Composition of accelerations to verify infinite heterogeneous systems

Sébastien Bardin and Alain Finkel

LSV, CNRS UMR 8643
ENS de Cachan
61 avenue du président Wilson
F-94235 CACHAN Cedex
FRANCE
{bardin,finkel}@lsv.ens-cachan.fr

Abstract. Symbolic representations and acceleration algorithms are emerging methods to extend model-checking to infinite state space systems. However until now, there is no general theory of acceleration, and designing acceleration algorithms for new data types is a complex task. On the other hand, protocols rarely manipulate new data types, rather new combinations of well-studied data types. For this reason, in this paper we focus on the automatic construction of symbolic representations and acceleration algorithms from existing ones.

Keywords: reachability set, unbounded heterogeneous data, composition of symbolic representations and acceleration methods.

1 Introduction

Context. Automatic verification of systems is a major issue considering the growing computerization of the society. At the same time, software based systems become more and more complex to verify, because of 1) unbounded data types 2) parametric reasoning 3) heterogeneous data interacting between each other. Standard model-checking techniques, well-suited to the verification of systems manipulating data ranging over finite domains, cannot handle these new challenges.

Unbounded data and parametric reasoning. During the past decade, promising methods have emerged to extend model-checking to infinite systems: 1) symbolic representations to represent infinite sets 2) widening or acceleration techniques to compute in one step infinite behaviors of the system. For several data types, symbolic representations with acceleration exist, tools are available and encouraging case-studies have been conducted: integers [Boi98,BFLP03,LAS], reals [AAB00,BJW01,BHJ03], lossy [ABJ98] or perfect [BGWW97,BH99,FPS03] channels, etc.

Heterogeneous data. On the other hand, very few work has been conducted on data heterogeneity. Of course, some approaches listed above consider heterogeneous variables. However none of these approaches consider data heterogeneity in general, but only particular combinations of data types. At the same time real systems naturally use many data types. For example, to model a communication protocol, we need words for the channel content while the maximum number of reemissions or clocks to abort connection require numeric variables. None of the specific symbolic representations listed above can handle all these features, *even if they are all available individually.* So when considering a system combining data types for which symbolic representations and accelerations are well known, we still have to design ad hoc symbolic representation and acceleration. This is neither trivial nor satisfactory. An interesting approach would be to *automate the construction of the symbolic representations and the acceleration algorithm* from the existing ones.

Related work. The Composite Symbolic Library [YKTB01] makes an attempt, but it considers only computing symbolic representations sound for post operation, not for acceleration which is much more difficult. Indeed in their specific case, the Cartesian product of representations is sufficient. Actually the library contains booleans (BDDs) and counters (convex sets or finite automata). TReX [ABS01] combines accelerations on lossy channels and numeric variables. However since they use a Cartesian product, they just obtain an upper approximation of the acceleration. Some approaches [Tav04] aim at making different tools working together. It is a combination at a very high level (tool), not the symbolic representation level. In [BH99], acceleration of different FIFO queues is done using synchronization between iteration variables. Here we extend the method and give a general framework for any data types, symbolic representations and accelerations matching some hypothesis. Finally [Bou01] proposes a framework for the verification of infinite systems based on languages and rewriting systems. This framework recovers accelerations presented here for queues and stacks, as well as accelerations for parameterized networks of identical processes. However combination of algorithms is not explored.

Our results. In this paper we explore the construction of new acceleration algorithms for heterogeneous data, from acceleration available on each data types. We propose a generic framework, based on the notions of *weak heterogeneous systems*, *Presburger symbolic representations* and *counting accelerations* to build automatically new symbolic representation and acceleration from existing ones.

1. First we show that for some reasonable subclasses of heterogeneous systems (*weak heterogeneous*), the Cartesian product of symbolic representations of each data type of the system is a good symbolic representation for the system. However the Cartesian product of accelerations is just an upper approximation.

2. Then we propose a subclasses of symbolic representations (*Presburger symbolic representation*) and acceleration (*counting acceleration*) for which the

construction of a symbolic representation and an acceleration algorithm of the Cartesian product can be automated. We then review existing symbolic representations and accelerations, showing our framework includes many existing work. Presburger symbolic representations and counting accelerations already exist for *unbounded counters, reals/clocks, perfect FIFO queues and stacks*.

3. Finally we discuss the architecture of a model-checker for infinite systems with arbitrary heterogeneous data. For any arbitrary weak heterogeneous system, data types are identified and specific symbolic representation and acceleration are automatically built from the ones available in a library. Then a generic semi-algorithm is used to compute the reachability set of the system. Checking safety properties is straightforward. So the user has only to provide symbolic representations and accelerations for data types *not available* in the library. All the rest is automatically done. This approach does not pretend to be more efficient than dedicated symbolic representations and algorithm, but it allows to reuse a maximum of work.

Outline. The rest of the paper is structured as follows. Section 2 lists some existing symbolic representations, used in the rest of the paper to illustrate the definitions. Section 3 provides basic definitions of heterogeneous systems and symbolic representations. Section 4 defines interesting subclasses of symbolic representations and accelerations techniques, from which acceleration on new data types can be automatically computed. Section 5 shows how these results can be used to design a tool for the verification of heterogeneous systems with arbitrary (unbounded) data types.

2 Existing symbolic representations

2.1 Notations

$\mathbb{N}, \mathbb{Z}, \mathbb{R}$ are the sets of non negative integers, integers and reals. For a set S, $\mathcal{P}(S)$ denotes the powerset of S, while $\mathcal{P}_f(S)$ denotes the set of finite subsets of S.

Let r_1, r_2 be two relations over $P_1 \times Q_1$ and $P_2 \times Q_2$. We define $r_1 \times r_2 \subseteq (P_1 \times P_2) \times (Q_1 \times Q_2)$ as $r_1 \times r_2 = \{((p_1,p_2),(q_1,q_2)), (p_1,q_1) \in r_1 \wedge (p_2,q_2) \in r_2\}$. The image of r_1, written $Im(r_1)$, is Q_1.

Let Σ be a finite alphabet. A linear regular expression (LRE) on Σ is a regular expression of the form $u_1^* v_1 u_2^* v_2 \dots u_n^* v_n$ with $u_i, v_i \in \Sigma^*$. A semi-linear regular expression (SLRE) [FPS03] on Σ is a finite union of LREs. A simple regular expression (SRE) [ABJ98] over Σ is a sum of products of $(a_i + \varepsilon)$ and $(a_1 + \dots + a_n)^*$ with $a_i \in \Sigma^*$.

2.2 Presburger arithmetics

Presburger arithmetics is the first order theory of $\mathbb{N}^m$, with predicates $\leq$ and $+$, denoted $< \mathbb{N}^m, \leq, + >$. Presburger arithmetics is decidable, and sets over $\mathbb{N}^m$ defined by a Presburger formula can be represented and manipulated by finite automata [BC96]. These constructions can be extended to $\mathbb{Z}$ and $\mathbb{R}$ (using weak deterministic Büchi automata for the real case). Actually weak deterministic Büchi automata represents more than Presburger arithmetics. They can encode the theory $< \mathbb{R}, \leq, +, \mathbb{Z} >$ [BJW01], where $\mathbb{Z}$ is a predicate stating if a real is an integer or not. In the following sections, we will write Presburger arithmetics on $\mathbb{R}$ for $< \mathbb{R}, \leq, +, \mathbb{Z} >$. Identifying a Presburger formula with the set of its solutions, we write $x \in \varphi$ instead of $x \models \varphi$.

2.3 Symbolic representations

We present here briefly several symbolic representations. This list does not attempt to be exhaustive. It will be used throughout the paper to show our general framework captures many existing works.

Boolean. BDDs [Bry92] is a compact representation for (large) sets of boolean values, using deterministic acyclic graphs.

Numeric variables. Convex polyhedra [HPR97] can be used to represent sets of reals, but the representation is not closed by union, and usually upper approximation are made (convex hull). NDDs [WB00]/UBAs [Ler03] is a symbolic representation of integers by automata. RVAs [BJW01] uses a similar encoding for reals. DBMs [Dil89] is a symbolic representation for clocks. CPDBM [AAB00] is an extension of DBM to handle unbounded integers. Basically the DBM is extended with parameters and (possibly non-linear) constraints.

Perfect FIFO Queues/stacks. QDDs [BGWW97] represents the content of a perfect FIFO queue by a regular expressions on the (finite) alphabet of messages. SLRE [FPS03] is based on same ideas, but they use SLRE instead of regular expressions. CQDDs [BH99] can be seen as a SLRE with variables attached to each '*' and a Presburger formula linking the variables. It is straightforward to use these representations for stacks [Bou01].

Lossy FIFO Queues. [ABJ98] uses Simple Regular Expressions (SRE) to represent the content of lossy FIFO channels.

3 Combination of symbolic representations

In this section, we handle the following issue: given a system with heterogeneous data types, and assuming we have symbolic representations for each data type, can we obtain easily a symbolic representation for the whole system? The answer is yes. However the method does not work anymore for acceleration.

3.1 Preliminaries

A *transition system* is a pair $(D, \rightarrow)$ where D is a set (the domain) and $\rightarrow \subseteq D \times D$ is the transition relation. In general, the relation $\rightarrow$ is not recursive. However in the following sections we will always suppose that $\to$ is *finitely presented* by a finite set of recursive relations over $D \times D$. Formally we suppose that there exist $m \geq 0$ recursive relations $r_i \subseteq D \times D$ such that $\to = (r_1, \ldots, r_m)^*$. Let $\mathcal{R} = \{r_1, \ldots, r_m\}$. A finitely presented transition system, shortly *a system*, will be written $(D, \mathcal{R})$ instead of $(D, \to)$.

Remark 1. Any transition system represented by a control graph labeled by actions on a finite set of variables follows this definition. $\mathcal{R}$ is then the set of transitions. In this paper we are mainly interested in systems manipulating heterogeneous variables, i.e. systems whose domain D can be written as a Cartesian product of elementary domains D_i. In the following sections, $D_1^{k_1} \times \ldots \times D_n^{k_n}$ is written $\times D_i^{k_i}$.

Definition 1 (symbolic representation). *A symbolic representation for a system $\mathcal{H} = (D, \mathcal{R})$ is a 5-tuple $\mathcal{S} = (\mathsf{S}, \gamma, \sqcup, \sqsubseteq, \mathsf{post})$ where S is the set of symbolic states, the concretization function $\gamma : \mathsf{S} \to \mathcal{P}(D)$ maps to each $\mathsf{s} \in \mathsf{S}$ a set of concrete states in D, $\sqcup$ and $\sqsubseteq$ are union and inclusion on S and $\mathsf{post} : \mathcal{R} \times \mathsf{S} \to \mathsf{S}$ computes the successors of a symbolic state, satisfying:*

1. $\gamma(\mathsf{s}_1 \sqcup \mathsf{s}_2) = \gamma(\mathsf{s}_1) \cup \gamma(\mathsf{s}_2)$ *(consistency of union);*
2. $\gamma(\mathsf{post}(r, \mathsf{s})) = r(\gamma(\mathsf{s}))$ *(consistency of* post*);*
3. $\mathsf{s}_1 \sqsubseteq \mathsf{s}_2 \Rightarrow \gamma(\mathsf{s}_1) \subseteq \gamma(\mathsf{s}_2)$ *(consistency of inclusion).*

A symbolic representation is *effective* if $\sqcup$ and post are computable and $\sqsubseteq$ is decidable.

Some examples. For example, UBAs, NDDs and RVAs are effective symbolic representations for $\mathbb{N}, \mathbb{Z}$, and $\mathbb{R}$ with Presburger relations. QDDs , SLRE and CQDDs are effective symbolic representations for perfect FIF0 queues or stacks. SRE is an effective symbolic representation for lossy FIFO channels. DBMs is an effective symbolic representation for clocks and bounded counters, CPDBM is a symbolic representation for clocks and counters, but it is not effective because basic operations require to solve non-linear arithmetics formula.

Definition 2 (Reachability set problem). *Given a system $\mathcal{H} = (D, \mathcal{R})$, let $\mathcal{S} = (\mathsf{S}, \gamma, \sqcup, \sqsubseteq, \mathsf{post})$ be a symbolic representation for $\mathcal{H}$ and an initial symbolic state $\mathsf{s}_0 \in \mathsf{S}$, the reachability set problem is to compute $\mathsf{s}' \in \mathsf{S}$ such that $\mathcal{R}^*(\gamma(\mathsf{s}_0)) = \gamma(\mathsf{s}')$.*

The reachability set problem is often undecidable. However semi-algorithms has been developed to solve it in practical cases. *Acceleration* allows to compute in one step the transitive closure of a relation. It differs from widening [CC77] in that widening aims at termination and loses exactness. Given a relation r and a set D, the acceleration of r applied to D is defined by $r^*(D) = \bigcup_{i \in \mathbb{N}} r^i(D)$.

Definition 3 (acceleration function). *Consider a system $\mathcal{H} = (D, \mathcal{R})$ and $\mathcal{S} = (\mathsf{S}, \gamma, \sqcup, \sqsubseteq, \mathsf{post})$ an effective symbolic representation for $\mathcal{H}$. An acceleration function for $(\mathcal{H}, \mathcal{S})$ is a computable totally defined function $\mathsf{post}^* : \mathcal{R}^* \times \mathsf{S} \to \mathsf{S}$ such that $\forall r, \mathsf{s} \in \mathcal{R}^* \times \mathsf{S}, \gamma(\mathsf{post}^*(r, \mathsf{s})) = r^*(\gamma(\mathsf{s}))$.*

Until now, there is no general theory of acceleration, and existing algorithms dramatically rely on the structure of the symbolic representation. There exist acceleration for SRE [ABJ98], QDDs [BGWW97] and SLRE [FPS03] with restricted operations on queues, CQDDs for queues [BH99] and stacks [Bou01], NDDs [Boi98] and UBAs [FL02] with restricted affine functions, RVAs [BHJ03] and restricted hybrid transitions. The acceleration defined on CPDBMs [AAB00] is not effective since it implies solving non-linear arithmetics formula.

We define here the class of systems, namely *flat systems*, for which having an acceleration function is sufficient to solve the reachability problem. Consider $\mathcal{R} = \{r_1, \dots, r_n\}$ a set of relations and $\bar{\mathcal{R}} = \{\bar{r_1}, \dots, \bar{r_n}\}$ its associated set of letters. We write $RegExp(\bar{\mathcal{R}})$ the set of regular expressions over $\bar{\mathcal{R}}$. We define inductively $\Psi : RegExp(\bar{\mathcal{R}}) \to \mathcal{R}^*$ by $\Psi(\emptyset) = \emptyset$, $\Psi(\varepsilon)$ is the identity relation, $\Psi(\bar{r_i} \cdot \bar{w}) = r_i \cdot \Psi(\bar{w})$, $\Psi(\bar{w_1} + \bar{w_2}) = \Psi(\bar{w_1}) \cup \Psi(\bar{w_2})$ and $\Psi(\bar{w}^*) = \Psi(\bar{w})^*$.

Definition 4 (flatness). *Consider a system $\mathcal{H} = (D, \mathcal{R})$ with $\mathcal{R} = \{r_1, \dots, r_n\}$. $\mathcal{H}$ is flat if there exists $L \in RegExp(\bar{\mathcal{R}})$ such that L is a SLRE and $\mathcal{R}^* = \Psi(L)$.*

Theorem 1. *Let $\mathcal{H} = (D, \mathcal{R})$ a flat system, $\mathcal{S} = (\mathsf{S}, \gamma, \sqcup, \sqsubseteq, \mathsf{post})$ a symbolic representation for $\mathcal{H}$ and assume there exists an acceleration function for $(\mathcal{H}, \mathcal{S})$. Then for any $\mathsf{s}_0 \in \mathsf{S}$, $\mathcal{R}^*(\mathsf{s}_0)$ is computable.*

Sketch of the proof.
Since $\mathcal{H} = (D, \mathcal{R})$ is flat, there exists $\bar{L}$ a SLRE on $\bar{\mathcal{R}}$ such that $\mathcal{R}^* = \Psi(\bar{L}) = L$. Then computing $\mathcal{R}^*(\mathsf{s}_0)$ comes down to computing $L(\mathsf{s}_0)$. Since $L = \Psi(\bar{L})$ and $\bar{L}$ is a SLRE over $\bar{\mathcal{R}}$ then L is of the form $\bigcup_i u_i$, where each u_i is of the form $u_i = u_{i_1}^* v_{i_1} \dots u_{i_{p_i}}^* v_{i_{p_i}}$, where all the u_{i_j}, v_{i_j} are relations in $\mathcal{R}$.

By hypothesis, S is closed by post for any function of $\mathcal{R}$ and post^* for any function of $\mathcal{R}^*$, and these are computable. Thus each $u_i(\mathsf{s}_0) = u_{i_1}^* \dots u_{i_p}^*(\mathsf{s}_0)$ exists and is computable. Then since S is closed by $\sqcup$, there exists $\mathsf{s}' \in \mathsf{S}$ such that $\mathsf{s}' = \sqcup_i \mathsf{s}_i = (\Psi(L))(\mathsf{s}_0) = \mathcal{R}^*(\mathsf{s}_0)$ and s' is computable. □

3.2 Weak heterogeneous systems

Definition 5 (weak heterogeneous systems). *Let $\mathcal{H} = (\times D_i^{k_i}, \mathcal{R})$ be a system. $\mathcal{H}$ is weakly heterogeneous if there exist n $\mathcal{R}_i$ finite sets of relations over $D_i^{k_i} \times D_i^{k_i}$ such that $\mathcal{R} \subseteq \times \mathcal{R}_i$. We write $\mathcal{H} = (\times D_i^{k_i}, \mathcal{R}, \times \mathcal{R}_i)$.*

Intuitively in a weak heterogeneous system, *data types are strongly encapsulated*. Each data has its own operations, and the whole system is built combining these operations. Since *this is consistent with modular or object oriented design*, weak heterogeneous systems are well-spread in practice.

For example, communication protocols using (possibly lossy) channels with finite sets of messages, (parameterized) maximum number of reemissions and clocks for abortion are weak heterogeneous systems. This includes many well-known protocols like the Alternating Bit Protocol, the Bounded Retransmission Protocol, the sliding window protocols, etc. Typical operations are of the form

```
(counter < MaxRetrans), (clock<MaxDelay) → (!mess), (counter++)
```
where counters, clocks and the channel are used with restricted interaction.

On the other hand, a weak heterogeneous system with queues and counters 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.*

In the following, we want to derive algorithms or properties on the whole system $\mathcal{H}$ from the study of projected systems $(D_i^{k_i}, \mathcal{R}_i)$.

Theorem 2. *Let* $\mathcal{H} = (\times D_i^{k_i}, \mathcal{R}, \times\mathcal{R}_i)$ *be a weak heterogeneous system. Assume that for all* $i \leq n$*, there exists* $(\mathsf{S}_i, \gamma_i, \sqcup_i, \sqsubseteq_i, \mathsf{post}_i)$ *an effective symbolic representation for* $(D_i^{k_i}, \mathcal{R}_i)$*. Then* $(\mathcal{P}_f(\times \mathsf{S}_i), \times\gamma_i, \sqcup, \times \sqsubseteq_i, \times\mathsf{post}_i)$ *is an effective symbolic representation for* $\mathcal{H}$*, where* $\sqcup$ *is the union operator on* $\mathcal{P}_f(\times\mathsf{S}_i)$*.*

Proof. We denote $\mathsf{s}_i \in \mathsf{S}_i$, $v = (\mathsf{s}_1, \ldots, \mathsf{s}_n) \in \times\mathsf{S}_i$, $p = \{v_1, \ldots, v_k\} \in \mathcal{P}_f(\times\mathsf{S}_i)$. With $\sqcup$, the *consistency of union (1)* is evident. *(2) consistency of* post. Let $r = \times r_i \in \mathcal{R}$. Then $r(\gamma(v)) = r(\times\gamma_i(\mathsf{s}_i))$. Since $\mathcal{H}$ is weak heterogeneous, it follows that $r(\gamma(v)) = \times r_i(\gamma_i(\mathsf{s}_i))$. Each $\mathcal{S}_i$ is a symbolic representation for $\mathcal{H}_i$. Then $r(\gamma(v)) = \times\gamma_i(\mathsf{post}_i(r_i, \mathsf{s}_i))$ which is equal to $\gamma(\times\mathsf{post}_i(r, \mathsf{s}))$. That proves the property for $\times\mathsf{S}_i$. The extension to $\mathcal{P}_f(\times\mathsf{S}_i)$ is straightforward. *(3) consistency of inclusion.* Assume that $v\times \subseteq_i v'$. Then by definition $\forall i, \mathsf{s}_i \subseteq_i \mathsf{s}'_i$. Each $\mathcal{S}_i$ is a symbolic representation for $\mathcal{H}_i$, then $\forall i, \gamma_i(\mathsf{s}_i) \subseteq_i \gamma_i(\mathsf{s}'_i)$. Thus $\times\gamma_i(v) \subseteq \times\gamma_i(v')$. The extension to $\mathcal{P}_f(\times\mathsf{S}_i)$ is defined by $p\times \subseteq_i p' \Leftrightarrow \forall i, \exists j, v_i \times \subseteq_i v'_j$. □

We just proved that, given arbitrary symbolic representation for certain domains and relations, their Cartesian product is a symbolic representation for the associated weak heterogeneous systems. A natural question is: does this hold also for acceleration? The answer is no. Indeed in general, *the Cartesian product of accelerations is an upper approximation.* For example TREX uses such approximation to combine counters/clocks with lossy FIFO channels.

Theorem 3. *Let* $\mathcal{H} = (\times D_i^{k_i}, \mathcal{R}, \times\mathcal{R}_i)$ *be a weak heterogeneous system. Assume that for all* $i \leq n$*, there exists* $\mathcal{S}_i$ *a symbolic representation for* $\mathcal{H}_i = (D_i^{k_i}, \mathcal{R}_i)$ *and an acceleration function* ${\mathsf{post}^*}_i$ *for* $(\mathcal{H}_i, \mathcal{S}_i)$*. Then* $\times{\mathsf{post}^*}_i$ *is an upper approximation of* post^* *on* $(\mathcal{H}, \times\mathcal{S}_i)$*.*

Sketch of the proof.
Let $r = r_1 \times r_2$ and $\mathsf{s} = (\mathsf{s}_1, \mathsf{s}_2)$. Intuitively, ${\mathsf{post}^*}_1(r_1, \mathsf{s}_1) \times {\mathsf{post}^*}_2(r_2, \mathsf{s}_2)$ represents $\bigcup_{i\in\mathbb{N}} \bigcup_{j\in\mathbb{N}} r_1^i(\mathsf{s}_1) \times r_2^j(\mathsf{s}_2)$ while $\mathsf{post}^*(r, \mathsf{s}) = \bigcup_{k\in\mathbb{N}} r_1^k(\mathsf{s}_1) \times r_2^k(\mathsf{s}_2)$ □

4 Combination of accelerations

We define in this section a particular class of symbolic representations, and a particular class of acceleration functions such that if there exist such symbolic representation and acceleration functions for domains $D_1, \ldots, D_n$ we can automatically build a symbolic representation and an acceleration function for a weak heterogeneous system on $D_1, \ldots, D_n$. These classes recover many existing symbolic representations and accelerations.

4.1 General framework

The main idea is to use symbolic representations which combine some finite part (a word in our definition) with a counting part, represented as a formula added to the finite part. Then for the acceleration, the counting parts are synchronized together. We consider constraints expressed in Presburger formula. Presburger must not be far to be a maximal logic for our construction, since in the proofs, the constraints are required to be closed under intersection and existential quantification, and satisfiability must be decidable. However we do not have investigated this question yet. For example it is easy to extend our construction to the whole logic defined by binary automata.

Definition 6 (Presburger symbolic representation (PSR)). *Let $\mathcal{H} = (D, \mathcal{R})$ be a system. A Presburger symbolic representation for $\mathcal{H}$ is an effective symbolic representation $\mathcal{S}p = (\mathcal{P}_f(\mathsf{Sp}), \gamma, \sqcup, \sqsubseteq, \mathsf{post})$ such that:*

1. Sp *is defined by a 4-tuple $(\Sigma, \mathfrak{L}, \mathcal{N}, \mathcal{V})$ where $\mathfrak{L}$ is a language over an alphabet Σ, $\mathcal{N}$ is a countable set of variables, $\mathcal{V} : \mathfrak{L} \to \mathcal{P}_f(\mathcal{N})$; representing the set of 2-tuples $\mathsf{sp} = (w, \Phi(\mathcal{V}(w)))$ with:*
   - *w is a word in $\mathfrak{L}$,*
   - *$\Phi(\mathcal{V}(w))$ is a Presburger formula whose free variables are in $\mathcal{V}(w)$; we will write $\Phi(w)$ or even Φ for convenience*
2. *the concretization function γ is defined as follows:*
   - *there exists a function $\gamma_a : (w : \mathfrak{L}) \times \mathbb{N}^{|\mathcal{V}(w)|} \to D$ such that $Im(\gamma_a(w))$ is not reduced to a singleton,*
   - *$\gamma((w, \Phi)) = \bigcup_{v \in \Phi} \gamma_a(w, v)$*
3. *$\mathsf{post}(r, (w, \Phi(w))) = (w', \exists w, \Phi(w) \wedge \varphi(w, w'))$ where w' and φ depend only on r and w. post is then extended to $\mathcal{P}_f(\mathsf{Sp})$ using the $\sqcup$ operator.*
4. *$(w, \Phi(w)) \sqsubseteq (w', \Phi'(w')) \Leftrightarrow w = w' \wedge \Phi \Rightarrow \Phi'$*

Requirement 2 means that the concretization function really takes into account the counting part of the representation. Requirement 3 means that a post operation modifies the word w and the formula Φ according to w, adding to Φ a formula φ linking old variables of w and new variables of w'. Many existing symbolic representations are Presburger symbolic representations.

Remark 2. These properties hold for Presburger symbolic representations:

- $\gamma((w, \mathit{False})) = \emptyset$;
- $\gamma((w, \Phi_1 \vee \Phi_2)) = \gamma((w, \Phi_1)) \cup \gamma((w, \Phi_2))$;
- $\gamma_a((w, (v_1, \ldots, v_n))) = \gamma((w, \bigwedge_i w_i = v_i))$;
- if $\mathsf{post}(r, (w, \Phi_1)) = (w', \exists w \Phi_1 \wedge \varphi)$ then $\mathsf{post}(r, (w, \Phi_2)) = (w', \exists w \Phi_2 \wedge \varphi)$.

Theorem 4. *NDDs, UBAs, RVAs and CQDDs are Presburger symbolic representations.*

Sketch of the proof.
For NDDs/UBAs/RVAs over 2 variables x, y take $\Sigma = \{\varepsilon\}, \mathcal{L} = \{\varepsilon\}, \mathcal{N} = \{x, y\}, \mathcal{V} : \varepsilon \to \{x, y\}$. For CQDDs over a finite set of messages Σ_{msg}, take $\Sigma = \Sigma_{msg} \cup \{*\}$, $\mathcal{L}$ the language of SLRE over Σ_{msg} (which are words over Σ) and Var a function which associates to each $*$ in a word in $\mathcal{L}$ a distinct variable. Notice that in this case, we use a restricted inclusion on CQDDs, only semi-exact. $\square$

BDD, QDD, SLRE, SRE are not Presburger symbolic representations, since there is no counting in these representations. CPDBMs are not PSR too since the formula used are out of Presburger arithmetics.

Definition 7 (Counting acceleration). *Let $\mathcal{H} = (D, \mathcal{R})$ be a system and $\mathcal{S}p = (\mathcal{P}_f(\mathsf{Sp}), \gamma, \sqcup, \sqsubseteq, \mathsf{post})$ a Presburger symbolic representation for $\mathcal{H}$. A counting acceleration for $(\mathcal{H}, \mathcal{S}p)$ is an acceleration function* post^* *for $(\mathcal{H}, \mathcal{P}_f(\mathcal{S}p))$ such that $\forall \mathsf{sp} = (w, \Phi(w)) \in \mathsf{Sp}, \forall r \in \mathcal{R}$,*

1. $\mathsf{post}^*(r, (w, \Phi)) = \bigsqcup_{i \leq k}(w'_i, \exists \theta \in \mathbb{N}, \exists w, \Phi(w) \wedge \varphi_i(w, w'_i, \theta))$ *where (k, w'_i, φ_i) depends only of r and w ;*
2. $\gamma((w', \exists \theta \in \mathbb{N}, \exists w, \Phi(w) \wedge \varphi(w, w', \theta) \wedge \theta = i)) = r^i(\gamma(w, \Phi))$.

post^* *is then extended to $\mathcal{P}_f(\mathsf{Sp})$ using the $\sqcup$ operator.*

Intuitively a counting acceleration keeps precisely the number of iterations, so that it is possible to manipulate it. Requirement 1 means that the post^* operation introduces an *iteration variable* and requirement 2 ensures that this variable keeps precisely the number of iterations.

Remark 3. If $\mathsf{post}^*(r, (w, \Phi_1)) = \bigsqcup_{i \leq k}(w'_i, \exists \theta \in \mathbb{N}, \exists w, \Phi_1 \wedge \varphi_i(w, w'_i, \theta))$ then $\mathsf{post}^*(r, (w, \Phi_2)) = \bigsqcup_{i \leq k}(w'_i, \exists \theta \in \mathbb{N}, \exists w, \Phi_2 \wedge \varphi_i(w, w'_i, \theta))$.

Theorem 5. *Counting accelerations exist for NDDs/UBAs, RVAs, and CQDDs.*

Sketch of the proof.
We cannot described in detail the algorithms used. See theorem 8.52 of [Boi98], proposition 2 of [FL02], theorem 5 of [BHJ03] and theorem 5.1 of [BH99]. $\square$

We define the *synchronized product* of PSRs. In this representation, the two PSRs are synchronized on shared variables. The consequence is that given two PSRs, the set of concrete states expressible with the cartesian product is strictly included in the one expressible with the synchronized product. This allows us to define an (exact) acceleration function for the synchronized product of PSRs.

Definition 8 (Synchronized product of PSR). *Let $\mathcal{S}p_1, \mathcal{S}p_2$ two Presburger symbolic representations $\mathcal{S}p_i = (\mathcal{P}_f(\mathsf{Sp}_i), \gamma_i, \sqcup_i, \sqsubseteq_i, \mathsf{post}_i)$ with $\mathsf{Sp}_i = (\Sigma_i, \mathcal{L}_i, \mathcal{N}_i, \mathcal{V}_i)$. The synchronized product $\mathcal{S}p_1 \otimes \mathcal{S}p_2$ is defined by $(\mathcal{P}_f(\mathsf{Sp}), \gamma, \sqcup, \sqsubseteq, \mathsf{post})$ where:*

- $\mathsf{Sp} = (\Sigma_1 \times \Sigma_2, \mathcal{L}_1 \times \mathcal{L}_2, \mathcal{N}_1 \times \mathcal{N}_2, \mathcal{V}_1 \times \mathcal{V}_2)$,
- $\gamma(w_1, w_2, \Phi) = \bigcup_{(v_1, v_2 \in \Phi)} \gamma_{1,a}(w_1, v_1) \times \gamma_{2,a}(w_2, v_2)$;

– $\sqcup$ *is the finite union on* $\mathcal{P}_f(\mathsf{Sp})$;
– $\mathsf{post}(w_1, w_2, \Phi) = (w'_1, w'_2, \exists w_1, w_2 \Phi(w_1, w_2) \wedge \varphi_1(w_1, w'_1) \wedge \varphi_2(w_2, w'_2))$;
– $(w_1, w_2, \Phi) \sqsubseteq (w'_1, w'_2, \Phi') \Leftrightarrow w_1 = w'_1 \wedge w_2 = w'_2 \wedge \Phi \Rightarrow \Phi'$.

Theorem 6. *Let* $\mathcal{H} = (\times D_i^{k_i}, \mathcal{R}, \times\mathcal{R}_i)$ *a weak heterogeneous system. Assume that* $\forall i$*, there exists* $\mathcal{S}p_i = (\mathcal{P}_f(\mathsf{Sp}_i), \gamma_i, \sqcup_i, \sqsubseteq_i, \mathsf{post}_i)$ *a Presburger symbolic representation for* $\mathcal{H}_i = (D_i^{k_i}, \mathcal{R}_i)$ *and* $\mathsf{post^*}_i$ *a counting acceleration for* $(\mathcal{H}_i, \mathcal{S}p_i)$*. Then*

1. $\bigotimes \mathcal{S}p_i$ *is a Presburger symbolic representation for* $\mathcal{H}$*,*
2. *there exists a counting acceleration for* $(\mathcal{H}, \bigotimes \mathcal{S}p_i)$*.*

Proof.
We prove statement 1 that $\bigotimes \mathcal{S}p_i$ is a PSR for $\mathcal{H}$. For the sake of conciseness, we give the proof for only 2 PSR. The extension to n PSR is clear. **Requirement 1** of a symbolic representation (*consistency of union*) is clear with $\sqcup$.

We prove **requirement 2** of a symbolic representation, the *consistency of inclusion*. Remark that if $\Phi \Rightarrow \Phi'$ then $\{v|\Phi(v)\} \subseteq \{v'|\Phi'(v')\}$. Using this and the definition of γ_i in terms of $\gamma_{i,a}$, it is then straightforward that $(w_1, w_2, \Phi) \sqsubseteq (w'_1, w'_2, \Phi') \Rightarrow \gamma((w_1, w_2, \Phi)) \subseteq \gamma((w'_1, w'_2, \Phi'))$.

Now we prove **requirement 3** of a symbolic representation, the *consistency of* post. Let us show that $r(\gamma(w_1, w_2, \Phi)) = \gamma(\mathsf{post}(r, (w_1, w_2, \Phi)))$.
By definition of γ, $r(\gamma(w_1, w_2, \Phi)) = r(\bigcup_{(v_1, v_2 \in \Phi)} \gamma_{1,a}(w_1, v_1) \times \gamma_{2,a}(w_2, v_2))$ $= \bigcup_{(v_1, v_2 \in \Phi)} r(\gamma_{1,a}(w_1, v_1) \times \gamma_{2,a}(w_2, v_2))$. Since $\mathcal{H}$ is a weak heterogeneous system, $r(\gamma(w_1, w_2, \Phi)) = \bigcup_{(v_1, v_2 \in \Phi)} r_1(\gamma_{1,a}(w_1, v_1)) \times r_2(\gamma_{2,a}(w_2, v_2))$, which can be written $r(\gamma(w_1, w_2, \Phi)) = \bigcup_{(v_1, v_2 \in \Phi)} r_1(\gamma_1(w_1, \bigwedge w_{1,i} = v_{1,i})) \times r_2(\gamma_2(w_2, \bigwedge w_{2,i} = v_{2,i}))$ using remark 2. Then $r(\gamma(w_1, w_2, \Phi)) = \bigcup_{(v_1, v_2 \in \Phi)} \gamma_1(\mathsf{post}_1(r_1, (w_1, \bigwedge w_{1,i} = v_{1,i}))) \times \gamma_2(\mathsf{post}_2(r_2, (w_2, \bigwedge w_{2,i} = v_{2,i})))$ because $\mathcal{S}_i$ are symbolic representations for $\mathcal{H}_i$. By definition of post, this is equal to $\bigcup_{(v_1, v_2 \in \Phi)} \gamma_1(w'_1, \exists w_1 \bigwedge w_{1,i} = v_{1,i} \wedge \varphi_1(w_1, w'_1)) \times \gamma_2((w'_2, \exists w_2 \bigwedge w_{2,i} = v_{2,i} \wedge \varphi_2(w_2, w'_2)))$. Using definitions of γ_1 and γ_2 we get that

$$
\begin{aligned}
& r(\gamma(w_1, w_2, \Phi)) \\
&= \textstyle\bigcup_{(v_1, v_2 \in \Phi)} (\bigcup_{v'_1 \in \exists w_1 \wedge w_1 = v_1 \wedge \varphi_1(w_1, w'_1)} \gamma_{1,a}(w'_1, v'_1)) \times (\bigcup_{v'_2 \in \exists w_2 \wedge w_2 = v_2 \wedge \varphi_2(w_2, w'_2)} \gamma_{2,a}(w'_2, v'_2)) \\
&= \textstyle\bigcup_{(v'_1, v'_2 \in \exists w_1, w_2 \Phi(w_1, w_2) \wedge \varphi_1(w_1, w'_1) \wedge \varphi_2(w_2, w'_2))} \gamma_{1,a}(w'_1, v'_1) \times \gamma_{2,a}(w'_2, v'_2) \\
&= \gamma(w'_1, w'_2, \exists w_1, w_2 \Phi(w_1, w_2) \wedge \varphi_1(w_1, w'_1) \wedge \varphi_2(w_2, w'_2))))
\end{aligned}
$$

Then $r(\gamma(w_1, w_2, \Phi)) = \gamma(\mathsf{post}(r, (w_1, w_2, \Phi)))$. This proves requirement 3, and the first statement, that $\bigotimes \mathcal{S}p_i$ is a PSR for $\mathcal{H}$.

We now prove statement 2, that there exists a counting acceleration for $(\mathcal{H}, \bigotimes \mathcal{S}p_i)$. We assume for conciseness that for each counting acceleration $\mathsf{post^*}_i$ on $(\mathcal{H}_i, \mathcal{S}p_i)$, the image by $\mathsf{post^*}_i$ of each $\mathsf{sp} \in \mathsf{Sp}_i$ is a singleton. The extension to $Im(\mathsf{post^*}_i) \subseteq \mathcal{P}_f(\mathsf{Sp}_i)$ is straightforward but cumbersome.

We define the acceleration $\mathsf{post^*}$ on $(\mathcal{H}, \mathcal{P}_f(\bigotimes \mathcal{S}p_i))$ the following. Assume that the $\mathsf{post^*}_i$ are defined by $\forall i$, $\mathsf{post^*}_i(r_i, (w_i, \Phi_i)) = (w'_i, \exists w_i, \theta, \Phi_i \wedge \varphi_i(w_i, w'_i, \theta))$. Then we define the acceleration $\mathsf{post^*}$ on the whole system $\mathcal{H}$ by
$\mathsf{post^*}(r_1 \times r_2, (w_1, w_2, \Phi(w_1, w_2))) = (w'_1, w'_2, \exists w_1, w_2, \theta \Phi(w_1, w_2) \wedge \varphi_1(w_1, w'_1, \theta) \wedge \varphi_2(w_2, w'_2, \theta))$.

First we show that $r^i(\gamma(w_1, w_2, \Phi)) = \gamma(w_1', w_2', \exists w_1, w_2, \theta \Phi \wedge \varphi_1 \wedge \varphi_2 \wedge \theta = i)$ $(*)$.

$r^i(\gamma(w_1, w_2, \Phi))$
$= r^i(\bigcup_{v_1,v_2 \in \Phi} \gamma_{1a}(w_1, v_1) \times \gamma_{2a}(w_2, v_2))$ by definition of γ
$= \bigcup_{v_1,v_2 \in \Phi} r^i(\gamma_{1a}(w_1, v_1) \times \gamma_{2a}(w_2, v_2))$
$= \bigcup_{v_1,v_2 \in \Phi} r_1^i(\gamma_{1a}(w_1, v_1)) \times r_2^i(\gamma_{2a}(w_2, v_2))$ since $\mathcal{H}$ is weak heterogeneous
$= \bigcup_{v_1,v_2 \in \Phi} r_1^i(\gamma_1(w_1, w_1 = v_1)) \times r_2^i(\gamma(w_2, w_2 = v_2))$ see remark 2
$= \bigcup_{v_1,v_2 \in \Phi} \gamma_1(w_1', \exists w_1, \theta\ w_1 = v_1 \wedge \varphi_1(w_1, w_1', \theta) \wedge \theta = i)) \times \gamma_2(w_2', \exists w_2, \theta\ w_2 = v_2 \wedge \varphi_2(w_2, w_2', \theta) \wedge \theta = i))$ see remark 3 and the $\mathsf{post^*}_i$ are counting accelerations)
$= \bigcup_{v_1',v_2' \in \exists w_1, w_2, \theta \Phi \wedge \varphi_1 \wedge \varphi_2 \wedge \theta = i} \gamma_1(w_1', v_1') \times \gamma_2(w_2', v_2')$ same idea than for post
$= \gamma(w_1', w_2', \exists w_1, w_2, \theta \Phi \wedge \varphi_1 \wedge \varphi_2 \wedge \theta = i)$.

Now let us prove that $\gamma(\mathsf{post}^*(r, (w_1, w_2, \Phi))) = r^*(\gamma(w_1, w_2, \Phi))$ $(**)$.

$r^*(\gamma(w_1, w_2, \Phi))$
$= \bigcup_i r^i(\gamma(w_1, w_2, \Phi))$
$= \bigcup_i \gamma(w_1', w_2', \exists w_1, w_2, \theta \Phi \wedge \varphi_1 \wedge \varphi_2 \wedge \theta = i))$ from (*) above
$= \bigcup_i \bigcup_{v_{1i}',v_{2i}' \in \exists w_1, w_2, \theta \Phi \wedge \varphi_1 \wedge \varphi_2 \wedge \theta = i)} \gamma_{1,a}(w_1', v_{1i}') \times \gamma_{2,a}(w_2', v_{2i}')$ definition of γ
$= \bigcup_{v_1',v_2' \in \exists w_1, w_2, \theta \Phi \wedge \varphi_1 \wedge \varphi_2} \gamma_{1,a}(w_1', v_1') \times \gamma_{2,a}(w_2', v_2')$
$= \gamma(w_1', w_2', \exists w_1, w_2, \theta \Phi \wedge \varphi_1 \wedge \varphi_2)$
$= \gamma(\mathsf{post}^*(r, (w_1, w_2, \Phi)))$

(**) proves we have an acceleration and (*) proves also this is a counting acceleration. □

4.2 Existing Presburger symbolic representations

Results on existing counting accelerations are summed up, to give the reader an idea of systems we can manipulate.

	data	effective symbolic representation	post^*/ counting
SRE	lossy queues	yes	yes/no
QDD	queues/stacks	yes	restricted/no
SLRE	queues/stacks	yes	restricted/no
BDD	boolean	yes	no
CQDD	**queues/stacks**	**yes**	**yes**
UBA/NDD	**counters**	**yes**	**yes**
RVA	**clocks and counters**	**yes**	**yes**
union of convexes	clocks and counters	yes	no
DBMs	clocks	yes	no
CPDBMs	clocks and counters	$\sqsubseteq$ semi-decidable	semi-decidable

Theorem 7. *Given the previous results, a symbolic representation and an acceleration function can be automatically computed for weak heterogeneous systems manipulating counters, clocks, perfect FIFO queues and stacks.* □

4.3 About termination

We can automatically combine acceleration functions. Now the issue is the insurance of termination. Of course it is undecidable in the generic case. But there are interesting particular cases. For example, if each projection of the system on each different data types is flat, i.e. is computable using an acceleration function, is the whole system flat or not (*)? We did not investigate this question deeply. However we sketch here briefly a first result of flatness for a whole system, from the studies of its projections.

We define the *projection* of a system $\mathcal{H}$ on a data type D_i as the system $\mathcal{H}'$ obtained from $\mathcal{H}$ projecting all the relations of $\mathcal{H}$ on D_i, and removing the identity relation. We introduce *controlled systems*, i.e. systems with a *control graph*. These are particular cases of systems, seeing their domain as a Cartesian product of arbitrary sets and a finite set of control states. Usual systems are controlled systems. A controlled system is *structurally flat* when its control graph is flat (no nested loop). *Structural flatness implies flatness.* It is straightforward that if each projection of a weak heterogeneous system is structurally flat, then the whole system is flat *(while not necessarily structurally flat).*

This is a weak version of (*), and we plan to work more on this aspect.

5 Toward a tool for generic weak heterogeneous systems

The previous results can be used to design a software for the verification of arbitrary weak heterogeneous systems. The idea is to have a generic algorithm working on generic PSR, and a library of PSR for different data. Then ad hoc symbolic representations are automatically computed using theorem 6.

5.1 The generic symbolic representation

We consider a generic Presburger symbolic representation (GPSR), which must provides the $\sqcup, \sqsubseteq$, post and post^* operators. Considering theorem 6, implementing the composition operator $GPSR \otimes GPSR \to GPSR$ is quite straightforward. Then given a weak heterogeneous system $\mathcal{H}$ with domains $D_1, \ldots, D_n$ and relations in $\mathcal{R}_1 \times \ldots \times \mathcal{R}_n$, given a library of Presburger symbolic representation PSR_i (implementing GPSR) available for each $(D_i, \mathcal{R}_i)$, we can automatically compute the class $\bigotimes_i PSR_i$ implementing GPSR for $\mathcal{H}$.

5.2 The generic algorithm

Basically, tools implementing effective accelerations use similar semi-algorithm to compute the reachability set. They use the classic iterative fixpoint computation, firing transitions one by one. The difference is to add *meta-transitions* to the set of transitions, and fire them with the post^* operation. The problem comes down to finding these meta-transitions. They may be provided by the user [LAS], or automatically found. In this case they can be computed statically [BFL04] or dynamically [ABS01]. The first approach is easy to extend to any data type, while the second needs a notion of “progress” difficult to define.

5.3 Architecture

We give the architecture (Fig. 1) of a tool operating on weak heterogeneous systems with arbitrary data. The tool takes a weak heterogeneous system in input. Specific symbolic representation and an acceleration technique are computed from symbolic representations available in the library, matching the data types of the system. Then the generic semi-algorithm for reachability set computation is instantiated with this specific symbolic representation. When the user wants to use unknown data, he has to provide only the PSR and the counting acceleration, instead of writing a new tool. And if the system uses any combination of data types available in the library, nothing is to be done.

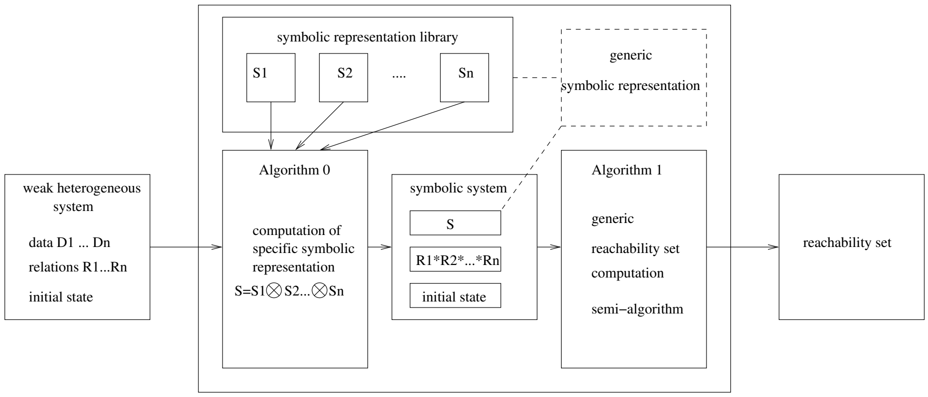

Fig. 1. Architecture of a tool for verification of arbitrary weak heterogeneous systems

6 Conclusion and Perspective

We focus here on the combination of existing symbolic representations and accelerations. We defined subclasses for which this combination can be automated. These subclasses turn out to be well-spread in practice and capture many existing work on numeric data, queues and stacks. Finally we have shown how these theoretical results can be used to design a tool for the verification of infinite systems with arbitrary data types. Future works include 1) theoretical work on getting more efficient combinations, heuristics and termination results on the whole system from the projections on each data types, as well as implementation of a prototype and case-studies; 2) defining which logics are suitable for our construction and isolating some maximal logics; 3) relaxing the hypothesis of exactness to combine a larger class of symbolic representations, in a more precise way than with the Cartesian product.

References

[AAB00] A. Annichini, E. Asarin, and A. Bouajjani. Symbolic techniques for parametric reasoning about counter and clock systems. In *Proc. 12th Int. Conf. Computer Aided Verification (CAV'2000), Chicago, IL, USA, July 2000*, volume 1855 of *Lecture Notes in Computer Science*, pages 419–434. Springer, 2000.

[ABJ98] Parosh Aziz Abdulla, Ahmed Bouajjani, and Bengt Jonsson. On-the-fly analysis of systems with unbounded, lossy FIFO channels. In *Proc. 10th Int. Conf. Computer Aided Verification (CAV'98), Vancouver, BC, Canada, June-July 1998*, volume 1427 of *Lecture Notes in Computer Science*, pages 305–318. Springer, 1998.

[ABS01] A. Annichini, A. Bouajjani, and M. Sighireanu. TReX: a Tool for Reachability analysis of Complex Systems. In *Proc. 13th Int. Conf. Computer Aided Verification (CAV'2001), Paris, France, July 2001*, volume 2102 of *Lecture Notes in Computer Science*, pages 368–372. Springer, 2001.

[BC96] Alexandre Boudet and Hubert Comon. Diophantine equations, Presburger arithmetic and finite automata. In *Proc. 21st Int. Coll. on Trees in Algebra and Programming (CAAP'96), Linköping, Sweden, Apr. 1996*, volume 1059 of *Lecture Notes in Computer Science*, pages 30–43. Springer, 1996.

[BFL04] S. Bardin, A. Finkel, and J. Leroux. FASTer acceleration of counter automata in practice. In *Proc. 10th Int. Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'2004), Barcelona, Spain, Apr. 2004*, volume 2988 of *Lecture Notes in Computer Science*, pages 576–590. Springer, 2004.

[BFLP03] S. Bardin, A. Finkel, J. Leroux, and L. Petrucci. FAST: Fast Acceleration of Symbolic Transition systems. In *Proc. 15th Int. Conf. Computer Aided Verification (CAV'2003), Boulder, CO, USA, July 2003*, volume 2725 of *Lecture Notes in Computer Science*, pages 118–121. Springer, 2003.

[BGWW97] Bernard Boigelot, Patrice Godefroid, Bernard Willems, and Pierre Wolper. The power of QDDs. In *Proc. of the 4th Static Analysis Symposium (SAS'1997), Paris, France, September. 1997*, volume 1302 of *Lecture Notes in Computer Science*, pages 172–186. Springer, 1997.

[BH99] Ahmed Bouajjani and Peter Habermehl. Symbolic reachability analysis of FIFO-channel systems with nonregular sets of configurations. *Theoretical Computer Science*, 221(1–2):211–250, 1999.

[BHJ03] B. Boigelot, F. Herbreteau, and S. Jodogne. Hybrid acceleration using real vector automata. In *Proc. 15th Int. Conf. Computer Aided Verification (CAV'2003), Boulder, CO, USA, July 2003*, volume 2725 of *Lecture Notes in Computer Science*, pages 193–205. Springer, 2003.

[BJW01] Bernard Boigelot, Sébastien Jodogne, and Pierre Wolper. On the use of weak automata for deciding linear arithmetic with integer and real variables. In *Proc. 1st Int, Joint Conf. , IJCAR 2001 Siena, Italy, June 18-23, 2001*, volume 2083 of *Lecture Notes in Computer Science*, pages 588–603. Springer, 2001.

[Boi98] B. Boigelot. *Symbolic Methods for Exploring Infinite State Spaces.* PhD thesis, Université de Liège, 1998.

[Bou01] Ahmed Bouajjani. Languages, rewriting systems, and verification of infinite-state systems. In *Proc. 28th Int. Coll. Automata, Languages, and Programming (ICALP'2001), Crete, Greece, July 2001*, volume 2076 of *Lecture Notes in Computer Science*, pages 24–39. Springer, 2001.

[Bry92] Randal E. Bryant. Symbolic boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys*, 24(3):293–318, 1992.

[CC77] Patrick Cousot and Radhia Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proc. 4th ACM Symp. Principles of Programming Languages, Los Angeles, CA, USA*, pages 238–253. ACM, January 1977.

[Dil89] David L. Dill. Timing assumption and verification of finite-state concurrent systems. In *Proc. Int. Workshop Automatic Verification Methods for Finite State Systems (CAV'89), Grenoble, June 1989*, volume 407 of *Lecture Notes in Computer Science*, pages 197–212. Springer, 1989.

[FL02] A. Finkel and J. Leroux. How to compose Presburger-accelerations: Applications to broadcast protocols. In *Proc. 22nd Conf. Found. of Software Technology and Theor. Comp. Sci. (FST&TCS'2002), Kanpur, India, Dec. 2002*, volume 2556 of *Lecture Notes in Computer Science*, pages 145–156. Springer, 2002.

[FPS03] Alain Finkel, S. Purushothaman Iyer, and Grégoire Sutre. Well-abstracted transition systems: Application to FIFO automata. *Information and Computation*, 181(1):1–31, 2003.

[HPR97] N. Halbwachs, Y.E. Proy, and P. Roumanoff. Verification of real-time systems using linear relation analysis. *Formal Methods in System Design*, 11(2):157–185, August 1997.

[LAS] LASH *homepage*. `http://www.montefiore.ulg.ac.be/~boigelot/research/lash/`.

[Ler03] J. Leroux. The affine hull of a binary automaton is computable in polynomial time. In *Proc. 5th Int. Workshop on Verification of Infinite State Systems (INFINITY'2003), Marseille, France, Sep. 2003*, Electronic Notes in Theor. Comp. Sci. Elsevier Science, 2003.

[Tav04] B. Tavernier. Calife: a generic graphical user interface for automota tools. In *Proc. of the 4th Workshop on Language Descriptions, Tools and Applications (LDTA'2004), Barcelona, Spain, April. 2004*, 2004.

[WB00] P. Wolper and B. Boigelot. On the construction of automata from linear arithmetic constraints. In *Proc. 6th Int. Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'2000), Berlin, Germany, Mar.-Apr. 2000*, volume 1785 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2000.

[YKTB01] Tuba Yavuz-Kahveci, Murat Tuncer, and Tevfik Bultan. A library for composite symbolic representations. In *Proc. 7th Int. Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'2001), Genova, Italy, Apr. 2001*, volume 2031 of *Lecture Notes in Computer Science*, pages 52–66. Springer, 2001.