The Well Structured Problem for Presburger Counter Machines

Alain Finkel[1] Ekanshdeep Gupta[2]

[1]LSV, ENS Paris-Saclay, CNRS, Université Paris-Saclay, IUF, France

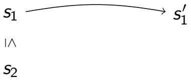
[2]Chennai Mathematical Institute, Chennai, India

FSTTCS'2019, Mumbai
13th December 2019

Well Structured Transition Systems [Finkel 87]

$(X, \to, \leq)$ is a well structured transition system iff

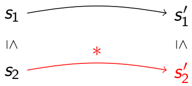
- $(X, \leq)$ is a well quasi order.
- $(X, \to, \leq)$ is monotone.
  - if there is a transition $s_1 \to s_1'$ and $s_2 \geq s_1$:



Well Structured Transition Systems [Finkel 87]

$(X, \rightarrow, \leq)$ is a well structured transition system iff

- $(X, \leq)$ is a well quasi order.
- $(X, \rightarrow, \leq)$ is monotone.
  - if there is a transition $s_1 \rightarrow s_1'$ and $s_2 \geq s_1$:

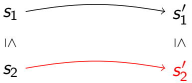


  - then there must exist $s_2' \geq s_1'$ such that $s_2 \xrightarrow{*} s_2'$.

Well Structured Transition Systems [Finkel 87]

$(X, \rightarrow, \leq)$ is a strong well structured transition system iff

- $(X, \leq)$ is a well quasi order.
- $(X, \rightarrow, \leq)$ is strongly monotone.
  - if there is a transition $s_1 \rightarrow s_1'$ and $s_2 \geq s_1$:



  - then there must exist $s_2' \geq s_1'$ such that $s_2 \rightarrow s_2'$.

Well Structured Transition Systems

Examples:

- Petri Nets
- Vector addition systems with states (VASS)
- VASS with reset/transfer/affine-ω extensions
- Finite state automata
- Lossy fifo systems and variants with time, data and priority
- ...

(Finkel, Schnoebelen, 2001)

Coverability

$(X, \to, \leq)$ an ordered transition system. $x, y \in X$.
y is coverable from x if $\exists x' \mid x \xrightarrow{*} x' \geq y$

Coverability

$(X, \rightarrow, \leq)$ an ordered transition system. $x, y \in X$.
y is coverable from x if $\exists x' \mid x \xrightarrow{*} x' \geq y$

Theorem

Coverability is decidable for WSTS.

Coverability

$(X, \rightarrow, \leq)$ an ordered transition system. $x, y \in X$.
y is coverable from x if $\exists x' \mid x \xrightarrow{*} x' \geq y$

Theorem

Coverability is decidable for WSTS.

- backward algorithm on upward closed sets.
  (Abdulla, Cerans et al., 1996)
- forward algorithm on downward closed sets.
  (Blondin, Finkel et al., 2017)

Motivation

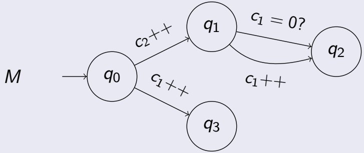
Minsky Machines

- not WSTS in general.
- coverability undecidable.

Motivation

Minsky Machines

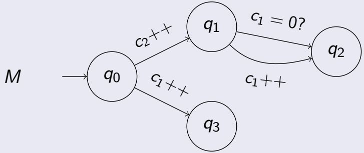
- not WSTS in general.
- coverability undecidable.



Motivation

Minsky Machines

- not WSTS in general.
- coverability undecidable.



Observe that M is (strongly) monotone.

Let us consider new problems about WSTS:

Well Structured Problems

Given an ordered transition system, is it a WSTS under:

- monotony (WSP)

Let us consider new problems about WSTS:

Well Structured Problems

Given an ordered transition system, is it a WSTS under:

- monotony (WSP)
- strong monotony (strong WSP)

Presburger Arithmetic

First order formulae over $(\mathbb{N}, +)$.

Presburger Arithmetic

First order formulae over $(\mathbb{N}, +)$.

- $lessthan(x, y) : \exists z(x + z = y)$
- $divby3(x) : \exists y(x = y + y + y)$
- $\phi(x, y) : y = x + x + x + 3$

Presburger Arithmetic

First order formulae over $(\mathbb{N}, +)$.

- $lessthan(x, y) : \exists z(x + z = y)$
- $divby3(x) : \exists y(x = y + y + y)$
- $\phi(x, y) : y = x + x + x + 3$

Theorem (Presburger, 1929)

Presburger arithmetic is decidable.

Presburger Counter Machines

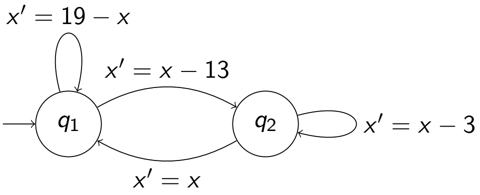
d-dimensional PCM $M = (Q, \rightarrow, \Phi)$:

- d counters.
- Q : set of control-states.
- Φ: set of Presburger formulae having $2d$ free variables.
- $\rightarrow \subseteq Q \times \Phi \times Q$.

Presburger Counter Machines

d-dimensional PCM $M = (Q, \rightarrow, \Phi)$:

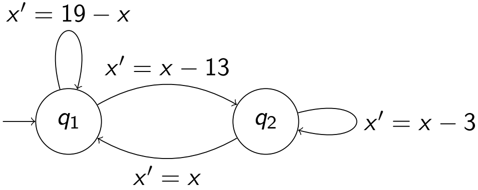
- d counters.
- Q : set of control-states.
- Φ: set of Presburger formulae having $2d$ free variables.
- $\rightarrow \subseteq Q \times \Phi \times Q$.



Presburger Counter Machines

d-dimensional PCM $M = (Q, \rightarrow, \Phi)$:

- d counters.
- Q : set of control-states.
- Φ: set of Presburger formulae having $2d$ free variables.
- $\rightarrow \subseteq Q \times \Phi \times Q$.



Sample run:
$(q_1, 0) \rightarrow (q_1, 19) \rightarrow (q_2, 6) \rightarrow (q_2, 3) \rightarrow (q_1, 3)$

Presburger Counter Machines

$M = (Q, \rightarrow, \Phi)$ a d dimensional PCM.

Presburger Counter Machines

$M = (Q, \rightarrow, \Phi)$ a d dimensional PCM.

- Functional PCMs: all formulae in Φ are functional.

Presburger Counter Machines

$M = (Q, \rightarrow, \Phi)$ a d dimensional PCM.

- Functional PCMs: all formulae in Φ are functional.
- Minsky machines: either translations or zero-tests.

Presburger Counter Machines

$M = (Q, \rightarrow, \Phi)$ a d dimensional PCM.

- Functional PCMs: all formulae in Φ are functional.
- Minsky machines: either translations or zero-tests.
- Affine VASS (AVASS): affine functions of the form $\vec{x'} = A\vec{x} + b$ with $A \in \mathbb{Z}^{d \times d}$, $b \in \mathbb{Z}^d$ with maximal domain.

Presburger Counter Machines

$M = (Q, \rightarrow, \Phi)$ a d dimensional PCM.

- Functional PCMs: all formulae in Φ are functional.
- Minsky machines: either translations or zero-tests.
- Affine VASS (AVASS): affine functions of the form $\vec{x'} = A\vec{x} + b$ with $A \in \mathbb{Z}^{d \times d}$, $b \in \mathbb{Z}^d$ with maximal domain.
- Positive AVASS (self-modified nets): AVASS where $A \in \mathbb{N}^{d \times d}$.

Presburger Counter Machines

$M = (Q, \rightarrow, \Phi)$ a d dimensional PCM.

- Functional PCMs: all formulae in Φ are functional.
- Minsky machines: either translations or zero-tests.
- Affine VASS (AVASS): affine functions of the form $\vec{x'} = A\vec{x} + b$ with $A \in \mathbb{Z}^{d\times d}$, $b \in \mathbb{Z}^d$ with maximal domain.
- Positive AVASS (self-modified nets): AVASS where $A \in \mathbb{N}^{d\times d}$.
- Totally Positive AVASS: Positive AVASS where $b \in \mathbb{N}^d$.

Well Structured Problem

Our Results:

	Well Structured Problem	Strong Well Structured Problem
PCM	U	D
Functional 1-PCM	U	D
2-AVASS	U	D
2-Minsky machines	U	D
1-AVASS	D	D

Well Structured Problem

Our Results:

	Well Structured Problem	Strong Well Structured Problem
PCM	U	D
Functional 1-PCM	U	D
2-AVASS	U	D
2-Minsky machines	U	D
1-AVASS	D	D

strong well-structuredness is Presburger expressible

Undecidability of WSP for 1-PCM

- Reduction from Minsky machine reachability.
- Given Minsky machine $M = (Q, \rightarrow, q_0)$, convert to 1-PCM N such that:

all states reachable in $M \iff N$ is a WSTS.

Undecidability of WSP for 1-PCM

- Reduction from Minsky machine reachability.
- Given Minsky machine $M = (Q, \rightarrow, q_0)$, convert to 1-PCM N such that:

$$\text{all states reachable in } M \iff N \text{ is a WSTS.}$$

Encode Minsky machine counters via Gödel encoding:

$$(m, n) \rightarrow 2^m 3^n c \text{ where } 2, 3 \text{ do not divide } c$$

Undecidability of WSP for 1-PCM

- Reduction from Minsky machine reachability.
- Given Minsky machine $M = (Q, \to, q_0)$, convert to 1-PCM N such that:

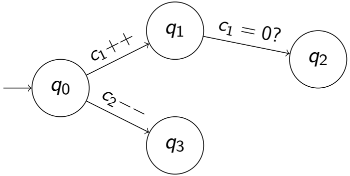
$$\text{all states reachable in } M \iff N \text{ is a WSTS.}$$

Encode Minsky machine counters via Gödel encoding:

$$(m, n) \to 2^m 3^n c \text{ where } 2, 3 \text{ do not divide } c$$

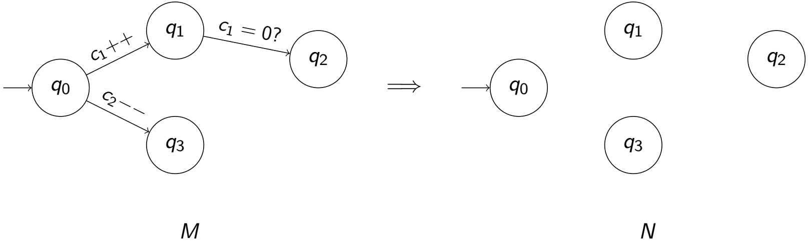
Consider (q, n) and (q, m) equivalent if $(\nu_2(n) = \nu_2(m)) \wedge (\nu_3(n) = \nu_3(m))$.

Given $M = (Q, \rightarrow, q_0)$:

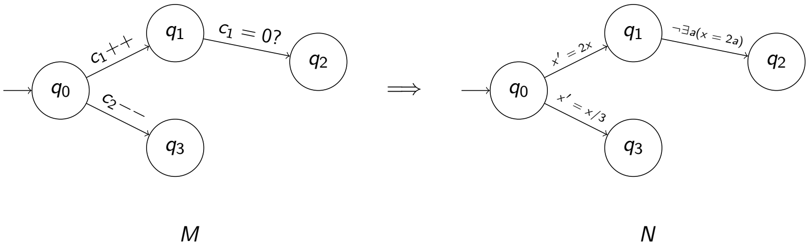


M

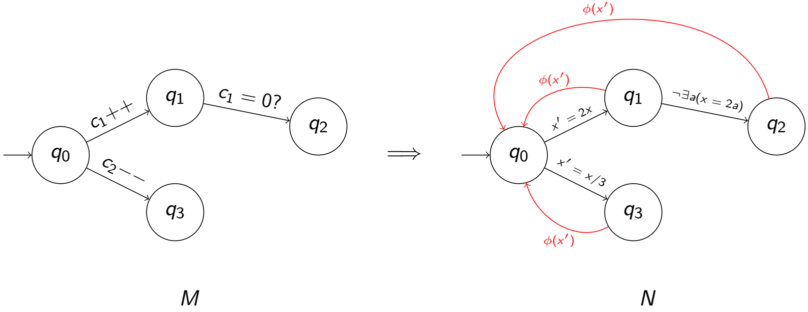
Given $M = (Q, \to, q_0)$:



Given $M = (Q, \rightarrow, q_0)$:



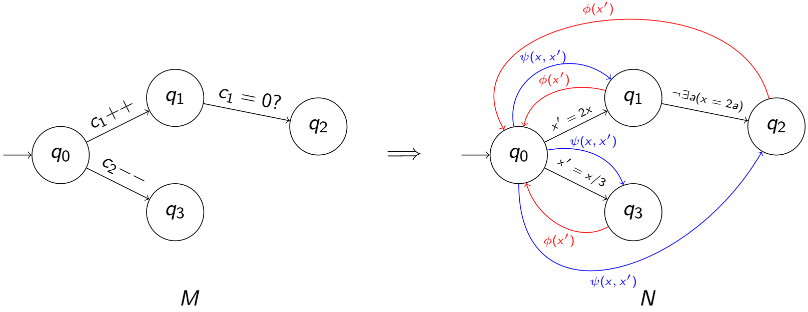
Given $M = (Q, \rightarrow, q_0)$:



$\phi(x') = (2 \nmid x') \wedge (3 \nmid x')$

all states reachable in $M \implies N$ is a WSTS.

Given $M = (Q, \rightarrow, q_0)$:



$\phi(x') = (2 \nmid x') \wedge (3 \nmid x')$
$\psi(x, x') = (x' = x = 0)$

all states reachable in $M \implies N$ is a WSTS.
all states reachable in $M \impliedby N$ is a WSTS.

Well Structured Problem

Theorem

WSP is undecidable for:

- *Functional* 1*-dim PCMs.*
- *2 counter Minsky machines.*
- *2 AVASS.*

Well Structured Problem

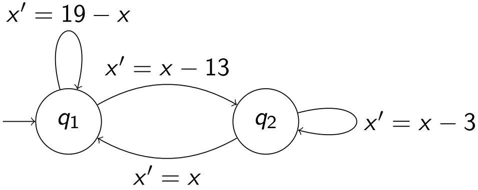
	Well Structured Problem	Strong Well Structured Problem
PCM	U	D
Functional 1-PCM	U	D
2-AVASS	U	D
2-Minsky machines	U	D
1-AVASS	D	D

1-AVASS

$M = (Q, \Phi, \rightarrow)$: 1 counter, transitions $x' = ax + b$, $a, b \in \mathbb{Z}$.

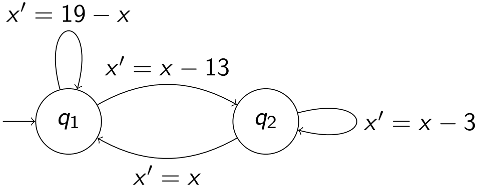
1-AVASS

$M = (Q, \Phi, \rightarrow)$: 1 counter, transitions $x' = ax + b$, $a, b \in \mathbb{Z}$.



1-AVASS

$M = (Q, \Phi, \rightarrow)$: 1 counter, transitions $x' = ax + b$, $a, b \in \mathbb{Z}$.



1-AVASS have the following properties:

- Reachability and coverability are decidable.
  (Finkel, Goller, Hasse, 2013)
- $Pre^*(q, n)$ is computable.
- WSP is decidable.

Pre^* algorithm

Given 1-AVASS $M = (Q, \rightarrow)$, configuration (q_f, n_f). Compute $Pre^*(q_f, n_f)$:

Pre^* algorithm

Given 1-AVASS $M = (Q, \rightarrow)$, configuration (q_f, n_f). Compute $Pre^*(q_f, n_f)$:

Key observations:

Pre^* algorithm

Given 1-AVASS $M = (Q, \rightarrow)$, configuration (q_f, n_f). Compute $Pre^*(q_f, n_f)$:

Key observations:

- $Pre^*(q_f, n_f)$ Presburger-expressible. So storing Presburger formulae sufficient.

Pre^* algorithm

Given 1-AVASS $M = (Q, \rightarrow)$, configuration (q_f, n_f). Compute $Pre^*(q_f, n_f)$:

Key observations:

- $Pre^*(q_f, n_f)$ Presburger-expressible. So storing Presburger formulae sufficient.
- Can compute all back-accelerations of cycles.

Pre^* algorithm

Given 1-AVASS $M = (Q, \rightarrow)$, configuration (q_f, n_f). Compute $Pre^*(q_f, n_f)$:

Key observations:

- $Pre^*(q_f, n_f)$ Presburger-expressible. So storing Presburger formulae sufficient.
- Can compute all back-accelerations of cycles.
- Computing back-accelerations of simple cycles sufficient - ie 1-AVASS are flattable.

Pre^* algorithm

Given 1-AVASS $M = (Q, \rightarrow)$, configuration (q_f, n_f). Compute $Pre^*(q_f, n_f)$:

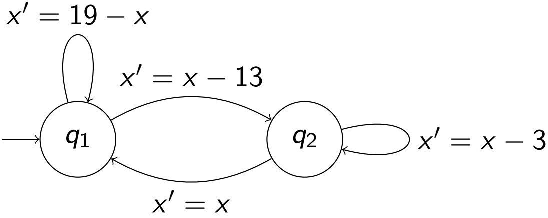
Key observations:

- $Pre^*(q_f, n_f)$ Presburger-expressible. So storing Presburger formulae sufficient.
- Can compute all back-accelerations of cycles.
- Computing back-accelerations of simple cycles sufficient - ie 1-AVASS are flattable.

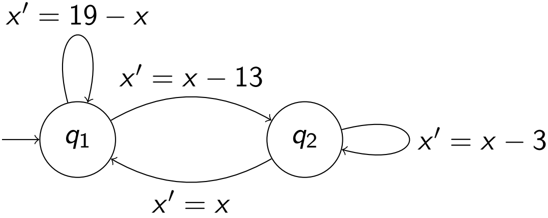
Idea:

- Maintain Presburger formula for each state q storing $\{n : (q, n) \in Pre^*(q_f, n_f)\}$.
- Iteratively backtrack from each transition and back-accelerate simple cycles.

Example: Compute $Pre^*(q_1, 19)$:



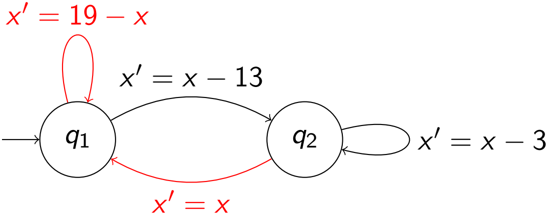
Example: Compute $Pre^*(q_1, 19)$:



ϕ_1:$n = 19$

ϕ_2:$\perp$

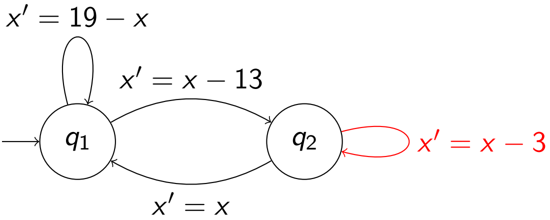
Example: Compute $Pre^*(q_1, 19)$:



ϕ_1:$n \in \{19, 0\}$

ϕ_2:$n \in \{19, 0\}$

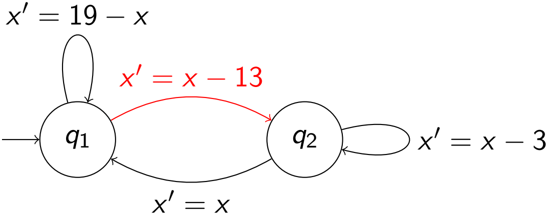
Example: Compute $Pre^*(q_1, 19)$:



ϕ_1:$n \in \{19, 0\}$

ϕ_2:$(n \geq 19 \wedge n =_3 1) \vee (n =_3 0)$

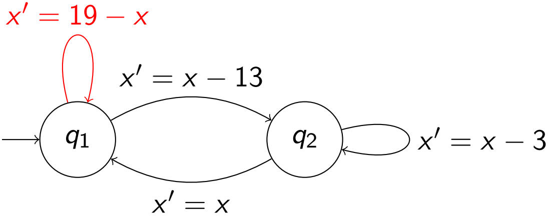
Example: Compute $Pre^*(q_1, 19)$:



ϕ_1:$n \in \{19, 0\} \vee$
$(n \geq 32 \wedge n =_3 2) \vee$
$(n \geq 13 \wedge n =_3 1)$

ϕ_2:$(n \geq 19 \wedge n =_3 1) \vee$
$(n =_3 0)$

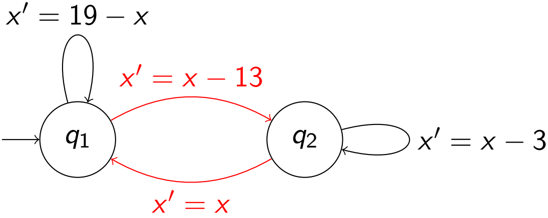
Example: Compute $Pre^*(q_1, 19)$:



ϕ_1: $n \in \{0, 3, 6, 19\} \vee$
$(n \geq 32 \wedge n =_3 2) \vee$
$(n \geq 13 \wedge n =_3 1)$

ϕ_2: $(n \geq 19 \wedge n =_3 1) \vee$
$(n =_3 0) \vee$
$(n \geq 32 \wedge n =_3 2)$

Example: Compute $Pre^*(q_1, 19)$:



ϕ_1:$n \in \{0, 3, 6, 19\} \vee$
$(n \geq 32 \wedge n =_3 2) \vee$
$(n \geq 13 \wedge n =_3 1) \vee$
$(n \geq 45 \wedge n =_3 0)$

ϕ_2:$(n \geq 19 \wedge n =_3 1) \vee$
$(n =_3 0) \vee$
$(n \geq 32 \wedge n =_3 2)$

Well Structured Problem

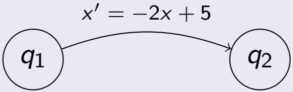
Theorem: WSP is decidable for 1-AVASS.

Well Structured Problem

Theorem: WSP is decidable for 1-AVASS.

Positive transitions are monotone.

For negative transitions:



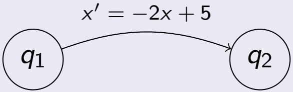
$(q_1, 0) \rightarrow (q_2, 5)$. So for WSTS, $(q_1, n) \xrightarrow{*} (q_2, m \geq 5)$.

Well Structured Problem

Theorem: WSP is decidable for 1-AVASS.

Positive transitions are monotone.

For negative transitions:



$(q_1, 0) \rightarrow (q_2, 5)$. So for WSTS, $(q_1, n) \xrightarrow{*} (q_2, m \geq 5)$.

- M is a WSTS, iff for all negative transitions $(q_1, (x' = ax + b), q_2)$, the set $\{q_1\} \times \mathbb{N}$ is a subset of $Pre^*(\uparrow(q_2, b))$.

Summary

	Well Structured Problem	Strong Well Structured Problem
PCM	U	D
Functional 1-PCM	U	D
2-AVASS	U	D
2-Minsky machines	U	D
1-AVASS	D	D

Further work

- Complexity analysis of the WSP on 1-AVASS.
- Complexity of the computation of Pre^* for 1-AVASS.
- Solve WSP for other models like pushdown counter machines, fifo automata, Petri net extensions.

Thank you!