

Counter Systems: The Quest for Pushing the Decidability Borders

Stéphane Demri

NYU & CNRS – Marie Curie Fellow

FROCOS & TABLEAUX, September 19th, 2013



Collaborations

- A survey paper and abstract written with Clark Barrett & Morgan Deters (NYU) in the proceedings.
- Some of the presented results are still subject to on-going research:
 - ACSys group, NYU.
 - Amit Dhar, Arnaud Sangnier (LIAFA).
 - Members of ANR REACHARD (LSV, LABRI).
- Less recent collaborations on the subject with M. Bersani, R. Gascon, V. Goranko, D. D'Souza, R. Lazić, etc.

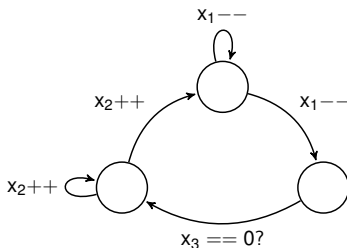
Overview

- 1 Presburger Counter Systems**
 - Definition
 - Decision Problems
 - Subclasses of PCS
- 2 Loops, Path Schemas and Flatness**
 - Loops & Path Schemas
 - Flatness
- 3 Verifying Temporal Properties**
 - Presburger LTL
 - Model-Checking Complexity
- 4 Path Schema Subsumption: An Overview**

Presburger Counter Systems

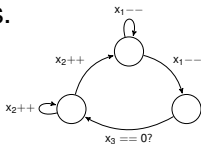
Integer Programs

- Finite-state automaton with counters interpreted by non-negative integers.



Integer Programs

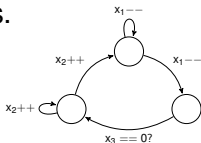
- Finite-state automaton with counters interpreted by non-negative integers.



- Many applications:
 - Broadcast protocols, Petri nets, . . .
 - Programs with pointer variables. [Bouajjani et al., CAV'06]
 - Replicated finite-state programs. [Kaiser & Kroening & Wahl, CAV'10]
 - Relationships with data logics. [Bojańczyk et al., TOCL 11]

Integer Programs

- Finite-state automaton with counters interpreted by non-negative integers.



- Many applications:
 - Broadcast protocols, Petri nets, . . .
 - Programs with pointer variables. [Bouajjani et al., CAV'06]
 - Replicated finite-state programs. [Kaiser & Kroening & Wahl, CAV'10]
 - Relationships with data logics. [Bojańczyk et al., TOCL 11]
- Techniques for model-checking infinite-state systems are required for formal verification.
- But, integer programs can simulate Turing machines.
- Checking safety or liveness properties is undecidable.

Taming Verification of Counter Machines

- Design of subclasses with decidable reachability problems
 - Vector addition systems (\approx Petri nets) [Kosaraju, STOC'82]
 - Flat relational counter machines. [Comon & Jurski, CAV'98]
 - Reversal-bounded counter machines. [Ibarra, JACM 78]
 - Flat affine counter machines with finite monoids.
[Boigelot, PhD 98; Finkel & Leroux, FSTTCS'02]

...

Taming Verification of Counter Machines

- Design of subclasses with decidable reachability problems
 - Vector addition systems (\approx Petri nets) [Kosaraju, STOC'82]
 - Flat relational counter machines. [Comon & Jurski, CAV'98]
 - Reversal-bounded counter machines. [Ibarra, JACM 78]
 - Flat affine counter machines with finite monoids.
[Boigelot, PhD 98; Finkel & Leroux, FSTTCS'02]
 - . . .
- Decision procedures
 - Translation into Presburger arithmetic.
[Fribourg & Olsén, CONCUR'97; Finkel & Leroux, FSTTCS'02]
 - Direct analysis on runs. [Rackoff, TCS 78]
 - Approximating reachability sets. [Karp & Miller, JCSS 69]
 - Well-structured transition systems.
[Finkel & Schnoebelen, TCS 01]

Taming Verification of Counter Machines

- Design of subclasses with decidable reachability problems
 - Vector addition systems (\approx Petri nets) [Kosaraju, STOC'82]
 - Flat relational counter machines. [Comon & Jurski, CAV'98]
 - Reversal-bounded counter machines. [Ibarra, JACM 78]
 - Flat affine counter machines with finite monoids.
[Boigelot, PhD 98; Finkel & Leroux, FSTTCS'02]
 - . . .
- Decision procedures
 - Translation into Presburger arithmetic.
[Fribourg & Olsén, CONCUR'97; Finkel & Leroux, FSTTCS'02]
 - Direct analysis on runs. [Rackoff, TCS 78]
 - Approximating reachability sets. [Karp & Miller, JCSS 69]
 - Well-structured transition systems.
[Finkel & Schnoebelen, TCS 01]
- Tools: FAST, LASH, TREX, FLATA, . . .

A Fundamental Decidable Theory

- First-order theory of $\langle \mathbb{N}, +, \leq \rangle$ introduced by Mojzesz Presburger (1929).
- Use in guards and in symbolic representations for sets of tuples.
- Many properties: decidability, quantifier elimination, quantifier-free fragment in NP, ...

A Fundamental Decidable Theory

- First-order theory of $\langle \mathbb{N}, +, \leq \rangle$ introduced by Mojzesz Presburger (1929).
- Use in guards and in symbolic representations for sets of tuples.
- Many properties: decidability, quantifier elimination, quantifier-free fragment in NP, ...
- Terms $t = a_1x_1 + \dots + a_nx_n + k$ where $a_1, \dots, a_n \in \mathbb{N}$, k is in \mathbb{N} and the x_i 's are variables.
- Presburger formulae: $\phi ::= t \leq t' \mid \neg\phi \mid \phi \wedge \phi \mid \exists x \phi$

Presburger Arithmetic

- **Linear fragment:** no quantification.
- Valuation $v : \text{VAR} \rightarrow \mathbb{N}$ + extension to all terms with

$$v(a_1x_1 + \dots + a_nx_n + k) \stackrel{\text{def}}{=} a_1v(x_1) + \dots + a_nv(x_n) + k$$

Presburger Arithmetic

- **Linear fragment:** no quantification.
- Valuation $v : \text{VAR} \rightarrow \mathbb{N}$ + extension to all terms with

$$v(a_1x_1 + \dots + a_nx_n + k) \stackrel{\text{def}}{=} a_1v(x_1) + \dots + a_nv(x_n) + k$$

- $v \models t \leq t'$ iff $v(t) \leq v(t')$; $v \models \phi \wedge \phi'$ iff $v \models \phi$ and $v \models \phi'$,
- $v \models \exists x \phi \stackrel{\text{def}}{\Leftrightarrow}$ there is $n \in \mathbb{N}$ such that $v[x \mapsto n] \models \phi$.

Presburger Arithmetic

- **Linear fragment:** no quantification.
- Valuation $v : \text{VAR} \rightarrow \mathbb{N} +$ extension to all terms with

$$v(a_1x_1 + \dots + a_nx_n + k) \stackrel{\text{def}}{=} a_1v(x_1) + \dots + a_nv(x_n) + k$$

- $v \models t \leq t'$ iff $v(t) \leq v(t')$; $v \models \phi \wedge \phi'$ iff $v \models \phi$ and $v \models \phi'$,
- $v \models \exists x \phi \stackrel{\text{def}}{\Leftrightarrow}$ there is $n \in \mathbb{N}$ such that $v[x \mapsto n] \models \phi$.
- Formula $\phi(x_1, \dots, x_n)$ with $n \geq 1$ free variables:

$$\llbracket \phi(x_1, \dots, x_n) \rrbracket \stackrel{\text{def}}{=} \{ \langle v(x_1), \dots, v(x_n) \rangle \in \mathbb{N}^n : v \models \phi \}.$$

- ϕ is satisfiable $\stackrel{\text{def}}{\Leftrightarrow}$ there is v such that $v \models \phi$.

Decision Procedures and Tools

- Quantifier elimination and refinements

[Cooper, ML 72; Reddy & Loveland, STOC'78]

- Tools dealing with quantifier-free PA, full PA or quantifier elimination: Z3, CVC4, Alt-Ergo, Yices2, Omega test.

Decision Procedures and Tools

- Quantifier elimination and refinements

[Cooper, ML 72; Reddy & Loveland, STOC'78]

- Tools dealing with quantifier-free PA, full PA or quantifier elimination: Z3, CVC4, Alt-Ergo, Yices2, Omega test.

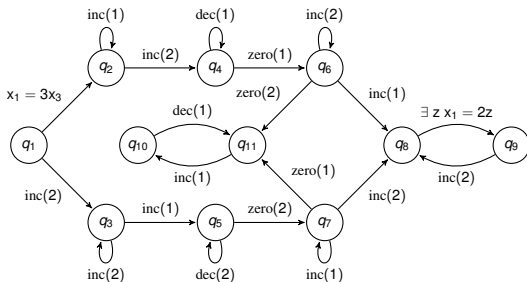
- Automata-based approach.

[Büchi, ZML 60; Boudet & Comon, CAAP'96]

- Automata-based tools for Presburger arithmetic: LIRA, suite of libraries TAPAS, MONA, and LASH.

Presburger Counter Systems (PCS)

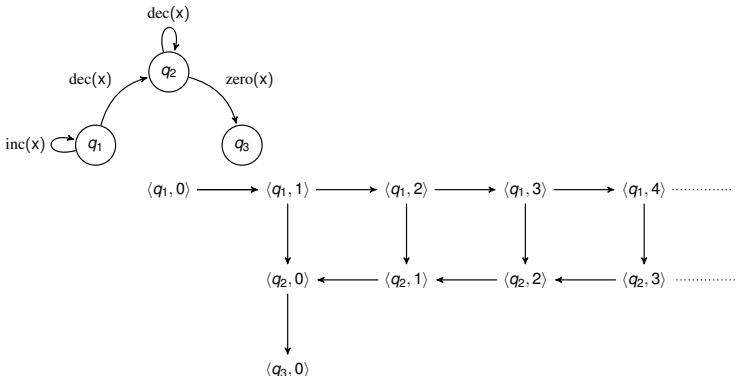
- Presburger Counter System $C = \langle Q, n, \delta \rangle$ of dimension n :
 - Q is a nonempty finite set of control states.
 - $n \geq 1$ is the dimension.
 - $\delta =$ finite set of transitions of the form $t = \langle q, \phi, q' \rangle$ where $q, q' \in Q$ and ϕ is a Presburger formula with free variables $x_1, \dots, x_n, x'_1, \dots, x'_n$.



- Configuration $\langle q, \mathbf{x} \rangle \in \mathfrak{S} = Q \times \mathbb{N}^n$.

Transition System $\mathcal{T}(C)$

- Transition system $\mathcal{T}(C) = \langle \mathcal{G}, \rightarrow \rangle$:
 - $\langle q, \mathbf{x} \rangle \rightarrow \langle q', \mathbf{x}' \rangle \stackrel{\text{def}}{\iff}$ there is $t = \langle q, \phi, q' \rangle$ such that $v[\bar{x} \leftarrow \mathbf{x}, \bar{x}' \leftarrow \mathbf{x}'] \models \phi$



- \rightarrow^* : reflexive and transitive closure of \rightarrow .
(sometimes written Reach_C)

Decision Problems

- Reachability problem:

Input: PCS C , $\langle q_0, \mathbf{x}_0 \rangle$ and $\langle q_f, \mathbf{x}_f \rangle$.

Question: $\langle q_0, \mathbf{x}_0 \rangle \xrightarrow{*} \langle q_f, \mathbf{x}_f \rangle$?

Decision Problems

- Reachability problem:
Input: PCS C , $\langle q_0, \mathbf{x}_0 \rangle$ and $\langle q_f, \mathbf{x}_f \rangle$.
Question: $\langle q_0, \mathbf{x}_0 \rangle \xrightarrow{*} \langle q_f, \mathbf{x}_f \rangle$?
- Control state reachability problem:
Input: PCS C , $\langle q_0, \mathbf{x}_0 \rangle$ and q_f .
Question: $\exists \mathbf{x}_f \langle q_0, \mathbf{x}_0 \rangle \xrightarrow{*} \langle q_f, \mathbf{x}_f \rangle$?

Decision Problems

- Reachability problem:

Input: PCS C , $\langle q_0, \mathbf{x}_0 \rangle$ and $\langle q_f, \mathbf{x}_f \rangle$.

Question: $\langle q_0, \mathbf{x}_0 \rangle \xrightarrow{*} \langle q_f, \mathbf{x}_f \rangle$?

- Control state reachability problem:

Input: PCS C , $\langle q_0, \mathbf{x}_0 \rangle$ and q_f .

Question: $\exists \mathbf{x}_f \langle q_0, \mathbf{x}_0 \rangle \xrightarrow{*} \langle q_f, \mathbf{x}_f \rangle$?

- Control state repeated reachability problem:

Input: PCS C , $\langle q_0, \mathbf{x}_0 \rangle$ and q_f .

Question: is there an infinite run starting from $\langle q_0, \mathbf{x}_0 \rangle$ such that the control state q_f is repeated infinitely often?

Decision Problems

- Reachability problem:
Input: PCS C , $\langle q_0, \mathbf{x}_0 \rangle$ and $\langle q_f, \mathbf{x}_f \rangle$.
Question: $\langle q_0, \mathbf{x}_0 \rangle \xrightarrow{*} \langle q_f, \mathbf{x}_f \rangle$?
- Control state reachability problem:
Input: PCS C , $\langle q_0, \mathbf{x}_0 \rangle$ and q_f .
Question: $\exists \mathbf{x}_f \langle q_0, \mathbf{x}_0 \rangle \xrightarrow{*} \langle q_f, \mathbf{x}_f \rangle$?
- Control state repeated reachability problem:
Input: PCS C , $\langle q_0, \mathbf{x}_0 \rangle$ and q_f .
Question: is there an infinite run starting from $\langle q_0, \mathbf{x}_0 \rangle$ such that the control state q_f is repeated infinitely often?
- Boundedness problem:
Input: PCS C and $\langle q_0, \mathbf{x}_0 \rangle$.
Question: is $\text{Reach}_C(\langle q_0, \mathbf{x}_0 \rangle)$ finite?

Subclasses of Presburger Counter Systems

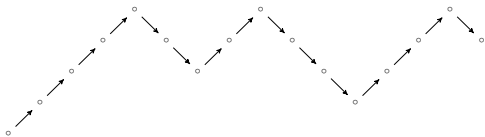
- **Counter systems (CS):** transitions $q \xrightarrow{\phi_g \wedge \phi_u} q' \in \delta$ s.t.
 - ϕ_g is a Boolean combination of atomic formulae of the form $t \leq t'$ built over x_1, \dots, x_n (ex: $2x_1 + x_2 = 3 \vee x_3 \geq 0$),
 - $\phi_u = \bigwedge_{i \in [1, n]} x'_i = x_i + \mathbf{b}(i)$ where $\mathbf{b} \in \mathbb{Z}^n$.
- Minsky machines are counter systems.
- **Vector addition systems with states (VASS):** all the transitions are of the form $q \xrightarrow{\top \wedge \phi_u} q'$.
(\approx Minsky machines without tests)

Decidability/Complexity Issues for VASS

- The reachability problem is decidable.
[Mayr, STOC'81; Kosaraju, STOC'82; Leroux, LICS'09]
 - No primitive recursive algorithm is known.
(use of well quasi-orderings)
 - EXPSPACE-hardness [Lipton, TR 76].
- Boundedness problem for VASS is EXPSPACE-complete.
[Lipton, TR 76; Rackoff, TCS 78]
- Checking equality between accessibility sets of two configurations is undecidable [Hack, TCS 76].

Reversal-Bounded Counter Systems

- Reversal: Alternation from nonincreasing mode to nondecreasing mode and vice-versa.



- Set \mathbb{T} : finite set of terms including $\{x_1, \dots, x_n\}$.
- Atomic formulae in guards are of the form $t \leq k$ or $t \geq k$ with $k \in \mathbb{Z}$ and t is of the form $\sum_i a_i x_i$ with the a_i 's in \mathbb{Z} .
- A run is **r - \mathbb{T} -reversal-bounded** whenever the number of reversals of each term in $\mathbb{T} \leq r$ times.

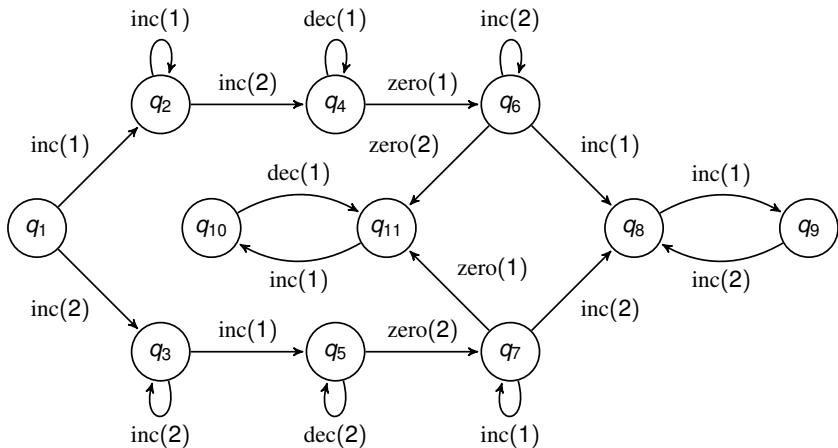
Reversal-Boundedness Leads to Semilinearity

- Given a CS C , $\mathbb{T}_C \stackrel{\text{def}}{=} \text{the set of terms } t \text{ occurring in } t \sim k$ with $\sim \in \{\leq, \geq\}$ + counters in $\{x_1, \dots, x_n\}$.
- $\langle C, \langle q_0, \mathbf{x}_0 \rangle \rangle$ is **reversal-bounded** $\stackrel{\text{def}}{\Leftrightarrow}$ there is $r \geq 0$ such that every run from $\langle q_0, \mathbf{x}_0 \rangle$ is $r\text{-}\mathbb{T}_C$ -reversal-bounded.
- When $\mathbb{T} = \{x_1, \dots, x_n\}$, \mathbb{T} -reversal-boundedness is equivalent to reversal-boundedness from [Ibarra, JACM 78].

Reversal-Boundedness Leads to Semilinearity

- Given a CS C , $\mathbb{T}_C \stackrel{\text{def}}{=} \text{the set of terms } t \text{ occurring in } t \sim k$ with $\sim \in \{\leq, \geq\}$ + counters in $\{x_1, \dots, x_n\}$.
- $\langle C, \langle q_0, \mathbf{x}_0 \rangle \rangle$ is **reversal-bounded** $\stackrel{\text{def}}{\Leftrightarrow}$ there is $r \geq 0$ such that every run from $\langle q_0, \mathbf{x}_0 \rangle$ is $r\text{-}\mathbb{T}_C$ -reversal-bounded.
- When $\mathbb{T} = \{x_1, \dots, x_n\}$, \mathbb{T} -reversal-boundedness is equivalent to reversal-boundedness from [Ibarra, JACM 78].
- Given a CS C , $r \geq 0$ and $q, q' \in Q$, one can effectively compute a Presburger formula $\phi_{q,q'}(\bar{x}, \bar{y})$ such that for all v , propositions below are equivalent:
 - $v \models \phi_{q,q'}(\bar{x}, \bar{y})$,
 - there is an $r\text{-}\mathbb{T}_C$ -reversal-bounded run from $\langle q, \langle v(x_1), \dots, v(x_n) \rangle \rangle$ to $\langle q', \langle v(y_1), \dots, v(y_n) \rangle \rangle$.

[Ibarra, JACM 78; Demri & Bersani, FRODOS'11]



$$\phi = (x_1 \geq 2 \wedge x_2 \geq 1 \wedge (x_2 + 1 \geq x_1)) \vee (x_2 \geq 2 \wedge x_1 \geq 1 \wedge x_1 + 1 \geq x_2)$$

$$\llbracket \phi \rrbracket = \{ \mathbf{y} \in \mathbb{N}^2 : \langle q_1, \mathbf{0} \rangle \xrightarrow{*} \langle q_9, \mathbf{y} \rangle \}$$

PCS with Octagonal Constraints

- **Octagonal constraint:** conjunction of atomic guards of the form

$$\pm y \pm z \leq k$$

where y, z are in $x_1, \dots, x_n, x'_1, \dots, x'_n$, $k \in \mathbb{Z}$ and $\pm y$ is either y or $-y$.

[Bozga & Girlea & Iosif, TACAS'09]

- **Difference bounds constraint:** conjunction of atomic guards of the form

$$y - z \leq k$$

where y, z are in $x_1, \dots, x_n, x'_1, \dots, x'_n$ and $k \in \mathbb{Z}$.

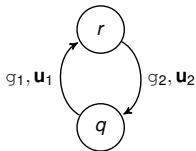
[Comon & Jurski, CAV'98]

- Guards on transitions in CS are Boolean combinations of linear constraints and therefore are incomparable with the above classes of guards.

Loops, Path Schemas and Flatness

Repeated Loop Effect

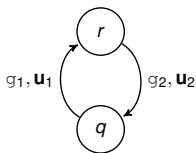
- **Path** p : finite sequence of transitions from δ corresponding to a path in its control graph.
- **Loop**: non-empty path \perp starting and ending by the same control state.



$$\perp = t_1 t_2$$

Repeated Loop Effect

- **Path** p : finite sequence of transitions from δ corresponding to a path in its control graph.
- **Loop**: non-empty path \perp starting and ending by the same control state.



$$\perp = t_1 t_2$$

- **Effect** $\text{effect}(\perp)$:

$$\{\langle \mathbf{x}, \mathbf{x}' \rangle \in \mathbb{N}^n \times \mathbb{N}^n : \langle \text{first}(\perp), \mathbf{x} \rangle \xrightarrow{\perp} \langle \text{last}(\perp), \mathbf{x}' \rangle\}$$

- **Repeated effect** $\text{effect}^{<\omega}(\perp)$ (a.k.a **acceleration**):

$$\{\langle \mathbf{x}, \mathbf{x}' \rangle \in \mathbb{N}^n \times \mathbb{N}^n : \langle \text{first}(\perp), \mathbf{x} \rangle \xrightarrow{\perp^i} \langle \text{last}(\perp), \mathbf{x}' \rangle, i \geq 0\}$$

Loop Effects and Presburger Arithmetic

- Reachability problem for loops:
 - Input:** PCS C , loop l , counter values $\mathbf{x}_0, \mathbf{x}_f$.
 - question:** Is $\langle \mathbf{x}_0, \mathbf{x}_f \rangle \in \text{effect}^{<\omega}(l)$?
- The repeated effect of loops from counter systems is not necessarily definable in Presburger arithmetic.
(take a loop whose effect is to multiply by 2 a counter)

Loop Effects and Presburger Arithmetic

- Reachability problem for loops:

Input: PCS C , loop l , counter values $\mathbf{x}_0, \mathbf{x}_f$.

question: Is $\langle \mathbf{x}_0, \mathbf{x}_f \rangle \in \text{effect}^{<\omega}(l)$?

- The repeated effect of loops from counter systems is not necessarily definable in Presburger arithmetic.
(take a loop whose effect is to multiply by 2 a counter)
- The reachability problem for loops for PCS is undecidable.
- The repeated effect of loops made of octagonal constraints is effectively definable in Presburger arithmetic.

[Comon & Jurski, CAV'98; Bozga & Girlea & Iosif, TACAS'09]

Counting Iteration

- **Counting iteration** of $R \subseteq \mathbb{N}^{2n}$: $R_{\text{CI}} \subseteq \mathbb{N}^n \times \mathbb{N} \times \mathbb{N}^n$ s.t.
 $\langle \mathbf{x}, i, \mathbf{y} \rangle \in R_{\text{CI}} \stackrel{\text{def}}{\iff} \mathbf{y}$ can be reached from \mathbf{x} in i steps.
- R has a Presburger counting iteration if its counting iteration is Presburger-definable.
- $\langle \mathbf{x}, \mathbf{y} \rangle \in R^*$ iff there is $i \in \mathbb{N}$ such that $\langle \mathbf{x}, i, \mathbf{y} \rangle \in R_{\text{CI}}$.

Counting Iteration

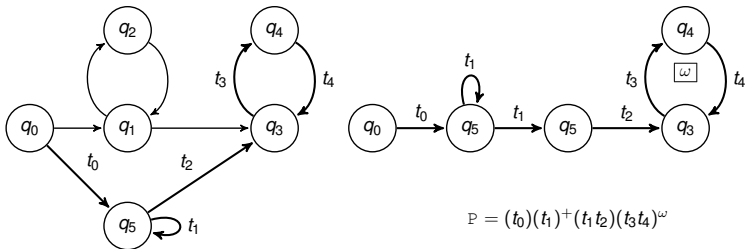
- **Counting iteration** of $R \subseteq \mathbb{N}^{2n}$: $R_{\text{CI}} \subseteq \mathbb{N}^n \times \mathbb{N} \times \mathbb{N}^n$ s.t. $\langle \mathbf{x}, i, \mathbf{y} \rangle \in R_{\text{CI}} \stackrel{\text{def}}{\iff} \mathbf{y}$ can be reached from \mathbf{x} in i steps.
- R has a Presburger counting iteration if its counting iteration is Presburger-definable.
- $\langle \mathbf{x}, \mathbf{y} \rangle \in R^*$ iff there is $i \in \mathbb{N}$ such that $\langle \mathbf{x}, i, \mathbf{y} \rangle \in R_{\text{CI}}$.
- $\{\langle \alpha, \alpha + 1 \rangle \in \mathbb{N}^2 : \alpha \in \mathbb{N}\}$ has a Presburger counting iteration.
- $R = \{\langle \alpha, 2\alpha \rangle \in \mathbb{N}^2 : \alpha \in \mathbb{N}\}$ has not.
($R^* = \{\langle \alpha, 2^\beta \alpha \rangle \in \mathbb{N}^2 : \alpha, \beta \in \mathbb{N}\}$)

Counting Iteration

- **Counting iteration** of $R \subseteq \mathbb{N}^{2n}$: $R_{\text{CI}} \subseteq \mathbb{N}^n \times \mathbb{N} \times \mathbb{N}^n$ s.t. $\langle \mathbf{x}, i, \mathbf{y} \rangle \in R_{\text{CI}} \stackrel{\text{def}}{\iff} \mathbf{y}$ can be reached from \mathbf{x} in i steps.
- R has a Presburger counting iteration if its counting iteration is Presburger-definable.
- $\langle \mathbf{x}, \mathbf{y} \rangle \in R^*$ iff there is $i \in \mathbb{N}$ such that $\langle \mathbf{x}, i, \mathbf{y} \rangle \in R_{\text{CI}}$.
- $\{\langle \alpha, \alpha + 1 \rangle \in \mathbb{N}^2 : \alpha \in \mathbb{N}\}$ has a Presburger counting iteration.
- $R = \{\langle \alpha, 2\alpha \rangle \in \mathbb{N}^2 : \alpha \in \mathbb{N}\}$ has not.
($R^* = \{\langle \alpha, 2^\beta \alpha \rangle \in \mathbb{N}^2 : \alpha, \beta \in \mathbb{N}\}$)
- A class of PCS **satisfies the property** (\star) when, for every loop l , $\text{effect}(l)$ has the Presburger counting iteration and its Presburger formula is computable.

From Loops to Path Schemas

- Infinitary path schema: alternation of non-loop and loop segments, ending by a loop, and representing a potentially infinite set of infinite runs.



- Infinitary path schema = ω -regular expression of the form $p_1 l_1^+ p_2 l_2^+ \dots p_k l_k^\omega$ over alphabet δ .
- Finitary path schema: $p_1 l_1^+ p_2 l_2^+ \dots p_k$ (no last loop).

Runs and Path Schemas

- Run ρ **respects** a path schema when its sequence of transitions belongs to the language of the path schema.
- **Good** path schemas are those that are *minimal* (no loop is multiple of smaller loops, no segment contains a loop, etc.).
- When (\star) holds, $\{\langle \mathbf{x}, \mathbf{x}' \rangle : \langle q, \mathbf{x} \rangle \xrightarrow{*} \langle q', \mathbf{x}' \rangle \text{ respects } \mathbb{P}\}$ is effectively Presburger-definable (\mathbb{P} is finitary).

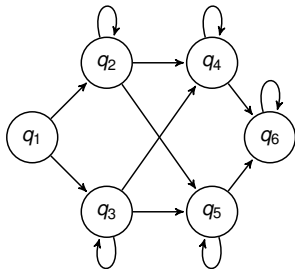
Runs and Path Schemas

- Run ρ **respects** a path schema when its sequence of transitions belongs to the language of the path schema.
- **Good** path schemas are those that are *minimal* (no loop is multiple of smaller loops, no segment contains a loop, etc.).
- When (\star) holds, $\{\langle \mathbf{x}, \mathbf{x}' \rangle : \langle q, \mathbf{x} \rangle \xrightarrow{*} \langle q', \mathbf{x}' \rangle \text{ respects } \mathbb{P}\}$ is effectively Presburger-definable (\mathbb{P} is finitary).
- The class of PCS with octagonal constraints enjoys (\star) .
See also the tool FLATA. [Bozga & Girlea & Iosif, TACAS'09]
- The class CS also enjoys (\star) .

Flat Presburger Counter Systems

- Every state belongs to at most one simple cycle.

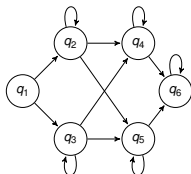
[Fribourg & Olsen, CONCUR'97; Comon & Jurski, CAV'98]



Flat Presburger Counter Systems

- Every state belongs to at most one simple cycle.

[Fribourg & Olsen, CONCUR'97; Comon & Jurski, CAV'98]

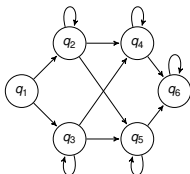


- In a flat counter system, the number of good infinitary path schemas is bounded by $\text{card}(\delta)^{(2 \times \text{card}(\delta))}$.
- Let \mathcal{C} be a class of PCS that enjoys (\star) . Then, for every flat PCS from \mathcal{C} , the relation $\text{Reach}_{\mathcal{C}}$ is Presburger-definable.
- Let \mathcal{C} be a class of PCS that enjoys (\star) . The reachability problem for \mathcal{C} is decidable.

Flat Presburger Counter Systems

- Every state belongs to at most one simple cycle.

[Fribourg & Olsen, CONCUR'97; Comon & Jurski, CAV'98]



- In a flat counter system, the number of good infinitary path schemas is bounded by $\text{card}(\delta)^{(2 \times \text{card}(\delta))}$.
- Let \mathcal{C} be a class of PCS that enjoys (\star) . Then, for every flat PCS from \mathcal{C} , the relation $\text{Reach}_{\mathcal{C}}$ is Presburger-definable.
- Let \mathcal{C} be a class of PCS that enjoys (\star) . The reachability problem for \mathcal{C} is decidable.
- The reachability problem for flat CS is NP-complete.

Flattable Systems

- Flat PCS are not always directly available.
- Relaxed version: reachability captured by a flat unfolding.
- $\langle C, \langle q_0, \mathbf{x}_0 \rangle \rangle$ is **initially flattable** iff there is a finite set of finitary path schemas generating the same configurations.
- Flattable PCS are everywhere. [Leroux & Sutre, ATVA'05]
 - Initialized reversal-bounded CS are initially flattable.
 - Initialized gainy counter automata are initially flattable.
 - Etc.

Flattable Systems

- Flat PCS are not always directly available.
- Relaxed version: reachability captured by a flat unfolding.
- $\langle C, \langle q_0, \mathbf{x}_0 \rangle \rangle$ is **initially flattable** iff there is a finite set of finitary path schemas generating the same configurations.
- Flattable PCS are everywhere. [Leroux & Sutre, ATVA'05]
 - Initialized reversal-bounded CS are initially flattable.
 - Initialized gainy counter automata are initially flattable.
 - Etc.
- Flat unfolding of a PCS provides less runs and it can be used as an underapproximation method.
- For VASS, Presburger-definable reachability set iff initially flattable. [Leroux, LICS'13]

We Want More: to Verify Temporal Properties

- Checking how configurations are temporally organized.
- Semilinearity of reachability sets no longer sufficient.
How to combine it with other proof techniques?
- Acceleration methods not designed to verify temporal properties.
- How to take advantage of advances in the development of SMT solvers and decision procedures for Presburger arithmetic?

Verifying Temporal Properties

Specifying Existence of Runs in Temporal Logic

- Repeated reachability can be obviously expressed by $G F q_f$.

Specifying Existence of Runs in Temporal Logic

- Repeated reachability can be obviously expressed by $G F q_f$.
- Initialized VASS is unbounded iff there is a run $\langle q, \mathbf{z} \rangle \xrightarrow{*} \langle q', \mathbf{y} \rangle \xrightarrow{*} \langle q', \mathbf{y}' \rangle$ with $\mathbf{y} \prec \mathbf{y}'$ for some q' .
- In temporal logic lingua:

$$\langle q, \mathbf{z} \rangle \models E \exists y_1, \dots, y_n F \left(\bigwedge_{i=1}^n x_i = y_i \wedge X F \left(\bigwedge_{i=1}^n x_i \geq y_i \wedge \bigvee_{i=1}^n x_i > y_i \right) \right)$$

Specifying Existence of Runs in Temporal Logic

- Repeated reachability can be obviously expressed by $G F q_f$.
- Initialized VASS is unbounded iff there is a run $\langle q, \mathbf{z} \rangle \xrightarrow{*} \langle q', \mathbf{y} \rangle \xrightarrow{*} \langle q', \mathbf{y}' \rangle$ with $\mathbf{y} \prec \mathbf{y}'$ for some q' .
- In temporal logic lingua:

$$\langle q, \mathbf{z} \rangle \models E \exists y_1, \dots, y_n F \left(\bigwedge_{i=1}^n x_i = y_i \wedge X F \left(\bigwedge_{i=1}^n x_i \geq y_i \wedge \bigvee_{i=1}^n x_i > y_i \right) \right)$$

- Linear-time temporal logics offer genericity and fragments can be easily designed.

Introduction to Presburger LTL

- First-order quantification over counter values, e.g.
 $\exists y G(x_1 \leq y)$. \approx “Along the run, counter 1 is bounded.”
- $\text{VAR}^P = \{y_1, y_2, \dots\}$: set of integer variables.
- $\text{VAR} = \{x_1, x_2, \dots\}$: set of counter variables.
- $\mathcal{Q} = \{q_1, q_2, \dots\}$: set of control state symbols.

Introduction to Presburger LTL

- First-order quantification over counter values, e.g.
 $\exists y G(x_1 \leq y)$. \approx “Along the run, counter 1 is bounded.”
- $\text{VAR}^P = \{y_1, y_2, \dots\}$: set of integer variables.
- $\text{VAR} = \{x_1, x_2, \dots\}$: set of counter variables.
- $\mathcal{Q} = \{q_1, q_2, \dots\}$: set of control state symbols.
- Formulae:

$$\phi ::= \psi \mid q \mid \phi \wedge \phi \mid \neg \phi \mid X\phi \mid \phi U \phi \mid \exists y \phi$$

- ψ : Boolean combination of linear constraints over $\text{VAR} \cup \text{VAR}^P$.
- $q \in \mathcal{Q}$.

Satisfaction Relation

- Environment \mathfrak{E} : partial map $\text{VAR}^P \rightarrow \mathbb{N}$.

$$Q \times \mathbb{N}^n \ni \rho = \langle q_0, \mathbf{x}_0 \rangle, \langle q_1, \mathbf{x}_1 \rangle \cdots \langle q_k, \mathbf{x}_k \rangle \cdots$$

- $\rho, i \models_{\mathfrak{E}} q \stackrel{\text{def}}{\Leftrightarrow} q = q_i$.
- $\rho, i \models_{\mathfrak{E}} \psi \stackrel{\text{def}}{\Leftrightarrow} v_i \models \psi$ with v_i extends \mathfrak{E} s.t. $v_i(x_j) = \mathbf{x}_i(j)$.
- $\rho, i \models_{\mathfrak{E}} X\phi \stackrel{\text{def}}{\Leftrightarrow} \rho, i+1 \models_{\mathfrak{E}} \phi$.
- $\rho, i \models_{\mathfrak{E}} \exists y \phi$ iff there is $k \in \mathbb{N}$ such that $\rho, i \models_{\mathfrak{E}[y \mapsto k]} \phi$.

Decision Problems for Presburger LTL

- Semi-closed formula: no variable from VAR^P is free.
 $F(x_1 = y)$ is not semi-closed unlike $G(x_1 > x_2)$ and $\exists y G(x_1 \leq y)$.

Decision Problems for Presburger LTL

- Semi-closed formula: no variable from VAR^P is free.
 $F(x_1 = y)$ is not semi-closed unlike $G(x_1 > x_2)$ and $\exists y G(x_1 \leq y)$.
- Satisfiability Problem
 - Input:** An Presburger LTL semi-closed formula ϕ with free counter variables x_1, \dots, x_n .
 - Question:** Is there a model $\rho \in (\mathbb{Q} \times \mathbb{N}^n)$ s.t. $\rho, 0 \models_{\emptyset} \phi$?
- Existential Model-Checking Problem
 - Input:** PCS $C = \langle Q, n, \delta \rangle, \langle q_0, \mathbf{x}_0 \rangle$ and semi-closed formula ϕ with free variables in $\{x_1, \dots, x_n\}$.
 - Question:** Is there an infinite run ρ starting at $\langle q_0, \mathbf{x}_0 \rangle$ such that $\rho, 0 \models_{\emptyset} \phi$?

(Infinite runs of PCS are Presburger LTL models)

Temporal Logics with Arithmetical Constraints

- Rich literature:
 - Constraints on the number of event occurrences.
[Bouajjani et al., LICS'95; Laroussinie et al., TIME'10]
 - Constraints on XML documents.
[Dal Zilio & Lugiez, RTA'03; Seidl et al., ICALP'04]
 - Temporal semantics of imperative programs.
[Manna & Pnueli, 1992]
- Program variable x never decreases below its initial value:

$$\exists y (x = y) \wedge G(x \geq y)$$

- Graded modal logics.
See e.g. <http://www.cs.man.ac.uk/~ezolin/ml/>

Temporal Logics with Arithmetical Constraints

- Rich literature:
 - Constraints on the number of event occurrences.
[Bouajjani et al., LICS'95; Laroussinie et al., TIME'10]
 - Constraints on XML documents.
[Dal Zilio & Lugiez, RTA'03; Seidl et al., ICALP'04]
 - Temporal semantics of imperative programs.
[Manna & Pnueli, 1992]
- Program variable x never decreases below its initial value:

$$\exists y (x = y) \wedge G(x \geq y)$$

- Graded modal logics.
See e.g. <http://www.cs.man.ac.uk/~ezolin/ml/>
- Model-checking restricted to $LTL(\mathbb{Q})$ is already undecidable.

A Fragment CLTL

- $\psi(X^{i_1}x_{j_1}, \dots, X^{i_k}x_{j_k})$ as a shortcut for
 $(\exists y_1, \dots, y_k X^{i_1}(y_1 = x_{j_1}) \wedge \dots \wedge X^{i_k}(y_k = x_{j_k}) \wedge \psi(y_1, \dots, y_k))$,
- $X^i x$ understood as the value of x at the i th next state.
- First counter remains constant: $G(x_1 = Xx_1)$.
- CLTL: fragment of Presburger LTL such that first-order quantification at the level of temporal formulae is restricted to formulae $\psi(X^{i_1}x_{j_1}, \dots, X^{i_k}x_{j_k})$.

Satisfiability Problems

- Fragment $\tilde{\mathcal{F}}_0 \ni \phi ::= x_i < x_j \mid x_i = x_j \mid x_i \leq k$.
- Satisfiability problem for CLTL($\tilde{\mathcal{F}}_0$) is PSPACE-complete.
(but not ω -regularity of symbolic models)

[Demri & D'Souza, I&C 07]

Satisfiability Problems

- Fragment $\mathfrak{F}_0 \ni \phi ::= x_i < x_j \mid x_i = x_j \mid x_i \leq k$.
- Satisfiability problem for CLTL(\mathfrak{F}_0) is PSPACE-complete.
(but not ω -regularity of symbolic models)

[Demri & D'Souza, I&C 07]

- Fragment $\mathfrak{F}_1 \ni \phi ::= x_i \sim x_j + d \mid x_i \sim d, d \in \mathbb{Z}$ and $\sim \in \{<, >, \leq, \geq, =\}$.
- Satisfiability problem for CLTL $_1^2(\mathfrak{F}_1)$ (1 var. + X-length=2) or for CLTL $_2^1(\mathfrak{F}_1)$ is undecidable.

See e.g. [Comon & Cortier, CSL'00; Demri & Gascon, TCS 09]

Satisfiability Problems

- Fragment $\mathfrak{F}_0 \ni \phi ::= x_i < x_j \mid x_i = x_j \mid x_i \leq k$.
- Satisfiability problem for $\text{CLTL}(\mathfrak{F}_0)$ is PSPACE-complete.
(but not ω -regularity of symbolic models)
[Demri & D'Souza, I&C 07]
- Fragment $\mathfrak{F}_1 \ni \phi ::= x_i \sim x_j + d \mid x_i \sim d, d \in \mathbb{Z}$ and $\sim \in \{<, >, \leq, \geq, =\}$.
- Satisfiability problem for $\text{CLTL}_1^2(\mathfrak{F}_1)$ (1 var. + X-length=2) or for $\text{CLTL}_2^1(\mathfrak{F}_1)$ is undecidable.
See e.g. [Comon & Cortier, CSL'00; Demri & Gascon, TCS 09]
- Satisfiability problem for $\text{CCTL}^*(\mathfrak{F}_0)$ (branching-time version of $\text{CLTL}(\mathfrak{F}_0)$) is decidable.
(use of weak MSO with bounding quantifier B)
[Carapelle & Kartzow & Lohrey, CONCUR'13]

EXPSPACE Upper Bound for VASS

- Control state repeated reachability problem restricted to VASS can be solved in exponential space.
- Model-checking problem restricted to $LTL(\mathcal{Q})$ and to VASS is EXPSPACE-complete.

[Habermehl, ICATPN 97]

- Decidability/undecidability results for linear-time temporal logic on Petri nets can be found in [Esparza, CAAP'94]; e.g., $LTL(\mathcal{Q}) + x_i = 0$ is undecidable.

What About Reversal-Bounded Counter Systems?

- Control state repeated reachability problem restricted to reversal-bounded counter systems is decidable.

See e.g.[Dang & Ibarra & San Pietro, FSTTCS'01]

- Problem RBMC:

Input: a CS C , $\langle q_0, \mathbf{x}_0 \rangle$, a CLTL formula ϕ , a bound $r \in \mathbb{N}$ (in binary),

Question: Is there an infinite run ρ from $\langle q_0, \mathbf{x}_0 \rangle$ such that $\rho, 0 \models \phi$ and ρ is r - \mathbb{T} -reversal-bounded with $\mathbb{T} = \mathbb{T}_C \cup \mathbb{T}_\phi$?

What About Reversal-Bounded Counter Systems?

- Control state repeated reachability problem restricted to reversal-bounded counter systems is decidable.

See e.g. [Dang & Ibarra & San Pietro, FSTTCS'01]

- Problem RBMC:

Input: a CS C , $\langle q_0, \mathbf{x}_0 \rangle$, a CLTL formula ϕ , a bound $r \in \mathbb{N}$ (in binary),

Question: Is there an infinite run ρ from $\langle q_0, \mathbf{x}_0 \rangle$ such that $\rho, 0 \models \phi$ and ρ is r -T-reversal-bounded with $T = T_C \cup T_\phi$?

- RBMC is NEXPTIME-complete.

[Howell & Rosier, JCSS 87]

[Bersani & Demri, FRODOS'11, Hague & Lin, CAV'11]

(Proof plan: RBMC \leq repeated reachability \leq reachability)

- Global model-checking is also possible for RBMC.

Flat CS and LTL with Arithmetical Constraints

[Demri & Dhar & Sangnier, IJCAR'12]

- Flat CS and arithmetical constraints \approx guards.

$$\phi ::= q \mid g \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \mathbf{X}\phi \mid \phi \mathbf{U}\phi \mid \mathbf{X}^{-1}\phi \mid \phi \mathbf{S}\phi$$

- Model-checking flat CS with this fragment is NP-complete.

Main Ingredients

- An algorithm in NP starts first by guessing a good infinitary path schema

$$p_1 l_2^+ p_3 l_4^+ \dots l_{k-1}^+ p_k l_k^\omega$$

Main Ingredients

- An algorithm in NP starts first by guessing a good infinitary path schema

$$p_1 l_2^+ p_3 l_4^+ \dots l_{k-1}^+ p_k l_k^\omega$$

- Ingredients of the proof aim at bounding the numbers of times loops are visited.

Main Ingredients

- An algorithm in NP starts first by guessing a good infinitary path schema

$$p_1 l_2^+ p_3 l_4^+ \dots l_{k-1}^+ p_k l_k^\omega$$

- Ingredients of the proof aim at bounding the numbers of times loops are visited.
 - 1 Eliminating disjunctions in guards.
... or how to flatten multiple loops with identical updates.

Main Ingredients

- An algorithm in NP starts first by guessing a good infinitary path schema

$$p_1 l_2^+ p_3 l_4^+ \dots l_{k-1}^+ p_k l_k^\omega$$

- Ingredients of the proof aim at bounding the numbers of times loops are visited.
 - 1 Eliminating disjunctions in guards.
... or how to flatten multiple loops with identical updates.
 - 2 To bound the numbers of times loops are visited when guards are conjunctions of linear constraints.
(loops may be visited an exponential number of times)

Main Ingredients

- An algorithm in NP starts first by guessing a good infinitary path schema

$$p_1 l_2^+ p_3 l_4^+ \dots l_{k-1}^+ p_k l_k^\omega$$

- Ingredients of the proof aim at bounding the numbers of times loops are visited.

- 1 Eliminating disjunctions in guards.
... or how to flatten multiple loops with identical updates.
- 2 To bound the numbers of times loops are visited when guards are conjunctions of linear constraints.
(loops may be visited an exponential number of times)
- 3 Stuttering Theorem for Past LTL

How to Internalize the Choices for Path Schemas

- How to deal with the nondeterminism related to the choice of the infinitary path schema using Boolean formulae?
- How to deal with the nondeterminism related to the unfolding of the path schemas for eliminating disjunctions?

How to Internalize the Choices for Path Schemas

- How to deal with the nondeterminism related to the choice of the infinitary path schema using Boolean formulae?
- How to deal with the nondeterminism related to the unfolding of the path schemas for eliminating disjunctions?
- How to deal with the nondeterminism related to the number of times loops are visited?
- May be related to how to find good accelerations?

See e.g. [Finkel & Leroux, FSTTCS'02; Gonnord, PhD 06]

Other Logics on Flat CS

- Model-checking flat counter systems with FO or linear μ -calculus is PSPACE-complete.
(arithmetical constraints are still allowed)

[Demri & Dhar & Sangnier, ICALP'13]

Other Logics on Flat CS

- Model-checking flat counter systems with FO or linear μ -calculus is PSPACE-complete.
(arithmetical constraints are still allowed)

[Demri & Dhar & Sangnier, ICALP'13]

- Model-checking flat CS with Presburger CTL* is decidable.

[Demri & Finkel & Goranko & van Drimmelen, JANCL 10]

- By reduction into Presburger arithmetic: runs respecting a path schema are encoded as tuples of natural numbers by counting how many times loops are visited.

Other Logics on Flat CS

- Model-checking flat counter systems with FO or linear μ -calculus is PSPACE-complete.
(arithmetical constraints are still allowed)

[Demri & Dhar & Sangnier, ICALP'13]

- Model-checking flat CS with Presburger CTL* is decidable.
[Demri & Finkel & Goranko & van Drimmelen, JANCL 10]
- By reduction into Presburger arithmetic: runs respecting a path schema are encoded as tuples of natural numbers by counting how many times loops are visited.
- *Open question*: decidability status of model-checking flat CS beyond CTL*.

Path Schema Subsumption: An Overview

Why Path Schema Enumeration?

- A finite set of path schemas is a simple way to represent a (potentially) infinite set of runs.
- Enumerating path schemas as a way to underapproximate the set of runs (bounded model-checking).

Why Path Schema Enumeration?

- A finite set of path schemas is a simple way to represent a (potentially) infinite set of runs.
- Enumerating path schemas as a way to underapproximate the set of runs (bounded model-checking).
- How to generate path schemas in a structured and controlled fashion?
- How to find a finite set of path schemas that fully captures the behavior of a PCS, if possible?

Why Path Schema Enumeration?

- A finite set of path schemas is a simple way to represent a (potentially) infinite set of runs.
- Enumerating path schemas as a way to underapproximate the set of runs (bounded model-checking).
- How to generate path schemas in a structured and controlled fashion?
- How to find a finite set of path schemas that fully captures the behavior of a PCS, if possible?
- Strategy in which we have a clear way of detecting whether we have enumerated sufficiently many path schemas.

Consistency and Subsumption

- Finitary path schemas P_1, \dots, P_α, P . All the path schemas start and end by the same control states.
- **Consistency** of P wrt the initial condition $\phi_{\text{init}}(y_1, \dots, y_n)$:

$$\exists x_1, \dots, x_n \exists x'_1, \dots, x'_n \phi_{\text{init}}(x_1, \dots, x_n) \wedge \\ \varphi_P(x_1, \dots, x_n, x'_1, \dots, x'_n)$$

- Existence of formula φ_P guaranteed by property (\star).
- For the class of CS, the consistency problem is NP-complete.

Consistency and Subsumption

- Finitary path schemas P_1, \dots, P_α, P . All the path schemas start and end by the same control states.
- **Consistency** of P wrt the initial condition $\phi_{\text{init}}(\mathbf{y}_1, \dots, \mathbf{y}_n)$:

$$\exists \mathbf{x}_1, \dots, \mathbf{x}_n \exists \mathbf{x}'_1, \dots, \mathbf{x}'_n \phi_{\text{init}}(\mathbf{x}_1, \dots, \mathbf{x}_n) \wedge \\ \varphi_P(\mathbf{x}_1, \dots, \mathbf{x}_n, \mathbf{x}'_1, \dots, \mathbf{x}'_n)$$

- Existence of formula φ_P guaranteed by property (\star).
- For the class of CS, the consistency problem is NP-complete.
- $\{P_1, \dots, P_\alpha\}$ **subsumes** P wrt $\phi_{\text{init}}(\mathbf{y}_1, \dots, \mathbf{y}_n)$:

$$\forall \mathbf{x}_1, \dots, \mathbf{x}_n \forall \mathbf{x}'_1, \dots, \mathbf{x}'_n (\phi_{\text{init}}(\mathbf{x}_1, \dots, \mathbf{x}_n) \wedge \varphi_P(\mathbf{x}_1, \dots, \mathbf{x}_n, \mathbf{x}'_1, \dots, \mathbf{x}'_n))$$

$$\Rightarrow \bigvee_{i \in [1, \alpha]} \exists \mathbf{z}_1, \dots, \mathbf{z}_n \phi_{\text{init}}(\mathbf{z}_1, \dots, \mathbf{z}_n) \wedge \varphi_{P_i}(\mathbf{z}_1, \dots, \mathbf{z}_n, \mathbf{x}'_1, \dots, \mathbf{x}'_n)$$

General Subsumption (one step beyond reachability)

- $[P]_{\phi_{\text{init}}} \stackrel{\text{def}}{=} \{ \mathbf{x}_f : \langle q_0, \mathbf{x}_0 \rangle \xrightarrow{*} \langle q_f, \mathbf{x}_f \rangle \text{ respects } P \text{ and } \mathbf{x}_0 \models \phi_{\text{init}} \}$.

General Subsumption (one step beyond reachability)

- $[P]_{\phi_{\text{init}}} \stackrel{\text{def}}{=} \{ \mathbf{x}_f : \langle q_0, \mathbf{x}_0 \rangle \xrightarrow{*} \langle q_f, \mathbf{x}_f \rangle \text{ respects } P \text{ and } \mathbf{x}_0 \models \phi_{\text{init}} \}$.
- **Pattern** ϕ_{pat} : formula in Presburger LTL without FO quantification and with free occurrences of y_1, \dots, y_α .

$$[P]_{\phi_{\text{pat}}, \phi_{\text{init}}} \stackrel{\text{def}}{=} \{ \mathfrak{E} : \rho = \langle q_0, \mathbf{x}_0 \rangle \xrightarrow{*} \langle q_f, \mathbf{x}_f \rangle \text{ respects } P, \\ \mathbf{x}_0 \models \phi_{\text{init}} \ \& \ \rho, \mathbf{0} \models_{\mathfrak{E}} \phi_{\text{pat}} \}$$

- $[P]_{\phi_{\text{init}}}$ above corresponds to $[P]_{\phi_{\text{pat}}, \phi_{\text{init}}}$ with

$$\phi_{\text{pat}} \stackrel{\text{def}}{=} F(\mathbf{x}_1 = y_1 \wedge \dots \wedge \mathbf{x}_n = y_n \wedge \neg \mathbf{X}T)$$

General Subsumption (one step beyond reachability)

- **Pattern** ϕ_{pat} : formula in Presburger LTL without FO quantification and with free occurrences of y_1, \dots, y_α .

$$[P]_{\phi_{\text{pat}}, \phi_{\text{init}}} \stackrel{\text{def}}{=} \{ \mathcal{E} : \rho = \langle q_0, \mathbf{x}_0 \rangle \xrightarrow{*} \langle q_f, \mathbf{x}_f \rangle \text{ respects } P, \\ \mathbf{x}_0 \models \phi_{\text{init}} \ \& \ \rho, 0 \models_{\mathcal{E}} \phi_{\text{pat}} \}$$

- $[P]_{\phi_{\text{init}}}$ above corresponds to $[P]_{\phi_{\text{pat}}, \phi_{\text{init}}}$ with

$$\phi_{\text{pat}} \stackrel{\text{def}}{=} F(\mathbf{x}_1 = \mathbf{y}_1 \wedge \dots \wedge \mathbf{x}_n = \mathbf{y}_n \wedge \neg \mathbf{X}\top)$$

- $\{P_1, \dots, P_\alpha\}$ **subsumes** P wrt $\phi_{\text{init}}(\mathbf{y}_1, \dots, \mathbf{y}_n)$ and the **pattern** $\phi_{\text{pat}} \stackrel{\text{def}}{\Leftrightarrow} [P]_{\phi_{\text{pat}}, \phi_{\text{init}}} \subseteq [P_1]_{\phi_{\text{pat}}, \phi_{\text{init}}} \cup \dots \cup [P_\alpha]_{\phi_{\text{pat}}, \phi_{\text{init}}}$.
- For any class of PCS satisfying (\star) , there is a reduction from the generalized path schema subsumption problem to the validity problem for (PA).

How to Deal with Quantifiers

- Subsumption tests contain quantifiers.
- Most well-known Satisfiability Modulo Theories (SMT) solvers deal with quantifier-free formulae.
- Dealing with quantifiers is usually a difficult task for SMT solvers.
- Examples of techniques to deal with quantifiers:
 - Quantifier elimination. See e.g. [Reddy & Loveland, STOC'78]
 - Heuristic instantiation. See e.g. [Reynolds et al, CADE'13]
 - Lazy approach to quantifier elimination. [Monniaux, CAV'10]
- Challenge: How to use SMT solvers to enumerate path schemas and to perform subsumption?

A Few Words about the Enumeration Algorithm

- Paper contains a sketch of the algorithm for enumerating path schemas.
- Subsumption tests parameterized by patterns and cycles are generated with cycle schemas.

A Few Words about the Enumeration Algorithm

- Paper contains a sketch of the algorithm for enumerating path schemas.
- Subsumption tests parameterized by patterns and cycles are generated with cycle schemas.
- Generation of path schemas without arithmetical constraints is complete, stratified and takes advantage of the generation of cycle schemas.
- With subsumption on counter values, a complete version of the algorithm can be obtained if cycles are generated independently of cycle schemas.

Concluding Remarks

- Verification of temporal properties on PCS in its infancy.
- Need for methods to deal with (full) Presburger arithmetic or for proof systems dealing with model-checking.
- How to take advantage of recent developments on SAT/SMT solvers to deal with nondeterminism, quantified formulae etc.?

Concluding Remarks

- Verification of temporal properties on PCS in its infancy.
- Need for methods to deal with (full) Presburger arithmetic or for proof systems dealing with model-checking.
- How to take advantage of recent developments on SAT/SMT solvers to deal with nondeterminism, quantified formulae etc.?
- Other related trends include SMT solvers for model-checking infinite-state systems, branching VASS, complexity for VASS, relationships between CS and data logics, etc.

IJCAR'14

<http://vs12014.at/ijcar/>

- 7th International Joint Conference on Automated Reasoning, Vienna, Austria.
- Dates
 - Submission **January 15th 2014**
 - Notification **March 31st, 2014**
 - Conference **July 19th to July 22nd 2014**
 - Affiliated workshops **July 17, 18, 23, 24**