Revisiting Parameter Synthesis for One-Counter Automata

Ritam Raha [1,2]

(Joint work with Guillermo Alberto Perez [1])

[1]University of Antwerp, Antwerp, Belgium

[2]LaBRI, Université de Bordeaux, France

OFCOURSE talk series, MPI-SWS, Germany

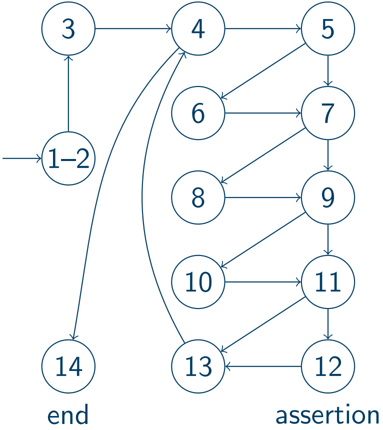
Universiteit Antwerpen

Using the control flow graph

```
x = 0
i = 0
i += x
while i >= 0:
    if i == 0:
        print("Hello world!")
    if i == 1:
        print("Lockdown = pain")
    if i == 2:
        print("P=NP!")
    if i >= 3:
        assert(False)
    i -= 1
# end program
```

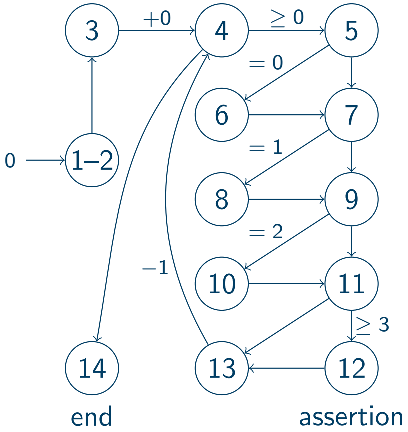
Using the control flow graph

```python
x = 0
i = 0
i += x
while i >= 0:
    if i == 0:
        print("Hello world!")
    if i == 1:
        print("Lockdown = pain")
    if i == 2:
        print("P=NP!")
    if i >= 3:
        assert(False)
    i -= 1
# end program
```



Extending the CFG with a counter

```
x = 0
i = 0
i += x
while i >= 0:
    if i == 0:
        print("Hello world!")
    if i == 1:
        print("Lockdown = pain")
    if i == 2:
        print("P=NP!")
    if i >= 3:
        assert(False)
    i -= 1
# end program
```

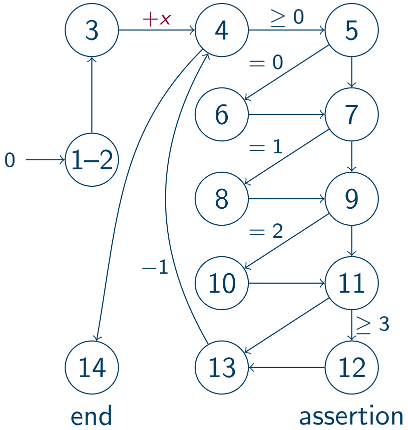


Parametric one-counter automata

```
def funprint(x):
    i = 0
    i += x
    while i >= 0:
        if i == 0:
            print("Hello world!")
        if i == 1:
            print("Lockdown = pain")
        if i == 2:
            print("P=NP!")
        if i >= 3:
            assert(False)
        i -= 1
    # end program
```

Parametric one-counter automata

```python
def funprint(x):
    i = 0
    i += x
    while i >= 0:
        if i == 0:
            print("Hello world!")
        if i == 1:
            print("Lockdown = pain")
        if i == 2:
            print("P=NP!")
        if i >= 3:
            assert(False)
        i -= 1
    # end program
```



Outline

Synthesis Problems for One-Counter Automata

Presburger Arithmetic with Divisibility

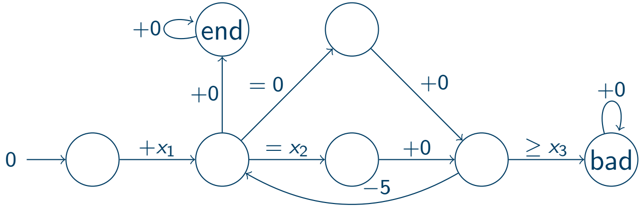
Encoding Synthesis Problems into PAD

Going back to logic: BIL

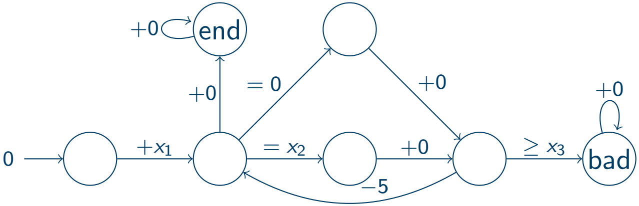
Parametric One-Counter Automata

Natural-valued parameters

$X = \{x_1, \ldots, x_n\}$



Parametric One-Counter Automata



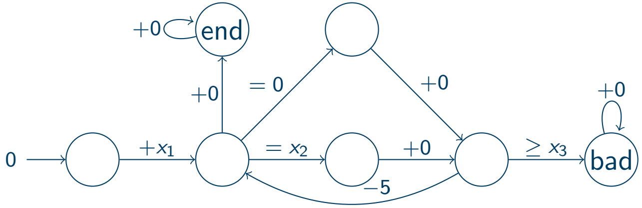
Natural-valued parameters

$X = \{x_1, \ldots, x_n\}$

Succinct OCA with Parameters

$\mathcal{A} = (Q, q_0, T, \delta, X)$, where $\delta : T \to Op$ with Op the union of

Parametric One-Counter Automata



Natural-valued parameters

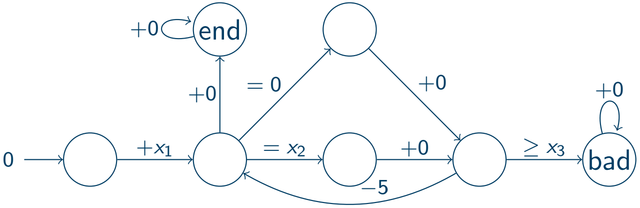
$X = \{x_1, \ldots, x_n\}$

Succinct OCA with Parameters

$\mathcal{A} = (Q, q_0, T, \delta, X)$, where $\delta : T \to Op$ with Op the union of

- $CU := \{+a, -a : a \in \mathbb{N}\}$, $CT := \{= a, \geq a : a \in \mathbb{N}\}$

Parametric One-Counter Automata



Natural-valued parameters

$X = \{x_1, \ldots, x_n\}$

Succinct OCA with Parameters

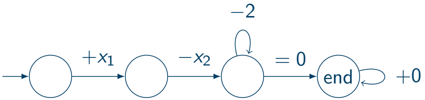
$\mathcal{A} = (Q, q_0, T, \delta, X)$, where $\delta : T \to Op$ with Op the union of

- $CU := \{+a, -a : a \in \mathbb{N}\}$, $CT := \{= a, \geq a : a \in \mathbb{N}\}$
- $PU := \{+x, -x : x \in X\}$, $PT := \{= x, \geq x : x \in X\}$

Synthesis problems

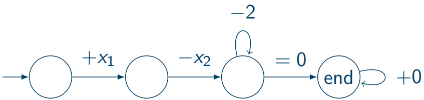
Definition (Parameter-value synthesis)

Is there some valuation $V : X \to \mathbb{N}$ such that all (infinite) runs of $\mathcal{A}$ satisfy a given ω-regular property?



	LTL	Reachability	Safety	Büchi	coBüchi
Lower bound	**PSPACE**-hard	**coNP**-hard	— $\mathbf{NP}^{\mathbf{NP}}$-hard —		
Upper bound	in **N3EXP**	— in **N2EXP** —			

Haase-Lechner approach via Logic



Logical formula: $x_1 \geq 0 \wedge x_1 \geq x_2 \wedge 2 | x_1 - x_2$

Presburger Arithmetic with divisibility!

Outline

Presburger arithmetic with divisibility (PAD)

- Presburger arithmetic (PA) $= \mathrm{FO}(\mathbb{Z}, 0, 1, +, <)$

Presburger arithmetic with divisibility (PAD)

- Presburger arithmetic (PA) $= \mathrm{FO}(\mathbb{Z}, 0, 1, +, <)$
- Presburger arithmetic with divisibility (PAD) $=$ PA $+ \mid$
  $(a \mid b \iff \exists c \in \mathbb{Z} : b = ac)$

Presburger arithmetic with divisibility (PAD)

- Presburger arithmetic (PA) $= \mathrm{FO}(\mathbb{Z}, 0, 1, +, <)$
- Presburger arithmetic with divisibility (PAD) $=$ PA $+ \mid$
  $(a \mid b \iff \exists c \in \mathbb{Z} : b = ac)$

Theorem (Robinson'49, Lipshitz'81)

Full PAD is undecidable; one alternation suffices for undecidability.

Presburger arithmetic with divisibility (PAD)

- Presburger arithmetic (PA) $= \mathrm{FO}(\mathbb{Z}, 0, 1, +, <)$
- Presburger arithmetic with divisibility (PAD) = PA $+ \mid$
  $(a \mid b \iff \exists c \in \mathbb{Z} : b = ac)$

Theorem (Robinson'49, Lipshitz'81)

Full PAD is undecidable; one alternation suffices for undecidability.

Theorem (Lipshitz'78, Lechner-Ouaknine-Worrell'15)

The existential fragment of PAD (EPAD) is decidable and in **NEXP**.

Restriction: $\forall\exists_R$PAD

- $\forall\exists_R\text{PAD} = \forall x_1 \ldots \forall x_n \exists y_1 \ldots \exists y_m . \varphi(\mathbf{x}, \mathbf{y})$
  - in φ, divisibilities of the form $f(\mathbf{x}) \mid g(\mathbf{x}, \mathbf{y})$

Restriction: $\forall\exists_R$PAD

- $\forall\exists_R\text{PAD} = \forall x_1 \ldots \forall x_n \exists y_1 \ldots \exists y_m.\, \varphi(\mathbf{x}, \mathbf{y})$
  - in φ, divisibilities of the form $f(\mathbf{x}) \mid g(\mathbf{x}, \mathbf{y})$
- $\forall\exists_R\text{PAD}^+ = \forall\exists_R\text{PAD}$ with $\neg$ not allowed before divisibility
  - A negation normal form where $\mid$ cannot be negated

Restriction: $\forall\exists_R$PAD

- $\forall\exists_R\text{PAD} = \forall x_1 \ldots \forall x_n \exists y_1 \ldots \exists y_m.\, \varphi(\mathbf{x}, \mathbf{y})$
  - in φ, divisibilities of the form $f(\mathbf{x}) \mid g(\mathbf{x}, \mathbf{y})$
- $\forall\exists_R\text{PAD}^+ = \forall\exists_R\text{PAD}$ with $\neg$ not allowed before divisibility
  - A negation normal form where $\mid$ cannot be negated

Claim (Bozga-Iosif'05, Lechner'15)

The synthesis problems for SOCAP are decidable via an encoding into $\forall\exists_R\text{PAD}^+$.

Outline

Encoding chronology

What we need?

Formulas $\varphi(\mathbf{x})$ such that $V : X \to \mathbb{N}$ satisfies φ iff $\mathcal{A}$ has a V-run reaching t from s.

Encoding chronology

What we need?

Formulas $\varphi(\mathbf{x})$ such that $V : X \to \mathbb{N}$ satisfies φ iff $\mathcal{A}$ has a V-run reaching t from s.

Witness or Certificates (Reachability Certificates)

Encoding chronology

What we need?

Formulas $\varphi(\mathbf{x})$ such that $V : X \to \mathbb{N}$ satisfies φ iff $\mathcal{A}$ has a V-run reaching t from s.

Witness or Certificates (Reachability Certificates) : three types

Encoding chronology

What we need?

Formulas $\varphi(\mathbf{x})$ such that $V : X \to \mathbb{N}$ satisfies φ iff $\mathcal{A}$ has a V-run reaching t from s.

Witness or Certificates (Reachability Certificates) : three types

- no positive cycles (type 1)
- begins with a positive cycle and ends with a negative cycle (type 2)
- no negative cycles (type 3)

Encoding chronology

What we need?

Formulas $\varphi(\mathbf{x})$ such that $V : X \to \mathbb{N}$ satisfies φ iff $\mathcal{A}$ has a V-run reaching t from s.

Witness or Certificates (Reachability Certificates) : three types

- no positive cycles (type 1)
- begins with a positive cycle and ends with a negative cycle (type 2)
- no negative cycles (type 3)

Theorem (Haase et. al. '09)

If $(q, c) \rightsquigarrow (q', c')$ without zero-test, then $(q, c) \xrightarrow{\pi_1\pi_2\pi_3}$ such that, π_1, π_2 and π_3 are type 1, type 2 and type 3 reachability certificates.

Encoding chronology

What we need?

Formulas $\varphi(\mathbf{x})$ such that $V : X \to \mathbb{N}$ satisfies φ iff $\mathcal{A}$ has a V-run reaching t from s.

Witness or Certificates (Reachability Certificates) : three types

- no positive cycles (type 1)
- begins with a positive cycle and ends with a negative cycle (type 2)
- no negative cycles (type 3)

Theorem (Haase et. al. '09)

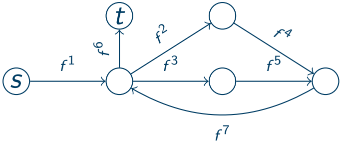
If $(q, c) \rightsquigarrow (q', c')$ without zero-test, then $(q, c) \xrightarrow{\pi_1\pi_2\pi_3}$ such that, π_1, π_2 and π_3 are type 1, type 2 and type 3 reachability certificates.

Now what do we need?

PAD formulas that encodes certificates

Now what do we need?

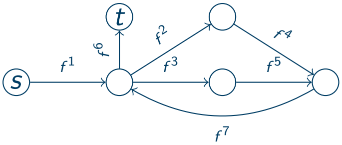
PAD formulas that encodes certificates



$$\bigwedge_{q \notin \{s,t\}} \left(\sum_{p \in T(\cdot,q)} f(p,q) = \sum_{r \in T(q,\cdot)} f(q,r) \right) \quad (1)$$

Now what do we need?

PAD formulas that encodes certificates



$$\bigwedge_{q \notin \{s,t\}} \left(\sum_{p \in T(\cdot,q)} f(p,q) = \sum_{r \in T(q,\cdot)} f(q,r) \right) \qquad (1)$$

Theorem (Euler's theorem for digraphs)

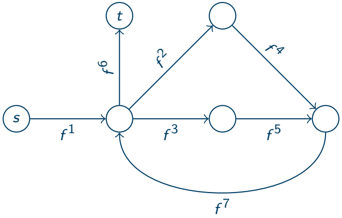
There is an s–t path iff there is a valuation of the f_i such that

- *the subgraph induced by the support and $\{(t,s)\}$ are strongly connected,*
- *it satisfies (1) and $\sum_{p \in T(\cdot,s)} f(p,s) - \sum_{r \in T(s,\cdot)} f(s,r) = 1$.*

From paths to runs

A formula per subgraph

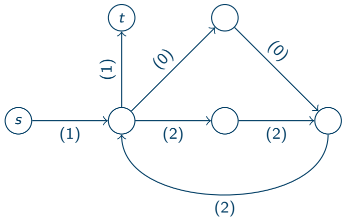
Now $\varphi_{\text{flow}}(\mathbf{f})$ is a disjunction of the flow constraints over all subgraphs which satisfy the support condition.



From paths to runs

A formula per subgraph

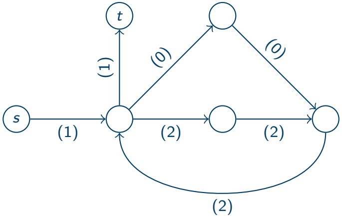
Now $\varphi_{\text{flow}}(\mathbf{f})$ is a disjunction of the flow constraints over all subgraphs which satisfy the support condition.



From paths to runs

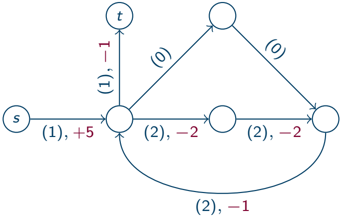
A formula per subgraph

Now $\varphi_{\text{flow}}(\mathbf{f})$ is a disjunction of the flow constraints over all subgraphs which satisfy the support condition.



A formula per subgraph

Now $\varphi_{\text{flow}}(\mathbf{f})$ is a disjunction of the flow constraints over all subgraphs which satisfy the support condition.



From paths to runs

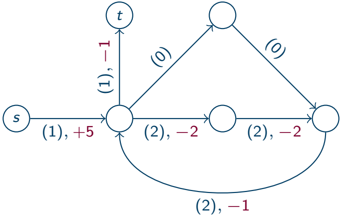
A formula per subgraph

Now $\varphi_{\text{flow}}(\mathbf{f})$ is a disjunction of the flow constraints over all subgraphs which satisfy the support condition.

A path is not a run

Not every path can be lifted to a run because:

- of equality tests,
- the lower-bound tests,
- the counter value cannot go negative



Decomposed flows

Flow Decomposition

We want $f_1, f_2, \ldots, f_{|Q|}$ such that $f = \sum_{i=1}^{|Q|} f_i$ and:

Decomposed flows

Flow Decomposition

We want $f_1, f_2, \ldots, f_{|Q|}$ such that $f = \sum_{i=1}^{|Q|} f_i$ and:

- f_i is a flow witnessing a q_i–q_{i+1} path,

Decomposed flows

Flow Decomposition

We want $f_1, f_2, \ldots, f_{|Q|}$ such that $f = \sum_{i=1}^{|Q|} f_i$ and:

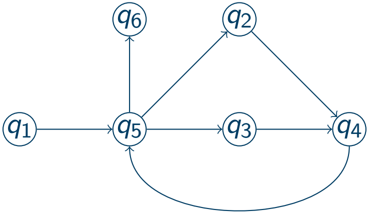
- f_i is a flow witnessing a q_i–q_{i+1} path,
- $f_j(p, q_i) = 0$ for all $i \leq j$ and all $(p, q_i) \in T$.

Decomposed flows

Flow Decomposition

We want $f_1, f_2, \ldots, f_{|Q|}$ such that $f = \sum_{i=1}^{|Q|} f_i$ and:

- f_i is a flow witnessing a q_i–q_{i+1} path,
- $f_j(p, q_i) = 0$ for all $i \leq j$ and all $(p, q_i) \in T$.

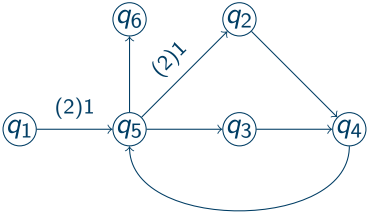


Decomposed flows

Flow Decomposition

We want $f_1, f_2, \ldots, f_{|Q|}$ such that $f = \sum_{i=1}^{|Q|} f_i$ and:

- f_i is a flow witnessing a q_i–q_{i+1} path,
- $f_j(p, q_i) = 0$ for all $i \leq j$ and all $(p, q_i) \in T$.

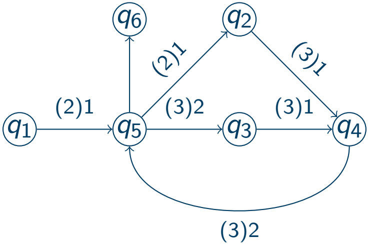


Decomposed flows

Flow Decomposition

We want $f_1, f_2, \ldots, f_{|Q|}$ such that $f = \sum_{i=1}^{|Q|} f_i$ and:

- f_i is a flow witnessing a q_i–q_{i+1} path,
- $f_j(p, q_i) = 0$ for all $i \leq j$ and all $(p, q_i) \in T$.

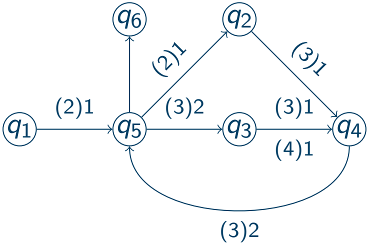


Decomposed flows

Flow Decomposition

We want $f_1, f_2, \ldots, f_{|Q|}$ such that $f = \sum_{i=1}^{|Q|} f_i$ and:

- f_i is a flow witnessing a q_i–q_{i+1} path,
- $f_j(p, q_i) = 0$ for all $i \leq j$ and all $(p, q_i) \in T$.

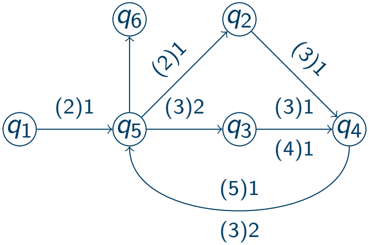


Decomposed flows

Flow Decomposition

We want $f_1, f_2, \ldots, f_{|Q|}$ such that $f = \sum_{i=1}^{|Q|} f_i$ and:

- f_i is a flow witnessing a q_i–q_{i+1} path,
- $f_j(p, q_i) = 0$ for all $i \leq j$ and all $(p, q_i) \in T$.

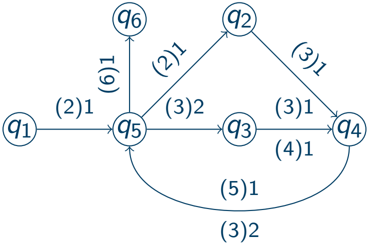


Decomposed flows

Flow Decomposition

We want $f_1, f_2, \ldots, f_{|Q|}$ such that $f = \sum_{i=1}^{|Q|} f_i$ and:

- f_i is a flow witnessing a q_i–q_{i+1} path,
- $f_j(p, q_i) = 0$ for all $i \leq j$ and all $(p, q_i) \in T$.

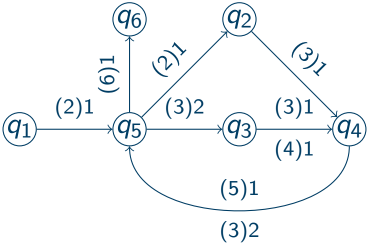


Decomposed flows

Flow Decomposition

We want $f_1, f_2, \ldots, f_{|Q|}$ such that $f = \sum_{i=1}^{|Q|} f_i$ and:

- f_i is a flow witnessing a q_i–q_{i+1} path,
- $f_j(p, q_i) = 0$ for all $i \leq j$ and all $(p, q_i) \in T$.



type 1: No positive cycles

To check the path can be lifted to a run:

$$\bigwedge_{m=1}^{|Q|} \sum_{i=1}^{m} \sum_{t \in T} f_i(t)\delta(t) \geq 0$$

Weight formulas using divisibility

A formula per case (1, 2, 3)

Now $\varphi_{\text{weight}}(\mathbf{x}, \mathbf{f})$ is a disjunction of the weight constraints over all decompositions.

Weight constraints use multiplication! $\sum f_i x_i \geq 0$

Weight formulas using divisibility

A formula per case (1, 2, 3)

Now $\varphi_{\mathrm{weight}}(\mathbf{x}, \mathbf{f})$ is a disjunction of the weight constraints over all decompositions.

Weight constraints use multiplication! $\sum f_i x_i \geq 0$

Divisibility to the rescue

Replace fx with a product variable z_{fx}:

$$(x_i \mid z_{f_i x_i}) \wedge (x_i > 0 \leftrightarrow z_{f_i x_i} > 0) \wedge \left(\sum z_{f_i x_i} \geq 0\right)$$

Final PAD encoding

A formula per case

For each case, we get formulas like the following:

$$\varphi(\mathbf{x}) \equiv \exists z_1 z_2 \cdots \bigvee_{\substack{\text{subgraphs}\\ \text{decomp}}} \bigwedge_{i \in I} (g_i(\mathbf{x}) \mid h_i(\mathbf{z})) \wedge \varphi_{nopos}(\mathbf{x}) \wedge \mathbf{x}, \mathbf{z} \geq \mathbf{0}$$

Final PAD encoding

A formula per case

For each case, we get formulas like the following:

$$\varphi(\mathbf{x}) \equiv \exists z_1 z_2 \cdots \bigvee_{\substack{\text{subgraphs} \\ \text{decomp}}} \bigwedge_{i \in I} (g_i(\mathbf{x}) \mid h_i(\mathbf{z})) \wedge \varphi_{nopos}(\mathbf{x}) \wedge \mathbf{x}, \mathbf{z} \geq \mathbf{0}$$

The safety synthesis problem

- Positive answer if $\forall \mathbf{x}.\Phi(\mathbf{x})$ is false
- $\forall \mathbf{x}.\Phi(\mathbf{x})$ is a sentence in $\forall\exists_R \mathrm{PAD}^+$

Outline

Synthesis Problems for One-Counter Automata

Presburger Arithmetic with Divisibility

Encoding Synthesis Problems into PAD

Going back to logic: BIL

Restriction: $\forall\exists_R$PAD

- $\forall\exists_R\text{PAD} = \forall x_1 \ldots \forall x_n \exists y_1 \ldots \exists y_m.\, \varphi(\mathbf{x}, \mathbf{y})$
  - in φ, divisibilities of the form $f(\mathbf{x}) \mid g(\mathbf{x}, \mathbf{y})$
- $\forall\exists_R\text{PAD}^+ = \forall\exists_R$PAD with $\neg$ not allowed before divisibility
  - A negation normal form where $\mid$ cannot be negated

Claim (Bozga-Iosif'05, Lechner'15)

The synthesis problems for SOCAP are decidable via an encoding into $\forall\exists_R\text{PAD}^+$.

$\forall\exists_R$PAD & Undecidability

Theorem (Bozga-Iosif'05)

$\forall\exists_R PAD$ is undecidable.

Theorem (Bozga-Iosif'05)

$\forall\exists_R PAD$ is undecidable.

Idea: LCM $\Rightarrow$ Square (x^2) $\Rightarrow$ Multiplication

Theorem (Bozga-Iosif'05)

$\forall\exists_R$PAD is undecidable.

Idea: LCM $\Rightarrow$ Square (x^2) $\Rightarrow$ Multiplication

- $\forall\exists_R\text{PAD}^+ \equiv \forall\exists_R\text{PAD}$:

$\forall\exists_R$PAD & Undecidability

Theorem (Bozga-Iosif'05)

$\forall\exists_R PAD$ is undecidable.

Idea: LCM $\Rightarrow$ Square (x^2) $\Rightarrow$ Multiplication

- $\forall\exists_R\text{PAD}^+ \equiv \forall\exists_R\text{PAD}$:

$$\neg(a \mid b) \iff \exists q \exists r (b = aq + r) \land (0 < r < b)$$

Theorem (Bozga-Iosif'05)

$\forall\exists_R PAD$ is undecidable.

Idea: LCM $\Rightarrow$ Square (x^2) $\Rightarrow$ Multiplication

- $\forall\exists_R$PAD$^+$ $\equiv$ $\forall\exists_R$PAD :

Undecidable!

$$\neg(a \mid b) \iff \exists q \exists r (b = aq + r) \wedge (0 < r < b)$$

A stronger restriction of $\forall\exists_R$PAD

- $\forall\exists_R\text{PAD} = \forall x_1 \dots \forall x_n \exists y_1 \dots \exists y_m . \varphi(\mathbf{x}, \mathbf{z})$ with divisibilities $f(\mathbf{x}) \mid g(\mathbf{x}, \mathbf{y})$

A stronger restriction of $\forall\exists_R$PAD

- $\forall\exists_R\text{PAD} = \forall x_1 \ldots \forall x_n \exists y_1 \ldots \exists y_m.\, \varphi(\mathbf{x}, \mathbf{z})$ with divisibilities $f(\mathbf{x}) \mid g(\mathbf{x}, \mathbf{y})$
- The Bozga-Iosif-Lechner (BIL) fragment of $\forall\exists_R$PAD:

$$\forall x_1 \ldots \forall x_n \in \mathbb{N},\ \exists y_1 \ldots \exists y_m \bigvee_{i \in I} \bigwedge_{j \in J_i} \Big(\underline{f_j(\mathbf{x}) \mid g_j(\mathbf{x}, \mathbf{y})} \wedge f_j(x) > 0 \Big) \wedge \underline{\varphi_i(\mathbf{x})} \wedge \mathbf{y} \geq \mathbf{0}$$

A stronger restriction of $\forall\exists_R$PAD

- $\forall\exists_R\text{PAD} = \forall x_1 \ldots \forall x_n \exists y_1 \ldots \exists y_m . \varphi(\mathbf{x}, \mathbf{z})$ with divisibilities $f(\mathbf{x}) \mid g(\mathbf{x}, \mathbf{y})$
- The Bozga-Iosif-Lechner (BIL) fragment of $\forall\exists_R$PAD:

$$\forall x_1 \ldots \forall x_n \in \mathbb{N},\ \exists y_1 \ldots \exists y_m \bigvee_{i \in I} \bigwedge_{j \in J_i} \Big(\underline{f_j(\mathbf{x}) \mid g_j(\mathbf{x}, \mathbf{y})} \wedge f_j(x) > 0 \Big) \wedge \underline{\varphi_i(\mathbf{x})} \wedge \mathbf{y} \geq \mathbf{0}$$

Theorem

The BIL fragment is decidable and in **coN2EXP**.

A stronger restriction of $\forall\exists_R$PAD

- $\forall\exists_R\mathrm{PAD} = \forall x_1 \ldots \forall x_n \exists y_1 \ldots \exists y_m.\, \varphi(\mathbf{x}, \mathbf{z})$ with divisibilities $f(\mathbf{x}) \mid g(\mathbf{x}, \mathbf{y})$
- The Bozga-Iosif-Lechner (BIL) fragment of $\forall\exists_R$PAD:

$$\forall x_1 \ldots \forall x_n \in \mathbb{N},\ \exists y_1 \ldots \exists y_m \bigvee_{i \in I} \bigwedge_{j \in J_i} \left(\underline{f_j(\mathbf{x}) \mid g_j(\mathbf{x}, \mathbf{y})} \wedge f_j(x) > 0 \right) \wedge \underline{\varphi_i(\mathbf{x})} \wedge \mathbf{y} \geq \mathbf{0}$$

Theorem

The BIL fragment is decidable and in **coN2EXP**.

Idea. Quantifier elimination (Generalized Chinese Remainder Theorem!) - Similar idea to Bozga & Iosif's work!

BIL is decidable!

Theorem (Generalized CRT)

Let $m_i \in \mathbb{N}_{>0}$, $a_i, r_i \in \mathbb{Z}$ *for* $1 \leq i \leq n$. *Then,*

$$\exists x \in \mathbb{Z}, \bigwedge_{i=1}^{n} m_i \mid (a_i x - r_i) \Leftrightarrow \bigwedge_{1 \leq i,j \leq n} \gcd(a_i m_j, a_j m_i) \mid (a_i r_j - a_j r_i) \wedge \bigwedge_{i=1}^{n} \gcd(a_i, m_i) \mid r_i$$

The solution for x *is unique modulo* $\mathrm{LCM}(m'_1, \ldots, m'_n)$, *where* $m'_i = \frac{m_i}{\gcd(a_i, m_i)}$.

BIL is decidable!

Example:

$$\forall x \exists y \bigvee_{i \in I} \bigwedge_{j \in J_i} (f_j(x) \mid (\beta_j(x) + \alpha_j(y)) \wedge f_j(x) > 0) \wedge \varphi_i(x) \wedge y \geq 0 \Rightarrow$$

BIL is decidable!

Example:

$$\forall x \exists y \bigvee_{i \in I} \bigwedge_{j \in J_i} (f_j(x) \mid (\beta_j(x) + \alpha_j(y)) \wedge f_j(x) > 0) \wedge \varphi_i(x) \wedge y \geq 0 \Rightarrow$$

$$\forall x \bigvee_{i \in I} \left(\exists y \bigwedge_{j \in J_i} (f_j(x) \mid (\alpha_j(y) - (-\beta_j(x))) \wedge y \geq 0 \right) \wedge \varphi'_i(x) \Rightarrow$$

$$\forall x \bigvee_{i \in I} \left(\bigwedge_{j,k \in J_i} \gcd(\alpha_k f_j(x), \alpha_j f_k(x)) \mid (\alpha_j \beta_k(x) - \alpha_k \beta_j(x)) \wedge \bigwedge_{j \in J_i} \gcd(\alpha_j, f_j(x)) \mid \beta_j(x) \right)$$

$$\wedge\ \varphi'_i(x)$$

BIL is decidable!

Example:

$$\forall x \exists y \bigvee_{i\in I} \bigwedge_{j\in J_i} (f_j(x) \mid (\beta_j(x) + \alpha_j(y)) \wedge f_j(x) > 0) \wedge \varphi_i(x) \wedge y \geq 0 \Rightarrow$$

$$\forall x \bigvee_{i\in I} \left(\exists y \bigwedge_{j\in J_i} (f_j(x) \mid (\alpha_j(y) - (-\beta_j(x))) \wedge y \geq 0 \right) \wedge \varphi'_i(x) \Rightarrow$$

$$\forall x \bigvee_{i\in I} \left(\bigwedge_{j,k\in J_i} \gcd(\alpha_k f_j(x), \alpha_j f_k(x)) \mid (\alpha_j \beta_k(x) - \alpha_k \beta_j(x)) \wedge \bigwedge_{j\in J_i} \gcd(\alpha_j, f_j(x)) \mid \beta_j(x) \right)$$

$$\wedge\ \varphi'_i(x)$$

∀PAD!

Synthesis Problem to BIL!

- Synthesis of SOCA is encodable in BIL fragment.

Synthesis Problem to BIL!

- Synthesis of SOCA is encodable in BIL fragment.
  - Idea. Careful encoding of "Reachability Certificates" to BIL- Similar to previous encoding!

Synthesis Problem to BIL!

- Synthesis of SOCA is encodable in BIL fragment.
  - Idea. Careful encoding of "Reachability Certificates" to BIL- Similar to previous encoding!

Theorem

The reachability, Büchi, coBüchi, and safety parameter synthesis problems for SOCA are all decidable in **N2EXP**. *The LTL synthesis problem for SOCA is decidable in* **N3EXP**.

Synthesis Problem to BIL!

- Synthesis of SOCA is encodable in BIL fragment.
  - Idea. Careful encoding of "Reachability Certificates" to BIL- Similar to previous encoding!

Theorem

The reachability, Büchi, coBüchi, and safety parameter synthesis problems for SOCA are all decidable in **N2EXP**. *The LTL synthesis problem for SOCA is decidable in* **N3EXP**.

- Restrict to Parametric Tests & Constant Updates: synthesis is in **PSPACE**.

Synthesis Problem to BIL!

- Synthesis of SOCA is encodable in BIL fragment.
  - Idea. Careful encoding of "Reachability Certificates" to BIL- Similar to previous encoding!

Theorem

The reachability, Büchi, coBüchi, and safety parameter synthesis problems for SOCA are all decidable in **N2EXP**. *The LTL synthesis problem for SOCA is decidable in* **N3EXP**.

- Restrict to Parametric Tests & Constant Updates: synthesis is in **PSPACE**.
  - Idea. Reduction to Alternating 2-way automata (using idea from Bollig et.al'19)

Conclusion

	LTL	Reachability	Safety	Büchi	coBüchi
Lower bound	**PSPACE**-hard	**coNP**-hard	— $\mathbf{NP}^{\mathbf{NP}}$-hard —		
Upper bound	in **N3EXP**	— in **N2EXP** —			

Conclusion

	LTL	Reachability	Safety	Büchi	coBüchi
Lower bound	**PSPACE**-hard	**coNP**-hard	— $\mathbf{NP}^{\mathbf{NP}}$-hard —		
Upper bound	in **N3EXP**	— in **N2EXP** —			

Summary.

- BIL: largest known decidable fragment of one alternation PAD!

Conclusion

	LTL	Reachability	Safety	Büchi	coBüchi
Lower bound	**PSPACE**-hard	**coNP**-hard	— $\mathbf{NP}^{\mathbf{NP}}$-hard —		
Upper bound	in **N3EXP**	— in **N2EXP** —			

Summary.

- BIL: largest known decidable fragment of one alternation PAD!
- Parameter Synthesis for SOCA is decidable!

Conclusion

	LTL	Reachability	Safety	Büchi	coBüchi
Lower bound	**PSPACE**-hard	**coNP**-hard	— $\mathbf{NP}^{\mathbf{NP}}$-hard —		
Upper bound	in **N3EXP**	— in **N2EXP** —			

Summary.

- BIL: largest known decidable fragment of one alternation PAD!
- Parameter Synthesis for SOCA is decidable!

Open Questions.

- Exact lower bounds: both for BIL & Synthesis problems!

Conclusion

	LTL	Reachability	Safety	Büchi	coBüchi
Lower bound	**PSPACE**-hard	**coNP**-hard	— $\mathbf{NP}^{\mathbf{NP}}$-hard —		
Upper bound	in **N3EXP**	— in **N2EXP** —			

Summary.

- BIL: largest known decidable fragment of one alternation PAD!
- Parameter Synthesis for SOCA is decidable!

Open Questions.

- Exact lower bounds: both for BIL & Synthesis problems!
- BIL to Synthesis: the opposite side reduction?

Thank you for your attention!