Reachability for Bounded Branching VASS

Filip Mazowiecki[1] and Michał Pilipczuk[2]

[1]University of Bordeaux

[2]University of Warsaw

CONCUR 2019

Reachability for Bounded Branching VASS

Filip Mazowiecki[1] and Michał Pilipczuk[2]

[1]University of Bordeaux

[2]University of Warsaw

CONCUR 2019

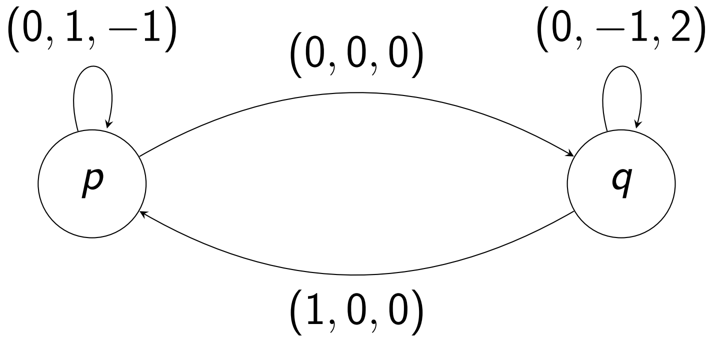
Vector addition systems with states (d-VASS)

(Q, Δ), where $\Delta \subseteq Q \times \mathbb{Z}^d \times Q$

Vector addition systems with states (d-VASS)

(Q, Δ), where $\Delta \subseteq Q \times \mathbb{Z}^d \times Q$

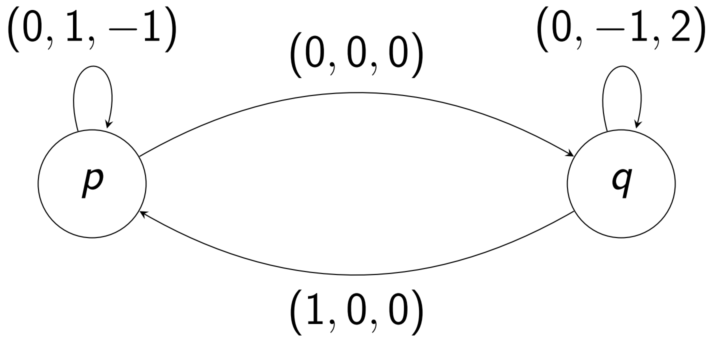
Example: $d = 3$, $Q = \{p, q\}$



Vector addition systems with states (d-VASS)

(Q, Δ), where $\Delta \subseteq Q \times \mathbb{Z}^d \times Q$

Example: $d = 3$, $Q = \{p, q\}$

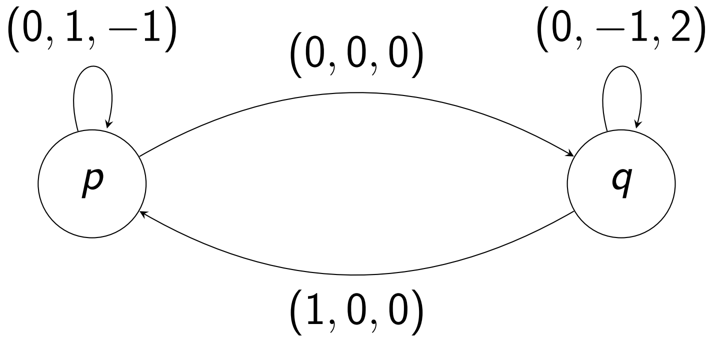


Configurations $p(\boldsymbol{v}) = (p, \boldsymbol{v}) \in Q \times \mathbb{N}^d$

Vector addition systems with states (d-VASS)

(Q, Δ), where $\Delta \subseteq Q \times \mathbb{Z}^d \times Q$

Example: $d = 3$, $Q = \{p, q\}$



Configurations $p(\boldsymbol{v}) = (p, \boldsymbol{v}) \in Q \times \mathbb{N}^d$

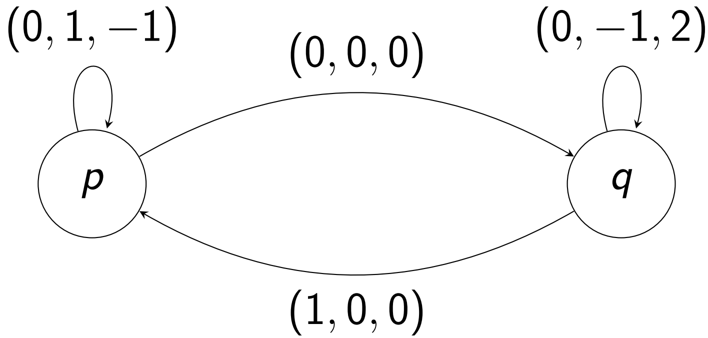
Example run:

$p(0, 0, 1) \to p(0, 1, 0) \to q(0, 1, 0) \to q(0, 0, 2)$

Vector addition systems with states (d-VASS)

(Q, Δ), where $\Delta \subseteq Q \times \mathbb{Z}^d \times Q$

Example: $d = 3$, $Q = \{p, q\}$



Configurations $p(\mathbf{v}) = (p, \mathbf{v}) \in Q \times \mathbb{N}^d$

Example run:

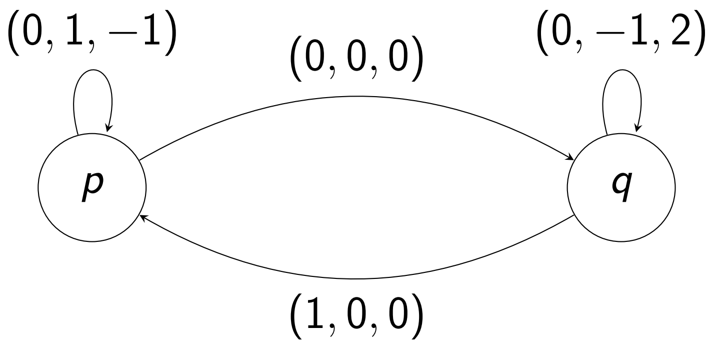
$p(0, 0, 1) \to p(0, 1, 0) \to q(0, 1, 0) \to q(0, 0, 2)$

Notation: $p(0, 0, 1) \to^* q(0, 0, 2)$

Vector addition systems with states (d-VASS)

(Q, Δ), where $\Delta \subseteq Q \times \boxed{\mathbb{Z}^d} \times Q$ $\boxed{\text{unary/binary}}$

Example: $d = 3$, $Q = \{p, q\}$



Configurations $p(\boldsymbol{v}) = (p, \boldsymbol{v}) \in Q \times \mathbb{N}^d$

Example run:

$p(0,0,1) \to p(0,1,0) \to q(0,1,0) \to q(0,0,2)$

Notation: $p(0,0,1) \to^* q(0,0,2)$

Bounded d-VASS (d-BoVASS)

(Q, Δ, B), where (Q, Δ) is a d-VASS

and $B \in \mathbb{N}$ (in binary)

Bounded d-VASS (d-BoVASS)

(Q, Δ, B), where (Q, Δ) is a d-VASS

and $B \in \mathbb{N}$ (in binary)

Configurations $p(\boldsymbol{v}) = (p, \boldsymbol{v}) \in Q \times \{0, \ldots, B\}^d$

Bounded d-VASS (d-BoVASS)

(Q, Δ, B), where (Q, Δ) is a d-VASS
and $B \in \mathbb{N}$ (in binary)

Configurations $p(\boldsymbol{v}) = (p, \boldsymbol{v}) \in Q \times \{0, \ldots, B\}^d$

For example $p(0, B, B) \xrightarrow{(1,1,-1)}$ not allowed $(B + 1 > B)$

Bounded d-VASS (d-BoVASS)

(Q, Δ, B), where (Q, Δ) is a d-VASS

and $B \in \mathbb{N}$ (in binary)

Configurations $p(\boldsymbol{v}) = (p, \boldsymbol{v}) \in Q \times \{0, \ldots, B\}^d$

For example $p(0, B, B) \xrightarrow{(1,1,-1)}$ not allowed $(B + 1 > B)$

Why is it interesting?

Counters: $c_1, \ldots, c_d$

Bounded d-VASS (d-BoVASS)

(Q, Δ, B), where (Q, Δ) is a d-VASS

and $B \in \mathbb{N}$ (in binary)

Configurations $p(\boldsymbol{v}) = (p, \boldsymbol{v}) \in Q \times \{0, \dots, B\}^d$

For example $p(0, B, B) \xrightarrow{(1,1,-1)}$ not allowed $(B + 1 > B)$

Why is it interesting?

Counters: $c_1, \dots, c_d$

- Before we could encode $c_i \geq n$ tests

First $(0, \dots, 0, -n, 0, \dots, 0)$ then $(0, \dots, 0, n, 0, \dots, 0)$

Bounded d-VASS (d-BoVASS)

(Q, Δ, B), where (Q, Δ) is a d-VASS
and $B \in \mathbb{N}$ (in binary)

Configurations $p(\boldsymbol{v}) = (p, \boldsymbol{v}) \in Q \times \{0, \ldots, B\}^d$

For example $p(0, B, B) \xrightarrow{(1,1,-1)}$ not allowed $(B + 1 > B)$

Why is it interesting?

Counters: $c_1, \ldots, c_d$

- Before we could encode $c_i \geq n$ tests

  First $(0, \ldots, 0, -n, 0, \ldots, 0)$ then $(0, \ldots, 0, n, 0, \ldots, 0)$

- Now we can also encode $c_i \leq n$ tests

  First $(0, \ldots, 0, B - n, 0, \ldots, 0)$ then $(0, \ldots, 0, n - B, 0, \ldots, 0)$

Decision problem

Reachability problem:

GIVEN: d-VASS (Q, Δ) and configurations $p(\boldsymbol{u}), q(\boldsymbol{v})$

(or d-BoVASS (Q, Δ, B))

DECIDE: whether $p(\boldsymbol{u}) \rightarrow^* q(\boldsymbol{v})$?

Decision problem

Reachability problem:

GIVEN: d-VASS (Q, Δ) and configurations $p(\boldsymbol{u}), q(\boldsymbol{v})$

(or d-BoVASS (Q, Δ, B))

DECIDE: whether $p(\boldsymbol{u}) \rightarrow^* q(\boldsymbol{v})$?

State of art:

- d-VASS: in Ackermann [Leroux and Schmitz, 2019]
  and TOWER-hard [Czerwiński et al., 2019]

Decision problem

Reachability problem:

GIVEN: d-VASS (Q, Δ) and configurations $p(\boldsymbol{u}), q(\boldsymbol{v})$

(or d-BoVASS (Q, Δ, B))

DECIDE: whether $p(\boldsymbol{u}) \rightarrow^* q(\boldsymbol{v})$?

State of art:

- d-VASS: in Ackermann [Leroux and Schmitz, 2019]
  and TOWER-hard [Czerwiński et al., 2019]
- 1-VASS: NL-complete (in unary) [Valiant and Paterson, 1975]
  NP-complete (in binary) [Haase et al., 2009]

Decision problem

Reachability problem:

GIVEN: d-VASS (Q, Δ) and configurations $p(\boldsymbol{u}), q(\boldsymbol{v})$

(or d-BOVASS (Q, Δ, B))

DECIDE: whether $p(\boldsymbol{u}) \rightarrow^* q(\boldsymbol{v})$?

State of art:

- d-VASS: in Ackermann [Leroux and Schmitz, 2019]
  and TOWER-hard [Czerwiński et al., 2019]
- 1-VASS: NL-complete (in unary) [Valiant and Paterson, 1975]
  NP-complete (in binary) [Haase et al., 2009]
- d-BOVASS: in PSPACE [Obvious]
  PSPACE-hard even for $d = 1$ [Fearnley and Jurdziński, 2013]

2-**BoVASS example**

Copying c_1 to c_2

2-BoVASS example

Copying c_1 to c_2

Let: $c_1 = n_1$, $c_2 = n_2$

$0 \leq n_1 \leq M$ and $B = M(M+2)$

Goal: $c_1 = c_2 = n_1$

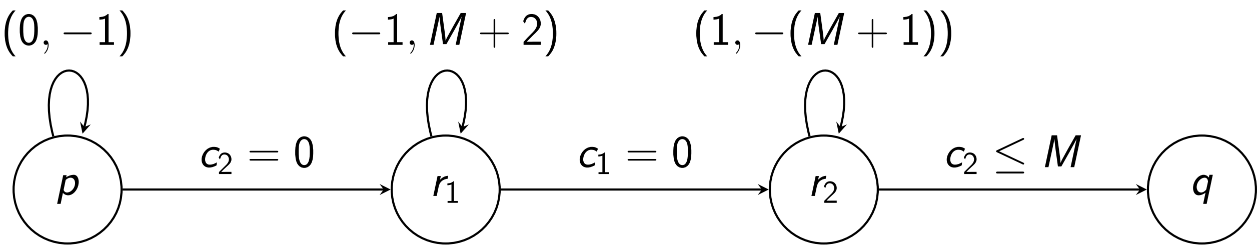
2-BoVASS example

Copying c_1 to c_2

Let: $c_1 = n_1$, $c_2 = n_2$

$0 \leq n_1 \leq M$ and $B = M(M+2)$

Goal: $c_1 = c_2 = n_1$



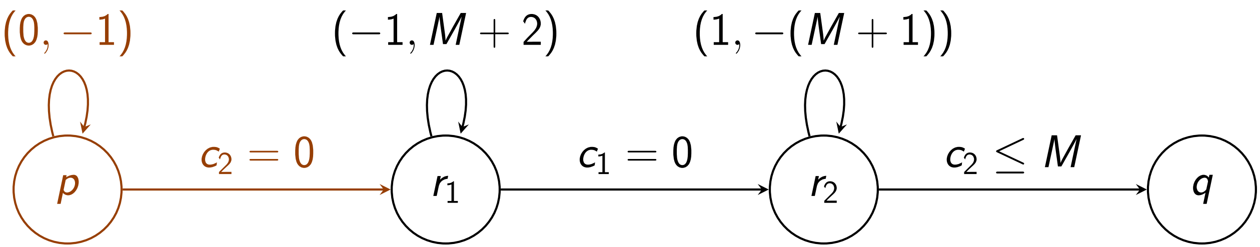
2-BoVASS example

Copying c_1 to c_2

Let: $c_1 = n_1$, $c_2 = n_2$

$$0 \leq n_1 \leq M \text{ and } B = M(M+2)$$

Goal: $c_1 = c_2 = n_1$



- $p(n_1, n_2) \to^* r_1(n_1, 0)$

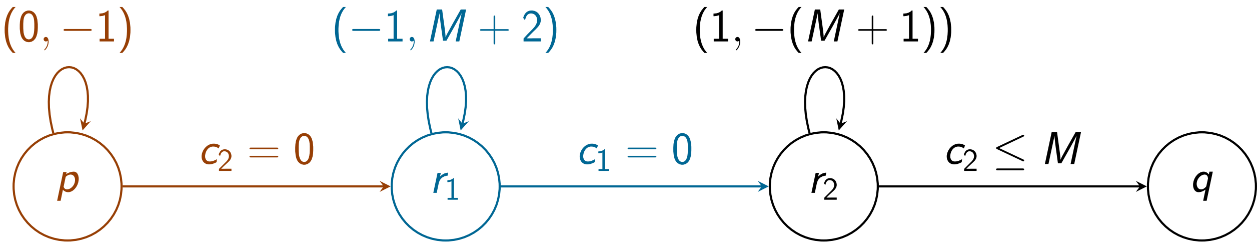
2-**BoVASS example**

Copying c_1 to c_2

Let: $c_1 = n_1$, $c_2 = n_2$

$$0 \leq n_1 \leq M \text{ and } B = M(M+2)$$

Goal: $c_1 = c_2 = n_1$



- $p(n_1, n_2) \to^* r_1(n_1, 0)$
- $r_1(n_1, 0) \to^* r_2(0, (M+2)n_1)$

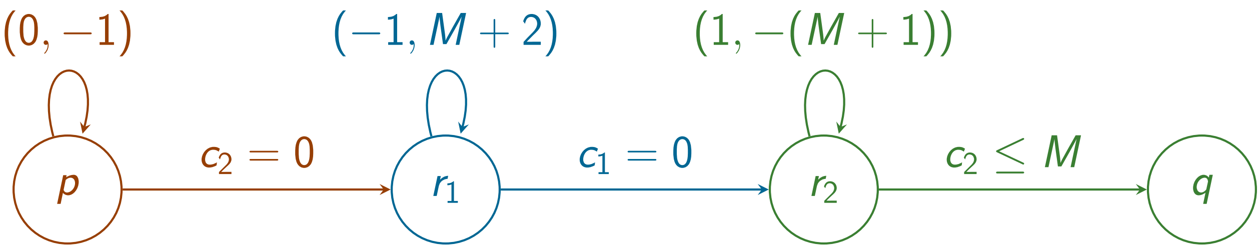
2-BoVASS example

Copying c_1 to c_2

Let: $c_1 = n_1$, $c_2 = n_2$

$$0 \leq n_1 \leq M \text{ and } B = M(M+2)$$

Goal: $c_1 = c_2 = n_1$



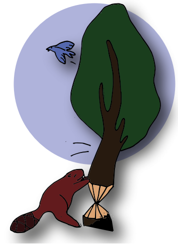
- $p(n_1, n_2) \to^* r_1(n_1, 0)$
- $r_1(n_1, 0) \to^* r_2(0, (M+2)n_1)$
- $r_2(0, (M+2)n_1) \to^* q(n_1, n_1)$

Branching d-VASS (d-BrVASS)

$(Q, \Delta_1, \Delta_2, q_0)$

where $\Delta_1 \subseteq Q \times \mathbb{Z} \times Q$, and $\Delta_2 \subseteq Q^3$

Branching d-VASS (d-BrVASS)

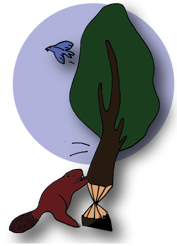


$(Q, \Delta_1, \Delta_2, q_0)$

where $\Delta_1 \subseteq Q \times \mathbb{Z} \times Q$, and $\Delta_2 \subseteq Q^3$

Configurations: $Q \times \mathbb{N}^d$ (the same)

Branching d-VASS (d-BrVASS)



$(Q, \Delta_1, \Delta_2, q_0)$

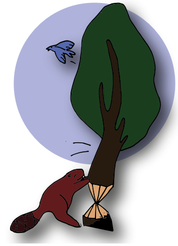
where $\Delta_1 \subseteq Q \times \mathbb{Z} \times Q$, and $\Delta_2 \subseteq Q^3$

Configurations: $Q \times \mathbb{N}^d$ (the same)

Computations are trees:

- leaves labelled: $(q_0, \mathbf{0})$

Branching d-VASS (d-BrVASS)



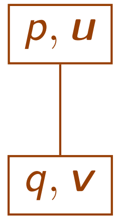
$(Q, \Delta_1, \Delta_2, q_0)$

where $\Delta_1 \subseteq Q \times \mathbb{Z} \times Q$, and $\Delta_2 \subseteq Q^3$

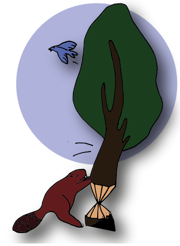
Configurations: $Q \times \mathbb{N}^d$ (the same)

Computations are trees:

- leaves labelled: $(q_0, \mathbf{0})$
- inner nodes
  - ▷ with one child: $(p, z, q) \in \Delta_1$, $\boldsymbol{u} + \boldsymbol{z} = \boldsymbol{v}$



Branching d-VASS (d-BrVASS)



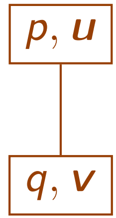
$(Q, \Delta_1, \Delta_2, q_0)$

where $\Delta_1 \subseteq Q \times \mathbb{Z} \times Q$, and $\Delta_2 \subseteq Q^3$

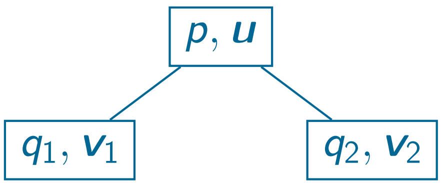
Configurations: $Q \times \mathbb{N}^d$ (the same)

Computations are trees:

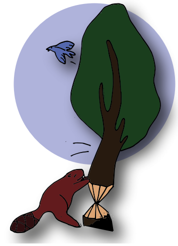
- leaves labelled: $(q_0, \mathbf{0})$
- inner nodes
  - ▷ with one child: $(p, z, q) \in \Delta_1$, $\boldsymbol{u} + \boldsymbol{z} = \boldsymbol{v}$



  - ▷ with two children: $(p, q_1, q_2) \in \Delta_2$, $\boldsymbol{u} = \boldsymbol{v}_1 + \boldsymbol{v}_2$



Branching d-VASS (d-BrVASS)



$(Q, \Delta_1, \Delta_2, q_0)$

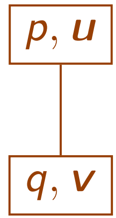
where $\Delta_1 \subseteq Q \times \mathbb{Z} \times Q$, and $\Delta_2 \subseteq Q^3$

Configurations: $Q \times \mathbb{N}^d$ (the same)

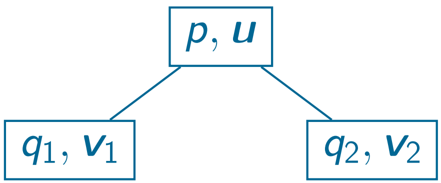
If $\Delta_2 = \emptyset$
then it's a d-VASS

Computations are trees:

- leaves labelled: $(q_0, \mathbf{0})$
- inner nodes
  - ▷ with one child: $(p, z, q) \in \Delta_1$, $\boldsymbol{u} + \boldsymbol{z} = \boldsymbol{v}$



  - ▷ with two children: $(p, q_1, q_2) \in \Delta_2$, $\boldsymbol{u} = \boldsymbol{v}_1 + \boldsymbol{v}_2$



1-BrVASS example

Fix n, b $\qquad (n = 3,\ b = 3)$

1-BrVASS example

Fix n, b $\qquad (n = 3,\ b = 3)$

- states $Q = \{q_1 \dots q_n\} \cup \{q_0, q_F\}$

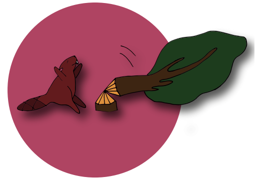
1-BrVASS example

Fix n, b $\quad (n = 3,\ b = 3)$

- states $Q = \{q_1 \ldots q_n\} \cup \{q_0, q_F\}$

$\boxed{q_0, 0}$ $\quad$ $\boxed{q_0, 0}$ $\quad$ $\boxed{q_0, 0}$ $\quad$ $\boxed{q_0, 0}$

1-BrVASS example



Fix n, b $\qquad (n = 3,\ b = 3)$

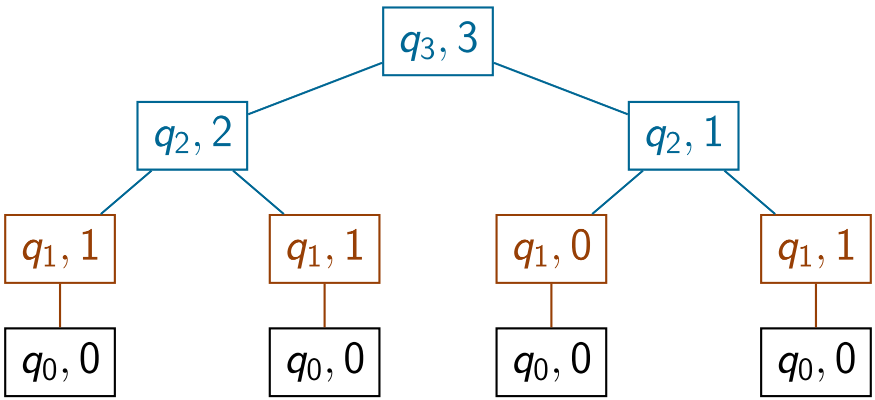
- states $Q = \{q_1 \dots q_n\} \cup \{q_0, q_F\}$
- three types of transitions:
  - $(q_1, 0, q_0), (q_1, -1, q_0) \in \Delta_1$ (initialize)

$$\begin{array}{cccc} \boxed{q_1, 1} & \boxed{q_1, 1} & \boxed{q_1, 0} & \boxed{q_1, 1} \\ | & | & | & | \\ \boxed{q_0, 0} & \boxed{q_0, 0} & \boxed{q_0, 0} & \boxed{q_0, 0} \end{array}$$

1-BrVASS example

Fix n, b $\quad$ $(n = 3,\ b = 3)$

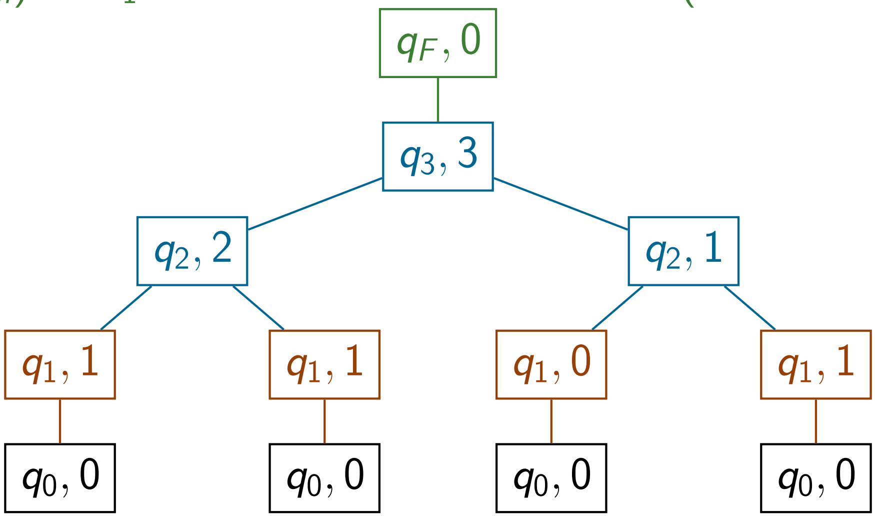
- states $Q = \{q_1 \dots q_n\} \cup \{q_0, q_F\}$
- three types of transitions:
  - $(q_1, 0, q_0), (q_1, -1, q_0) \in \Delta_1$ (initialize)
  - $(q_{i+1}, q_i, q_i) \in \Delta_2$ for all $i < n$ (build tree)



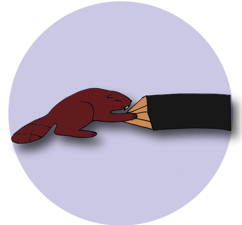
1-BrVASS example

Fix n, b $\qquad (n = 3,\ b = 3)$

- states $Q = \{q_1 \ldots q_n\} \cup \{q_0, q_F\}$
- three types of transitions:
  - $(q_1, 0, q_0), (q_1, -1, q_0) \in \Delta_1$ (initialize)
  - $(q_{i+1}, q_i, q_i) \in \Delta_2$ for all $i < n$ (build tree)
  - $(q_F, b, q_n) \in \Delta_1$ (check value)



Decision problem for d-BrVASS

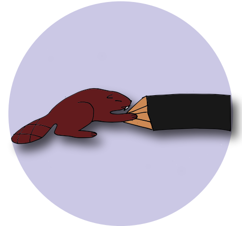


Reachability problem:

GIVEN: d-BrVASS $(Q, \Delta_1, \Delta_2, q_0)$ and a configuration $p(\boldsymbol{u})$

DECIDE: is there a computation tree with $p(\boldsymbol{u})$ in the root?

Decision problem for d-BrVASS



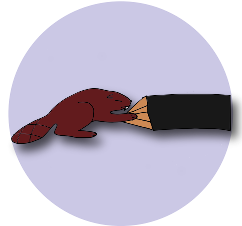
Reachability problem:

GIVEN: d-BrVASS $(Q, \Delta_1, \Delta_2, q_0)$ and a configuration $p(\boldsymbol{u})$

DECIDE: is there a computation tree with $p(\boldsymbol{u})$ in the root?

Previous example: Reachable iff $0 \leq b \leq 2^{n-1}$

Decision problem for d-BrVASS



Reachability problem:

GIVEN: d-BRVASS $(Q, \Delta_1, \Delta_2, q_0)$ and a configuration $p(\boldsymbol{u})$

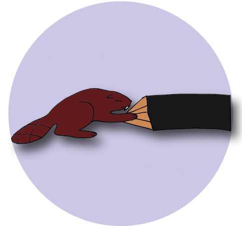
DECIDE: is there a computation tree with $p(\boldsymbol{u})$ in the root?

Previous example: Reachable iff $0 \leq b \leq 2^{n-1}$

State of art:

- d-BRVASS: open for $d \geq 2$

Decision problem for d-BrVASS



Reachability problem:

GIVEN: d-BRVASS $(Q, \Delta_1, \Delta_2, q_0)$ and a configuration $p(\boldsymbol{u})$

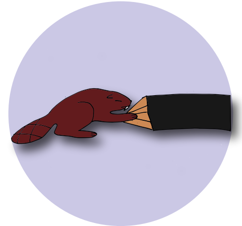
DECIDE: is there a computation tree with $p(\boldsymbol{u})$ in the root?

Previous example: $\boxed{\text{Reachable iff } 0 \leq b \leq 2^{n-1}}$

State of art:

- d-BRVASS: open for $d \geq 2$
- 1-BRVASS: PTIME-complete (in unary) [Göller et al., 2016]
  PSPACE-complete (in binary) [Figueira et al., 2017]

Decision problem for d-BrVASS



Reachability problem:

GIVEN: d-BRVASS $(Q, \Delta_1, \Delta_2, q_0)$ and a configuration $p(\boldsymbol{u})$

DECIDE: is there a computation tree with $p(\boldsymbol{u})$ in the root?

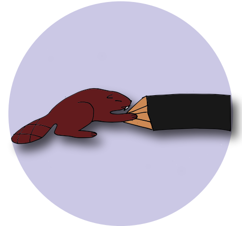
Previous example: Reachable iff $0 \leq b \leq 2^{n-1}$

State of art:

- d-BRVASS: open for $d \geq 2$
- 1-BRVASS: PTIME-complete (in unary) [Göller et al., 2016]
  PSPACE-complete (in binary) [Figueira et al., 2017]

- Bounded d-BRVASS (d-BOBRVASS): this paper!

Decision problem for d-BrVASS



Reachability problem:

GIVEN: d-BRVASS $(Q, \Delta_1, \Delta_2, q_0)$ and a configuration $p(\boldsymbol{u})$

DECIDE: is there a computation tree with $p(\boldsymbol{u})$ in the root?

Previous example: Reachable iff $0 \leq b \leq 2^{n-1}$

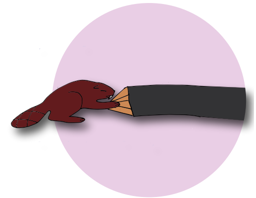
State of art:

- d-BRVASS: open for $d \geq 2$
- 1-BRVASS: PTIME-complete (in unary) [Göller et al., 2016]
  PSPACE-complete (in binary) [Figueira et al., 2017]

- Bounded d-BRVASS (d-BOBRVASS): this paper!

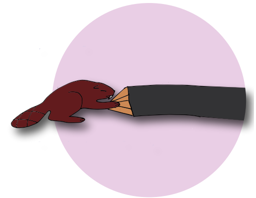
in EXPTIME [Obvious]

Results for reachability for d-BoBrVASS



For $d = 1$

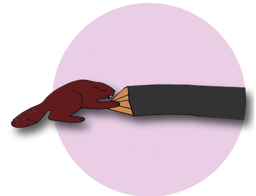
Results for reachability for d-BoBrVASS



For $d = 1$

	unary	binary	with bound
VASS	NL-complete	NP-complete	PSPACE-complete

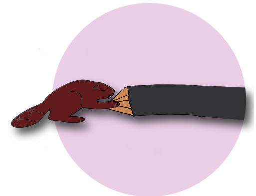
Results for reachability for d-BoBrVASS



For $d = 1$

	unary	binary	with bound
VASS	NL-complete	NP-complete	PSPACE-complete
BrVASS	P-complete	PSPACE-complete	in EXPTIME

Results for reachability for d-BoBrVASS

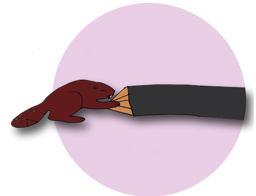


For $d = 1$

	unary	binary	with bound
VASS	NL-complete	NP-complete	PSPACE-complete
BrVASS	P-complete	PSPACE-complete	in EXPTIME

PSPACE-hard

Results for reachability for d-BoBrVASS



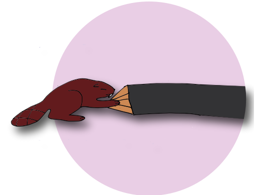
For $d = 1$

	unary	binary	with bound
VASS	NL-complete	NP-complete	PSPACE-complete
BrVASS	P-complete	PSPACE-complete	in EXPTIME

PSPACE-hard

Possibly misleading: branching is not alternation

Results for reachability for d-BoBrVASS



For $d = 1$

	unary	binary	with bound
VASS	NL-complete	NP-complete	PSPACE-complete
BrVASS	P-complete	PSPACE-complete	in EXPTIME

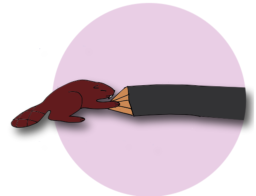
PSPACE-hard

Possibly misleading: branching is not alternation

For $d = 2$

	unary	binary	with bound
VASS	NL-complete	PSPACE-complete	PSPACE-complete

Results for reachability for d-BoBrVASS



For $d = 1$

	unary	binary	with bound
VASS	NL-complete	NP-complete	PSPACE-complete
BrVASS	P-complete	PSPACE-complete	in EXPTIME

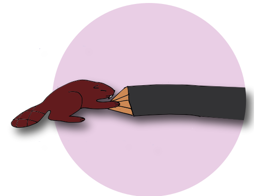
PSPACE-hard

Possibly misleading: branching is not alternation

For $d = 2$

	unary	binary	with bound
VASS	NL-complete	PSPACE-complete	PSPACE-complete
BrVASS	?	?	EXPTIME-complete

Results for reachability for d-BoBrVASS



For $d = 1$

	unary	binary	with bound
VASS	NL-complete	NP-complete	PSPACE-complete
BrVASS	P-complete	PSPACE-complete	in EXPTIME

PSPACE-hard

Possibly misleading: branching is not alternation

For $d = 2$

	unary	binary	with bound
VASS	NL-complete	PSPACE-complete	PSPACE-complete
BrVASS	?	?	EXPTIME-complete

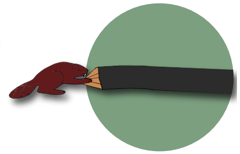
- Reachability for 1-BoBrVASS remains open

The 2-BoBrVASS we work with

$(Q, \Delta_1, \Delta_2, q_0)$

c_1 – the real counter, c_2 – buffer (usually $c_2 = 0$)

The 2-BoBrVASS we work with

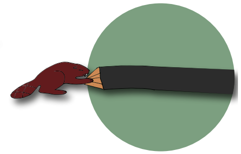


$(Q, \Delta_1, \Delta_2, q_0)$

c_1 – the real counter, c_2 – buffer (usually $c_2 = 0$)

- Test transitions: $c_i \geq n, c_i \leq n, c_i = n \in \Delta_1$

The 2-BoBrVASS we work with



$(Q, \Delta_1, \Delta_2, q_0)$

c_1 – the real counter, c_2 – buffer (usually $c_2 = 0$)

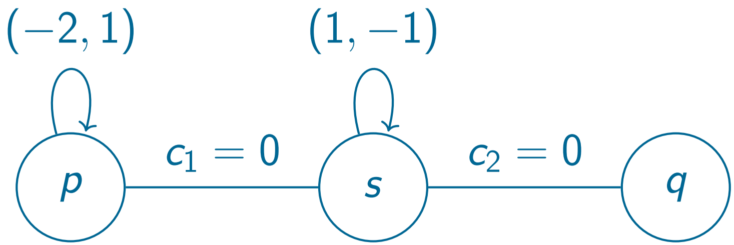
- Test transitions: $c_i \geq n$, $c_i \leq n$, $c_i = n \in \Delta_1$
- Division transitions: $c_1 \div 2 \in \Delta_1$, $\quad p(m, 0) \xrightarrow{c_1 \div 2} q(n, 0)$ iff $m = 2n$

The 2-BoBrVASS we work with

$(Q, \Delta_1, \Delta_2, q_0)$

c_1 – the real counter, c_2 – buffer (usually $c_2 = 0$)

- Test transitions: $c_i \geq n$, $c_i \leq n$, $c_i = n \in \Delta_1$
- Division transitions: $c_1 \div 2 \in \Delta_1$, $\quad p(m, 0) \xrightarrow{c_1 \div 2} q(n, 0)$ iff $m = 2n$

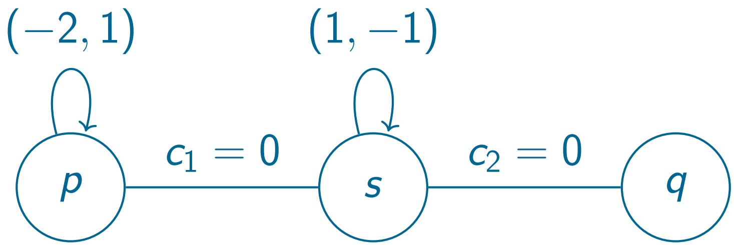


The 2-BoBrVASS we work with

$(Q, \Delta_1, \Delta_2, q_0)$

c_1 – the real counter, c_2 – buffer (usually $c_2 = 0$)

- Test transitions: $c_i \geq n$, $c_i \leq n$, $c_i = n \in \Delta_1$
- Division transitions: $c_1 \div 2 \in \Delta_1$, $\quad p(m, 0) \xrightarrow{c_1 \div 2} q(n, 0)$ iff $m = 2n$



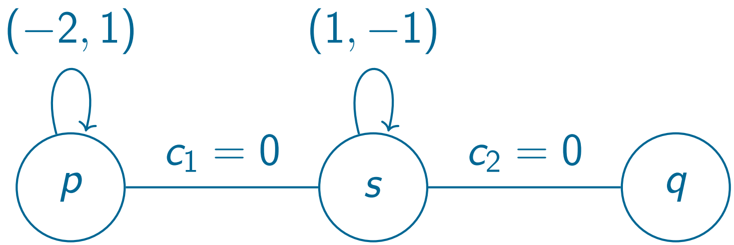
Notation: 1-$\textsc{BoBrVASS}^{\div 2}$

The 2-BoBrVASS we work with

$(Q, \Delta_1, \Delta_2, q_0)$

c_1 – the real counter, c_2 – buffer (usually $c_2 = 0$)

- Test transitions: $c_i \geq n$, $c_i \leq n$, $c_i = n \in \Delta_1$
- Division transitions: $c_1 \div 2 \in \Delta_1$, $\quad p(m, 0) \xrightarrow{c_1 \div 2} q(n, 0)$ iff $m = 2n$

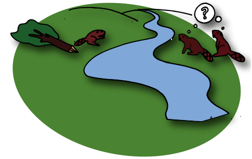


Notation: 1-BoBrVASS$^{\div 2}$

Remark

$$\text{1-BoBrVASS}^{\div 2} = \text{1-BoBrVASS}^{\times 2} = \text{1-BoBrVASS}^{\times 2, \div 2}$$

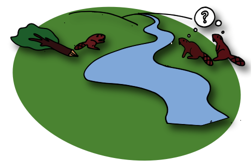
Two gadgets for 1-BoBrVASS$^{\div 2}$



Fix $M = 2^n$ and suppose $c_1 = x, \quad 0 \leq x \leq M-1$

Bound $B = M^4$

Two gadgets for 1-BoBrVASS$^{\div 2}$



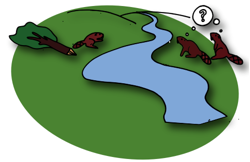
Fix $M = 2^n$ and suppose $c_1 = x, \quad 0 \leq x \leq M - 1$

Bound $B = M^4$

- Gadget 1 (G1): $x \to x + Mx$.

$$\begin{tikzcd}
\boxed{p} \arrow[loop above, "M+M^2+M^3"] \arrow[r, "c_1 \leq M^4"] & \boxed{r} \arrow[loop above, "-1-M^3"] \arrow[r, "c_1 \leq M^3 + M^2"] & \boxed{q_0} \arrow[r, "\div 2"] & \cdots \arrow[r, "\div 2"] & \boxed{q_n}
\end{tikzcd}$$

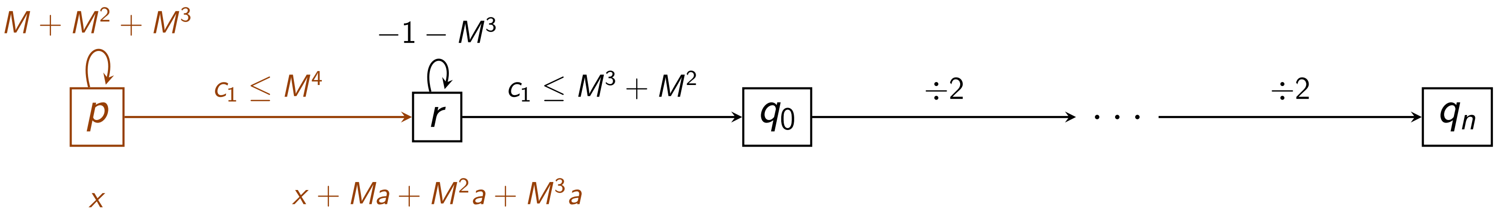
Two gadgets for 1-BoBrVASS$^{\div 2}$



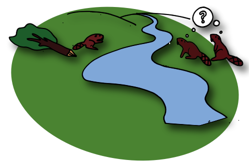
Fix $M = 2^n$ and suppose $c_1 = x, \quad 0 \leq x \leq M - 1$

Bound $B = M^4$

- Gadget 1 (G1): $x \to x + Mx$.



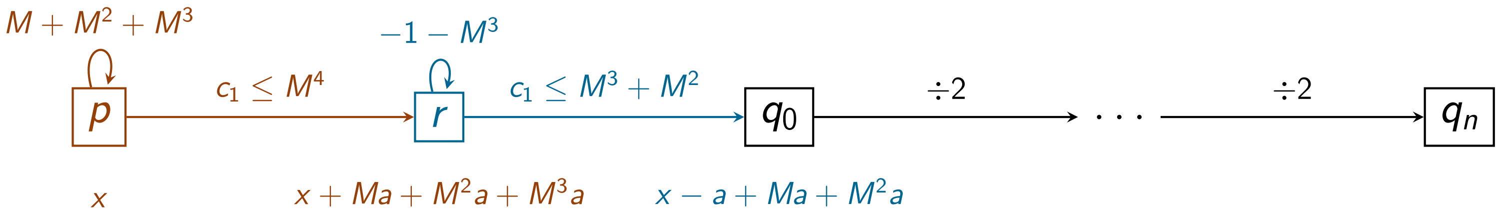
Two gadgets for 1-BoBrVASS$^{\div 2}$



Fix $M = 2^n$ and suppose $c_1 = x, \quad 0 \leq x \leq M - 1$

Bound $B = M^4$

- Gadget 1 (G1): $x \to x + Mx$.

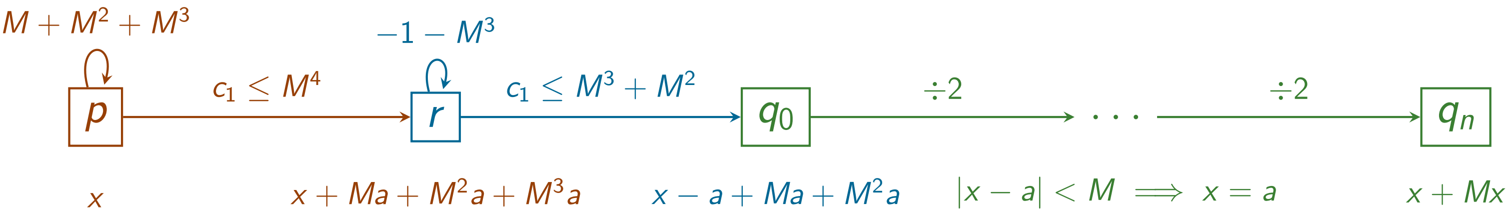


Two gadgets for 1-BoBrVASS$^{\div 2}$

Fix $M = 2^n$ and suppose $c_1 = x, \quad 0 \leq x \leq M - 1$

Bound $B = M^4$

- Gadget 1 (G1): $x \to x + Mx$.

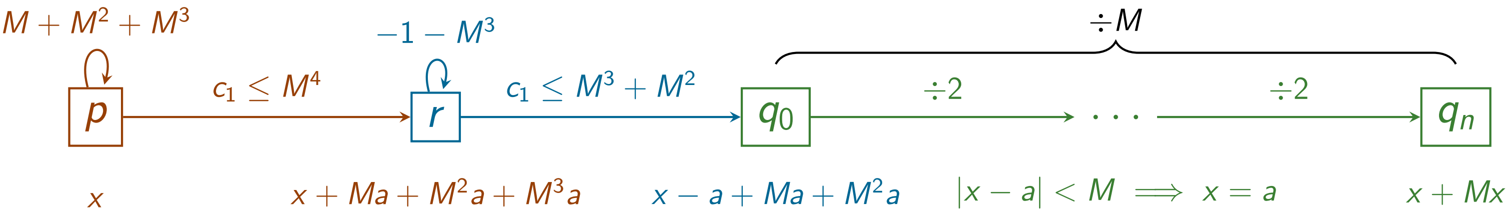


Two gadgets for 1-BoBrVASS$^{\div 2}$

Fix $M = 2^n$ and suppose $c_1 = x, \quad 0 \le x \le M - 1$

Bound $B = M^4$

- Gadget 1 (G1): $x \to x + Mx$.

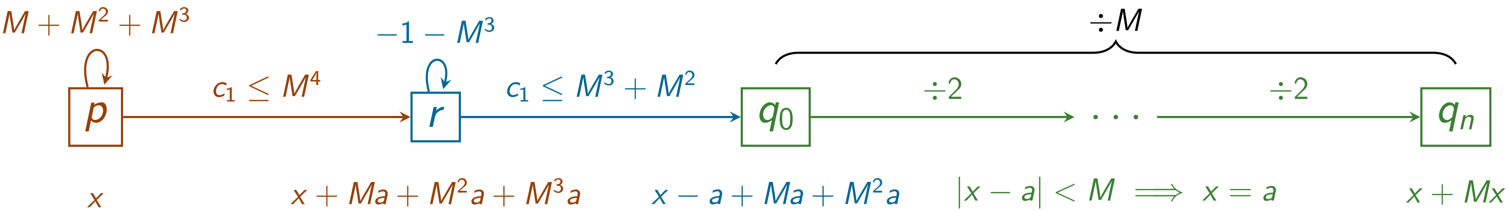


Two gadgets for 1-BoBrVASS$^{\div 2}$

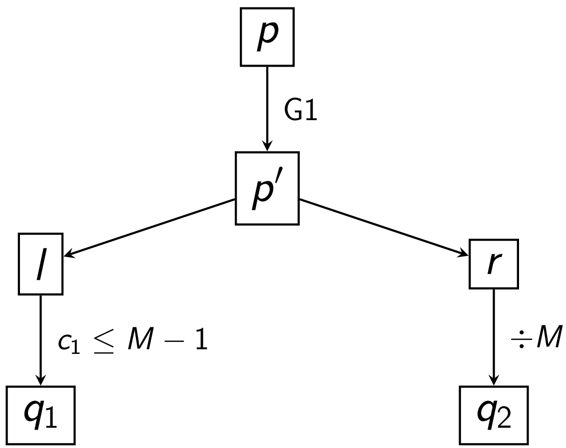
Fix $M = 2^n$ and suppose $c_1 = x$, $\quad 0 \leq x \leq M - 1$

Bound $B = M^4$

- Gadget 1 (G1): $x \to x + Mx$.



- Gadget 2 (G2): $\begin{array}{ccc} & x & \\ \swarrow & & \searrow \\ x & & x \end{array}$

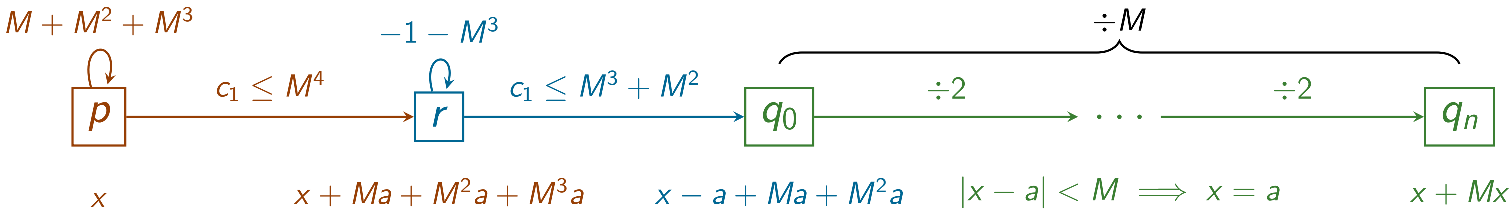


Two gadgets for 1-BoBrVASS$^{\div 2}$

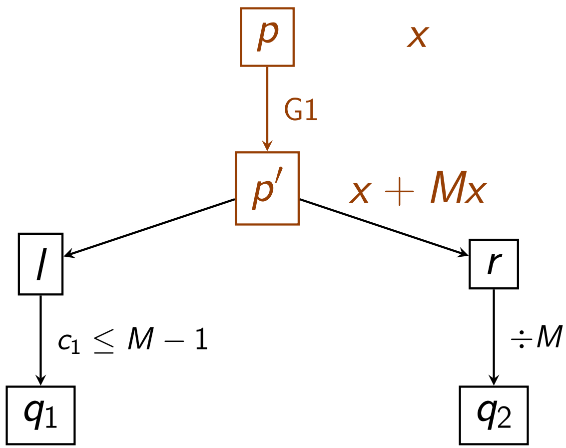
Fix $M = 2^n$ and suppose $c_1 = x, \quad 0 \leq x \leq M - 1$

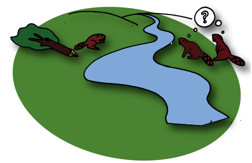
Bound $B = M^4$

- Gadget 1 (G1): $x \to x + Mx$.



- Gadget 2 (G2): $\begin{array}{ccc} & x & \\ \swarrow & & \searrow \\ x & & x \end{array}$



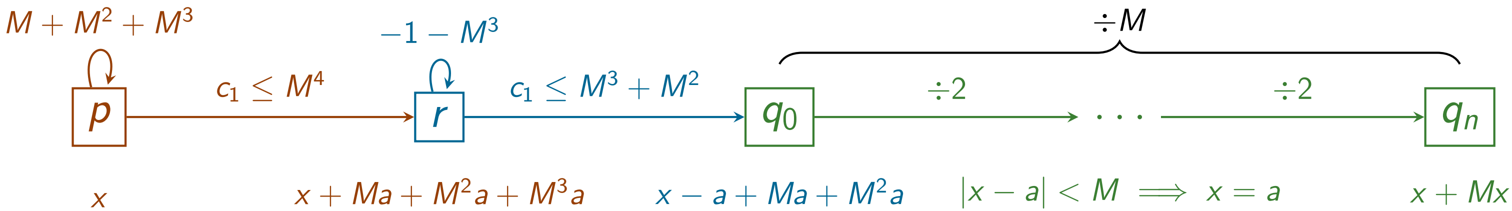


Two gadgets for 1-BoBrVASS$^{\div 2}$

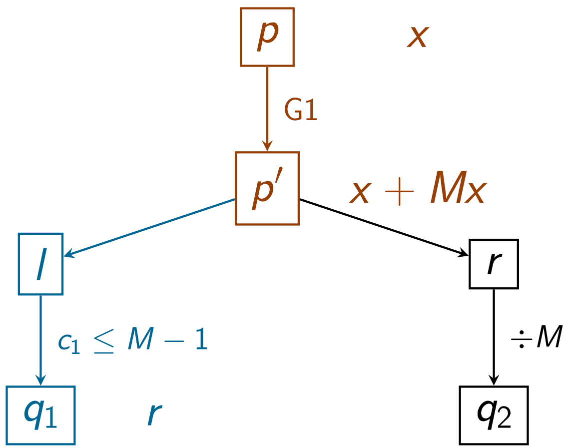
Fix $M = 2^n$ and suppose $c_1 = x, \quad 0 \leq x \leq M - 1$

Bound $B = M^4$

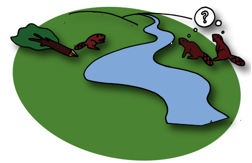
- Gadget 1 (G1): $x \to x + Mx$.



- Gadget 2 (G2): $\begin{array}{ccc} & x & \\ \swarrow & & \searrow \\ x & & x \end{array}$



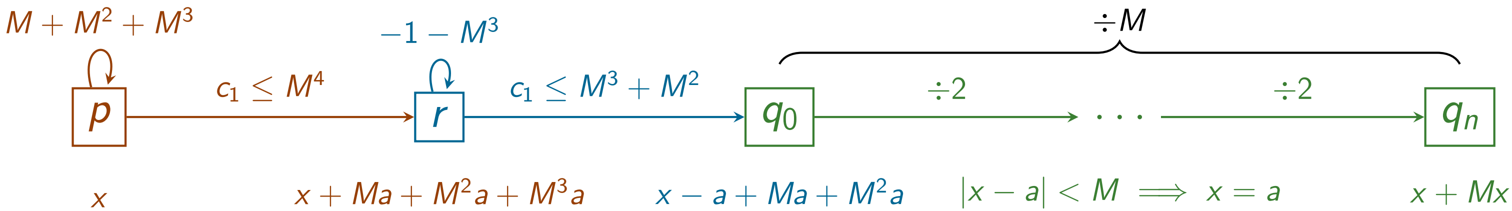
Two gadgets for 1-BoBrVASS$^{\div 2}$



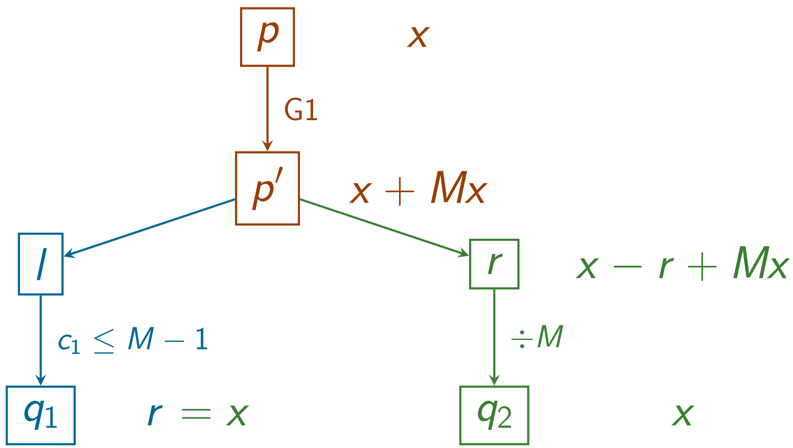
Fix $M = 2^n$ and suppose $c_1 = x, \quad 0 \leq x \leq M - 1$

Bound $B = M^4$

- Gadget 1 (G1): $x \to x + Mx$.



- Gadget 2 (G2): $\begin{tikzcd}[column sep=tiny] & x \arrow[dl] \arrow[dr] & \\ x & & x \end{tikzcd}$



Countdown games

$(S, T), \quad S = S_\exists \uplus S_\forall$

$T \subseteq S \times \mathbb{N} \setminus \{0\} \times S$ (in binary)

Countdown games

$(S, T), \quad S = S_\exists \uplus S_\forall$

$T \subseteq S \times \mathbb{N} \setminus \{0\} \times S$ (in binary)

- Starting configuration $s(c)$, $s \in S$, $c \in \mathbb{N}$

Countdown games

$(S, T), \quad S = S_\exists \uplus S_\forall$

$T \subseteq S \times \mathbb{N} \setminus \{0\} \times S$ (in binary)

- Starting configuration $s(c)$, $s \in S$, $c \in \mathbb{N}$
- Two players: $\exists$ and $\forall$. If $s \in S_q$ it's q's turn

Countdown games

$(S, T), \quad S = S_\exists \uplus S_\forall$

$T \subseteq S \times \mathbb{N} \setminus \{0\} \times S$ (in binary)

- Starting configuration $s(c)$, $s \in S$, $c \in \mathbb{N}$
- Two players: $\exists$ and $\forall$. If $s \in S_q$ it's q's turn
- q chooses $t = (s, n, s')$ and moves to $s'(c - n)$

  (we can assume each s has two outgoing transitions)

Countdown games

$(S, T), \quad S = S_\exists \uplus S_\forall$

$T \subseteq S \times \mathbb{N} \setminus \{0\} \times S$ (in binary)

- Starting configuration $s(c)$, $s \in S$, $c \in \mathbb{N}$
- Two players: $\exists$ and $\forall$. If $s \in S_q$ it's q's turn
- q chooses $t = (s, n, s')$ and moves to $s'(c - n)$

  (we can assume each s has two outgoing transitions)
- If $c - n = 0$ then $\exists$ wins,

  if $c - n < 0$ then $\forall$ wins, otherwise continue

Countdown games

$(S, T), \quad S = S_\exists \uplus S_\forall$

$T \subseteq S \times \mathbb{N} \setminus \{0\} \times S$ (in binary)

- Starting configuration $s(c)$, $s \in S$, $c \in \mathbb{N}$
- Two players: $\exists$ and $\forall$. If $s \in S_q$ it's q's turn
- q chooses $t = (s, n, s')$ and moves to $s'(c - n)$

  (we can assume each s has two outgoing transitions)
- If $c - n = 0$ then $\exists$ wins,

  if $c - n < 0$ then $\forall$ wins, otherwise continue

Lemma (Jurdziński et al., 2008)

GIVEN: (S, T) and $s(c)$ DETERMINE: does $\exists$ have a winning strategy

is EXPTIME-complete.

Encoding

Given (S, T) and $s(c)$ $\quad (c < M = 2^n)$

Encoding

Given (S, T) and $s(c)$ $\quad (c < M = 2^n)$

Construction of $(Q, \Delta_1, \Delta_2, q_0)$ (with $\div 2$):

- $S \subseteq Q$

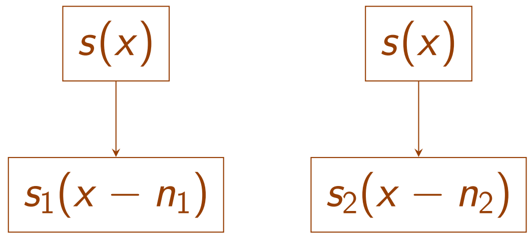
Encoding

Given (S, T) and $s(c)$ $\quad (c < M = 2^n)$

Construction of $(Q, \Delta_1, \Delta_2, q_0)$ (with $\div 2$):

- $S \subseteq Q$
- For every $s \in S_\exists$ let $(s, n_1, s_1), (s, n_2, s_2) \in T$

  Add $(s, -n_1, s_1), (s, -n_2, s_2) \in \Delta_1$



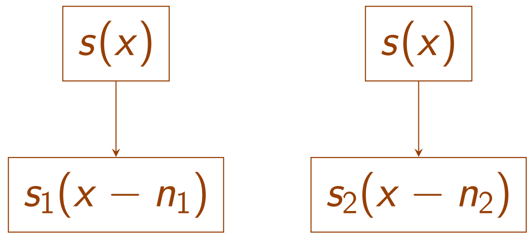
Encoding

Given (S, T) and $s(c)$ $\quad (c < M = 2^n)$

Construction of $(Q, \Delta_1, \Delta_2, q_0)$ (with $\div 2$):

- $S \subseteq Q$
- For every $s \in S_\exists$ let $(s, n_1, s_1), (s, n_2, s_2) \in T$

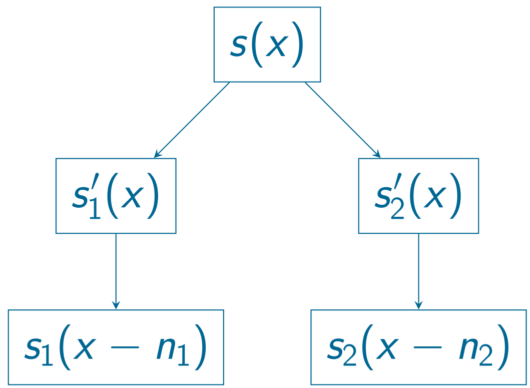
  Add $(s, -n_1, s_1), (s, -n_2, s_2) \in \Delta_1$



- For every $s \in S_\forall$ let $(s, n_1, s_1), (s, n_2, s_2) \in T$

  Add G2 that copies value in s to s'_1 and s'_2

  Add $(s'_1, -n_1, s_1), (s'_2, -n_2, s_2) \in \Delta_1$



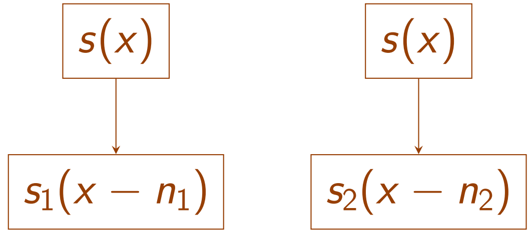
Encoding

Given (S, T) and $s(c)$ $\quad (c < M = 2^n)$

Construction of $(Q, \Delta_1, \Delta_2, q_0)$ (with $\div 2$):

- $S \subseteq Q$
- For every $s \in S_\exists$ let $(s, n_1, s_1), (s, n_2, s_2) \in T$

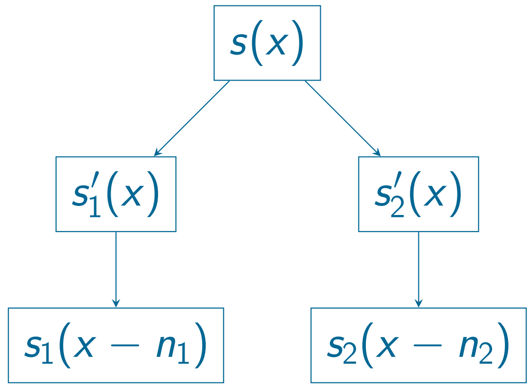
  Add $(s, -n_1, s_1), (s, -n_2, s_2) \in \Delta_1$



- For every $s \in S_\forall$ let $(s, n_1, s_1), (s, n_2, s_2) \in T$

  Add G2 that copies value in s to s_1' and s_2'

  Add $(s_1', -n_1, s_1), (s_2', -n_2, s_2) \in \Delta_1$



- For every $s \in S$

  Add $(s, 0, q_0) \in \Delta_1$

Approaches for 1-BoBrVASS

Can we simulate $\div 2$ or $\times 2$ in 1-BoBrVASS?

Approaches for 1-BoBrVASS

Can we simulate $\div 2$ or $\times 2$ in 1-BoBrVASS?

We prove that no

Approaches for 1-BoBrVASS

Can we simulate $\div 2$ or $\times 2$ in 1-$\textsc{BoBrVASS}$?

We prove that no

- Is it EXPTIME-hard?

  Then one should encode a full binary tree of exponential height

Approaches for 1-BoBrVASS

Can we simulate $\div 2$ or $\times 2$ in 1-BoBrVASS?

We prove that no

- Is it EXPTIME-hard?

  Then one should encode a full binary tree of exponential height

  A gadget like G2 would be useful

Approaches for 1-BoBrVASS

Can we simulate $\div 2$ or $\times 2$ in 1-BoBrVASS?

We prove that no

- Is it EXPTIME-hard?

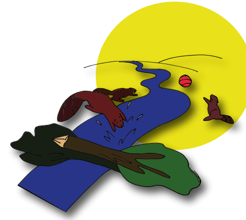
  Then one should encode a full binary tree of exponential height

  A gadget like G2 would be useful

- Is it in PSPACE?

  It suffices to show that computation trees can be of exponential size

Approaches for 1-BoBrVASS



Can we simulate $\div 2$ or $\times 2$ in 1-BoBrVASS?

We prove that no

- Is it EXPTIME-hard?

  Then one should encode a full binary tree of exponential height

  A gadget like G2 would be useful

- Is it in PSPACE?

  It suffices to show that computation trees can be of exponential size

  That's how PSPACE upper bound for 1-BrVASS is proved

  Nothing from [Figueira et al., 2017] work for 1-BoBrVASS

Conclusion

- 2-BoBrVASS is EXPTIME-complete

  What about 1-BoBrVASS (In EXPTIME, PSPACE-hard)?

Conclusion

- 2-BoBrVASS is EXPTIME-complete

  What about 1-BoBrVASS (In EXPTIME, PSPACE-hard)?

- Connections with timed automata

  1-BoVASS: timed automata [Haase et al., 2012]

  1-BoBrVASS: timed pushdown automata [Clemente et al., 2017]

Conclusion

- 2-BoBrVASS is EXPTIME-complete

  What about 1-BoBrVASS (In EXPTIME, PSPACE-hard)?

- Connections with timed automata

  1-BoVASS: timed automata [Haase et al., 2012]

  1-BoBrVASS: timed pushdown automata [Clemente et al., 2017]

- Connections with VASS

  1-BoVASS $\leq$ 2-VASS [Blondin et al., 2015]

  1-BoBrVASS $\leq$ 2-BrVASS [this paper]

Conclusion

- 2-BoBrVASS is EXPTIME-complete

  What about 1-BoBrVASS (In EXPTIME, PSPACE-hard)?

- Connections with timed automata

  1-BoVASS: timed automata [Haase et al., 2012]

  1-BoBrVASS: timed pushdown automata [Clemente et al., 2017]

- Connections with VASS

  1-BoVASS $\leq$ 2-VASS [Blondin et al., 2015]

  1-BoBrVASS $\leq$ 2-BrVASS [this paper]

- Some explanation: 'bobr' is 'beaver' in polish