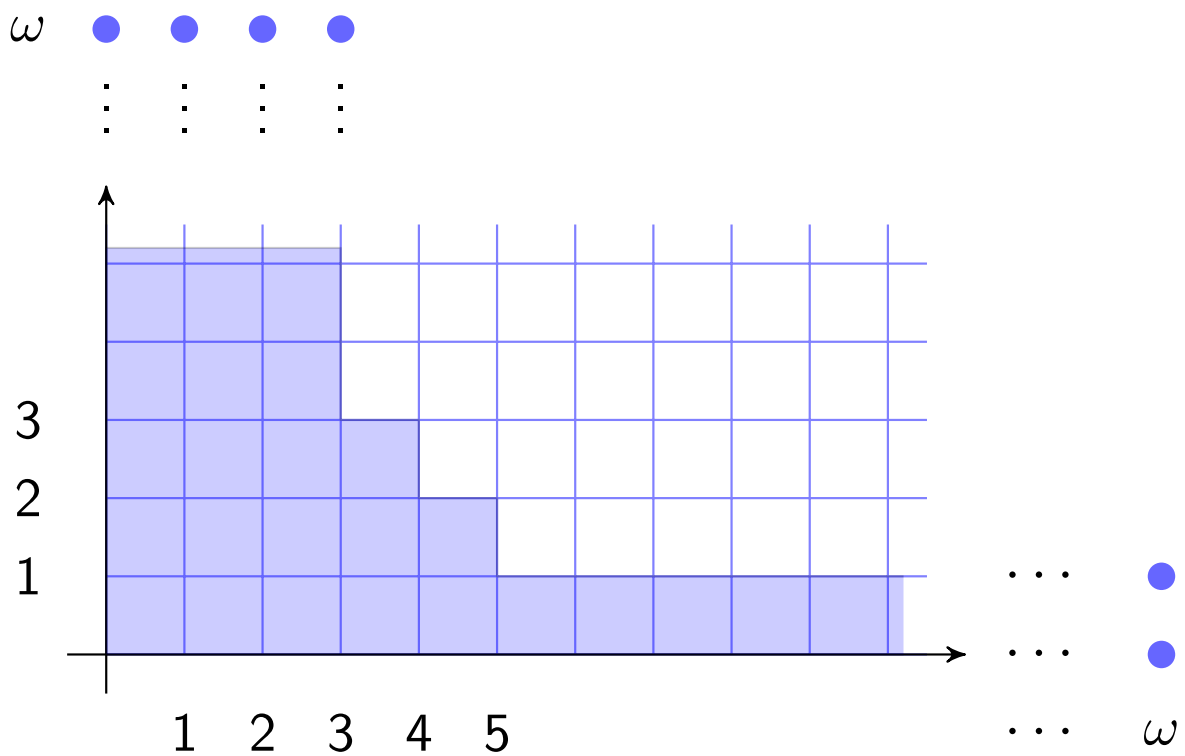
Let $\mathcal{V}$ be a VAS. Given $f \in \mathbb{N}_\omega^d$, one can compute a basis of $\Downarrow_f \mathrm{Reach}(\mathcal{V})$.

Bases for downward closed sets: example



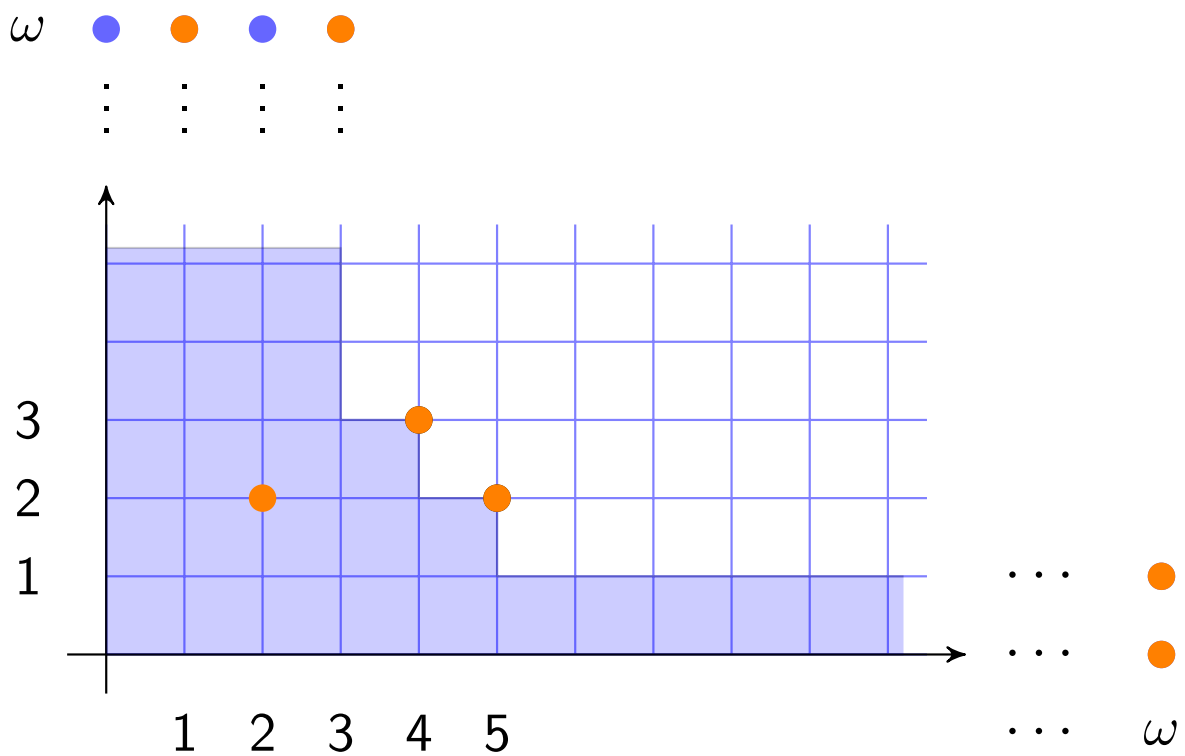
- D

Bases for downward closed sets: example



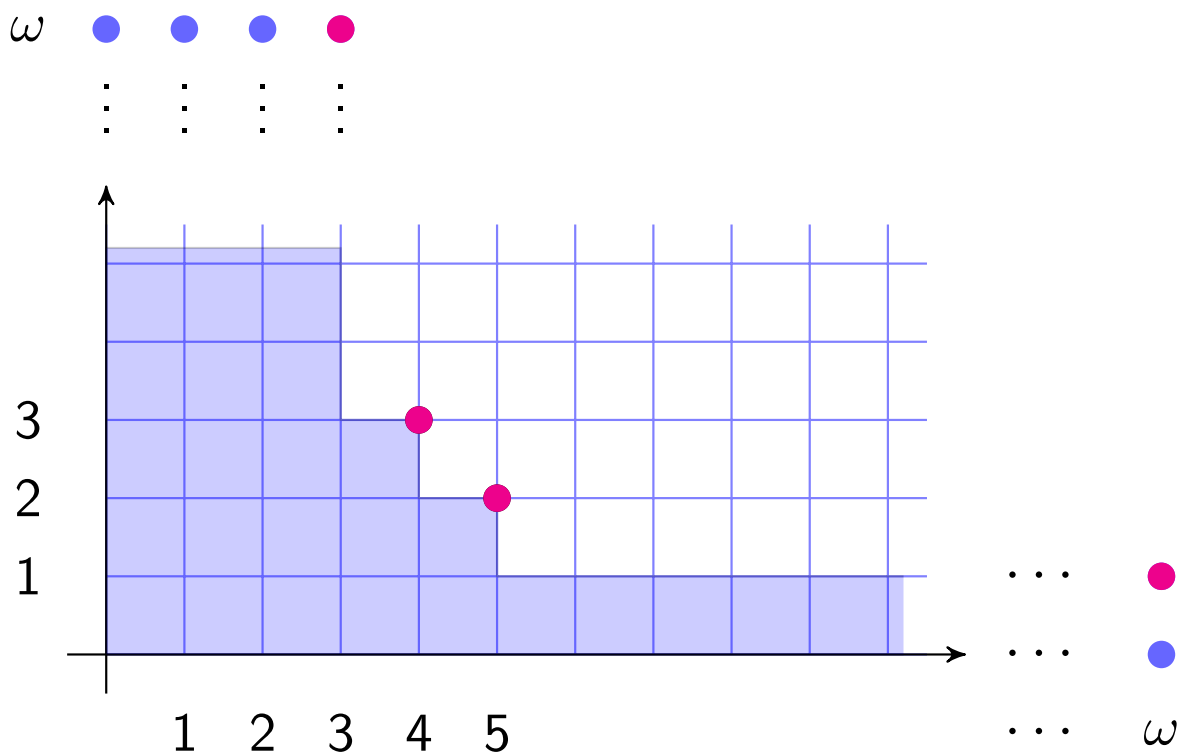
- D
- $\mathrm{Lim}\, D$

Bases for downward closed sets: example



- D
- Lim D
- A basis.

Bases for downward closed sets: example



- D
- Lim D
- A basis.
- The minimal, canonical basis.

Adapted Karp-Miller algorithm: principle

Theorem (KM progress theorem)

Let $\mathcal{V}$ be a VAS. Given $\boldsymbol{f} \in \mathbb{N}_\omega^d$, one can compute a basis of $\Downarrow_{\boldsymbol{f}} \mathrm{Reach}(\mathcal{V})$.

- We use KM progress theorem to compute the cover of a VASS_0 $\mathcal{V}_z$.
- We apply it only with $\boldsymbol{f} = (0, \omega, \omega, \ldots, \omega)$.
- KM progress theorem allows to perform meta steps.
- A meta steps computes in one shot an information similar to that computed by usual KM algorithm, preserving 0 value on component 1.

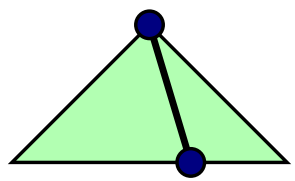
Adapted Karp-Miller algorithm: principle

Theorem (KM progress theorem)

Let $\mathcal{V}$ be a VAS. Given $f \in \mathbb{N}_\omega^d$, one can compute a basis of $\Downarrow_f \mathrm{Reach}(\mathcal{V})$.

- We use KM progress theorem to compute the cover of a VASS_0 $\mathcal{V}_z$.
- We apply it only with $f = (0, \omega, \omega, \ldots, \omega)$.
- KM progress theorem allows to perform meta steps.
- A meta steps computes in one shot an information similar to that computed by usual KM algorithm, preserving 0 value on component 1.

Meta step (no zero-test)



Adapted Karp-Miller algorithm: principle

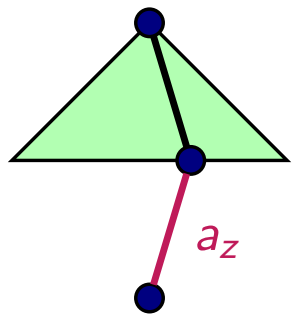
Theorem (KM progress theorem)

Let $\mathcal{V}$ be a VAS. Given $\boldsymbol{f} \in \mathbb{N}_\omega^d$, one can compute a basis of $\Downarrow_{\boldsymbol{f}} \mathrm{Reach}(\mathcal{V})$.

- We use KM progress theorem to compute the cover of a VASS_0 $\mathcal{V}_z$.
- We apply it only with $\boldsymbol{f} = (0, \omega, \omega, \ldots, \omega)$.
- KM progress theorem allows to perform meta steps.
- A meta steps computes in one shot an information similar to that computed by usual KM algorithm, preserving 0 value on component 1.

Meta step (no zero-test)

Zero-test



Adapted Karp-Miller algorithm: principle

Theorem (KM progress theorem)

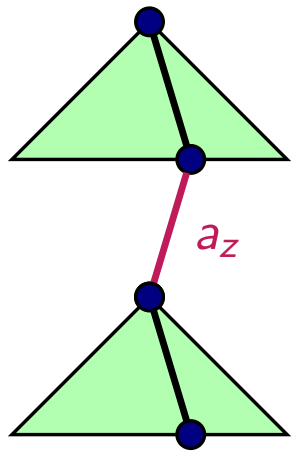
Let $\mathcal{V}$ be a VAS. Given $\boldsymbol{f} \in \mathbb{N}_\omega^d$, one can compute a basis of $\Downarrow_{\boldsymbol{f}} \mathrm{Reach}(\mathcal{V})$.

- We use KM progress theorem to compute the cover of a VASS_0 $\mathcal{V}_z$.
- We apply it only with $\boldsymbol{f} = (0, \omega, \omega, \ldots, \omega)$.
- KM progress theorem allows to perform meta steps.
- A meta steps computes in one shot an information similar to that computed by usual KM algorithm, preserving 0 value on component 1.

Meta step (no zero-test)

Zero-test

Meta step (no zero-test)

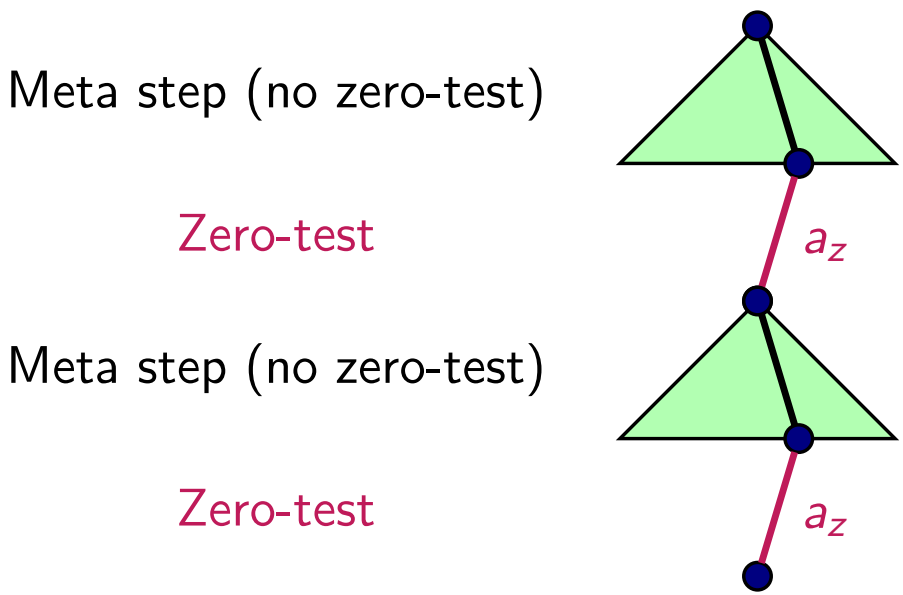


Adapted Karp-Miller algorithm: principle

Theorem (KM progress theorem)

Let $\mathcal{V}$ be a VAS. Given $f \in \mathbb{N}_\omega^d$, one can compute a basis of $\Downarrow_f \mathrm{Reach}(\mathcal{V})$.

- We use KM progress theorem to compute the cover of a VASS_0 $\mathcal{V}_z$.
- We apply it only with $f = (0, \omega, \omega, \ldots, \omega)$.
- KM progress theorem allows to perform meta steps.
- A meta steps computes in one shot an information similar to that computed by usual KM algorithm, preserving 0 value on component 1.



Adapted Karp-Miller algorithm: principle

Theorem (KM progress theorem)

Let $\mathcal{V}$ be a VAS. Given $f \in \mathbb{N}_\omega^d$, one can compute a basis of $\Downarrow_f \mathrm{Reach}(\mathcal{V})$.

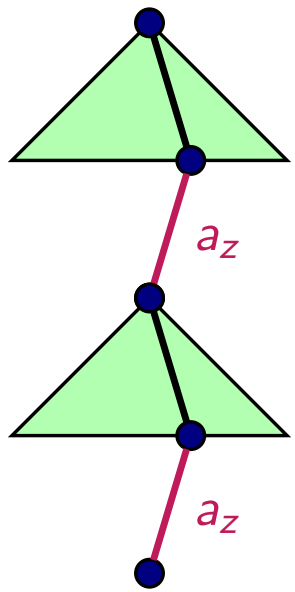
- We use KM progress theorem to compute the cover of a VASS_0 $\mathcal{V}_z$.
- We apply it only with $f = (0, \omega, \omega, \ldots, \omega)$.
- KM progress theorem allows to perform meta steps.
- A meta steps computes in one shot an information similar to that computed by usual KM algorithm, preserving 0 value on component 1.

Meta step (no zero-test)

Zero-test

Meta step (no zero-test)

Zero-test



Corresponds to factorizations

$u_1 a_z u_2 a_z \ldots$

Adapted Karp-Miller algorithm: outline

- We start with $x_{in} \in \{0\} \times \mathbb{N}^{d-1}$ and $\delta(a_z) \in \{0\} \times \mathbb{Z}^{d-1}$.
- Algorithm first computes a basis of $\Downarrow_f \mathrm{Reach}(\mathcal{V}_z)$.
- As in Karp-Miller's algorithm, build a tree whose nodes are labeled.

1. Start with tree $\mathcal{T}$ single root labeled by x_{in}.
2. For each unprocessed leaf n labeled x:
   - 2.1 Perform standard acceleration at n. Call y the new label.
   - 2.2 Fire the zero-test at n, y, if possible.
   - 2.3 Expand tree at n with the minimal basis of $\Downarrow_f \mathrm{Reach}(\mathcal{V}_z(y))$.

- Acceleration guarantees termination.

Adapted Karp-Miller algorithm (finish off)

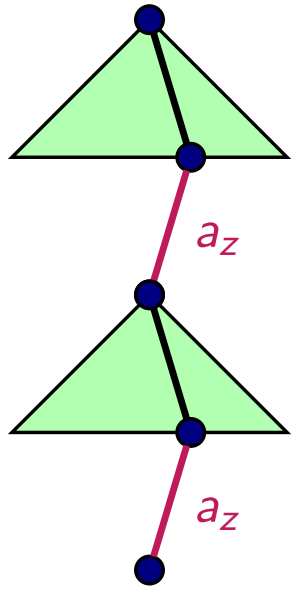
- The algorithm computes only a basis of $\Downarrow_f \mathrm{Reach}(\mathcal{V}_z)$.
- To compute the cover $\mathrm{Cover}(\mathcal{V}_z)$, we have to run the usual Karp-Miller algorithm from the obtained set.

Meta step (no zero-test)

Zero-test

Meta step (no zero-test)

Zero-test



Corresponds to factorizations

$u_1 a_z u_2 a_z$

Adapted Karp-Miller algorithm (finish off)

- The algorithm computes only a basis of $\Downarrow_f \mathrm{Reach}(\mathcal{V}_z)$.
- To compute the cover $\mathrm{Cover}(\mathcal{V}_z)$, we have to run the usual Karp-Miller algorithm from the obtained set.

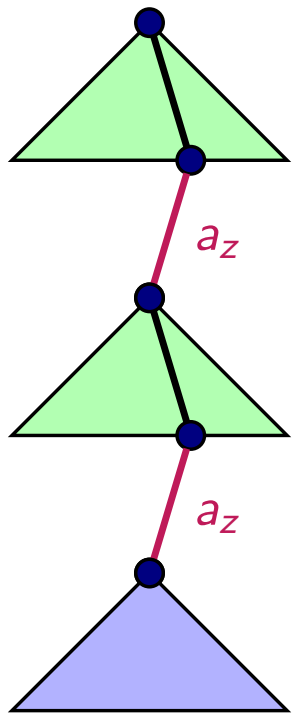
Meta step (no zero-test)

Zero-test

Meta step (no zero-test)

Zero-test

Last stage



Corresponds to factorizations

$u_1 a_z u_2 a_z u$

Proof outline of KM progress theorem

We work with $M = \mathrm{Lim}\,\mathrm{Reach}(\mathcal{V})$.

1. M is (effectively) recursive.

2. Implies that $\left\{(P, \boldsymbol{y}) \in 2^{\{1,\ldots,d\}} \times \mathbb{N}_\omega^d \mid \boldsymbol{y} \in \downarrow_{\leqslant_P} M\right\}$ is recursive.
   For given P, build $\mathcal{V}_P$ from $\mathcal{V}$ with $\mathrm{Reach}(\mathcal{V}_P) = \mathrm{Cover}_{\leqslant_P}(\mathcal{V})$, making non-$P$ positions lossy.

3. Implies that $\left\{(\boldsymbol{f}, \boldsymbol{y}) \in \mathbb{N}_\omega^d \times \mathbb{N}_\omega^d \mid \boldsymbol{y} \in \Downarrow_{\boldsymbol{f}} M\right\}$ is recursive.

4. Implies, by Valk & Jantzen's Lemma, that $\Downarrow_{\boldsymbol{f}} M$ has computable basis.

A note on limit sets

- For $M \subseteq \mathbb{N}^d$, an algorithm for membership in $\operatorname{Lim} M$ yields an algorithm for membership in $M = \mathbb{N}^d \cap \operatorname{Lim} M$.
- Conversely, an algorithm for $\operatorname{Lim} M$ gives in general more information.

1. It may happen that M recursive, and $\operatorname{Lim} M$ not recursive.

   $T_0, T_1, \ldots$ enumeration of TM, and $M = \{(k, \ell, \alpha(k,\ell)) \mid k, \ell \geqslant 0\}$

   with $\alpha(k,\ell) = \big|\{j \leqslant k \mid T_j \text{ halts in at most } \ell \text{ steps on } \varepsilon\}\big|$

2. Even if $\operatorname{Lim} M$ is recursive, it may be impossible to derive an algorithm for membership in $\operatorname{Lim} M$ from one such algorithm for M.

   M: reachability set of a lossy counter machine.

   Effective algorithm for M, and $\operatorname{Lim} M$ recursive,

   No algorithm deciding $x \in \operatorname{Lim} M$ from input LCM and $x \in \mathbb{N}_\omega^d$.

Limits of reachable states are computable for VAS

Theorem (BFLZ'10)

Given a VAS $\mathcal{V}$ and $x \in \mathbb{N}_\omega^d$, one can decide if x is a limit of reachable states.

Proof Main ingredients

- Reachability algorithm as an oracle.
- Limits witnessed with pumping argument.
- Pumping argument proved using Higman's Lemma.

Limits of reachable states are computable for VAS

Theorem (BFLZ'10)

Given a VAS $\mathcal{V}$ and $x \in \mathbb{N}_\omega^d$, one can decide if x is a limit of reachable states.

- Easy direction: $\operatorname{Lim}\operatorname{Reach}(\mathcal{V})$ is (effectively) co-RE.
- For $y \in \mathbb{N}_\omega^d$, let $y[\ell]$ be obtained from y by replacing all ω's by ℓ.

Lemma (BFLZ'10)

From $\mathcal{V}$ and y, one can build $\mathcal{V}_y$ st.

$$y \notin \operatorname{Lim}\operatorname{Reach}(\mathcal{V}) \iff \exists \ell \in \mathbb{N},\ y_\ell \notin \operatorname{Reach}(\mathcal{V}_y).$$

Proof.

Construction of $\mathcal{V}_y$: just make ω positions lossy. □

Limits of reachable states for VAS

Theorem (BFLZ'10)

Given a VAS $\mathcal{V}$ and $x \in \mathbb{N}_\omega^d$, one can decide if x is a limit of reachable states.

Limits of reachable states for VAS

Theorem (BFLZ'10)

Given a VAS $\mathcal{V}$ and $x \in \mathbb{N}_\omega^d$, one can decide if x is a limit of reachable states.

- Difficult direction: $\operatorname{Lim}\operatorname{Reach}(\mathcal{V})$ is (effectively) RE.
- Want recursively enumerable witnesses.
- Witnesses are productive sequences.

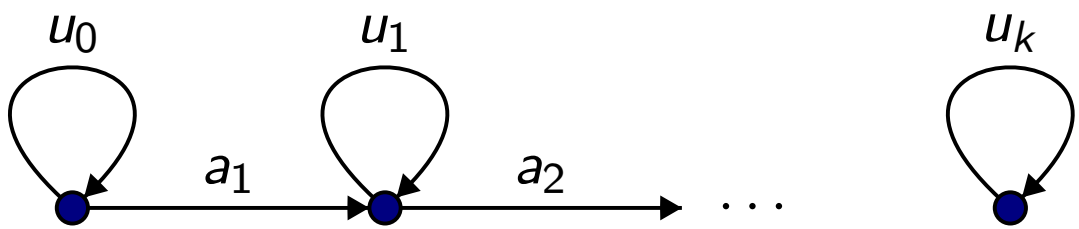
Productive sequences

Productive sequence

$\pi = (u_i)_{0 \leqslant i \leqslant k}$ is productive for a word $a_1 \cdots a_k$ if the words

$$u_0^n a_1 u_1^n \cdots a_k u_k^n, \qquad n \geqslant 1$$

are all fireable.



Equivalently

- the partial sums $\delta(u_0) + \cdots + \delta(u_j)$ are nonnegative.
- the word $u_0 a_1 u_1 \cdots a_k u_k$ is fireable.

Limits of reachable states for VAS

Proposition

$$\text{Lim Reach}(\mathcal{V}) = \left\{ x_{in} + \delta(v) + \omega\delta(\pi) \mid \exists v \in A^*, \exists \pi \text{ productive in } \mathcal{V} \text{ for } v \right\}.$$

- Higman's Lemma on nontrivial ordering on reachable states.
- Allow to find loops of productive witness, and pump.

$$x_{in} \xrightarrow{a_1} y_1 \cdots \xrightarrow{a_k} y_k = x_m$$

$$x_{in} \xrightarrow{u_0 a_1} z_1 \cdots \xrightarrow{u_{k-1} a_k} z_k \xrightarrow{u_k} x_n$$

- Consequence: $\text{Lim Reach}(\mathcal{V})$ is (effectively) RE.

From LTL to Büchi condition

- A control state q is Büchi if there is a run visiting q infinitely often.
- Labeled VASS_0: each action a has an associated value $\gamma(a) \in \Sigma^*$.
  A run $w = a_1 a_2 \cdots \in A^\omega$ induces a trace $\gamma(a_1)\gamma(a_2)\cdots \in \Sigma^\omega$.
- LTL model-checking: given $\varphi \in \mathrm{LTL}_\Sigma$ and labeled VASS_0 $\mathcal{V}$, does there exist a trace w such that $w \models \varphi$?
- In synchronized product $\mathcal{A}_\varphi \times \mathcal{V}$, check if a final control state is Büchi.

From Büchi condition to increasing loops

- A q_f-loop is a pair $(x, y) \in \{0\} \times \mathbb{N}^{d-1}$ such that $(q_f, x) \xrightarrow{+} (q_f, y)$.
- A q_f-loop is
  - increasing if $x \leqslant y$,
  - reachable if $(q_{in}, x_{in}) \xrightarrow{*} (q_f, x)$,
  - ℓ-bounded, for $\ell \in \{0\} \times \mathbb{N}_\omega^{d-1}$, if $x \leqslant \ell$.
- Remark. A reachable increasing q_f-loop witnesses that q_f is Büchi.

Deciding Büchi condition

Proposition (R. Bonnet)

There are reductions $1 \leqslant 2 \leqslant 3$.

1. q_f is Büchi along a run where the zero-test is fired ∞-often.
2. There is a reachable increasing q_f-loop.
3. There is an ℓ-bounded increasing q_f-loop, where (q_f, ℓ) is in the minimal basis of $\Downarrow_f \mathsf{Reach}(\mathcal{V})$.

Theorem R. Bonnet [RP'11]

Testing if a state is Büchi is decidable.

Proof.

Property 3 is decidable. But proof uses reachability of VASS_0 ☹. □

Conclusion

- Generalization of KM algorithm, via limit set of reachable states.
- Several algorithms for VASS_0: LTL, place boundedness, regularity.

- "Classical" properties of VASS are preserved for VASS_0.
- Open LTL algorithm relies on reachability for VASS (KM, unavoidable) and on reachability for VASS_0. Can we remove this last dependency?
- Open One stack?