

Combination of accelerations.

Sébastien Bardin
Joint work with Alain Finkel

LSV - CNRS & ÉNS de Cachan

Université Libre de Bruxelles, 29/11/2004

Safe software?

- New technologies spread quickly in many fields of every-day life: mobile phones, cars, e-administration, online banking, etc.
- More and more critical data are processed automatically.
- Software verification is crucial

Standard methods does not work

- Simulation and test: exponential effort in program size.
- Program size increases exponentially with time.
- Too expensive!!

Need for computer-aided verification

Different methods

- Theorem proving.
- Test.
- Model-checking.

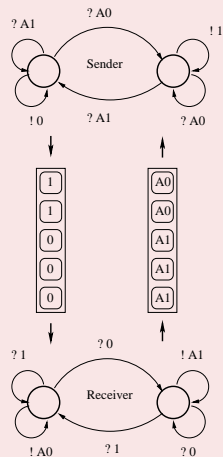
Model-checking finite systems (1)

finite system

- n finite variables
 $(x_1, \dots, x_n) \in F_1 \times \dots \times F_n$
- a finite control structure
 - set Q of locations q_1, \dots, q_m
 - finite set of transitions
 $(Q \times (F_1 \times \dots \times F_n))^2$

Example

booleans, enumerations, bounded queues, bounded integers, ...



Model-checking finite systems (2)

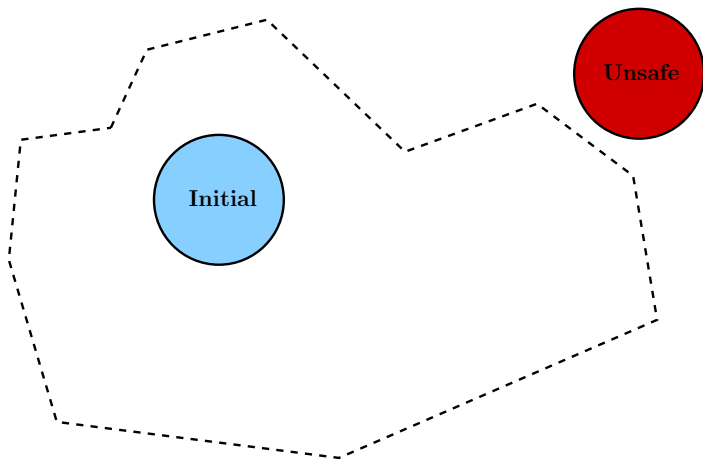
Reachability set

- configurations $c \in Q \times (F_1 \times \dots \times F_n)$
- $c' \in \text{post}(c)$ iff there exists a transition T such that cTc' .
- $c' \in \text{post}^*(c)$ iff there exist T_1, \dots, T_n such that $cT_1c_1T_2c_2 \dots T_nc'$.
- $\text{post}^*(X_0)$ is the reachability set from initial configuration X_0 .

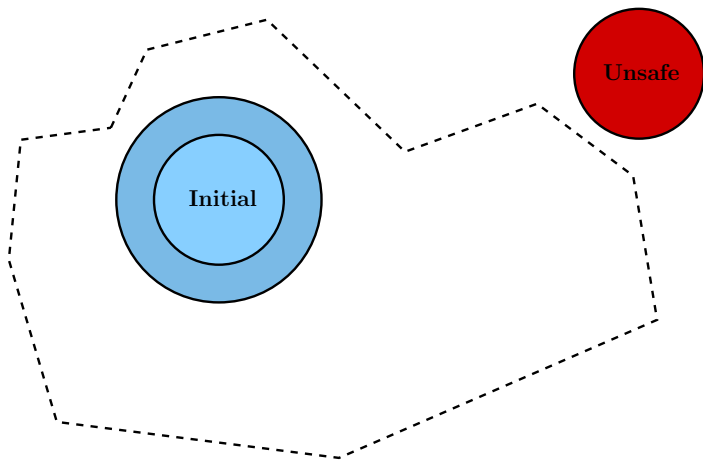
Safety properties = properties on reachable configurations.

- interesting: mutual exclusion, deadlock freedom, ...
- can be checked easily from the reachability set.

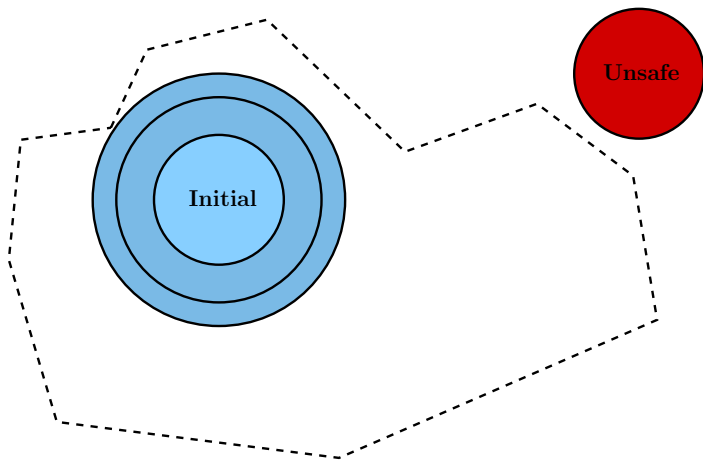
Model-checking finite systems (3)



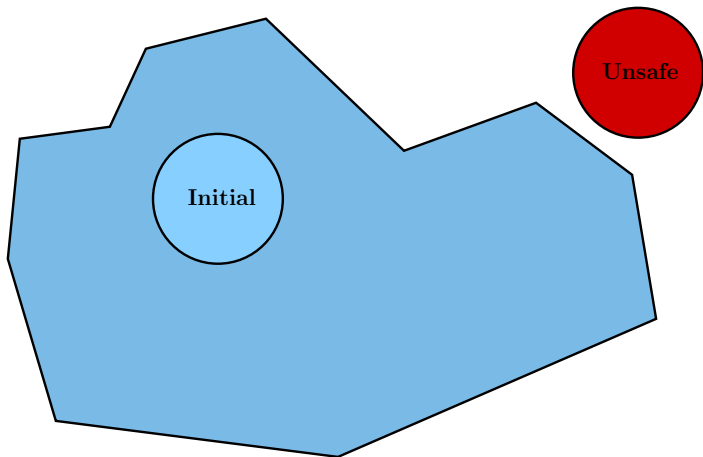
Model-checking finite systems (3)



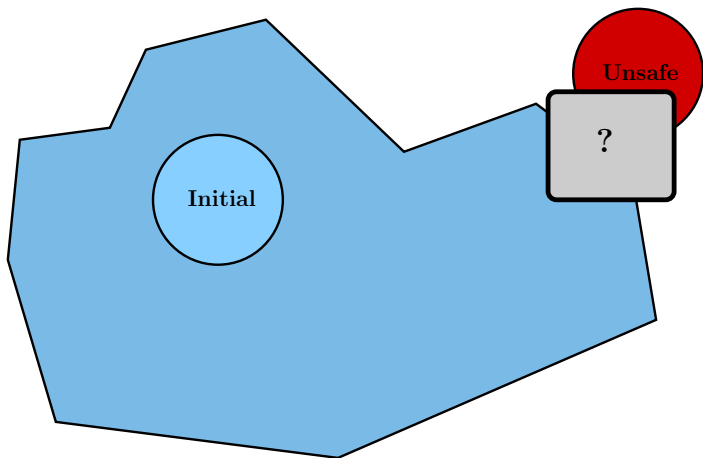
Model-checking finite systems (3)



Model-checking finite systems (3)



Model-checking finite systems (3)



Model-checking finite systems (4)

Model checking finite state systems

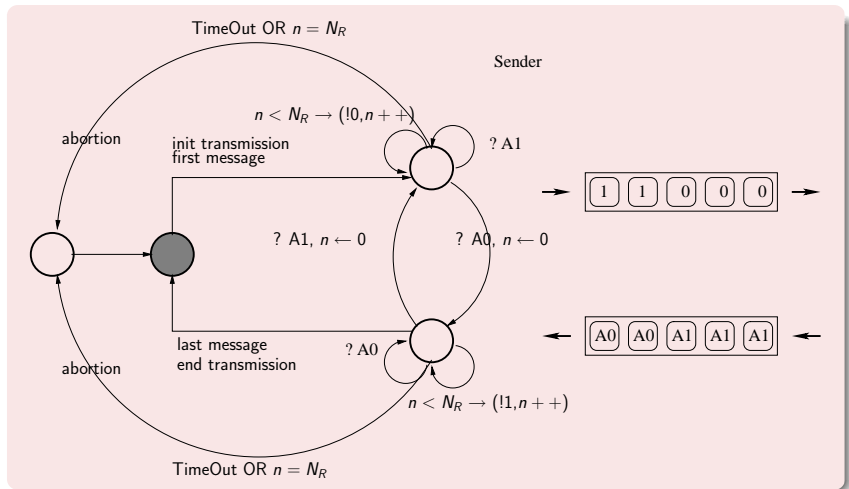
Model-checking is a very successful approach for systems with a finite state space.

- Build the state space, then decide the property.
- Automatic.

In practice

- Efficient tools
- Industrial use.

The Bounded Retransmission Protocol



Finite systems are not realistic enough

New generation of complex protocols

Bounded Retransmission Protocol, CES Philips, TTP ...

Difficult because of

- 1 data ranging over infinite domains (integers, clocks, queues, etc.)
- 2 heterogeneous data types
- 3 parametric reasoning (ex: number of agents)

Goal

We focus on systems with

- 1 infinite state space
- 2 **different sources of infinity.**

We do not consider infinite control structure.

existing work

- 1 one cause of infinity: many works (see after)
- 2 **many causes of infinity: ??**

Outline

- 1 Motivations.
- 2 Tackling infinite state space.
- 3 Tackling heterogeneity.
- 4 Acceleration and heterogeneity.
- 5 Weak heterogeneous systems.
- 6 Formal framework.
- 7 Symbolic representations and heterogeneity.
- 8 Symbolic representations for better composition.
- 9 Conclusion and Perspectives.

Extend model-checking to infiniteness

Several ways to handle infinite systems

Decidable subclasses

- Rarely expressive enough
- OR decidable procedures are too expensive (Petri Nets, timed automata)

Other approach

- finite abstractions
- semi-algorithms, efficient in practice (widening, automated abstractions)

Symbolic model-checking

Infinite reachability set is one of the problems

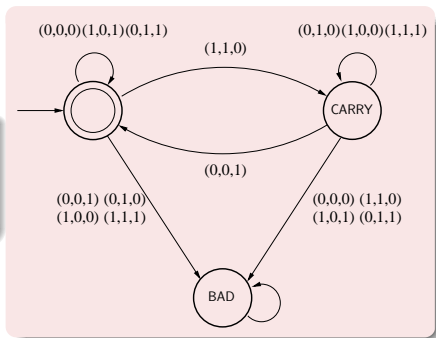
Solution

- manipulate symbolic configurations, i.e. infinite sets of configurations.
- symbolic configurations are encoded by symbolic representations.
- symbolic representations must provide symbolic operations for post, \cup , \subseteq .

Symbolic representations

Systems with counters

- automata (NDDs, UBAs)
- polyhedra



Symbolic representations

perfect queues

- 1 regular expressions (QDDs)
- 2 semi-linear regular expressions (SLRE)
- 3 constraint SLRE (CQDDs)

1 $(a + bc^*d)^*$

2 $u_1 w_1^* \dots u_n w_n^*$

3 $u_1 w_1^{\theta_1} \dots u_n w_n^{\theta_n}$ and
 $\varphi(\theta_1, \dots, \theta_n)$

Symbolic representations

lossy queues

simple regular expressions (SRE)

$$(a_1 + \varepsilon)b_1^* \dots (a_n + \varepsilon)b_n^*$$

Symbolic representations

Hybrids systems

- 1 difference bound matrices (DBMs)
- 2 constraint difference bound matrices (CPDBM)
- 3 weak Büchi automata (RVAs)
- 4 polyhedra

1

0	$-x \leq c_1$	$-y \leq c_2$
$x \leq c_3$	0	$x - y \leq c_4$
$y \leq c_5$	$y - x \leq c_6$	0

2

0	$-x \leq p_1$	$-y \leq p_2$
$x \leq p_3$	0	$x - y \leq p_4$
$y \leq p_5$	$y - x \leq p_6$	0

and $\varphi(p_1, \dots, p_6)$

Limits

Problem

The iterative firing of transitions does not converge to the reachability set in general.

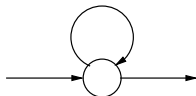
Acceleration

Acceleration

Compute in one step the infinite iteration of some transitions of the system.

- exact widening
- computation of the transitive closure of a transition

If $x \geq 0$ then $x \leftarrow x + 2$



Standard iteration

If $S_0 = \{0\}$ then

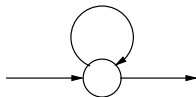
Acceleration

Acceleration

Compute in one step the infinite iteration of some transitions of the system.

- exact widening
- computation of the transitive closure of a transition

If $x \geq 0$ then $x \leftarrow x + 2$



Standard iteration

If $S_0 = \{0\}$ then $Reach \supseteq \{0\}$.

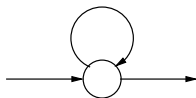
Acceleration

Acceleration

Compute in one step the infinite iteration of some transitions of the system.

- exact widening
- computation of the transitive closure of a transition

If $x \geq 0$ then $x \leftarrow x + 2$



Standard iteration

If $S_0 = \{0\}$ then $Reach \supseteq \{0, 2\}$.

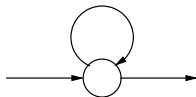
Acceleration

Acceleration

Compute in one step the infinite iteration of some transitions of the system.

- exact widening
- computation of the transitive closure of a transition

If $x \geq 0$ then $x \leftarrow x + 2$



Standard iteration

If $S_0 = \{0\}$ then $Reach \supseteq \{0, 2, 4\}$.

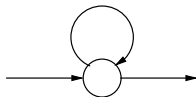
Acceleration

Acceleration

Compute in one step the infinite iteration of some transitions of the system.

- exact widening
- computation of the transitive closure of a transition

If $x \geq 0$ then $x \leftarrow x + 2$



Standard iteration

If $S_0 = \{0\}$ then *Reach* $\supseteq \{0, 2, \dots, 2.k\}$ and so on

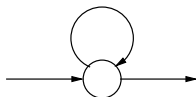
Acceleration

Acceleration

Compute in one step the infinite iteration of some transitions of the system.

- exact widening
- computation of the transitive closure of a transition

If $x \geq 0$ then $x \leftarrow x + 2$



With **acceleration**

If $S_0 = \{0\}$ then

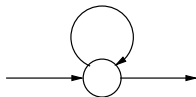
Acceleration

Acceleration

Compute in one step the infinite iteration of some transitions of the system.

- exact widening
- computation of the transitive closure of a transition

If $x \geq 0$ then $x \leftarrow x + 2$



With **acceleration**

If $S_0 = \{0\}$ then $Reach = 2.\mathbb{N}$ in one step.

State of the art

A promising approach

- Acceleration for numeric variables, clocks, communication channels
- Tools: FAST, LASH, TRES.
- Many encouraging case-studies.

Problem

- no general theory of acceleration
- acceleration algorithms deeply depend of the symbolic representation
- defining symbolic representations and accelerations are difficult tasks.

Outline

- 1 Motivations.
- 2 Tackling infinite state space.
- 3 Tackling heterogeneity.
- 4 Acceleration and heterogeneity.
- 5 Weak heterogeneous systems.
- 6 Formal framework.
- 7 Symbolic representations and heterogeneity.
- 8 Symbolic representations for better composition.
- 9 Conclusion and Perspectives.

Related work on heterogeneity

Very few work has been conducted in this direction.

Example

- specific approach (counters and clocks, stacks and counters) with *dedicated* acceleration
- the Composite Symbolic Library, or algebraic BDDs: nice methods for post, but acceleration is not investigated.
- upper approximation of post* by Cartesian product (TReX)

Outline

- 1 Motivations.
- 2 Tackling infinite state space.
- 3 Tackling heterogeneity.
- 4 **Acceleration and heterogeneity.**
- 5 Weak heterogeneous systems.
- 6 Formal framework.
- 7 Symbolic representations and heterogeneity.
- 8 Symbolic representations for better composition.
- 9 Conclusion and Perspectives.

Problem

Heterogeneous systems

- many basic data types interacting between each others
- symbolic representations and accelerations are available for each data types

Problem

We do not know how to use the knowledge on basic data types to build easily symbolic representations and acceleration algorithms on the whole system.

Main issue

Composition of accelerations

Given symbolic representations and accelerations for different data types \mathcal{D}_1 and \mathcal{D}_2 , we want to deduce a symbolic representation and an acceleration for $\mathcal{D}_1 \times \mathcal{D}_2$

Results

- we show that the cartesian product does not conserve acceleration,
- we give a construction for a specific subclass of symbolic representations,
- this subclass subsumes works on integers, clocks, stacks and queues.

Outline

- 1 Motivations
- 2 Tackling infinite state space.
- 3 Tackling heterogeneity.
- 4 Acceleration and heterogeneity.
- 5 **Weak heterogeneous systems.**
- 6 Formal framework.
- 7 Symbolic representations and heterogeneity.
- 8 Symbolic representations for better composition.
- 9 Conclusion and Perspectives.

Transition systems

Definition (transition system)

A transition system is a pair (D, \rightarrow^*) where D is a set (the domain) and $\rightarrow^* \subseteq D \times D$ is the global reachability relation of the system.

finitely presented

Usually \rightarrow^* is *finitely presented* by a finite number of relations $\mathcal{R} = \{r_1, \dots, r_m\}$ such that $\rightarrow^* = (r_1, \dots, r_m)^*$. We write (D, \mathcal{R}) for (D, \rightarrow) .

heterogeneous

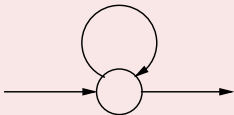
A transition system is *heterogeneous* if there exist $n \geq 2, k_1, \dots, k_n \in \mathbb{N}$ and n different sets \mathcal{D}_i such that $\mathcal{D} = \mathcal{D}_1^{k_1} \times \dots \times \mathcal{D}_n^{k_n}$ written $\times \mathcal{D}_i^{k_i}$.

Weak heterogeneous systems

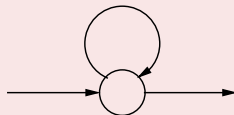
Weak heterogeneous systems

- *Data types are strongly encapsulated.* Each data has its own operations, and the whole system is built combining these operations.
- *Consistent with modular or object oriented design.*

If $x \geq 0$ then $x \leftarrow x + 2$



If ? a then ! b

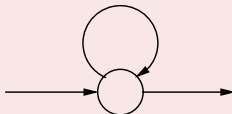


Weak heterogeneous systems

Weak heterogeneous systems

- *Data types are strongly encapsulated.* Each data has its own operations, and the whole system is built combining these operations.
- *Consistent with modular or object oriented design.*

If $(x \geq 0 \text{ and } ?a)$ then $(x \leftarrow x + 2, !b)$



Weak heterogeneous systems

Definition (Formally)

Let $\mathcal{H} = (\times \mathcal{D}_i^{k_i}, \mathcal{R})$ be a finitely presented heterogeneous system. \mathcal{H} is weakly heterogeneous w.r.t. $\times \mathcal{D}_i^{k_i}$ if there exist n \mathcal{R}_i finite sets of relations over $\mathcal{D}_i^{k_i} \times \mathcal{D}_i^{k_i}$ such that $\mathcal{R} \subseteq \times \mathcal{R}_i$. We write $\mathcal{H} = (\times \mathcal{D}_i^{k_i}, \mathcal{R}, \times \mathcal{R}_i)$.

Modeling power

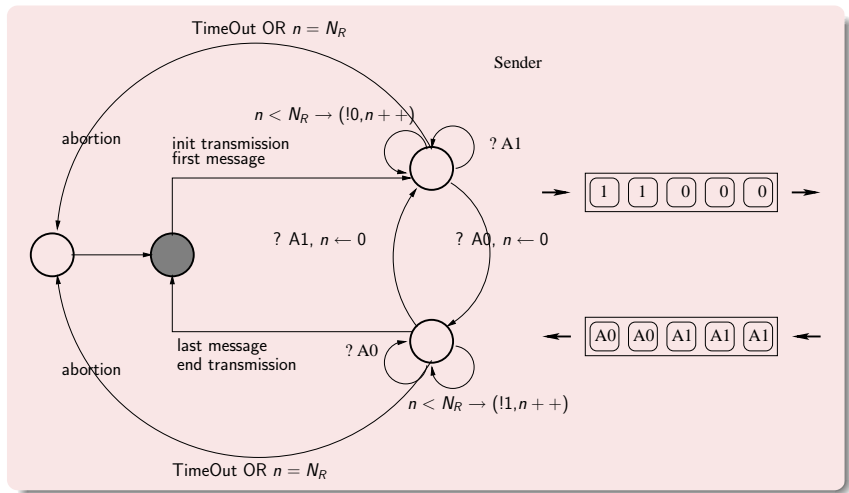
Example

Can model communication protocols using channels with finite sets of messages, maximum number of reemissions and clocks for abortion (ABP, BRP, ...)

Limit

Cannot model writing the value of a counter into a queue, since it implies *mixing the structures of the data types, and not only their operations.*

Modeling power



Outline

- 1 Motivations
- 2 Tackling infinite state space.
- 3 Tackling heterogeneity.
- 4 Acceleration and heterogeneity.
- 5 Weak heterogeneous systems.
- 6 **Formal framework.**
- 7 Symbolic representations and heterogeneity.
- 8 Symbolic representations for better composition.
- 9 Conclusion and Perspectives.

Symbolic representations

Definition (symbolic representation)

A symbolic representation for a finitely presented transition system $\mathcal{H} = (D, \mathcal{R})$ is a 5-uplet $\mathcal{S} = (C, \gamma, \sqcup, \sqsubseteq, \text{post})$ where

- C the set of symbolic states (configurations),
- $\gamma : C \rightarrow \mathcal{P}(\mathcal{D})$ the concretization function,
- \sqcup and \sqsubseteq symbolic union and inclusion on C ,
- and $\text{post} : \mathcal{R} \times C \rightarrow C$ the symbolic successor,

- 1 $\gamma(c_1 \sqcup c_2) = \gamma(c_1) \cup \gamma(c_2)$ (consistency of union);
- 2 $\gamma(\text{post}(r, c)) = r(\gamma(c))$ (consistency of post);
- 3 $c_1 \sqsubseteq c_2 \Rightarrow \gamma(c_1) \subseteq \gamma(c_2)$ (consistency of inclusion).

Some examples of symbolic representations

Example

- numeric variables: UBAs (\mathbb{N}), NDDs (\mathbb{Z}) and RVAs (\mathbb{R}).
- clocks and counters: CPDBM, RVAs.
- perfect fifo queues, stacks: QDDs , SLRE and CQDDs.
- lossy fifo channels: SRE
- ...

Symbolic reachability set problem

Definition (symbolic reachability set problem)

- *Input*
 - a finitely presented transition system $\mathcal{H} = (\mathcal{D}, \mathcal{R})$,
 - $\mathcal{S} = (\mathcal{C}, \gamma, \sqcup, \sqsubseteq, \text{post})$ a symbolic representation for \mathcal{H} ,
 - an initial symbolic state $c_0 \in \mathcal{C}$,
- *Output:* $c' \in \mathcal{C}$ such that $\mathcal{R}^*(\gamma(c_0)) = \gamma(c')$ if it exists, no otherwise.

Acceleration function

Definition (acceleration function)

Consider $\mathcal{H} = (\mathcal{D}, \mathcal{R})$ a finitely presented transition system and $\mathcal{S} = (\mathcal{C}, \gamma, \sqcup, \sqsubseteq, \text{post})$ a symbolic representation for \mathcal{H} . An acceleration function for $(\mathcal{H}, \mathcal{S})$ is a computable totally defined function $\text{post}^* : \mathcal{R}^* \times \mathcal{C} \rightarrow \mathcal{C}$ such that

$$\forall r, c \in \mathcal{R}^* \times \mathcal{C}, \gamma(\text{post}^*(r, c)) = r^*(\gamma(c))$$

Outline

- 1 Motivations
- 2 Tackling infinite state space.
- 3 Tackling heterogeneity.
- 4 Acceleration and heterogeneity.
- 5 Weak heterogeneous systems.
- 6 Formal framework.
- 7 **Symbolic representations and heterogeneity.**
- 8 Symbolic representations for better composition.
- 9 Conclusion and Perspectives.

First results

Cartesian product of symbolic representations

We define the Cartesian product of $\mathcal{S}_1 = (\mathcal{C}_1, \gamma_1, \sqcup_1, \sqsubseteq_1, \text{post}_1)$ and $\mathcal{S}_2 = (\mathcal{C}_2, \gamma_2, \sqcup_2, \sqsubseteq_2, \text{post}_2)$ by

$$\mathcal{S}_1 \times \mathcal{S}_2 = (\mathcal{P}_f(\mathcal{C}_1 \times \mathcal{C}_2), \gamma_1 \times \gamma_2, \sqcup, \sqsubseteq_1 \times \sqsubseteq_2, \text{post}_1 \times \text{post}_2).$$

Theorem

If \mathcal{S}_1 is a symbolic representation for $(\mathcal{D}_1^{k_1}, \mathcal{R}_1)$ and \mathcal{S}_2 is a symbolic representation for $(\mathcal{D}_2^{k_2}, \mathcal{R}_2)$ then $\mathcal{S}_1 \times \mathcal{S}_2$ is a symbolic representation for all weak heterogeneous systems $(\mathcal{D}_1^{k_1} \times \mathcal{D}_2^{k_2}, \mathcal{R}, \mathcal{R}_1 \times \mathcal{R}_2)$.

Result used in the Composite Symbolic Library (ALV).

About acceleration

Does not hold for acceleration

The Cartesian product of acceleration is not an acceleration.

Proof.

$$\textcircled{1} \quad r_1^*(d_1) \times r_2^*(d_2) = \bigcup_{i \in \mathbb{N}} \bigcup_{j \in \mathbb{N}} r_1^i(d_1) \times r_2^j(d_2) \quad (1)$$

$$\textcircled{2} \quad (r_1 \times r_2)^*(d_1, d_2) = \bigcup_{k \in \mathbb{N}} r_1^k(d_1) \times r_2^k(d_2) \quad (2)$$

$$\textcircled{3} \quad (2) \subseteq (1) \text{ but } (2) \neq (1) \text{ in general}$$



Outline

- 1 Motivations
- 2 Tackling infinite state space.
- 3 Tackling heterogeneity.
- 4 Acceleration and heterogeneity.
- 5 Weak heterogeneous systems.
- 6 Formal framework.
- 7 Symbolic representations and heterogeneity.
- 8 **Symbolic representations for better composition.**
- 9 Conclusion and Perspectives.

Ideas

The previous proof relies on the non synchronization of the numbers of iterations.

Idea

- The symbolic representation must model explicitly the number of iterations.
- Then we define a variant of Cartesian product, synchronizing the representations of iterations.

Presburger symbolic representation

Definition (Presburger symbolic representation)

A Presburger symbolic representation for \mathcal{H} is a symbolic representation $\mathcal{S}p = (Cp, \gamma, \sqcup, \sqsubseteq, \text{post})$ such that:

- Cp is a set of 2-uplets $cp = (w, \Phi(\bar{w}))$ with:
 - w is a word over a language \mathcal{L} ,
 - \bar{w} is a finite set of variables associated to $w \in \mathcal{L}$,
 - $\Phi(\bar{w})$ is a Presburger formula whose free variables are in \bar{w} .
- other hypothesis on γ and post

Presburger symbolic representation

Example (Common in practice)

Following representations are OK

- NDDs, UBAs (counters)
- RVAs (counters and clocks)
- CPDBMs (counters and clocks)
- CQDDs (queues, stacks)

BUT not the following ones ...

- SRE
- SLRE
- QDDs
- convex polyhedra

Presburger symbolic representation

SLRE

- 1 $u_1 w_1^* \dots u_n w_n^*$
- 2 **NO**, cause θ does not appear.

CQDD

- 1 $u_1 w_1^{k_1} \dots u_n w_n^{k_n}$ and $\varphi(k_1, \dots, k_n)$
- 2 **YES**, cause θ will appear in φ .

Counting acceleration

Definition (Counting acceleration)

A counting acceleration for (\mathcal{H}, Sp) is an acceleration function post^* for (\mathcal{H}, Sp) such that

- $(w, \Phi(\bar{w})) \xrightarrow{\text{post}^* r} (w', \exists \theta \in \mathbb{N}. \exists \bar{w}. \Phi(\bar{w}) \wedge \varphi(\bar{w}, \bar{w}', \theta))$
- θ is the number of iterations
- $\exists \bar{w}. \Phi(\bar{w})$ keeps the old constrains
- φ links old (\bar{w}) , new (\bar{w}') and iterations θ .
- (w', φ) depends only of r and w ;
- one more hypothesis about γ ...

Counting acceleration

Example (Common in practice)

Standard accelerations for the following representations are counting:

- NDDs, UBAs (counters)
- RVAs (counters and clocks)
- CPDBMs (counters and clocks)
- CQDDs (queues, stacks)

Synchronized product

Synchronized product of Presburger symbolic representations

- Input: $cp_1 = (w_1, \Phi_1(\overline{w_1}))$ and $cp_2 = (w_2, \Phi_2(\overline{w_2}))$
- Output: $cp_1 \otimes cp_2 = (w_1, w_2, \Phi(\overline{w_1}, \overline{w_2}))$.

\otimes is extended to $\mathcal{S}p_1$ and $\mathcal{S}p_2$.

We can define an acceleration for it.

Synchronized acceleration

A counting acceleration is defined by

$$\text{post}^* : w_1 \xrightarrow{r} (w', \varphi(\overline{w}, \overline{w}', \theta)).$$

Synchronized product of counting accelerations

- Input:

- 1 $\text{post}^*_1 : w_1 \xrightarrow{r_1} (w'_1, \varphi_1(\overline{w}_1, \overline{w}'_1, \theta)),$

- 2 $\text{post}^*_2 : w_2 \xrightarrow{r_2} (w'_2, \varphi_2(\overline{w}_2, \overline{w}'_2, \theta))$

- Output

$$\text{post}^*_1 \otimes \text{post}^*_2 :$$

$$(w_1, w_2) \xrightarrow{r_1 \times r_2} ((w'_1, w'_2), \varphi_1(\overline{w}_1, \overline{w}'_1, \theta) \wedge \varphi_2(\overline{w}_2, \overline{w}'_2, \theta))$$

Synchronized product and Acceleration

Theorem

Let $\mathcal{H} = (\times \mathcal{D}_i^{k_i}, \mathcal{R} \subseteq \times \mathcal{R}_i)$ a weak heterogeneous system. Assume that for all i , there exists Sp_i a Presburger symbolic representation for $\mathcal{H}_i = (\mathcal{D}_i^{k_i}, \mathcal{R}_i)$ and post^*_i a counting acceleration for (\mathcal{H}_i, Sp_i) . Then

- $\bigotimes Sp_i$ is a Presburger symbolic representation for \mathcal{H} ,
- $\bigotimes \text{post}^*_i$ is a counting acceleration for $(\mathcal{H}, \bigotimes Sp_i)$.

Existing symbolic representations

	data	counting acceleration?
SRE	lossy queues	no
QDD	queues/stacks	no
SLRE	queues/stacks	no
CQDD	queues/stacks	yes
UBA/NDD	counters	yes
RVA	clocks and counters	yes
CPDBMs	clocks and counters	semi-decidable

Theorem

A symbolic representation and an acceleration function can be computed automatically for weak heterogeneous systems manipulating counters, clocks, perfect FIFO queues and stacks.

Conclusion

Summary

- Heterogeneity of systems has hardly been investigated.
- We propose a generic approach to build automatically symbolic representations and accelerations from existing ones.

Results

- Our framework recovers many existing work,
- it can be used to quickly verify systems with many data types.

Perspective

- heuristics and termination results from the projections on each data type,
- theoretical work on getting more efficient combinations,
- exact computation imposes a heavy framework, relax it and check for a best upper-approximation?

A generic tool for symbolic model-checking with acceleration

- generic modules for symbolic representations and accelerations,
- a module implementing the synchronized product.

Toward a generic tool

