

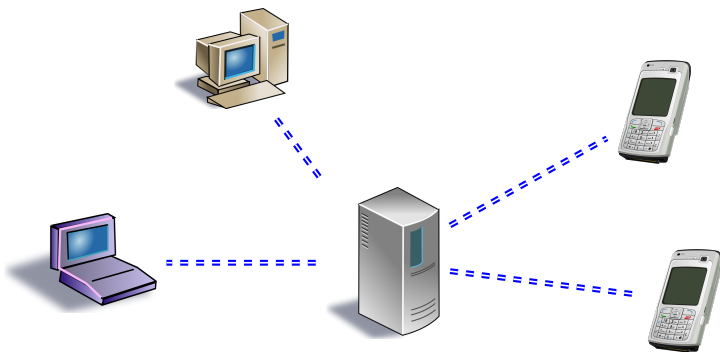
On the Context-Freeness Problem for Vector Addition Systems

Jérôme Leroux, Vincent Penelle, M. Praveen and Grégoire Sutre

Univ. Bordeaux & CNRS, LaBRI, UMR 5800, Talence, France

LIFO Seminar, University of Orléans, May 12th 2014

Verification of Concurrent Systems (1)



- ⇒ Concurrent systems everywhere!
- ⇒ Hard to design correctly
 - Complex and **unforeseen** interactions between components
 - Need for to automated verification tools

Model-checking

$$\mathcal{M} \stackrel{?}{\models} \varphi$$

Need models \mathcal{M} with

- enough expressive power to represent the system to verify
- support for automatic verification (model-checking decidable)

Vector Addition Systems \simeq Petri nets

- Classical model for (parametrized) concurrent systems
 - ▶ Rendez-vous synchronization
 - ▶ Asynchronous communication via unbounded unordered buffers
 - ▶ Dynamic process creation
- Fundamental class that is often used as a toolbox

Table of Contents

- 1 Vector Addition Systems
- 2 Decidability of the Context-Freeness Problem for VAS
- 3 Simple Witnesses of Non-Context-Freeness
- 4 A Relational Trace Logic for VAS with EXPSPACE Solvability
- 5 Conclusion

Table of Contents

- 1 Vector Addition Systems
- 2 Decidability of the Context-Freeness Problem for VAS
- 3 Simple Witnesses of Non-Context-Freeness
- 4 A Relational Trace Logic for VAS with EXPSPACE Solvability
- 5 Conclusion

Vector Addition Systems with States in a Nutshell

A vector addition system with states is a finite-state automaton that is

- equipped with finitely many **counters** x, y, \dots
- counters range over the set \mathbb{N} of **natural numbers**
- counter operations are:

- ▶ increment:

$$x := x + 1$$

- ▶ guarded decrement:

$$\text{assert}(x > 0) ; x := x - 1$$

Vector addition systems with states are similar to **Petri nets**

Definition

A **vector addition system** is a pair $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ where

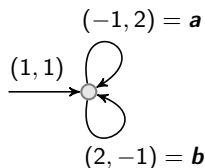
- $\mathbf{v}_{\text{init}} \in \mathbb{N}^d$: **initial vector**
- $\mathbf{A} \subseteq \mathbb{Z}^d$: finite set of **actions**

Definition

A **vector addition system** is a pair $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ where

- $\mathbf{v}_{\text{init}} \in \mathbb{N}^d$: **initial vector**
- $\mathbf{A} \subseteq \mathbb{Z}^d$: finite set of **actions**

$$\begin{aligned}\mathbf{A} &= \{(-1, 2), (2, -1)\} \\ \mathbf{v}_{\text{init}} &= (1, 1)\end{aligned}$$



Vector Addition Systems — Semantics

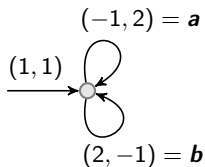
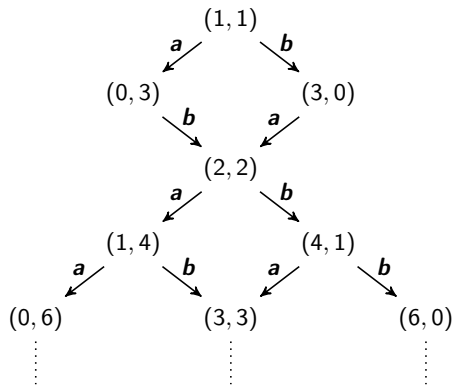
The **semantics** of a VAS $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ is the transition system $\langle \mathbb{N}^d, \mathbf{v}_{\text{init}}, \rightarrow \rangle$ whose transition relation \rightarrow is given by

$$\frac{\mathbf{a} \in \mathbf{A} \wedge \mathbf{v}' = \mathbf{v} + \mathbf{a} \geq \mathbf{0}}{\mathbf{v} \rightarrow \mathbf{v}'}$$

Vector Addition Systems — Semantics

The **semantics** of a VAS $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ is the transition system $\langle \mathbb{N}^d, \mathbf{v}_{\text{init}}, \rightarrow \rangle$ whose transition relation \rightarrow is given by

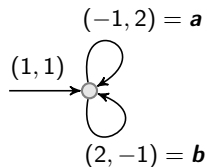
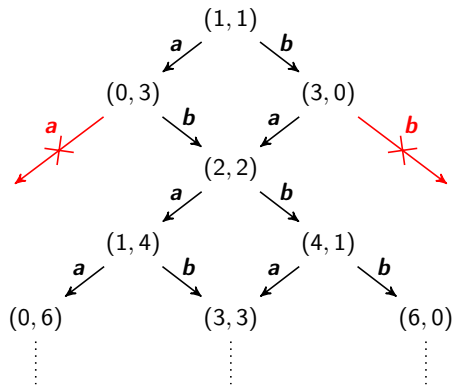
$$\frac{\mathbf{a} \in \mathbf{A} \wedge \mathbf{v}' = \mathbf{v} + \mathbf{a} \geq \mathbf{0}}{\mathbf{v} \rightarrow \mathbf{v}'}$$



Vector Addition Systems — Semantics

The **semantics** of a VAS $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ is the transition system $\langle \mathbb{N}^d, \mathbf{v}_{\text{init}}, \rightarrow \rangle$ whose transition relation \rightarrow is given by

$$\frac{\mathbf{a} \in \mathbf{A} \wedge \mathbf{v}' = \mathbf{v} + \mathbf{a} \geq \mathbf{0}}{\mathbf{v} \rightarrow \mathbf{v}'}$$



Definition

A **vector addition system with states** is a tuple $\langle Q, q_{\text{init}}, \mathbf{v}_{\text{init}}, \Delta \rangle$ where

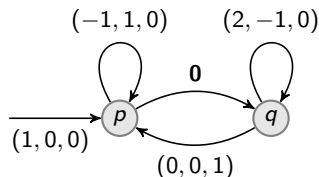
- Q : finite set of **states**
- $q_{\text{init}} \in Q$: **initial state**
- $\mathbf{v}_{\text{init}} \in \mathbb{N}^d$: **initial vector**
- $\Delta \subseteq Q \times \mathbb{Z}^d \times Q$: finite set of transition **rules**

Definition

A **vector addition system with states** is a tuple $\langle Q, q_{\text{init}}, \mathbf{v}_{\text{init}}, \Delta \rangle$ where

- Q : finite set of **states**
- $q_{\text{init}} \in Q$: **initial state**
- $\mathbf{v}_{\text{init}} \in \mathbb{N}^d$: **initial vector**
- $\Delta \subseteq Q \times \mathbb{Z}^d \times Q$: finite set of transition **rules**

$$\begin{aligned}
 Q &= \{p, q\} \\
 q_{\text{init}} &= p \\
 \mathbf{v}_{\text{init}} &= (1, 0, 0) \\
 \Delta &= \{(p, (-1, 1, 0), p), \dots\}
 \end{aligned}$$



Definition

A **vector addition system with states** is a tuple $\langle Q, q_{\text{init}}, \mathbf{v}_{\text{init}}, \Delta \rangle$ where

- Q : finite set of **states**
- $q_{\text{init}} \in Q$: **initial state**
- $\mathbf{v}_{\text{init}} \in \mathbb{N}^d$: **initial vector**
- $\Delta \subseteq Q \times \mathbb{Z}^d \times Q$: finite set of transition **rules**

$$\begin{array}{ccc}
 \text{VAS} & & \text{VASS} \\
 \langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle & \simeq & \langle \{q\}, q, \mathbf{v}_{\text{init}}, \{q\} \times \mathbf{A} \times \{q\} \rangle
 \end{array}$$

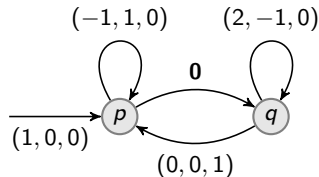
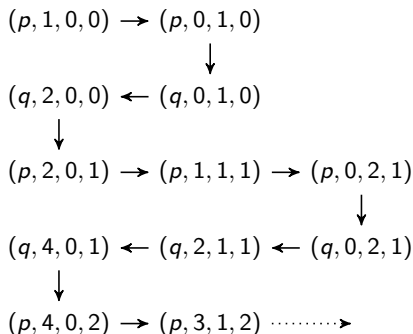
The **semantics** of a VASS $\langle Q, q_{\text{init}}, \mathbf{v}_{\text{init}}, \Delta \rangle$ is the transition system $\langle Q \times \mathbb{N}^d, (q_{\text{init}}, \mathbf{v}_{\text{init}}), \rightarrow \rangle$ whose transition relation \rightarrow is given by

$$\frac{(q, \mathbf{a}, q') \in \Delta \wedge \mathbf{v}' = \mathbf{v} + \mathbf{a} \geq \mathbf{0}}{(q, \mathbf{v}) \rightarrow (q', \mathbf{v}')}$$

Vector Addition Systems with States — Semantics

The **semantics** of a VASS $\langle Q, q_{\text{init}}, \mathbf{v}_{\text{init}}, \Delta \rangle$ is the transition system $\langle Q \times \mathbb{N}^d, (q_{\text{init}}, \mathbf{v}_{\text{init}}), \rightarrow \rangle$ whose transition relation \rightarrow is given by

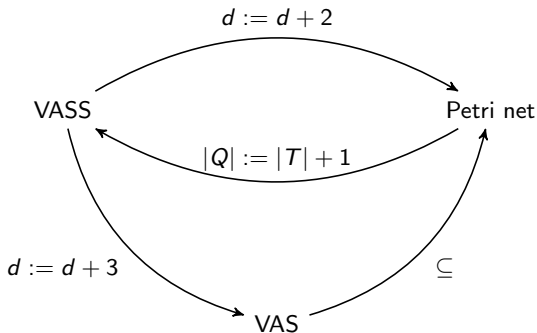
$$\frac{(q, \mathbf{a}, q') \in \Delta \wedge \mathbf{v}' = \mathbf{v} + \mathbf{a} \geq \mathbf{0}}{(q, \mathbf{v}) \rightarrow (q', \mathbf{v}')}$$



VAS \simeq Petri nets \simeq VASS

Additional Feature of Petri nets

Test $x \geq cst$ without modifying x



State of the Art — Reachability, Coverability, ...

Input: A VAS $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ and a final vector $\mathbf{v}_{\text{final}}$

Reachability Problem — Whether $\mathbf{v}_{\text{init}} \xrightarrow{*} \mathbf{v}_{\text{final}}$

Decidable [Mayr 81, Kosaraju 82], EXPSPACE-hard [Lipton 76]

Coverability Problem — Whether $\exists \mathbf{v} : \mathbf{v}_{\text{init}} \xrightarrow{*} \mathbf{v} \geq \mathbf{v}_{\text{final}}$

Decidable [Karp&Miller 69], EXPSPACE-complete [Rackoff 78]

Boundedness Problem — Whether $\{\mathbf{v} \in \mathbb{N}^d \mid \mathbf{v}_{\text{init}} \xrightarrow{*} \mathbf{v}\}$ is finite

Decidable [Karp&Miller 69], EXPSPACE-complete [Rackoff 78]

- Many EXPSPACE-complete problems (place boundedness, ...)
- Some undecidable problems (equality of reachability sets, ...)

Trace Language

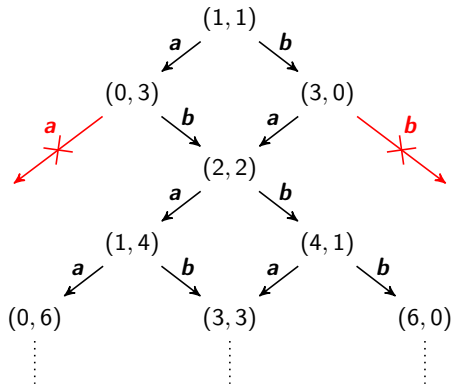
A **trace** of a VAS $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ is a sequence $\mathbf{a}_1, \dots, \mathbf{a}_n$ such that

$$\mathbf{v}_{\text{init}} \xrightarrow{\mathbf{a}_1} \mathbf{v}_1 \cdots \mathbf{v}_{n-1} \xrightarrow{\mathbf{a}_n} \mathbf{v}_n$$

Trace Language

A **trace** of a VAS $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ is a sequence $\mathbf{a}_1, \dots, \mathbf{a}_n$ such that

$$\mathbf{v}_{\text{init}} \xrightarrow{\mathbf{a}_1} \mathbf{v}_1 \cdots \mathbf{v}_{n-1} \xrightarrow{\mathbf{a}_n} \mathbf{v}_n$$

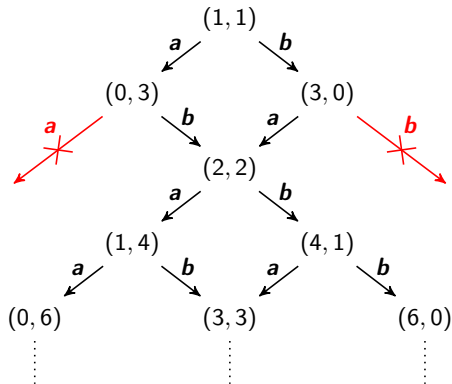


- **abb** is a trace
- **aa** is **not** a trace

Trace Language

A **trace** of a VAS $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ is a sequence $\mathbf{a}_1, \dots, \mathbf{a}_n$ such that

$$\mathbf{v}_{\text{init}} \xrightarrow{\mathbf{a}_1} \mathbf{v}_1 \cdots \mathbf{v}_{n-1} \xrightarrow{\mathbf{a}_n} \mathbf{v}_n$$



- **abb** is a trace
- **aa** is **not** a trace

Definition

The **trace language** is the set of all traces

The Regularity and Context-Freeness Problems

Definition

Input: A VAS $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$

Output: Is the trace language of $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ regular/context-free?

Regularity

- Decidable [Valk&Vidal-Naquet 81, Ginzburg&Yoeli 80]
- EXPSPACE-complete [Demri 10]
 - ▶ Witness: trace $u_1\sigma_1 \cdots u_k\sigma_k$ such that $L \cap (u_1\sigma_1^* \cdots u_k\sigma_k^*)$ is not regular
 - ▶ Length at most doubly-exponential (in $|\mathbf{A}|$)

The Regularity and Context-Freeness Problems

Definition

Input: A VAS $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$

Output: Is the trace language of $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ regular/context-free?

Regularity

- Decidable [Valk&Vidal-Naquet 81, Ginzburg&Yoeli 80]
- EXPSPACE-complete [Demri 10]
 - ▶ Witness: trace $u_1\sigma_1 \cdots u_k\sigma_k$ such that $L \cap (u_1\sigma_1^* \cdots u_k\sigma_k^*)$ is not regular
 - ▶ Length at most doubly-exponential (in $|\mathbf{A}|$)

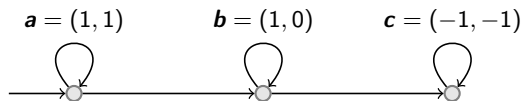
Context-Freeness

- Decidable [Schwer 92]
 - ▶ Complex criterion, based on the coverability graph
 - ▶ Intricate proof with flaws

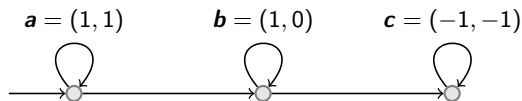
Table of Contents

- 1 Vector Addition Systems
- 2 Decidability of the Context-Freeness Problem for VAS**
- 3 Simple Witnesses of Non-Context-Freeness
- 4 A Relational Trace Logic for VAS with EXPSPACE Solvability
- 5 Conclusion

Example 1



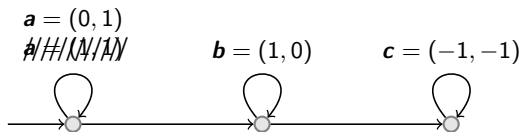
Example 1



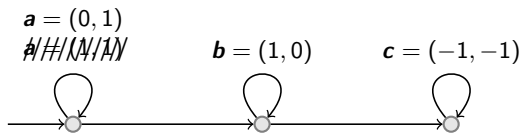
Its trace language is context-free:

$$L = \{\mathbf{a}^n \mathbf{b}^m \mathbf{c}^p \mid n \geq p \wedge m \geq 0\}$$

Example 2



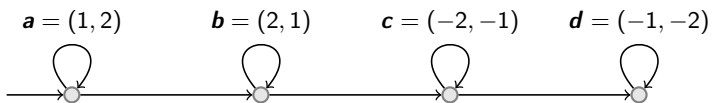
Example 2



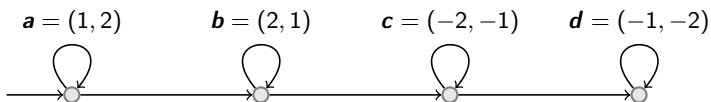
Its trace language is **not** context-free:

$$L = \{ \mathbf{a}^n \mathbf{b}^m \mathbf{c}^p \mid n \geq p \wedge \mathbf{a} \neq \emptyset \}$$

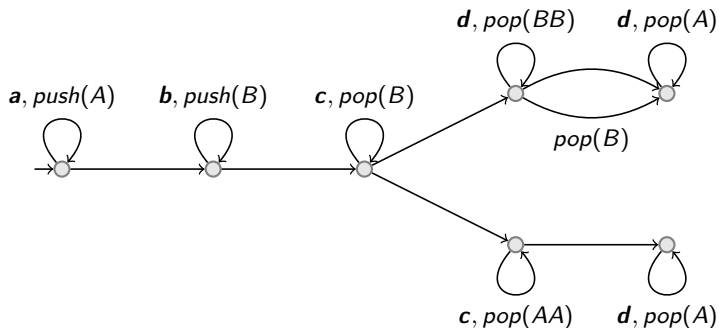
Example 3



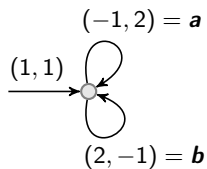
Example 3



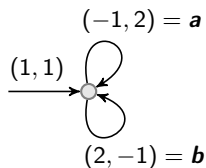
Its trace language is context-free:



Example 4 (Running Example)



Example 4 (Running Example)



Its trace language is **not** context-free:

$$L \cap (ab)^* a^* b^* = \{(ab)^n a^m b^p \mid n + 1 \geq m \wedge n + 1 + 2m \geq p\}$$

Simulation of a VAS by a Pushdown Automaton: Main Ideas

Goal: recognize the traces of a VAS with a pushdown automaton

Use the stack to store the values of the counters

For each vector read from the input tape

- Positive vector \rightarrow Push it onto the stack
- Non-positive vector \rightarrow Match it with the stack

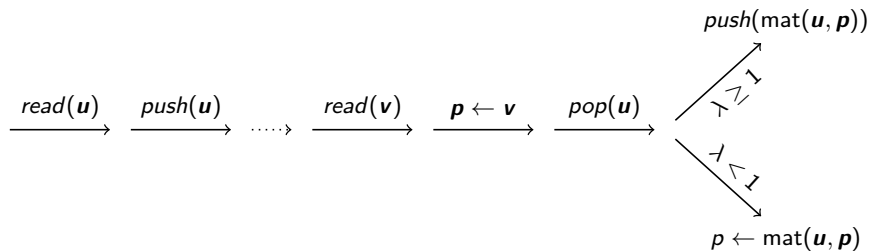
Problem: some components may be **lost** by matching

Match and Remainder

$\mathbf{u} \geq \mathbf{0}$, $\mathbf{v} \not\geq \mathbf{0}$, and λ the greatest rational such that $\mathbf{u} + \lambda\mathbf{v} \geq \mathbf{0}$,

$$\text{mat}(\mathbf{u}, \mathbf{v}) = \begin{cases} (1 - \lambda) \cdot \mathbf{v} & \text{if } \lambda < 1 \\ (1 - \frac{1}{\lambda}) \cdot \mathbf{u} & \text{if } \lambda \geq 1 \end{cases}$$

$$\text{rem}(\mathbf{u}, \mathbf{v}) = \mathbf{u} + \mathbf{v} - \text{mat}(\mathbf{u}, \mathbf{v}) \geq \mathbf{0}$$



Match and Remainder: Example with $\lambda < 1$

Take $\mathbf{u} = (1, 2, 1)$ and $\mathbf{v} = (-1, -3, 2)$

The greatest λ such that $\mathbf{u} + \lambda\mathbf{v} \geq \mathbf{0}$ is $\lambda = \frac{2}{3}$

After matching \mathbf{u} and \mathbf{v} , we are left with

$$\text{mat}(\mathbf{u}, \mathbf{v}) = (1 - \lambda) \cdot \mathbf{v} = \left(-\frac{1}{3}, -1, \frac{2}{3}\right)$$

The remainder is

$$\text{rem}(\mathbf{u}, \mathbf{v}) = \mathbf{u} + \lambda\mathbf{v} = \left(\frac{1}{3}, 0, \frac{7}{3}\right)$$

Simulation of a VAS by a Pushdown Machine (Variables)

Global variables

- local buffer $\mathbf{w} \in \mathbf{A}^*$
- accumulated remainder $\mathbf{r} \in \mathbb{Q}_{\geq 0}^d$
- **stack** of extracted vectors in $\mathbb{Q}_{\geq 0}^d$

They represent the VAS configuration:

$$\mathbf{v}_{\text{init}} + \Delta(\mathbf{w}) + \mathbf{r} + \Delta(\mathbf{stack})$$

Initialize ()

- 1 $\mathbf{w} \leftarrow \varepsilon$
- 2 $\mathbf{r} \leftarrow \mathbf{0}$
- 3 **stack** $\leftarrow \varepsilon$

$$\Delta(\mathbf{a}_1 \cdots \mathbf{a}_n) = \mathbf{a}_1 + \cdots + \mathbf{a}_n$$

Simulation of a VAS by a Pushdown Machine (Read)

VAS configuration: $\mathbf{v}_{\text{init}} + \Delta(\mathbf{w}) + \mathbf{r} + \Delta(\text{stack})$

Read ($\mathbf{a} \in \mathbf{A}$)

```
1  if  $\mathbf{v}_{\text{init}} + \Delta(\mathbf{w}) + \mathbf{r} + \Delta(\text{stack}) + \mathbf{a} \geq 0$  then
2       $\mathbf{w} \leftarrow \mathbf{w} \cdot \mathbf{a}$ 
3      Simplify ()
4  else
5      reject
```

Simulation of a VAS by a Pushdown Machine (Simplify)

Simplify ()

```
1  while  $\Delta(\sigma) + \mathbf{r} + \Delta(\text{stack}) \geq \mathbf{0}$  for some suffix  $\sigma \neq \varepsilon$  of  $\mathbf{w}$  do
2      Pick such a suffix  $\sigma$ 
3       $\mathbf{w} \leftarrow \mathbf{w} \cdot \sigma^{-1}$ 
4       $\mathbf{p} \leftarrow \Delta(\sigma)$ 
5      while  $\mathbf{p} \not\geq \mathbf{0}$  do
6          if  $\text{stack}$  is empty then
7              fail
8          else
9              pop  $\gamma$  from  $\text{stack}$ 
10              $(\mathbf{p}, \mathbf{r}) \leftarrow (\text{mat}(\gamma, \mathbf{p}), \mathbf{r} + \text{rem}(\gamma, \mathbf{p}))$ 
11             if  $\mathbf{p}(i) > 0 \Rightarrow \mathbf{r}(i) > 0$  for every index  $i$  then
12                  $\mathbf{r} \leftarrow \mathbf{r} + \mathbf{p}$ 
13             else
14                 push  $\mathbf{p}$  onto  $\text{stack}$ 
```

Properties of the Simulation

Proposition

*If **fail** is not reachable, then the pushdown machine recognizes the trace language of the VAS*

Proof: easy

Proposition

The language recognized by the pushdown machine is context-free

Proof: next slide

Corollary

*If **fail** is not reachable, the trace language of the VAS is context-free*

Proof: The Pushdown Machine Recognizes a CFL

Proposition

The set of reachable values of w is finite

Proposition

The reachable alphabet for $stack$ is finite

Proof: The Pushdown Machine Recognizes a CFL

Proposition

The set of reachable values of w is finite

Proposition

The reachable alphabet for $stack$ is finite

- ⇨ Standard PDA, with stack alphabet $\Gamma \subseteq \mathbb{Q}_{\geq 0}^d$, augmented with:
- counters $\mathbf{r} \in \mathbb{Q}_{\geq 0}^d$, updated by assignments $\mathbf{r} \leftarrow \mathbf{r} + \mathbf{v}$ with $\mathbf{v} \geq \mathbf{0}$
 - tests: $\mathbf{r}(i) + \Delta(\mathit{stack})(i) \# z$ and $\mathbf{r}(i) \# z$ where $\# \in \{\leq, \geq\}$

Proof: The Pushdown Machine Recognizes a CFL

Proposition

The set of reachable values of \mathbf{w} is finite

Proposition

The reachable alphabet for \mathbf{stack} is finite

- ⇒ Standard PDA, with stack alphabet $\Gamma \subseteq \mathbb{Q}_{\geq 0}^d$, augmented with:
- counters $\mathbf{r} \in \mathbb{Q}_{\geq 0}^d$, updated by assignments $\mathbf{r} \leftarrow \mathbf{r} + \mathbf{v}$ with $\mathbf{v} \geq \mathbf{0}$
 - tests: $\mathbf{r}(i) + \Delta(\mathbf{stack})(i) \# z$ and $\mathbf{r}(i) \# z$ where $\# \in \{\leq, \geq\}$

Replace $\mathbb{Q}_{\geq 0}^d$ by $\mathbb{N}_{\geq 0}^d$ and let $K \in \mathbb{N}$ be the maximum z of the tests

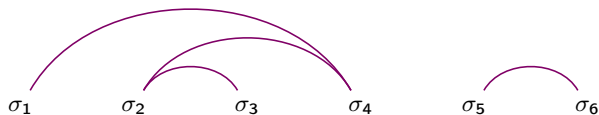
Abstract by \top components of \mathbf{r} and $\Delta(\mathbf{stack})$ larger than K

⇒ Store \mathbf{r} in the state and $\Delta(\mathbf{stack})$ within the stack

Table of Contents

- 1 Vector Addition Systems
- 2 Decidability of the Context-Freeness Problem for VAS
- 3 Simple Witnesses of Non-Context-Freeness**
- 4 A Relational Trace Logic for VAS with EXPSPACE Solvability
- 5 Conclusion

Matching Schemes



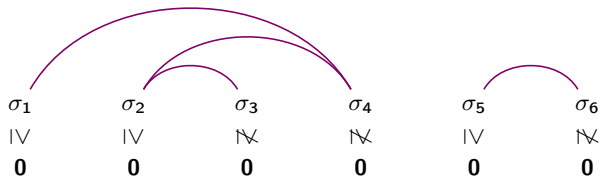
Definition

A **matching scheme** is a tuple $(\sigma_1, \dots, \sigma_k, U)$ such that

- $\sigma_1, \dots, \sigma_k$: words in \mathbf{A}^*
- U : binary relation on $\{1, \dots, k\}$ that is **nested**:

$$(s, t) \in U \quad \Rightarrow \quad s \leq t$$
$$(r, t) \in U \wedge (s, u) \in U \quad \Rightarrow \quad \neg(r < s < t < u)$$

Matching Schemes



Definition

A **matching scheme** is a tuple $(\sigma_1, \dots, \sigma_k, U)$ such that

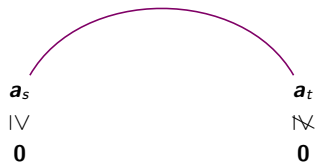
- $\sigma_1, \dots, \sigma_k$: words in \mathbf{A}^*
- U : binary relation on $\{1, \dots, k\}$ that is **nested**:

$$(s, t) \in U \Rightarrow s \leq t$$
$$(r, t) \in U \wedge (s, u) \in U \Rightarrow \neg(r < s < t < u)$$

- For every $(s, t) \in U$, $\Delta(\sigma_s) \geq \mathbf{0}$ and $\Delta(\sigma_t) \not\geq \mathbf{0}$

Loss by Matching $(s, t) \in U$

Consider the case of single actions: $\sigma_s = \mathbf{a}_s$ and $\sigma_t = \mathbf{a}_t$

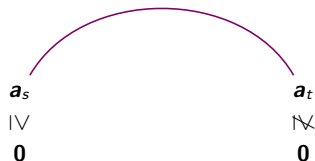


$$\begin{aligned}\mathbf{a}_s &= (1, 1) \\ \mathbf{a}_t &= (-1, -2)\end{aligned}$$

Use one \mathbf{a}_s to do as many \mathbf{a}_t as possible. What do we lose?

Loss by Matching $(s, t) \in U$

Consider the case of single actions: $\sigma_s = \mathbf{a}_s$ and $\sigma_t = \mathbf{a}_t$



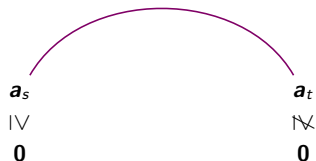
$$\begin{aligned}\mathbf{a}_s &= (1, 1) \\ \mathbf{a}_t &= (-1, -2) \\ \lambda_{s,t} &= \frac{1}{2}\end{aligned}$$

Use one \mathbf{a}_s to do as many \mathbf{a}_t as possible. What do we lose?

Ratio: $\lambda_{s,t} = \max \{ \lambda \in \mathbb{Q} \mid \mathbf{a}_s + \lambda \mathbf{a}_t \geq \mathbf{0} \}$

Loss by Matching $(s, t) \in U$

Consider the case of single actions: $\sigma_s = \mathbf{a}_s$ and $\sigma_t = \mathbf{a}_t$



$$\mathbf{a}_s = (1, 1)$$

$$\mathbf{a}_t = (-1, -2)$$

$$\lambda_{s,t} = \frac{1}{2}$$

$$\text{lost}(s, t) = \{1\}$$

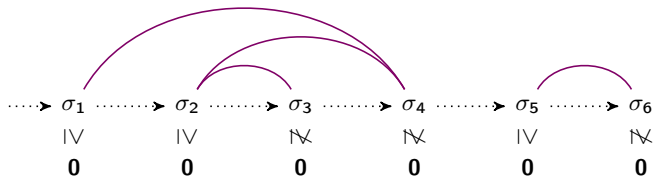
Use one \mathbf{a}_s to do as many \mathbf{a}_t as possible. What do we lose?

Ratio:
$$\lambda_{s,t} = \max \{ \lambda \in \mathbb{Q} \mid \mathbf{a}_s + \lambda \mathbf{a}_t \geq \mathbf{0} \}$$

Definition

$$\text{lost}(s, t) = \|\mathbf{a}_s + \lambda_{s,t} \mathbf{a}_t\|^+ = \|\text{rem}(\mathbf{a}_s, \mathbf{a}_t)\|^+$$

Witnesses of Non-Context-Freeness



Definition

A **witness of non-context-freeness** is a tuple $(\sigma_1, \dots, \sigma_k, U)$ such that

- $(\sigma_1, \dots, \sigma_k, U)$: matching scheme
- $u_1\sigma_1 \cdots u_k\sigma_k$ is a trace for some u_1, \dots, u_k
- $\Delta(\sigma_k) \not\leq \mathbf{0}$ and $\|\Delta(\sigma_k)\|^- \subseteq \bigcup_{(s,t) \in U} \text{lost}(s, t)$
- $\|\Delta(\sigma_t)\|^- \subseteq \|\Delta(\sigma_{s_{\min}(t)})\|^+$ for all $(\cdot, t) \in U$ with $t < k$

Characterization of Context-Freeness through Witnesses

Proposition

*If **fail** is reachable, then a witness of non-context-freeness can be constructed from a run reaching **fail***

Proposition

If a VAS admits a witness of non-context-freeness, then its trace language is not context-free

Proof: based on the characterization of bounded context-free languages by Ginsburg and Spanier

Characterization of Context-Freeness through Witnesses

Proposition

If *fail* is reachable, then a witness of non-context-freeness can be constructed from a run reaching *fail*

Proposition

If a VAS admits a witness of non-context-freeness, then its trace language is not context-free

Proof: based on the characterization of bounded context-free languages by Ginsburg and Spanier

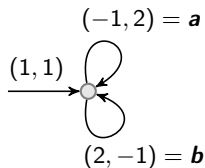
Theorem

The trace language of a VAS $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ is not context-free

\iff ***fail** is reachable*

\iff *$\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ admits a witness of non-context-freeness*

Witnesses of Non-Context-Freeness: Example



$L \cap (\mathbf{ab})^* \mathbf{a}^* \mathbf{b}^*$ is not context-free

Witness



- The largest rational λ such that $\Delta(\mathbf{ab}) + \lambda\Delta(\mathbf{a}) \geq \mathbf{0}$ is $\lambda_{1,2} = 1$
- $\text{lost}(1, 2) = \|\Delta(\mathbf{ab}) + 1\Delta(\mathbf{a})\|^+ = \|(0, 3)\|^+ = \{2\}$ contains $\|\mathbf{b}\|^-$

Definition

A **witness of non-context-freeness** is a tuple $(\sigma_1, \dots, \sigma_k, U)$ such that

- $\Delta(\sigma_k) \not\leq 0$ and $\|\Delta(\sigma_k)\|^- \subseteq \bigcup_{(s,t) \in U} \text{lost}(s, t)$
- ...

⇒ Bound k

⇒ **Non-linear** arithmetic constraints over the $\Delta(\sigma_j)$

$$i \in \text{lost}(s, t) \Leftrightarrow \begin{cases} \delta_t(i) > 0 \wedge (\delta_s(i) > 0 \vee \|\delta_t\|^- \subseteq \|\delta_s\|^+) \\ \delta_t(i) \leq 0 \wedge \bigvee_{j \neq i} \delta_s(i) \cdot \delta_t(j) < \delta_s(j) \cdot \delta_t(i) \end{cases}$$

(δ_j stands for $\Delta(\sigma_j)$)

Proposition

The trace language of $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ is not context-free iff $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ admits a witness of non-context-freeness $(\sigma_1, \dots, \sigma_k, U)$ such that

- $k \leq 3d + 1$
- for every $(s, t) \in U$, the ratio $\lambda_{s,t}$ is either 0 or 1

Proposition

The trace language of $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ is not context-free iff $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ admits a witness of non-context-freeness $(\sigma_1, \dots, \sigma_k, U)$ such that

- $k \leq 3d + 1$
- for every $(s, t) \in U$, the ratio $\lambda_{s,t}$ is either 0 or 1

Proof:

- Simplify U by keeping only pairs that add new lost components
- At most two pairs (s, t) and $(s_{\min}(t), t)$ for each $i \in \{1, \dots, d\}$
- Remove useless σ_j but keep σ_k

Proposition

The trace language of $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ is not context-free iff $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ admits a witness of non-context-freeness $(\sigma_1, \dots, \sigma_k, U)$ such that

- $k \leq 3d + 1$
- for every $(s, t) \in U$, the ratio $\lambda_{s,t}$ is either 0 or 1

Proof:

- Let $(s, t) \in U$ with positive ratio $\lambda_{s,t} = \frac{p}{q}$
- Replacing σ_s by $(\sigma_s)^q$ and σ_t by $(\sigma_t)^p$ yields the ratio $\frac{q}{p} \cdot \frac{p}{q} = 1$
- But this modifies the ratio of other pairs (s', t) and (s, t')
- Follow nesting of U to prevent conflicts

Encoding by Linear Arithmetic Constraints

The requirement that $\lambda_{s,t} \in \{0, 1\}$ simplifies the encoding of $\text{lost}(s, t)$

$$i \in \text{lost}(s, t) \Leftrightarrow i \in \|\delta_s + \lambda_{s,t}\delta_t\|^+$$

$$\begin{aligned} & \lambda_{s,t} = 0 \wedge \delta_s(i) > 0 \\ \Leftrightarrow & \vee \\ & \lambda_{s,t} = 1 \wedge \delta_s(i) + \delta_t(i) > 0 \end{aligned}$$

$$\lambda_{s,t} = 0 \Leftrightarrow \bigvee_{i=1}^d (\delta_s(i) = 0 \wedge \delta_t(i) < 0)$$

$$\lambda_{s,t} = 1 \Leftrightarrow \delta_s + \delta_t \geq \mathbf{0} \wedge \bigvee_{i=1}^d (\delta_s(i) + \delta_t(i) = 0 \wedge \delta_t(i) < 0)$$

Encoding by Linear Arithmetic Constraints

The requirement that $\lambda_{s,t} \in \{0, 1\}$ simplifies the encoding of $\text{lost}(s, t)$

$$i \in \text{lost}(s, t) \Leftrightarrow i \in \|\delta_s + \lambda_{s,t}\delta_t\|^+$$

$$\begin{aligned} & \lambda_{s,t} = 0 \wedge \delta_s(i) > 0 \\ \Leftrightarrow & \vee \\ & \lambda_{s,t} = 1 \wedge \delta_s(i) + \delta_t(i) > 0 \end{aligned}$$

$$\lambda_{s,t} = 0 \Leftrightarrow \bigvee_{i=1}^d (\delta_s(i) = 0 \wedge \delta_t(i) < 0)$$

$$\lambda_{s,t} = 1 \Leftrightarrow \delta_s + \delta_t \geq 0 \wedge \bigvee_{i=1}^d (\delta_s(i) + \delta_t(i) = 0 \wedge \delta_t(i) < 0)$$

⇒ Need **linear relations** between $\Delta(\sigma_1), \dots, \Delta(\sigma_k)$

Table of Contents

- 1 Vector Addition Systems
- 2 Decidability of the Context-Freeness Problem for VAS
- 3 Simple Witnesses of Non-Context-Freeness
- 4 A Relational Trace Logic for VAS with EXPSPACE Solvability
- 5 Conclusion

Yet Another Logic for VAS Traces: Syntax

Given a trace of the form $u_1 \sigma_1 \cdots u_k \sigma_k$

$\cdots \rightarrow \sigma_1 \cdots \rightarrow \sigma_2 \cdots \rightarrow \sigma_3 \cdots \rightarrow \sigma_4 \cdots \rightarrow \sigma_5 \cdots \rightarrow \sigma_6$

The logic expresses properties of $\Delta(\sigma_1), \dots, \Delta(\sigma_k)$

Definition

$$t ::= z \delta_j(i) \mid t + t \quad (z \in \mathbb{Z}, j \geq 1, 1 \leq i \leq d)$$

$$\phi ::= t \geq n \mid \phi \vee \phi \mid \phi \wedge \phi \quad (n \in \mathbb{N})$$

Variables δ_j are interpreted as $\Delta(\sigma_j)$

Yet Another Logic for VAS Traces: Semantics

Definition (Demri 10)

A trace $u_1 \sigma_1 \cdots u_k \sigma_k$ is **self-covering** when

$$\|\Delta(\sigma_j)\|^- \subseteq \|\Delta(\sigma_1)\|^+ \cup \cdots \cup \|\Delta(\sigma_{j-1})\|^+ \quad (\forall j \leq k)$$

Definition

$$\begin{aligned} u_1 \sigma_1 \cdots u_k \sigma_k \models \phi & \text{ if } \phi[\Delta(\sigma_j) / \delta_j] \text{ holds} \\ \langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle \models \phi & \text{ if } u_1 \sigma_1 \cdots u_k \sigma_k \models \phi \text{ for some s.-c. trace} \end{aligned}$$

- ⇒ The model-checking problem **would** become REACHABILITY-hard if
- arbitrary traces were allowed
 - intermediate steps were forbidden ($u_j = \varepsilon$)

A Few Example Properties

Unboundedness: $\bigvee_{i=1}^d \delta_1(i) \geq 1$

Place unboundedness: $\bigvee_{j=1}^d \delta_j(p) \geq 1$

Non-regularity: $\bigvee_{i=1}^d -\delta_{d+1}(i) \geq 1$

Non-context-freeness: $\bigvee_{k=1}^{3d+1} \bigvee_{\substack{U \subseteq \{1, \dots, k\} \\ \text{nested}}} \dots \delta_s(i) + \delta_t(i) = 0 \dots$

Small Model Property

Theorem

If there is a self-covering trace in $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle$ satisfying ϕ , then there is one of length at most

$$2^p(|\mathbf{A}|+|\phi|) \cdot c^{(d \cdot k(\phi))^3}$$

where p is a polynomial and c is a constant

Corollary

The model-checking problem $\langle \mathbf{v}_{\text{init}}, \mathbf{A} \rangle \stackrel{?}{\models} \phi$ is EXPSPACE-complete

Corollary

The context-freeness problem for VAS is EXPSPACE-complete

Proof: EXPSPACE-hardness by reduction from the boundedness problem

Table of Contents

- 1 Vector Addition Systems
- 2 Decidability of the Context-Freeness Problem for VAS
- 3 Simple Witnesses of Non-Context-Freeness
- 4 A Relational Trace Logic for VAS with EXPSPACE Solvability
- 5 Conclusion

Conclusion and Open Problems

New proof, simpler than the one of Schwer

- Characterization of non-context-freeness through simple witnesses
- The trace language of a VAS is context-free if, and only if, it has a context-free intersection with every bounded regular language

New logic for expressing properties of VAS traces

- Can express linear relations between cycles visited by a run
- Model-checking is EXPSpace -complete
- Incomparable with existing logics that are solvable in EXPSpace

Complexity of the context-freeness problem: EXPSpace -complete

Conclusion and Open Problems

New proof, simpler than the one of Schwer

- Characterization of non-context-freeness through simple witnesses
- The trace language of a VAS is context-free if, and only if, it has a context-free intersection with every bounded regular language

New logic for expressing properties of VAS traces

- Can express linear relations between cycles visited by a run
- Model-checking is EXPSPACE -complete
- Incomparable with existing logics that are solvable in EXPSPACE

Complexity of the context-freeness problem: EXPSPACE -complete

Open Problems

- Existing EXPSPACE logics are incomparable: try to unify them!
- Coverability and reachability for Pushdown VAS

There is hopefully still time for ...

Questions ?