Logiques et modèles pour la vérification de systèmes infinis

Florent Bouchy

LSV, ENS Cachan, CNRS

Soutenance de thèse de doctorat
10 novembre 2009
Cachan, France

Logics and Models for Verifying Infinite-state Systems

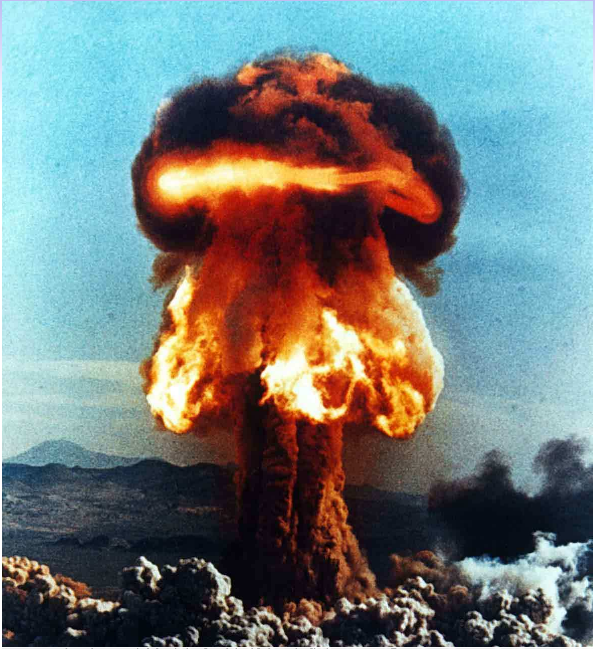
Florent Bouchy

LSV, ENS Cachan, CNRS

PhD defense
November 10^{th}, 2009
Cachan, France

Context

Context

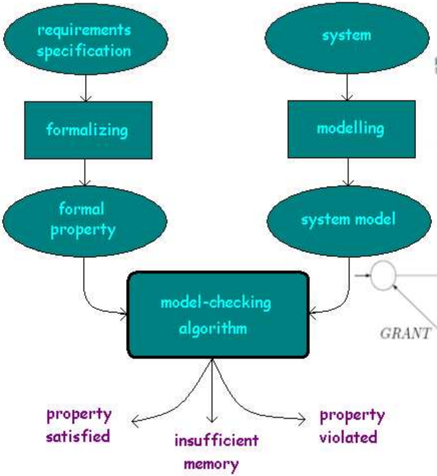


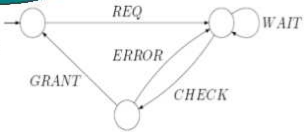
Model Checking

“Does a system achieve exactly what we want it to?”

"each request is eventually granted"

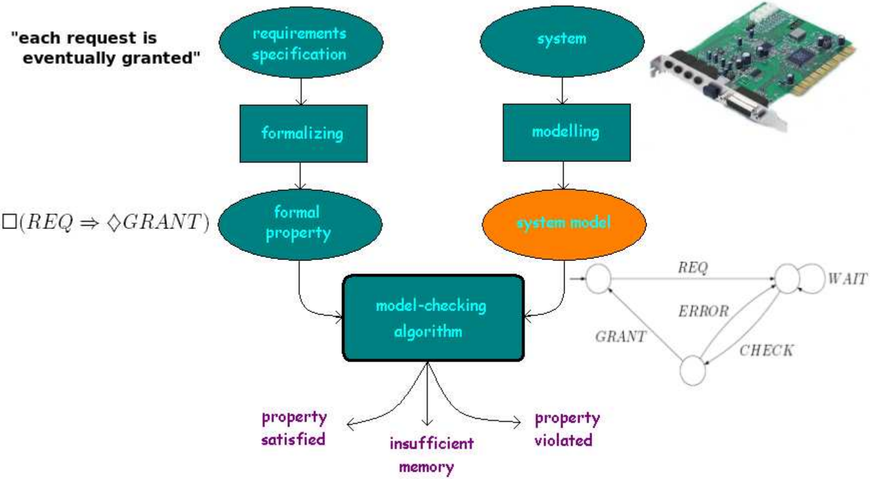
$\Box(REQ \Rightarrow \Diamond GRANT)$





Model Checking

"Does a system achieve exactly what we want it to?"



Modelling Systems with Automata

Capturing the behaviour of a system

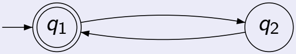
- Control state of the automaton = general "state" of the system
  (e.g. idle, printing, moving, waiting, etc.)
- Transition of the automaton = action/event of the system
  (e.g. message received, data computed, acknowledgment sent, etc.)

Modelling Systems with Automata

Capturing the behaviour of a system

- Control state of the automaton = general "state" of the system *(e.g. idle, printing, moving, waiting, etc.)*
- Transition of the automaton = action/event of the system *(e.g. message received, data computed, acknowledgment sent, etc.)*

Example: an automaton

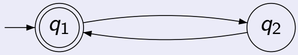


Modelling Systems with Automata

Capturing the behaviour of a system

- Control state of the automaton = general "state" of the system
  (e.g. idle, printing, moving, waiting, etc.)
- Transition of the automaton = action/event of the system
  (e.g. message received, data computed, acknowledgment sent, etc.)

Example: an automaton



Need for richer models, closer to reality

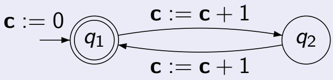
Extend automata with:
stacks, FIFO channels, integer or real variables, trees, etc.

Modelling Systems with Automata

Capturing the behaviour of a system

- Control state of the automaton = general "state" of the system
  (e.g. idle, printing, moving, waiting, etc.)
- Transition of the automaton = action/event of the system
  (e.g. message received, data computed, acknowledgment sent, etc.)

Example: an automaton with one counter



Need for richer models, closer to reality

Extend automata with:
stacks, FIFO channels, integer or real variables, trees, etc.

Reprensenting Sets with Arithmetical Logics

Extending automata with variables...

...generates an infinite set of configurations!

☞ How to check all these configurations?

Reprensenting Sets with Arithmetical Logics

Extending automata with variables...

...generates an infinite set of configurations!

☞ How to check all these configurations?

Capturing infinite behaviour of variables

Encode infinite set of values as solutions of arithmetical formula.

Reprensenting Sets with Arithmetical Logics

Extending automata with variables...

...generates an infinite set of configurations!

☞ How to check all these configurations?

Capturing infinite behaviour of variables

Encode infinite set of values as solutions of arithmetical formula.

Example: an arithmetical formula

$\phi(x) := (\exists y.\ x = y + y)$ defines the (infinite) set of even numbers: $\{x \mid \phi(x) \text{ holds}\}$.

Reprensenting Sets with Arithmetical Logics

Extending automata with variables...

...generates an infinite set of configurations!

☞ How to check all these configurations?

Capturing infinite behaviour of variables

Encode infinite set of values as solutions of arithmetical formula.

Example: an arithmetical formula

$\phi(x) := (\exists y.\ x = y + y)$ defines the (infinite) set of even numbers: $\{x \mid \phi(x) \text{ holds}\}$.

Definition

Two logics L, L' (resp. formulas F, F') are equivalent iff they define the same sets. We denote it by $L \equiv L'$ (resp. $F \equiv F'$).

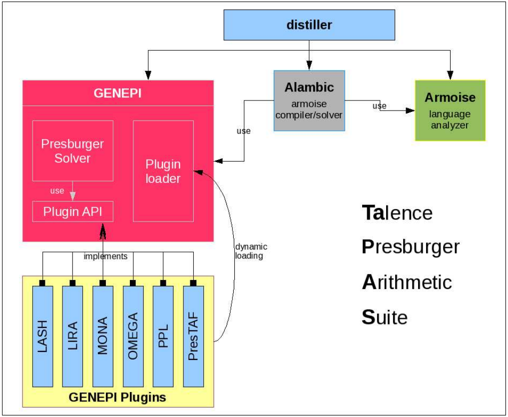
Outline

1. **Decomposing Mixed/Real Logics**
   - Principle of the Decomposition
   - Application over MLA and Extended MLA
2. **Timed Counter Systems**
   - Definitions
   - Region Graph
   - Decidable Subclasses
3. **Dense-choice Counter Machines**
   - Definitions
   - Logical Characterization

Outline

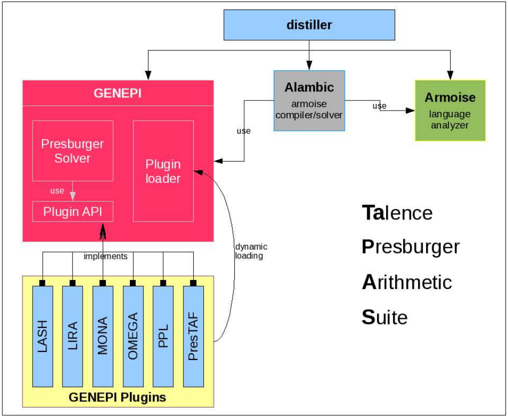
Motivation

Represent vectors of real values by using the modular framework of the GENEPI tool *[Leroux & Point, 06]*:



Motivation

Represent vectors of real values by using the modular framework of the GENEPI tool *[Leroux & Point, 06]*:



☞ Extract the integer component from reals, to use periodicity

Outline

Finite Unions of Sums "Integer+Decimal"

For any $A, B \subseteq \mathbb{R}$, let $A + B = \{a + b \mid a \in A, b \in B\}$.

Example: a real formula decomposed as "$\bigcup(int + dec)$"...

$$\Big(\{(x,y) \mid 0 \leq x < 10^6 \wedge y = 0\} + \{(x,y) \in [0,1]^2 \mid x = y\}\Big)$$
$$\cup\Big(\{(x,y) \mid x = 10^6 \wedge y = 0\} + \{(x,y) \in [0,1]^2 \mid x \geq y\}\Big)$$
$$\cup\Big(\{(x,y) \mid x > 10^6 \wedge y = 0\} + \{(x,y) \in [0,1]^2\}\Big)$$

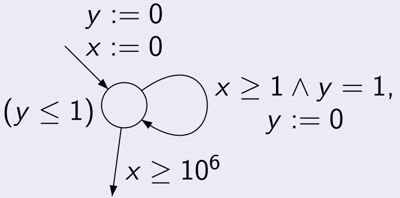
Finite Unions of Sums "Integer+Decimal"

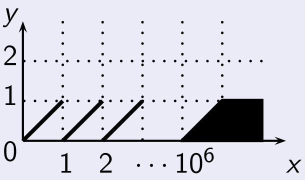
For any $A, B \subseteq \mathbb{R}$, let $A + B = \{a + b \mid a \in A, b \in B\}$.

Example: a real formula decomposed as "$\bigcup(int + dec)$"...

$$\Big(\{(x,y) \mid 0 \leq x < 10^6 \wedge y = 0\} + \{(x,y) \in [0,1]^2 \mid x = y\}\Big)$$
$$\cup\Big(\{(x,y) \mid x = 10^6 \wedge y = 0\} + \{(x,y) \in [0,1]^2 \mid x \geq y\}\Big)$$
$$\cup\Big(\{(x,y) \mid x > 10^6 \wedge y = 0\} + \{(x,y) \in [0,1]^2\}\Big)$$

...representing the clock values of a timed automaton:





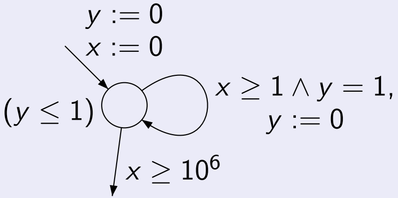
Finite Unions of Sums "Integer+Decimal"

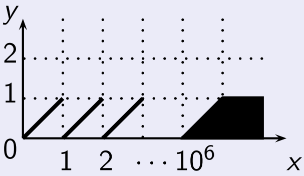
For any $A, B \subseteq \mathbb{R}$, let $A + B = \{a + b \mid a \in A, b \in B\}$.

Example: a real formula decomposed as "$\bigcup(int + dec)$"...

$$\Big(\{(x,y) \mid 0 \leq x < 10^6 \wedge y = 0\} + \diagup\Big)$$
$$\cup\Big(\{(x,y) \mid x = 10^6 \wedge y = 0\} + \{(x,y) \in [0,1]^2 \mid x \geq y\}\Big)$$
$$\cup\Big(\{(x,y) \mid x > 10^6 \wedge y = 0\} + \{(x,y) \in [0,1]^2\}\Big)$$

...representing the clock values of a timed automaton:





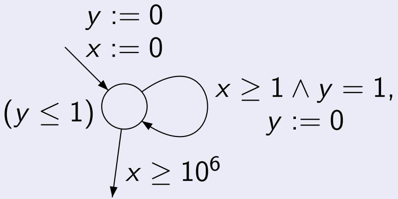
Finite Unions of Sums "Integer+Decimal"

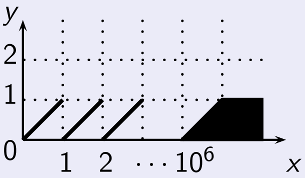
For any $A, B \subseteq \mathbb{R}$, let $A + B = \{a + b \mid a \in A, b \in B\}$.

Example: a real formula decomposed as "$\bigcup(int + dec)$"...

$$
\begin{aligned}
&\Big(\{(x,y) \mid 0 \leq x < 10^6 \wedge y = 0\} + \Big) \\
&\cup \Big(\{(x,y) \mid x = 10^6 \wedge y = 0\} + \Big) \\
&\cup \Big(\{(x,y) \mid x > 10^6 \wedge y = 0\} + \{(x,y) \in [0,1]^2\}\Big)
\end{aligned}
$$

...representing the clock values of a timed automaton:





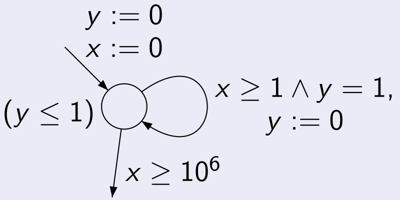
Finite Unions of Sums "Integer+Decimal"

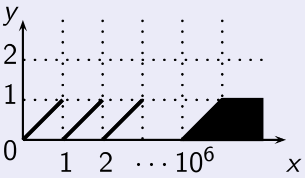
For any $A, B \subseteq \mathbb{R}$, let $A + B = \{a + b \mid a \in A, b \in B\}$.

Example: a real formula decomposed as "$\bigcup(int + dec)$"...

$$\Big(\{(x,y) \mid 0 \le x < 10^6 \wedge y = 0\} + \diagup\Big)$$
$$\cup\Big(\{(x,y) \mid x = 10^6 \wedge y = 0\} + \blacktriangleright\Big)$$
$$\cup\Big(\{(x,y) \mid x > 10^6 \wedge y = 0\} + \blacksquare\Big)$$

...representing the clock values of a timed automaton:





Our representation

- Let $\mathfrak{Z} \subseteq \mathrm{P}(\mathbb{Z}^n)$ and $\mathfrak{D} \subseteq \mathrm{P}(\mathbb{D}^n)$, where $\mathbb{D} = [0, 1[$

Our representation

- Let $\mathfrak{Z} \subseteq \mathrm{P}(\mathbb{Z}^n)$ and $\mathfrak{D} \subseteq \mathrm{P}(\mathbb{D}^n)$, where $\mathbb{D} = [0, 1[$
- Let $\mathfrak{Z} \uplus \mathfrak{D}$ be the class of every $R \subseteq \mathbb{R}^n$ such that $R = \bigcup_{i=1}^{p} (Z_i + D_i)$, where $(Z_i, D_i) \in \mathfrak{Z} \times \mathfrak{D}$ and $p \geq 1$

Our representation

- Let $\mathfrak{Z} \subseteq \mathrm{P}(\mathbb{Z}^n)$ and $\mathfrak{D} \subseteq \mathrm{P}(\mathbb{D}^n)$, where $\mathbb{D} = [0, 1[$
- Let $\mathfrak{Z} \uplus \mathfrak{D}$ be the class of every $R \subseteq \mathbb{R}^n$ such that $R = \bigcup_{i=1}^{p} (Z_i + D_i)$, where $(Z_i, D_i) \in \mathfrak{Z} \times \mathfrak{D}$ and $p \geq 1$
- Note that some R are not representable !

$$\text{Counter-example: } R = \bigcup_{j=1}^{\infty} \left(\{j\} + \left\{ \frac{1}{j+1} \right\} \right)$$

Our representation

- Let $\mathfrak{Z} \subseteq P(\mathbb{Z}^n)$ and $\mathfrak{D} \subseteq P(\mathbb{D}^n)$, where $\mathbb{D} = [0, 1[$
- Let $\mathfrak{Z} \uplus \mathfrak{D}$ be the class of every $R \subseteq \mathbb{R}^n$ such that $R = \bigcup_{i=1}^{p} (Z_i + D_i)$, where $(Z_i, D_i) \in \mathfrak{Z} \times \mathfrak{D}$ and $p \geq 1$
- Note that some R are not representable !

Definition

A class $\mathfrak{R} \subseteq \bigcup_{n \in \mathbb{N}} P(\mathbb{R}^n)$ is stable if it is closed under boolean combinations, cartesian product, projection, and permutation.

Our representation

- Let $\mathfrak{Z} \subseteq \mathrm{P}(\mathbb{Z}^n)$ and $\mathfrak{D} \subseteq \mathrm{P}(\mathbb{D}^n)$, where $\mathbb{D} = [0, 1[$
- Let $\mathfrak{Z} \uplus \mathfrak{D}$ be the class of every $R \subseteq \mathbb{R}^n$ such that $R = \bigcup_{i=1}^{p}(Z_i + D_i)$, where $(Z_i, D_i) \in \mathfrak{Z} \times \mathfrak{D}$ and $p \geq 1$
- Note that some R are not representable !

Definition

A class $\mathfrak{R} \subseteq \bigcup_{n\in\mathbb{N}} P(\mathbb{R}^n)$ is stable if it is closed under boolean combinations, cartesian product, projection, and permutation.

Definition

A class $\mathfrak{R} \subseteq \bigcup_{n\in\mathbb{N}} P(\mathbb{R}^n)$ is effectively representable if it is stable and if there exist algorithms computing the above operations.

Our representation (continued)

Theorem ($\uplus$-Decidability)

The SATisfiability problem is decidable for any logic encoding $\mathfrak{Z} \uplus \mathfrak{D}$ if $\mathfrak{Z}$ and $\mathfrak{D}$ are effectively representable.

Our representation (continued)

Theorem ($\uplus$-Decidability)

The SATisfiability problem is decidable for any logic encoding $\mathfrak{Z} \uplus \mathfrak{D}$ if $\mathfrak{Z}$ and $\mathfrak{D}$ are effectively representable.

Proof sketch

- $(Z_1 + D_1) \cap (Z_2 + D_2) = (Z_1 \cap Z_2) + (D_1 \cap D_2)$
- $(Z_1 + D_1) \backslash (Z_2 + D_2) = ((Z_1 \backslash Z_2) + D_1) \cup (Z_1 + (D_1 \backslash D_2))$
- similarly, projection and permutation distribute easily over $\uplus$

Our representation (continued)

Theorem ($\uplus$-Decidability)

The SATisfiability problem is decidable for any logic encoding $\mathfrak{Z} \uplus \mathfrak{D}$ if $\mathfrak{Z}$ and $\mathfrak{D}$ are effectively representable.

Proof sketch

- $(Z_1 + D_1) \cap (Z_2 + D_2) = (Z_1 \cap Z_2) + (D_1 \cap D_2)$
- $(Z_1 + D_1) \backslash (Z_2 + D_2) = ((Z_1 \backslash Z_2) + D_1) \cup (Z_1 + (D_1 \backslash D_2))$
- similarly, projection and permutation distribute easily over $\uplus$
- $\mathbb{R}^n$ can be written $\mathbb{Z}^n + \mathbb{D}^n$
- $R_+ = \{\mathbf{r} \in \mathbb{R}^3 \mid r_1 + r_2 = r_3\}$ can be written
  $$\bigcup_{c \in \{0,1\}} \{\mathbf{z} \in \mathbb{Z}^3 \mid z_1 + z_2 + c = z_3\} + \{\mathbf{d} \in \mathbb{D}^3 \mid d_1 + d_2 = d_3 + c\}$$
- similarly, $\emptyset$, $\leq$, $\mathbb{Z}^n$ are easily definable $\square$

Outline

1. **Decomposing Mixed/Real Logics**
   - Principle of the Decomposition
   - Application over MLA and Extended MLA
2. **Timed Counter Systems**
   - Definitions
   - Region Graph
   - Decidable Subclasses
3. **Dense-choice Counter Machines**
   - Definitions
   - Logical Characterization

A quick reminder about logics...

Presburger logic: $\mathbf{FO}\,(\mathbb{Z}, +, \leq)$

$$\begin{aligned}
\text{expression } E &::= 0 \mid 1 \mid x \mid E + E \mid E - E \\
\text{atomic formula } C &::= E = E \mid E < E \\
\text{formula } F &::= C \mid \neg F \mid F \wedge F \mid \exists x. F
\end{aligned}$$

A quick reminder about logics...

Presburger logic: $\mathrm{FO}(\mathbb{Z}, +, \leq)$

$$\begin{aligned} \text{expression } E &::= 0 \mid 1 \mid x \mid E + E \mid E - E \\ \text{atomic formula } C &::= E = E \mid E < E \\ \text{formula } F &::= C \mid \neg F \mid F \wedge F \mid \exists x.F \end{aligned}$$

Theorem *[Presburger, 29]*

The SATisfiability problem is decidable for the logic $\mathrm{FO}(\mathbb{Z}, +, \leq)$.

A quick reminder about logics...

Presburger logic: $\mathrm{FO}(\mathbb{Z}, +, \leq)$

$$\begin{aligned} \text{expression } E &::= 0 \mid 1 \mid x \mid E + E \mid E - E \\ \text{atomic formula } C &::= E = E \mid E < E \\ \text{formula } F &::= C \mid \neg F \mid F \wedge F \mid \exists x.F \end{aligned}$$

Theorem *[Presburger, 29]*

The SATisfiability problem is decidable for the logic $\mathrm{FO}(\mathbb{Z}, +, \leq)$.

Example: a Presburger-definable set

$\phi(x) := (\exists y.\ x = y + y)$ defines the set of even numbers.

A quick reminder about logics...

Presburger logic: $\mathrm{FO}(\mathbb{Z}, +, \leq)$

$$\begin{aligned} \text{expression } E &::= 0 \mid 1 \mid x \mid E + E \mid E - E \\ \text{atomic formula } C &::= E = E \mid E < E \\ \text{formula } F &::= C \mid \neg F \mid F \wedge F \mid \exists x.F \end{aligned}$$

Theorem *[Presburger, 29]*

The SATisfiability problem is decidable for the logic $\mathrm{FO}(\mathbb{Z}, +, \leq)$.

Example: a Presburger-definable set

$\phi(x) := (\exists y.\ x = y + y)$ defines the set of even numbers.

Example: a non-Presburger-definable set

There is no Presburger formula defining the set $\{2^n \mid n \in \mathbb{N}\}$.

A quick reminder about logics... (continued)

Mixed Linear Arithmetic (MLA): $\mathrm{FO}(\mathbb{R}, \mathbb{Z}, +, \leq)$

$$\begin{aligned} \text{expression } E &::= 0 \mid 1 \mid z \mid r \mid E + E \mid E - E \\ \text{atomic formula } C &::= E = E \mid E < E \\ \text{formula } F &::= C \mid \neg F \mid F \wedge F \mid \exists z.F \mid \exists r.F \end{aligned}$$

A quick reminder about logics... (continued)

Mixed Linear Arithmetic (MLA): $\mathrm{FO}(\mathbb{R}, \mathbb{Z}, +, \leq)$

$$\begin{aligned} \text{expression } E &::= 0 \mid 1 \mid z \mid r \mid E + E \mid E - E \\ \text{atomic formula } C &::= E = E \mid E < E \\ \text{formula } F &::= C \mid \neg F \mid F \wedge F \mid \exists z.F \mid \exists r.F \end{aligned}$$

Theorem *[Boigelot et al., 98; Weispfenning, 99]*

The SATisfiability problem is decidable for the logic $\mathrm{FO}(\mathbb{R}, \mathbb{Z}, +, \leq)$.

Decomposing MLA and Extended MLA

Theorem *[TIME'08]*

$$\mathrm{FO}\left(\mathbb{R},\mathbb{Z},+,\leq\right) \equiv \mathrm{FO}\left(\mathbb{Z},+,\leq\right) \uplus \mathrm{FO}\left(\mathbb{D},+,\leq\right)$$

Decomposing MLA and Extended MLA

Theorem *[TIME'08]*

$\mathrm{FO}(\mathbb{R}, \mathbb{Z}, +, \leq) \equiv \mathrm{FO}(\mathbb{Z}, +, \leq) \uplus \mathrm{FO}(\mathbb{D}, +, \leq)$

Proof idea

Application of the $\uplus$-Decidability theorem

Decomposing MLA and Extended MLA

Theorem *[TIME'08]*

$\mathrm{FO}(\mathbb{R}, \mathbb{Z}, +, \leq) \equiv \mathrm{FO}(\mathbb{Z}, +, \leq) \uplus \mathrm{FO}(\mathbb{D}, +, \leq)$

Proof idea

Application of the $\uplus$-Decidability theorem

Proposition *[TIME'08]*

$\mathrm{FO}(\mathbb{R}, \mathbb{Z}, +, \leq, X_b) \equiv \mathrm{FO}(\mathbb{Z}, +, \leq, V_b) \uplus \mathrm{FO}(\mathbb{D}, +, \leq, W_b)$

Decomposing MLA and Extended MLA

Theorem *[TIME'08]*

$$\mathrm{FO}(\mathbb{R}, \mathbb{Z}, +, \leq) \equiv \mathrm{FO}(\mathbb{Z}, +, \leq) \uplus \mathrm{FO}(\mathbb{D}, +, \leq)$$

Proof idea

Application of the $\uplus$-Decidability theorem

Proposition *[TIME'08]*

$$\mathrm{FO}(\mathbb{R}, \mathbb{Z}, +, \leq, X_b) \equiv \mathrm{FO}(\mathbb{Z}, +, \leq, V_b) \uplus \mathrm{FO}(\mathbb{D}, +, \leq, W_b)$$

Proof idea

- Application of the $\uplus$-Decidability theorem
- Mutual encodings of X_b, V_b, W_b

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$\mathrm{FO}\left(\mathbb{R}, \mathbb{Z}, +, \leq, X_b\right) \equiv \mathrm{FO}\left(\mathbb{Z}, +, \leq, V_b\right) \uplus \mathrm{FO}\left(\mathbb{D}, +, \leq, W_b\right)$

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$$\mathrm{FO}(\mathbb{R}, \mathbb{Z}, +, \leq, X_b) \equiv \mathrm{FO}(\mathbb{Z}, +, \leq, V_b) \uplus \mathrm{FO}(\mathbb{D}, +, \leq, W_b)$$

- "$X_b(x, u, a)$: in x's b-ary writing, the digit in position u is a"

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$$\mathrm{FO}\left(\mathbb{R}, \mathbb{Z}, +, \leq, X_b\right) \equiv \mathrm{FO}\left(\mathbb{Z}, +, \leq, V_b\right) \uplus \mathrm{FO}\left(\mathbb{D}, +, \leq, W_b\right)$$

- "$X_b(x, u, a)$: in x's b-ary writing, the digit in position u is a"
- "$V_b(x) = u$: in x's b-ary writing, the rightmost non-null integer digit is at position u"

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$$\mathrm{FO}\left(\mathbb{R}, \mathbb{Z}, +, \leq, X_b\right) \equiv \mathrm{FO}\left(\mathbb{Z}, +, \leq, V_b\right) \uplus \mathrm{FO}\left(\mathbb{D}, +, \leq, W_b\right)$$

- "$X_b(x, u, a)$: in x's b-ary writing, the digit in position u is a"
- "$V_b(x) = u$: in x's b-ary writing, the rightmost non-null integer digit is at position u"
- "$W_b(x) = u$: in x's b-ary writing, the leftmost non-null decimal digit is at position u"

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$$\mathrm{FO}\left(\mathbb{R}, \mathbb{Z}, +, \leq, X_b\right) \equiv \mathrm{FO}\left(\mathbb{Z}, +, \leq, V_b\right) \uplus \mathrm{FO}\left(\mathbb{D}, +, \leq, W_b\right)$$

Example illustrating X_b, V_b, W_b

$$x = (42.0625)_{10} = (0^*101010 \star 00010^\omega)_2$$

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$$\mathrm{FO}\left(\mathbb{R}, \mathbb{Z}, +, \leq, X_b\right) \equiv \mathrm{FO}\left(\mathbb{Z}, +, \leq, V_b\right) \uplus \mathrm{FO}\left(\mathbb{D}, +, \leq, W_b\right)$$

Example illustrating X_b, V_b, W_b

$$X_2(x, 2^4, 0)$$

$$x = (42.0625)_{10} = (0^* 101010 \star 00010^\omega)_2$$

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$$\mathrm{FO}\left(\mathbb{R}, \mathbb{Z}, +, \leq, X_b\right) \equiv \mathrm{FO}\left(\mathbb{Z}, +, \leq, V_b\right) \uplus \mathrm{FO}\left(\mathbb{D}, +, \leq, W_b\right)$$

Example illustrating X_b, V_b, W_b

$$X_2(x, 2^1, 1)$$

$$X_2(x, 2^4, 0)$$

$$x = (42.0625)_{10} = (0^* 101010 \star 00010^\omega)_2$$

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$$\mathrm{FO}\left(\mathbb{R}, \mathbb{Z}, +, \leq, X_b\right) \equiv \mathrm{FO}\left(\mathbb{Z}, +, \leq, V_b\right) \uplus \mathrm{FO}\left(\mathbb{D}, +, \leq, W_b\right)$$

Example illustrating X_b, V_b, W_b

$$X_2(x, 2^1, 1)$$

$$X_2(x, 2^4, 0) \qquad X_2(x, 2^{-2}, 0)$$

$$x = (42.0625)_{10} = (0^*101010 \star 00010^\omega)_2$$

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$$\mathrm{FO}\,(\mathbb{R}, \mathbb{Z}, +, \leq, X_b) \equiv \mathrm{FO}\,(\mathbb{Z}, +, \leq, V_b) \uplus \mathrm{FO}\,(\mathbb{D}, +, \leq, W_b)$$

Example illustrating X_b, V_b, W_b

$$X_2(x, 2^1, 1)$$

$$X_2(x, 2^4, 0) \qquad X_2(x, 2^{-2}, 0)$$

$$x = (42.0625)_{10} = (0^*101010 \star 00010^\omega)_2$$

$$V_2(x) = 2^1$$

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$$\mathrm{FO}\left(\mathbb{R}, \mathbb{Z}, +, \leq, X_b\right) \equiv \mathrm{FO}\left(\mathbb{Z}, +, \leq, V_b\right) \uplus \mathrm{FO}\left(\mathbb{D}, +, \leq, W_b\right)$$

Example illustrating X_b, V_b, W_b

$$X_2(x, 2^1, 1)$$

$$X_2(x, 2^4, 0) \qquad X_2(x, 2^{-2}, 0)$$

$$x = (42.0625)_{10} = (0^*101010 \star 00010^\omega)_2$$

$$V_2(x) = 2^1$$

$$W_2(x) = 2^{-4}$$

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$$\mathrm{FO}(\mathbb{R}, \mathbb{Z}, +, \leq, X_b) \equiv \mathrm{FO}(\mathbb{Z}, +, \leq, V_b) \uplus \mathrm{FO}(\mathbb{D}, +, \leq, W_b)$$

Proof idea: mutual encodings of X_b, V_b, W_b

$$X_b(x, u, a) \equiv \exists w \in \mathbb{D}, \exists y \in \mathbb{Z}, \exists z \in \mathbb{R},$$
$$x = w + y + z + au \wedge z < u$$
$$\wedge \Big(\big(u \geq 1 \wedge V_b(y) \geq bu \big) \vee \big(u < 1 \wedge W_b(w) \geq bu \big) \Big)$$

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$$\mathrm{FO}\,(\mathbb{R}, \mathbb{Z}, +, \leq, X_b) \equiv \mathrm{FO}\,(\mathbb{Z}, +, \leq, V_b) \uplus \mathrm{FO}\,(\mathbb{D}, +, \leq, W_b)$$

Proof idea: mutual encodings of X_b, V_b, W_b

$$X_b(x, u, a) \equiv \exists w \in \mathbb{D}, \exists y \in \mathbb{Z}, \exists z \in \mathbb{R},$$
$$x = w + y + z + au \wedge z < u$$
$$\wedge \Big(\big(u \geq 1 \wedge V_b(y) \geq bu \big) \vee \big(u < 1 \wedge W_b(w) \geq bu \big) \Big)$$

$$V_b(x) = y \equiv \exists a \in \mathbb{R},$$
$$X_b(x, y, a) \wedge a \neq 0 \wedge \forall z \in \mathbb{N} \big(z < y \Rightarrow X_b(x, z, 0) \big)$$

Extended MLA: X_b, V_b, W_b

Proposition *[TIME'08]*

$$\mathrm{FO}(\mathbb{R}, \mathbb{Z}, +, \leq, X_b) \equiv \mathrm{FO}(\mathbb{Z}, +, \leq, V_b) \uplus \mathrm{FO}(\mathbb{D}, +, \leq, W_b)$$

Proof idea: mutual encodings of X_b, V_b, W_b

$$X_b(x, u, a) \equiv \exists w \in \mathbb{D}, \exists y \in \mathbb{Z}, \exists z \in \mathbb{R},$$
$$x = w + y + z + au \wedge z < u$$
$$\wedge \Big(\big(u \geq 1 \wedge V_b(y) \geq bu \big) \vee \big(u < 1 \wedge W_b(w) \geq bu \big) \Big)$$

$$V_b(x) = y \equiv \exists a \in \mathbb{R},$$
$$X_b(x, y, a) \wedge a \neq 0 \wedge \forall z \in \mathbb{N} \big(z < y \Rightarrow X_b(x, z, 0) \big)$$

$$W_b(x) = y \equiv \exists a \in \mathbb{R},$$
$$X_b(x, y, a) \wedge a \neq 0 \wedge \forall z \in \mathbb{D} \big((bz < 1 \wedge z > y) \Rightarrow X_b(x, z, 0) \big)$$

Outline

Motivation

Initial observation

Motivation

Initial observation

- need to model time in formal verification;

Motivation

Initial observation

- need to model time in formal verification;
  Timed Automata: widespread and efficient way to model time

Motivation

Initial observation

- need to model time in formal verification;
  Timed Automata: widespread and efficient way to model time
- need for a richer and more general model;

Motivation

Initial observation

- need to model time in formal verification;
  Timed Automata: widespread and efficient way to model time
- need for a richer and more general model;
  counters: most used datatype in verification case studies

Motivation

Initial observation

- need to model time in formal verification;
  Timed Automata: widespread and efficient way to model time
- need for a richer and more general model;
  counters: most used datatype in verification case studies
- models using counters have several different definitions;

Motivation

Initial observation

- need to model time in formal verification;
  Timed Automata: widespread and efficient way to model time
- need for a richer and more general model;
  counters: most used datatype in verification case studies
- models using counters have several different definitions;
  Counter Systems: can be generalized to a unifying definition

Motivation

Initial observation

- need to model time in formal verification;
  Timed Automata: widespread and efficient way to model time
- need for a richer and more general model;
  counters: most used datatype in verification case studies
- models using counters have several different definitions;
  Counter Systems: can be generalized to a unifying definition

☞ *We combine Timed Automata and Counter Systems*

Motivation

Initial observation

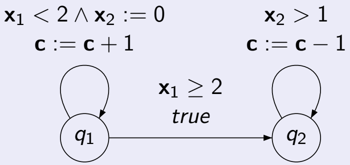
- need to model time in formal verification;
  Timed Automata: widespread and efficient way to model time
- need for a richer and more general model;
  counters: most used datatype in verification case studies
- models using counters have several different definitions;
  Counter Systems: can be generalized to a unifying definition

☞ *We combine Timed Automata and Counter Systems and we study their reachability matters*

Outline

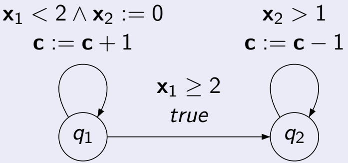
Definitions

Example: a Timed Counter System (TCS)



Definitions

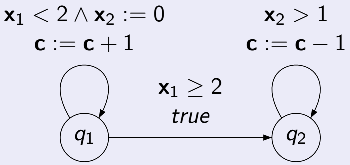
Example: a Timed Counter System (TCS)



X = a set of m real-valued variables, called clocks.
$\mathbf{x}$ = a valuation of the clocks, in $\mathbb{R}_+^m$.
R_X = resets and linear guards over clocks

Definitions

Example: a Timed Counter System (TCS)



X = a set of m real-valued variables, called clocks.
$\mathbf{x}$ = a valuation of the clocks, in $\mathbb{R}_+^m$.
R_X = resets and linear guards over clocks

C = a set of n integer-valued variables, called counters.
$\mathbf{c}$ = a valuation of the counters, in $\mathbb{Z}^n$.
R_C = Presburger-definable binary relations over counters

Definitions (continued)

Definition *[INFINITY'08]*

A Timed Counter System is a tuple $\langle Q, X, C, E \rangle$ where:

- Q is a finite set of control states
- $E \subseteq Q \times R_X \times R_C \times Q$ is a finite set of transitions

Definitions (continued)

Definition *[INFINITY'08]*

A Timed Counter System is a tuple $\langle Q, X, C, E \rangle$ where:

- Q is a finite set of control states
- $E \subseteq Q \times R_X \times R_C \times Q$ is a finite set of transitions

R_X and R_C are disjoint!

Definitions (continued)

Definition *[INFINITY'08]*

A Timed Counter System is a tuple $\langle Q, X, C, E \rangle$ where:

- Q is a finite set of control states
- $E \subseteq Q \times R_X \times R_C \times Q$ is a finite set of transitions

R_X and R_C are disjoint!

Definition

A *Timed Automaton* is a TCS where $C = \emptyset$.
A *Counter System* is a TCS where $X = \emptyset$.

Semantics of a TCS

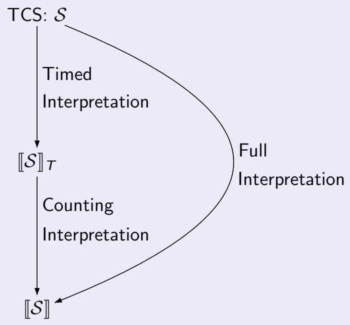
For any TCS $\mathcal{S}$, define its:

- Counting Transition System $[\![\mathcal{S}]\!]_C$
- Timed Transition System $[\![\mathcal{S}]\!]_T$
- (full) Transition System $[\![\mathcal{S}]\!]$

Semantics of a TCS

For any TCS $\mathcal{S}$, define its:

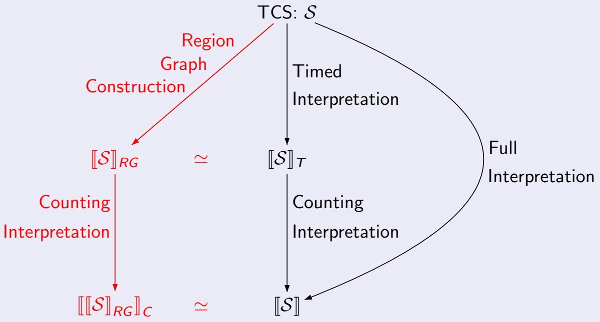
- Counting Transition System $[\![\mathcal{S}]\!]_C$
- Timed Transition System $[\![\mathcal{S}]\!]_T$
- (full) Transition System $[\![\mathcal{S}]\!]$



Semantics of a TCS

For any TCS $\mathcal{S}$, define its:

- Counting Transition System $[\![\mathcal{S}]\!]_C$
- Timed Transition System $[\![\mathcal{S}]\!]_T$ $\simeq$ Region Graph $[\![\mathcal{S}]\!]_{RG}$
- (full) Transition System $[\![\mathcal{S}]\!]$ $\simeq$ $[\![[\![\mathcal{S}]\!]_{RG}]\!]_C$



Reachability

Clocks are used for modelling temporal requirements; their *exact* value does not really matter.

Reachability

Clocks are used for modelling temporal requirements; their *exact* value does not really matter.

Counter Reachability Problem (CRP)

Inputs: A TCS $\mathcal{S}$, an initial configuration $(q_0, \mathbf{x}_0, \mathbf{c}_0)$ of $[\![\mathcal{S}]\!]$, and a configuration $(q, \mathbf{c})$ of $[\![\mathcal{S}]\!]_C$.
Question: Is there a clock valuation $\mathbf{x}$ such that $(q, \mathbf{x}, \mathbf{c})$ is reachable from $(q_0, \mathbf{x}_0, \mathbf{c}_0)$ in $[\![\mathcal{S}]\!]$?

Reachability

Clocks are used for modelling temporal requirements; their *exact* value does not really matter.

Counter Reachability Problem (CRP)

Inputs: A TCS $\mathcal{S}$, an initial configuration $(q_0, \mathbf{x}_0, \mathbf{c}_0)$ of $[\![\mathcal{S}]\!]$, and a configuration $(q, \mathbf{c})$ of $[\![\mathcal{S}]\!]_C$.

Question: Is there a clock valuation $\mathbf{x}$ such that $(q, \mathbf{x}, \mathbf{c})$ is reachable from $(q_0, \mathbf{x}_0, \mathbf{c}_0)$ in $[\![\mathcal{S}]\!]$?

CRP extends reachability problem of CS, known to be undecidable *[Minsky, 61]*; therefore CRP is undecidable for TCS.

Reachability

Clocks are used for modelling temporal requirements; their *exact* value does not really matter.

Counter+region Reachability Problem (CRP)

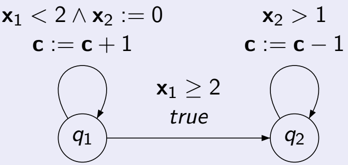
Inputs: A TCS $\mathcal{S}$, an initial configuration $(q_0, \rho_0, \mathbf{c}_0)$ of $[\![\mathcal{S}]\!]$, a region ρ, and a configuration $(q, \mathbf{c})$ of $[\![\mathcal{S}]\!]_C$.
Question: Is $(q, \rho, \mathbf{c})$ reachable from $(q_0, \rho_0, \mathbf{c}_0)$ in $[\![[\![\mathcal{S}]\!]_{RG}]\!]_C$?

CRP extends reachability problem of CS, known to be undecidable *[Minsky, 61]*; therefore CRP is undecidable for TCS.

Outline

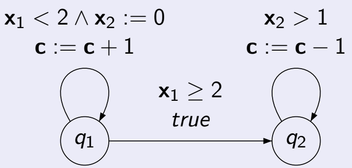
Regions

Timed Counter System $\mathcal{S}$

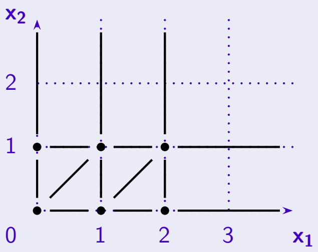


Regions

Timed Counter System $\mathcal{S}$



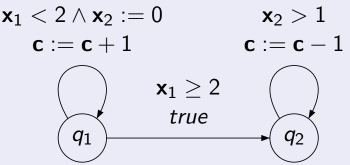
Clock regions of $\mathcal{S}$



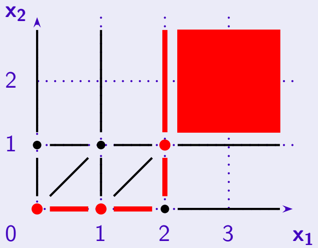
28 regions in total:
6 points, 9 line segments, 5 half-lines, 4 triangular closed areas, and 4 open areas

Regions

Timed Counter System $\mathcal{S}$



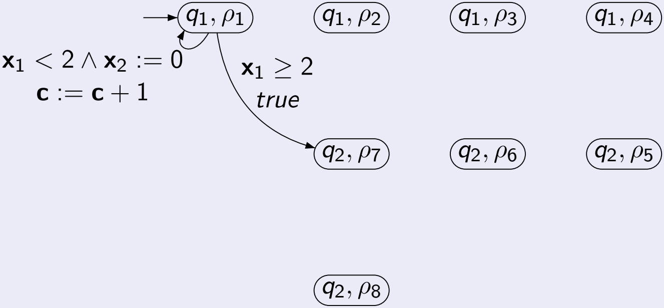
Reachable clock regions of $\mathcal{S}$



8 reachable regions (out of 28), considering the initial configuration $\left(q_1, \binom{0}{0}, 0\right)$

Region Graph

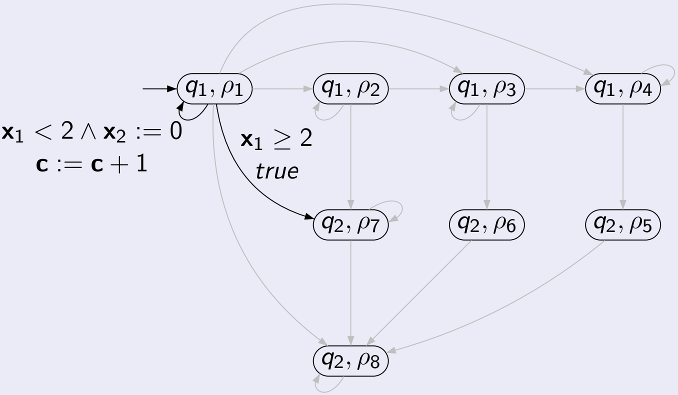
Region Graph $[\![\mathcal{S}]\!]_{RG}$



N.B.: ρ_i are the 8 reachable regions

Region Graph

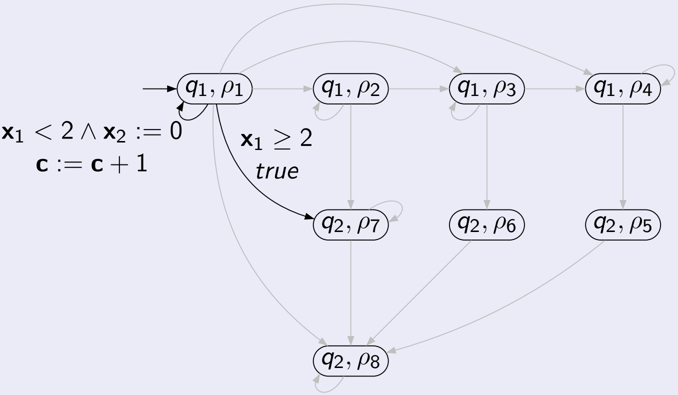
Region Graph $[\![\mathcal{S}]\!]_{RG}$



N.B.: ρ_i are the 8 reachable regions

Region Graph

Region Graph $[\![\mathcal{S}]\!]_{RG}$ is a Counter System!



N.B.: ρ_i are the 8 reachable regions

The Region Graph as a Counter System

Key idea of the analysis of TCS:

For a TCS $\mathcal{S}$, its region graph $[\![\mathcal{S}]\!]_{RG}$ is also a Counter System.

The Region Graph as a Counter System

Key idea of the analysis of TCS:

For a TCS $\mathcal{S}$, its region graph $[\![\mathcal{S}]\!]_{RG}$ is also a Counter System.

Let $\mathfrak{C}$ be a class of TCS such that there is an algorithm solving the reachability problem for $[\![\mathcal{S}]\!]_{RG}$, for any $\mathcal{S} \in \mathfrak{C}$.
(e.g. $\mathfrak{C}$ can be VASS/Petri Nets, bounded CS, etc.)

Theorem *[INFINITY'08]*

The Counter Reachability Problem is decidable for $\mathfrak{C}$.

The Region Graph as a Counter System

Key idea of the analysis of TCS:

For a TCS $\mathcal{S}$, its region graph $[\![\mathcal{S}]\!]_{RG}$ is also a Counter System.

Let $\mathfrak{C}$ be a class of TCS such that there is an algorithm solving the reachability problem for $[\![\mathcal{S}]\!]_{RG}$, for any $\mathcal{S} \in \mathfrak{C}$.
(e.g. $\mathfrak{C}$ can be VASS/Petri Nets, bounded CS, etc.)

Theorem *[INFINITY'08]*

The Counter Reachability Problem is decidable for $\mathfrak{C}$.

Proof sketch ▸ time-abstract bisimulation

- By definition, $[\![[\![\mathcal{S}]\!]_T]\!]_C = [\![[\![\mathcal{S}]\!]_C]\!]_T = [\![\mathcal{S}]\!]$.
- $[\![\mathcal{S}]\!]_{RG} \simeq [\![\mathcal{S}]\!]_T$ *[Alur & Dill, 90]*.
- Therefore $[\![[\![\mathcal{S}]\!]_{RG}]\!]_C \simeq [\![\mathcal{S}]\!]$. □

Outline

Subclasses of TCS

Decidability results

Model	Region Graph	Counter Reach. Prob.
TCS	CS	Undecidable
TVASS	VASS	Decidable
Reversal-bounded TCM	Reversal-bounded CM	Decidable
Bounded TCS	Bounded CS	Decidable

Subclasses of TCS

Decidability results

Model	Region Graph	Counter Reach. Prob.
TCS	CS	Undecidable
TVASS	VASS	Decidable
Reversal-bounded TCM	Reversal-bounded CM	Decidable
Bounded TCS	Bounded CS	Decidable

- Timed Counter Machine (TCM) = TCS whose relations on counters are translations with guards of the form $\mathbf{c} \geq k$ or $\mathbf{c} = k$, with $k \in \mathbb{N}^n$
- Timed VASS (TVASS) = TCM with only $\mathbf{c} \geq k$ guards
- Bounded TCS = TCS whose counter values are bounded
- Reversal-Bounded TCM = TCM whose counters do a bounded number of alternations between increasing and decreasing modes

Outline

1. **Decomposing Mixed/Real Logics**
   - Principle of the Decomposition
   - Application over MLA and Extended MLA
2. **Timed Counter Systems**
   - Definitions
   - Region Graph
   - Decidable Subclasses
3. **Dense-choice Counter Machines**
   - Definitions
   - Logical Characterization

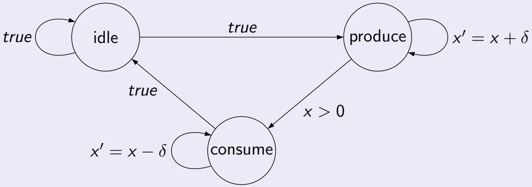
Motivation

- Extend integer-valued Counter Machines to real-valued Counter Machines
- Model hybrid systems (i.e. discrete automata featuring time, temperature, pressure, etc.)

Motivation

- Extend integer-valued Counter Machines to real-valued Counter Machines
- Model hybrid systems (i.e. discrete automata featuring time, temperature, pressure, etc.)

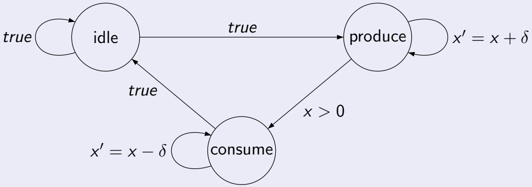
Example: a Dense-choice Counter Machine (DCM)



Motivation

- Extend integer-valued Counter Machines to real-valued Counter Machines
- Model hybrid systems (i.e. discrete automata featuring time, temperature, pressure, etc.)

Example: a Dense-choice Counter Machine (DCM)



A new value δ is chosen in $]0, 1[$ at each step!

Outline

1. **Decomposing Mixed/Real Logics**
   - Principle of the Decomposition
   - Application over MLA and Extended MLA

2. **Timed Counter Systems**
   - Definitions
   - Region Graph
   - Decidable Subclasses

3. **Dense-choice Counter Machines**
   - Definitions
   - Logical Characterization

Definitions

Guards $G = \{(x = 0), (x > 0), true\}$
Actions $A = \{1, \Delta\}$ (i.e. increments/decrements)
☞ 1 is "integer", Δ is "non-deterministically-chosen decimal"

DCM

A Dense-choice Counter Machine with $n > 0$ counters is a tuple $\mathcal{M} = \langle S, T \rangle$ where:

- S is a finite set of control states
- $T \subseteq S \times \Sigma \times S$ is a finite set of transitions, with $\Sigma = (G \times \mathbb{Z} \times A)^n$

Definitions

Guards $G = \{(x = 0), (x > 0), true\}$
Actions $A = \{1, \Delta\}$ (i.e. increments/decrements)
☞ 1 is "integer", Δ is "non-deterministically-chosen decimal"

DCM

A Dense-choice Counter Machine with $n > 0$ counters is a tuple $\mathcal{M} = \langle S, T\rangle$ where:

- S is a finite set of control states
- $T \subseteq S \times \Sigma \times S$ is a finite set of transitions, with $\Sigma = (G \times \mathbb{Z} \times A)^n$

Example

A transition $x > 0 \wedge x := x - 3\delta$ is written $(x = 0, -3, \Delta)$.

Definitions (continued)

Semantics of a DCM $\mathcal{M} = \langle S, T \rangle$

The transition system $TS(\mathcal{M}) = \langle C, \rightarrow \rangle$ is defined by:

- $C = S \times \mathbb{R}_+^n$ is the set of configurations
- $\rightarrow \subseteq C \times \Sigma \times C$ is the set of transitions, defined by:
  $(s, \mathbf{x}) \xrightarrow{\mathbf{g}, \boldsymbol{\lambda}, \mathbf{a}} (s', \mathbf{x}') \iff (s, (\mathbf{g}, \boldsymbol{\lambda}, \mathbf{a}), s') \in T \wedge \exists \delta \in \mathbb{R}$ such that:
  $0 < \delta < 1 \wedge \mathbf{x} \models \mathbf{g} \wedge \mathbf{x}' = \mathbf{x} + \boldsymbol{\lambda}\mathbf{d}$, with $\mathbf{d} = \mathbf{a}[\Delta \leftarrow \delta]$

Definitions (continued)

Semantics of a DCM $\mathcal{M} = \langle S, T \rangle$

The transition system $TS(\mathcal{M}) = \langle C, \rightarrow \rangle$ is defined by:

- $C = S \times \mathbb{R}_+^n$ is the set of configurations
- $\rightarrow \subseteq C \times \Sigma \times C$ is the set of transitions, defined by:
  $(s, \mathbf{x}) \xrightarrow{\mathbf{g},\boldsymbol{\lambda},\mathbf{a}} (s', \mathbf{x}') \iff (s, (\mathbf{g}, \boldsymbol{\lambda}, \mathbf{a}), s') \in T \wedge \exists \delta \in \mathbb{R}$ such that:
  $0 < \delta < 1 \wedge \mathbf{x} \models \mathbf{g} \wedge \mathbf{x}' = \mathbf{x} + \boldsymbol{\lambda}\mathbf{d}$, with $\mathbf{d} = \mathbf{a}[\Delta \leftarrow \delta]$

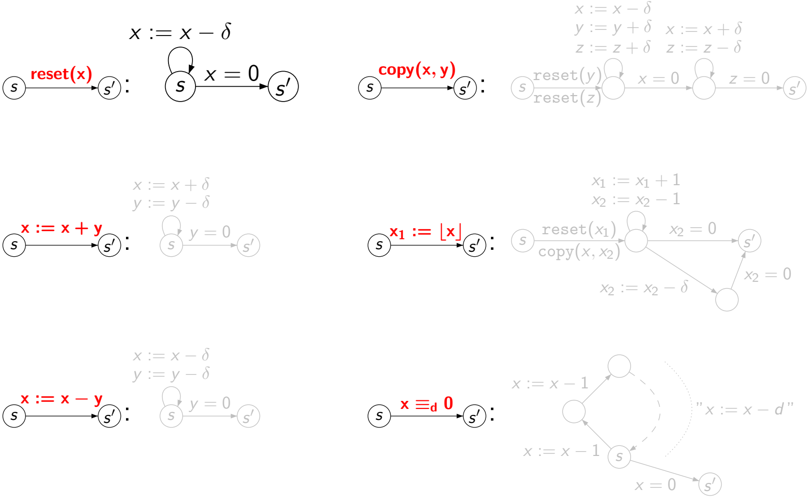
Subclasses of DCM

- purely-DCM: $a_i = \Delta$ for each counter $i \leq n$
- (discrete) CM: $a_i = 1$ for each counter $i \leq n$ (i.e. Minsky machine)
- reversal-bounded DCM: $\exists r \in \mathbb{N}$ s.t. on every run, each counter switches between increments and decrements at most r times
- bounded DCM: $\exists b \in \mathbb{N}$ s.t. on every run, no counter exceeds b

Encodings of macro-operations in DCM

$$s \xrightarrow{\textbf{reset(x)}} s' : \quad \begin{tikzcd} s \arrow[loop above, "x := x - \delta"] \arrow[r, "x = 0"] & s' \end{tikzcd}$$

Encodings of macro-operations in DCM



Outline

1. Decomposing Mixed/Real Logics
   - Principle of the Decomposition
   - Application over MLA and Extended MLA
2. Timed Counter Systems
   - Definitions
   - Region Graph
   - Decidable Subclasses
3. **Dense-choice Counter Machines**
   - Definitions
   - **Logical Characterization**

Reversal-Bounded DCM and MLA

Theorem *[Ibarra, 78]*

The reachability relation of a reversal-bounded CM is definable in Presburger logic.

Reversal-Bounded DCM and MLA

Theorem *[Ibarra, 78]*

The reachability relation of a reversal-bounded CM is definable in Presburger logic.

Theorem *[Ibarra & San Pietro et al., 03]*

The reachability relation of a reversal-bounded DCM is definable in MLA.

Reversal-Bounded DCM and MLA

Theorem *[Ibarra, 78]*

The reachability relation of a reversal-bounded CM is definable in Presburger logic.

Theorem *[Ibarra & San Pietro et al., 03]*

The reachability relation of a reversal-bounded DCM is definable in MLA.

Theorem *[INFINITY'09]*

Each MLA formula defines the reachability relation of a reversal-bounded DCM.

Contributions

(De)composition of logics

- Characterization of logics
- Decidability proofs for SATisfiability of logics

Contributions

(De)composition of logics

- Characterization of logics
- Decidability proofs for SATisfiability of logics

TCS

- New model mixing clocks and counters
- Analysis using region abstraction and subclasses identification

Contributions

(De)composition of logics

- Characterization of logics
- Decidability proofs for SATisfiability of logics

TCS

- New model mixing clocks and counters
- Analysis using region abstraction and subclasses identification

DCM

- Full logical characterization of reversal-bounded DCM with MLA

Future Work

Theory

- Study new logics through decomposition (for charac. & decidability)
- Extend decidable subclasses (flat TCS)
- Generalize TCS analysis technique to Timed *Heterogeneous* Systems
- Solve open decidability problems for subclasses of DCM
- Study complexity of decision procedures in DCM

Future Work

Theory

- Study new logics through decomposition (for charac. & decidability)
- Extend decidable subclasses (flat TCS)
- Generalize TCS analysis technique to Timed *Heterogeneous* Systems
- Solve open decidability problems for subclasses of DCM
- Study complexity of decision procedures in DCM

Implementation

- Enhance GENEPI prototype decomposing logics
- Implement region construction in FAST
- Design a tool for reversal-bounded DCM