Model Checking Lossy Vector Addition Systems

Ahmed Bouajjani *‡ Richard Mayr†‡

Abstract

Lossy VASS (vector addition systems with states) are defined as a subclass of VASS in analogy to lossy FIFO-channel systems. They can be used to model concurrent systems with unreliable communication. We analyze the decidability of model checking problems for lossy systems and several branching-time and linear-time temporal logics. We present an almost complete picture of the decidability of model checking for normal VASS, lossy VASS and lossy VASS with test for zero.

1 Introduction

Systems are usually modeled by finite control transition systems with different kinds of variables and data structures like counters, clocks, stacks, fifo-channels, etc. One of the widely used models of concurrent systems is the model of Petri nets which is equivalent to the model of vector addition systems with states (VASS for short). These models can be considered as particular cases of counter machines where tests to zero are forbidden (the addition of inhibitor arcs gives them the full power of counter machines). VASS's can model communicating systems through unbounded unordered buffers, and hence they can be seen as abstractions of fifo-channels systems, when the ordering between messages in the channels is not relevant but only their number. Actually, it is often the case that communicating systems must be analyzed under the assumption that they communicate through unreliable channels. Hence, it is natural to consider *lossy* models of communicating systems, i.e. models where messages can be lost. Several works have been

*Verimag, Centre Equation, 2 avenue de Vignate, 38610 Gières, France.
†Institut für Informatik, TU-München, Arcisstr. 21, D-80290 München, Germany.
‡`Ahmed.Bouajjani@imag.fr, mayrri@informatik.tu-muenchen.de`

devoted recently to the analysis of lossy unbounded fifo-channels systems [AJ93, AJ96, CFI96]. They have shown in particular that the reachability problem is decidable for these models, which implies the decidability of the verification problem for safety properties. They have also shown that, unfortunately, liveness properties cannot be checked for lossy fifo-channel systems, unless for very special ones like single eventualities. In particular, it is impossible to model check lossy channel systems under fairness conditions. In this paper, we study verification problems for VASS and VASS with inhibitor arcs (counter machines) under the assumption of lossiness. Here lossiness means that the contents of a place/counter can spontaneously get lower at any time.

Since verification is essentially based on reachability analysis, the first question we address is whether given a set S of configurations, it is possible to effectively construct the set $pre^*(S)$ of all its predecessors and the set $post^*(S)$ of all its successors. Using the approach introduced in [CFI96, ACJT96], it can be shown very easily that the pre^* image of any set of configurations is effectively constructible for lossy VASS even with inhibitor arcs, and that this set can be represented by simple linear constraints (SC for short), where integer variables can be compared only with constants. Moreover, we show that for lossy VASS, the $post^*$ images are SC definable and effectively constructible, but interestingly, for lossy VASS with inhibitor arcs these sets are not constructible although they are SC definable. This means that safety properties (at least) can be checked for VASS with both backward and forward reachability analysis, while they can only be checked using backward search for lossy VASS with inhibitor arcs (efficient but incomplete forward analysis procedures could be applied following the approach presented in [BGWW97, BH97, ABJ98]).

Then, the major part of the paper concerns model checking. We consider two versions of this problem: *local model checking*, or simply *model checking*, which consists in deciding whether a given configuration of a system satisfies a given formula of a temporal logic, and *global model checking* which consists in constructing the set of all configurations that satisfy a given formula. We address these problems for a variety of linear-time and branching-time properties. For the sake of generality, we express these properties in a temporal logic, called AL (Automata Logic), which is based on the use of automata on finite and infinite sequences to specify path properties (in the spirit of ETL [Wol83]), and the use of path quantifiers to express branching-time properties (like in ECTL* [CGK87, Tho89]). The basic state predicates in this logic are SC constraints.

Our main positive result is that for lossy VASS, the global model checking is decidable for the logic $\exists$AL with only upward closed constraints, and dually for $\forall$AL with downward closed constraints ($\forall$AL and $\exists$AL are the universal and existential positive fragments of AL. They subsume respectively the corresponding well-known fragments $\forall$CTL* and $\exists$CTL* [GL94] of the logic CTL*). Actually, we show that when only infinite paths are considered our decidability result also holds for normal VASS. A corollary of these results is that, since in $\forall$AL we use automata to define path properties, all ∞-regular (ω-regular) linear-time properties on finite and infinite paths (on infinite paths only) are decidable for lossy VASS (normal VASS), and even more, we can construct the set of all the configurations satisfying these properties. This generalizes the result concerning VASS and ω-regular properties given in [Esp94] where only model checking is considered. Notice also that $\forall$AL is strictly more expressive than all linear-time temporal logics.

These results are interesting since systems are often verified under abstractions, and it can be shown that, if a system $\mathcal{S}$ simulates a system $\mathcal{C}$, then if a $\forall$AL formula holds on the "abstract" system $\mathcal{S}$, it also holds on the "concrete" system $\mathcal{C}$. For instance, if we start with a (lossy) fifo-channel system, we can abstract it to a VASS or a lossy VASS, and then check the property we are interested in on the abstract model, and if it holds, we can deduce that it holds on the original model.

Then we show that these decidability results break down if we relax any of the restrictions we have on the system models and the logic: model checking becomes undecidable if we consider $\forall$AL or $\exists$AL formulas with both downward and upward closed constraints, or if we consider lossy VASS with inhibitor arcs. Also, even if we use only propositional constraints in the logic (i.e., only constraints on control locations) the use of negation must be restricted: model checking is undecidable for CTL and lossy VASS. However, it is decidable for the fragments EF and EG of CTL even for lossy VASS with inhibitor arcs, but surprisingly, global model checking is undecidable for EG and lossy VASS (while it is decidable for EF and lossy VASS with inhibitor arcs). We also obtain as a side effect of our results that normal VASS (Petri nets) and lossy VASS with inhibitor arcs (lossy counter machines) are two incomparable models.

The rest of the paper is organized as follows: In the next section we recall the definition of VASS. In Section 3 we introduce simple constraints and show how they can be used to represent sets of VASS configurations. In Section 4, we show how the backward and forward reachability sets of lossy VASS can be effectively computed by means of simple constraints. In Section 5 we recall the definitions of automata on finite and infinite sequences since they are used in the definition of our logic AL. In Section 5 we introduce the logic AL and its fragments. In Section 6 we give our results concerning model checking of lossy VASS and different fragments of AL. In Section 7, we examine how these results extend or break down in the case of lossy VASS with inhibitor arcs. Finally, we give concluding remarks and a table summarizing our results in Section 8.

2 Vector Addition Systems with States

Definition 2.1 *A n-dim VASS $\mathcal{S}$ is a tuple $(\Sigma, \mathcal{X}, Q, \delta)$ where*

- *Σ is a set of action labels,*
- *$\mathcal{X}$ is a set of variables such that $|\mathcal{X}| = n$,*
- *Q is a finite set of control states,*
- *δ is a finite set of transitions of the form (q_1, a, Δ, q_2) where $a \in \Sigma$, $\Delta \in \mathbb{Z}^n$.*

A *configuration* of $\mathcal{S}$ is a pair $\langle q, \vec{u}\rangle$ where $q \in Q$ and $\vec{u} \in \mathbb{N}^n$. Let $\mathcal{C}(\mathcal{S})$ be the set of configurations of $\mathcal{S}$. Given a configuration $s = \langle q, \vec{u}\rangle$, we let $State(s) = q$ and $Val(s) = \vec{u}$.

We define a *transition relation* $\longrightarrow$ on configurations as follows: $\langle q_1, \vec{u}_1\rangle \xrightarrow{a} \langle q_2, \vec{u}_2\rangle$ iff $\exists \tau = (q_1, a, \Delta, q_2) \in \delta$, $\vec{u}_2 = \vec{u}_1 + \Delta$. We let $post_\tau(\langle q_1, \vec{u}_1\rangle)$ (resp. $pre_\tau(\langle q_2, \vec{u}_2\rangle)$) denote the configuration $\langle q_2, \vec{u}_2\rangle$ (resp. $\langle q_1, \vec{u}_1\rangle$), i.e., the immediate successor (resp. predecessor) of $\langle q_1, \vec{u}_1\rangle$ (resp. $\langle q_2, \vec{u}_2\rangle$) by the transition τ. Then, we let *post* (resp. *pre*) denote the union of the $post_\tau$'s (resp. pre_τ's) for all the transitions $\tau \in \delta$. In other words, $post(\langle q, \vec{u}\rangle) = \{\langle q', \vec{u}'\rangle \ : \ \exists a \in \Sigma.\ \langle q, \vec{u}\rangle \xrightarrow{a} \langle q', \vec{u}'\rangle\}$, and $pre(\langle q, \vec{u}\rangle) = \{\langle q', \vec{u}'\rangle \ : \ \exists a \in \Sigma.\ \langle q', \vec{u}'\rangle \xrightarrow{a} \langle q, \vec{u}\rangle\}$. Let $post^*$ and pre^* be the reflexive-transitive closures of *post* and *pre*.

Given a configuration s, a *run* of the system $\mathcal{S}$ starting from s is a finite or infinite sequence $s_0 a_0 s_1 a_1 \ldots s_n$ such that $s = s_0$ and, for every $i \geq 0$, $s_i \xrightarrow{a_i} s_{i+1}$. We denote by $Run_f(s, \mathcal{S})$ (resp. $Run_\omega(s, \mathcal{S})$) the set of finite (resp. infinite) runs of $\mathcal{S}$ starting from s.

A *lossy VASS* is defined as a VASS with a *weak transition relation* $\Longrightarrow$ on configurations. We define the relation $\Longrightarrow$ as follows: $\langle q_1, \vec{u}_1\rangle \stackrel{a}{\Longrightarrow} \langle q_2, \vec{u}_2\rangle$ iff $\exists \vec{u}_1', \vec{u}_2' \in \mathbb{N}^n$, $\vec{u}_1 \geq \vec{u}_1'$, $\langle q_1, \vec{u}_1'\rangle \xrightarrow{a} \langle q_2, \vec{u}_2'\rangle$, and $\vec{u}_2' \geq \vec{u}_2$.

The definition of the weak transition relation induces corresponding notions of runs, successor and predecessor functions defined exactly as in the case of perfect VASS's by considering the weak transition relation $\Longrightarrow$ instead of the relation $\longrightarrow$.

3 Representing Sets of Configurations

Definition 3.1 *Let $\leq$ denote the usual ordering on natural numbers. We extend this relation to vectors of natural numbers in the standard way. Let $\vec{u} = (u_1, \ldots, u_n)$ and $\vec{v} = (v_1, \ldots, v_n)$ be two vectors in $(I\!N \cup \infty)^n$. Then, we have $\vec{u} \leq \vec{v}$ iff $\forall i \in \{1, \ldots, n\}.\ u_i \leq v_i$.*

Given a set $S \subseteq I\!N^n$, we denote by $min(S)$ the set of minimal elements of S w.r.t. the relation $\leq$.

Let $S \subseteq I\!N^n$. Then, S is upward *(resp.* downward*) closed iff $\forall \vec{u} \in I\!N^n.\ \vec{u} \in S \Rightarrow (\forall \vec{v} \in I\!N^n.\ \vec{v} \geq \vec{u}$ (resp. $\vec{v} \leq \vec{u}$) $\Rightarrow \vec{v} \in S)$. Given a set $S \subseteq I\!N^n$, we denote by $S\!\uparrow$ (resp. $S\!\downarrow$) the upward (resp. downward) closure of S, i.e., the smallest upward (resp. downward) closed set which contains S.*

Lemma 3.2 *Every set $S \subseteq I\!N^n$ has a finite number of minimal elements.*

Proof Let $S \subseteq I\!N^n$ and suppose that S has an infinite set of minimals $min(S) = \{\vec{u}_1, \vec{u}_2, \ldots\}$. Then, by Dickson's lemma, there are two indices i and j such that $\vec{u}_i \leq \vec{u}_j$, which contradicts the fact that $\vec{u}_i$ and $\vec{u}_j$ are both minimal elements. ■

Lemma 3.3 *A set is upward closed if and only if $S = min(S)\!\uparrow$.*

Lemma 3.4 *The union and the intersection of two upward (resp. downward) closed sets is an upward (resp. downward) closed set. The complement of an upward closed set is downward closed and vice-versa.*

Proof The cases of union and intersection are trivial. Let S be a downward closed set, and let $\overline{S} = I\!N^n \setminus S$. Then, let us consider two vectors $\vec{u} \in \overline{S}$ and $\vec{v} \in I\!N^n$ such that $\vec{v} \geq \vec{u}$. We have necessarily $\vec{v} \in \overline{S}$, because otherwise (i.e., if $\vec{v} \in S$) $\vec{u}$ must be also in S since it is downward closed, which contradicts the fact that $\vec{u} \in \overline{S}$. A symmetrical argument allows to show that the complement of an upward closed set is downward closed.

Definition 3.5 (Constraints) *Let $\mathcal{X} = \{x_1, \ldots, x_n\}$ be a set of variables ranging over $I\!N$.*

1. *A* simple constraint *over $\mathcal{X}$, SC for short, is any boolean combination of constraints of the form $x \geq c$ where $x \in \mathcal{X}$ and $c \in I\!N \cup \{\infty\}$.*

2. *An* upward closed *(resp.* downward closed*) constraint over $\mathcal{X}$, UC (resp. DC) for short, is any positive boolean combination of constraints of the form $x \geq c$ (resp. $x < c$) where $x \in \mathcal{X}$ and $c \in I\!N \cup \{\infty\}$.*

Constraints are interpreted in the standard way as a subset of $\mathbb{N}^n$ ($\leq$ is the usual ordering and $<$ is the strict inequality). Given a simple constraint ξ, we let $[\![\xi]\!]$ denote the set of vectors in $\mathbb{N}^n$ satisfying ξ. Notice that the constraints $x < 0$ and $x \geq \infty$ correspond to $\emptyset$ and that $x \geq 0$ and $x < \infty$ correspond to $\mathbb{N}$.

Definition 3.6 *A set S is SC (resp. UC, DC) definable if there exists an SC (resp. UC, DC) ξ such that $S = [\![\xi]\!]$.*

Definition 3.7 (Normal forms)

1. *A* canonical product *is a constraint of the form $\vec{\ell} \leq \vec{x} \leq \vec{u}$,*
2. *A* canonical upward closed product *is a constraint of the form $\vec{\ell} \leq \vec{x}$,*
3. *A* canonical downward closed product *is a constraint of the form $\vec{x} \leq \vec{u}$,*

where $\vec{\ell} \in \mathbb{N}^n$ and $\vec{u} \in (\mathbb{N} \cup \{\infty\})^n$.
A SC (resp. UC, DC) in normal form *is either $\emptyset$, or a finite disjunction of canonical (resp. canonical upward closed, canonical downward closed) products.*

Lemma 3.8 *Every SC (resp. UC, DC) is equivalent to a SC (resp. UC, DC) in normal form.*

Lemma 3.9 (Expressiveness) *SC definable sets are closed under boolean operations, and UC definable sets as well as DC definable sets are closed under union and intersection. The complement of a UC definable set is a DC definable set and vice-versa. A subset of $\mathbb{N}^n$ is UC definable (resp. DC definable) if and only if it is an upward (resp. downward) closed set. A set is SC definable if and only if it is a boolean combination of upward closed sets.*

Proof Follows from Lemmas 3.2, 3.3, and 3.4. ■

Let $\mathcal{S} = (\Sigma, \mathcal{X}, Q, \delta)$ be a n-dim VASS with $Q = \{q_1, \ldots, q_m\}$. Then, every set of configurations of $\mathcal{S}$ is defined as a union $C = \{q_1\} \times S_1 \cup \cdots \cup \{q_n\} \times S_m$ where the S_i's are sets of n-dim vectors of natural numbers. The set of configurations C is SC (resp. UC, DC) definable if all the S_i's are SC (resp. UC, DC) definable.

We represent SC definable sets by simple constraints in normal form coupled with control states. From now on, we consider a canonical product to be a pair of the form $\langle q, \vec{\ell} \leq \vec{x} \leq \vec{u} \rangle$ where $q \in Q$. A simple constraint is either $\emptyset$ or a finite disjunction of canonical products. We consider the same convention in the cases of upward closed and downward closed constraints. We use SC$(Q, \mathcal{X})$ (resp. UC$(Q, \mathcal{X})$, DC$(Q, \mathcal{X})$) to denote the set of simple constraints (resp. upward closed, downward closed constraints). We omit the parameters Q and $\mathcal{X}$ when they are known from the context.

4 Computing Successors and Predecessors

Proposition 4.1 *The class SC is effectively closed under the under the operations post and pre for any lossy VASS's.*

Proof First, notice that these operations are distributive w.r.t. union (disjunction). Hence, it suffices to consider separately each transition $\tau = (q, a, \Delta, q')$ and show how to perform them on canonical products:

1. $post_\tau(\langle q, \vec{\ell} \leq \vec{x} \leq \vec{u} \rangle) = \langle q', \vec{x} \leq \vec{u} + \Delta \rangle$.

2. $pre_\tau(\langle q', \vec{\ell} \leq \vec{x} \leq \vec{u} \rangle) = \langle q, (\vec{\ell} - \Delta) \sqcap \vec{0} \leq \vec{x} \rangle$,

where $\forall \vec{u}, \vec{v} \in I\!N^n$, $\vec{u} \sqcap \vec{v}$ is the vector such that $\forall i \in \{1, \ldots, n\}$. $(\vec{u} \sqcap \vec{v})_i = max(u_i, v_i)$. ■

Notice that for lossy VASS's, the *pre* image of any set of configurations is upward closed and its *post* image is downward closed. This also holds for pre^* and $post^*$.

Theorem 4.2 *For every n-dim lossy VASS $\mathcal{S}$, and every n-dim SC set S, the set $pre^*(S)$ is UC definable and effectively constructible.*

Proof Since the set $pre^*(S)$ is upward closed, by Proposition 3.9 we deduce that it is UC definable. The construction of this set is similar to the one given in [CFI96, ACJT96] for lossy channel systems. ■

Theorem 4.3 *For every n-dim lossy VASS $\mathcal{S}$, and every n-dim SC set S, the set $post^*(S)$ is DC definable and effectively constructible.*

Proof Since $post^*(S)$ is downward closed, by Proposition 3.9 we deduce that $post^*(S)$ is DC definable. This set can be constructed using the Karp-Miller algorithm for the construction of the coverability graph. ■

5 Automata and Automata Logic

In this section we define the finite automata we use to express properties of computations. These automata are rather standard, except that they are labeled on states and edges as well. This is natural in the context of specification, since state labels are associated with predicates on the configurations of a given system and edge labels are associated with the actions of the system.

Definition 5.1 *Let Λ and Σ be two finite alphabets. A labeled transition graph over (Λ, Σ) is a tuple $\mathcal{G} = (Q, q_{init}, \Pi, \delta)$ where*

- *Q is a finite set of states,*
- *q_{init} is the initial state,*
- *$\Pi : Q \to \Lambda$ is a state labeling function,*
- *$\delta \subseteq Q \times \Sigma \times Q$ is a finite set of labeled transitions.*

We write $q \xrightarrow{a} q'$ when $(q, a, q') \in \delta$.
Given a state q, a run of $\mathcal{G}$ starting from q is a finite or infinite sequence $q_0 a_0 q_1 a_1 q_2 \ldots$ such that $q_0 = q$ and $\forall i \geq 0.\ q_i \xrightarrow{a_i} q_{i+1}$.

Definition 5.2 (Automata on finite sequences) *A finite-state automaton over (Λ, Σ) on finite sequences is a tuple $\mathcal{A}_f = (Q, q_{init}, \Pi, \delta, F)$ where $(Q, q_{init}, \Pi, \delta)$ is a labeled transition graph over (Λ, Σ), and $F \subseteq Q$ is a set of final states. A finite sequence $\lambda_0 a_0 \lambda_1 a_1 \ldots \lambda_n \in \Lambda(\Sigma\Lambda)^*$ is* accepted *by $\mathcal{A}_f$ if there is a run $q_0 a_0 q_1 a_1 \ldots q_n$ of $\mathcal{A}_f$ starting from q_{init} such that $\forall i \in \{0, \ldots, n\}.\ \Pi(q_i) = \lambda_i$, and $q_n \in F$. We denote by $L(\mathcal{A}_f)$ the set of sequences in $\Lambda(\Sigma\Lambda)^*$ accepted by $\mathcal{A}_f$.*

Definition 5.3 (Büchi ω-automata) *A finite-state Büchi automaton over (Λ, Σ) is a tuple $\mathcal{A}_\omega = (Q, q_{init}, \Pi, \delta, F)$ where $(Q, q_{init}, \Pi, \delta)$ is a labeled transition graph over (Λ, Σ), and $F \subseteq Q$ is a set of repeating states. An infinite sequence $\lambda_0 a_0 \lambda_1 a_1 \ldots \lambda_n \in (\Lambda\Sigma)^\omega$ is* accepted *by $\mathcal{A}_\omega$ if there is a run $q_0 a_0 q_1 a_1 \ldots$ of $\mathcal{A}_\omega$ starting from q_{init} such that $\forall i \geq 0.\ \Pi(q_i) = \lambda_i$, and $\overset{\infty}{\exists}\, i \geq 0.\ q_i \in F$. We denote by $L(\mathcal{A}_\omega)$ the set of sequences in $(\Lambda\Sigma)^\omega$ accepted by $\mathcal{A}_\omega$.*

Definition 5.4 (Closed ω-automata) *A closed ω-automaton is a Büchi automaton $\mathcal{A}_{\omega c} = (Q, q_{init}, \Pi, \delta, F)$ such that $F = Q$.*

Remark 5.5 *[Tho90] Büchi automata define ω-regular sets of infinite sequences. They are closed under boolean operations. Closed ω-automata define closed ω-regular sets in the Cantor topology (the class F in the Borel hierarchy). They correspond to the class of ω-regular safety properties. Closed ω-automata are closed under intersection and union, but not under complementation.*

We introduce an automata-based branching-time temporal logic called AL (Automata Logic). This logic is defined in the spirit of the extended temporal logic ETL [Wol83] and is an extension of ECTL* [CGK87, Tho89] which allows to express temporal properties of (lossy) VASS's involving simple constraints. The logic AL is more expressive than the branching-time logics CTL and CTL*, and allows to express all ∞-regular linear-time properties on finite and infinite computations.

Definition 5.6 (Automata Logic) *Given a set of control states Q and a set of variables $\mathcal{X}$, we let $\mathcal{F}$ denote a subset of $SC(Q, \mathcal{X})$, and we let π range over elements of $\mathcal{F}$. Then, the set of AL($\mathcal{F}$) formulae is defined by the following grammar:*

$$\begin{array}{rl} \varphi ::= & \pi \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \exists\mathcal{A}_f(\varphi_1, \ldots, \varphi_m) \mid \forall\mathcal{A}_f(\varphi_1, \ldots, \varphi_m) \mid \\ & \exists\mathcal{A}_\omega(\varphi_1, \ldots, \varphi_m) \mid \forall\mathcal{A}_\omega(\varphi_1, \ldots, \varphi_m) \end{array}$$

where $\mathcal{A}_f$ (resp. $\mathcal{A}_\omega$) is a finite-state automaton on finite (resp. infinite) sequences over $(\Lambda = \{\lambda_1, \ldots, \lambda_m\}, \Sigma)$. We consider standard abbreviations like $\Rightarrow$.

Definition 5.7 *We use $\star$ to denote f or ω. Let $\mathcal{S} = (\Sigma, \mathcal{X}, Q, \delta)$ be a n-dim (lossy) VASS, We define a satisfaction relation between configurations of $\mathcal{S}$ and AL($\mathcal{F}$) as follows:*

$$\begin{array}{rcl} s \models (q, \xi) & \textit{iff} & State(s) = q \;\; and \;\; Val(s) \in [\![\xi]\!] \\ s \models \neg\varphi & \textit{iff} & s \not\models \varphi \\ s \models \varphi_1 \vee \varphi_2 & \textit{iff} & s \models \varphi_1 \; or \; s \models \varphi_2 \\ s \models \varphi_1 \wedge \varphi_2 & \textit{iff} & s \models \varphi_1 \; and \; s \models \varphi_2 \\ s \models \exists\mathcal{A}_\star(\varphi_1, \ldots, \varphi_m) & \textit{iff} & \exists\rho = s_0 a_0 \ldots \in Run_\star(s, \mathcal{S}).\ \exists\sigma = \lambda_{i_0} a_0 \ldots \in L(\mathcal{A}_\star). \\ & & |\sigma| = |\rho| \; and \; \forall j.\ 0 \leq j < |\rho|.\ s_j \models \varphi_{i_j} \\ s \models \forall\mathcal{A}_\star(\varphi_1, \ldots, \varphi_m) & \textit{iff} & \forall\rho = s_0 a_0 \ldots \in Run_\star(s, \mathcal{S}).\ \exists\sigma = \lambda_{i_0} a_0 \ldots \in L(\mathcal{A}_\star) \\ & & |\sigma| = |\rho| \; and \; \forall j.\ 0 \leq j < |\rho|.\ s_j \models \varphi_{i_j} \end{array}$$

For every formula φ, let $[\![\varphi]\!]_{\mathcal{S}} := \{s \in \mathcal{S} \mid s \models \varphi\}$.

Definition 5.8 (Fragments of AL) *$\exists$AL($\mathcal{F}$) is the fragment of AL that uses only constraints from $\mathcal{F}$, conjunction, disjunction and existential path quantification. $\forall$AL($\mathcal{F}$) is the fragment of AL that uses only constraints from $\mathcal{F}$, conjunction, disjunction and universal path quantification. Let X be (some fragment of) the logic AL. Then X_f (resp. X_ω, $X_{\omega c}$) denote the fragment of X where only automata on finite sequences (resp. Büchi, closed ω-automata) are used.*

AL is a weaker logic than the modal μ-calculus, but many widely known temporal logics are fragments of AL. Every propositional linear-time property, in particular LTL properties, can be expressed by means of automata [VW86, Tho90] and hence they can be expressed in AL. Then, it is easy to see that CTL* is a fragment of AL since every path formula in CTL* corresponds to an LTL formula. Thus, CTL is also a fragment of AL. Clearly, $\forall$AL and $\exists$AL subsume the positive universal and existential fragments of CTL* denoted $\forall$CTL* and $\exists$CTL* (notice that LTL is a fragment of $\forall$CTL*).

We consider two fragments of CTL called EF and EG. The logic EF uses SC predicates, boolean operators, the one-step next operator and the operator EF which is defined by $[\![EF\varphi]\!] = pre^*([\![\varphi]\!])$, i.e., a configuration s satisfies $EF\varphi$ if there is a reachable configuration from s which satisfies φ. The logic EG is defined like EF, except that the operator EF is replaced by the operator EG, which is defined as follows: $s \models EG\varphi$ iff there exists a complete run that starts at s and always satisfies φ. By a complete run we mean either an infinite run or a finite run ending in a deadlock.

We use the subscripts f or ω to denote the fragments of these logics obtained by interpreting their formulas on either finite or infinite paths only. Then, it can be seen that $\mathrm{EF} = \mathrm{EF}_f \subseteq \mathrm{CTL}_f \subseteq \mathrm{CTL}^*_f \subseteq \mathrm{AL}_f$. It can also be seen that EG_ω is a fragment of $\mathrm{AL}_{\omega c}$ but EG is not (due to the finite paths).

Figure 1 shows the relationship between several logics. An arc between two logics means that the higher logic in the graph is more expressive.

6 Model Checking

Definition 6.1 (Model checking and global model checking)

1. *The* model checking problem *is, given a configuration s and a formula φ, whether $s \in [\![\varphi]\!]_{\mathcal{S}}$,*

2. *The* global model checking problem *is whether for any formula φ the set $[\![\varphi]\!]_{\mathcal{S}}$ is effectively constructible.*

Lemma 6.2 *Let $\mathcal{S}$ be a lossy VASS. Then for every formula φ of the form $\exists\mathcal{A}_f(\pi_1,\ldots,\pi_m)$ where all the π_i are SC, the set $[\![\varphi]\!]_{\mathcal{S}}$ is SC definable and effectively constructible.*

Proof First we compute the product of $\mathcal{S}$ and $\mathcal{A}_f$ and obtain a new lossy VASS $\mathcal{S}'$. All states in the finite control of $\mathcal{S}'$ have the form (q,q') where q is a state in the finite control of $\mathcal{S}$ and q' is a state in $\mathcal{A}_f$. The initial state is the

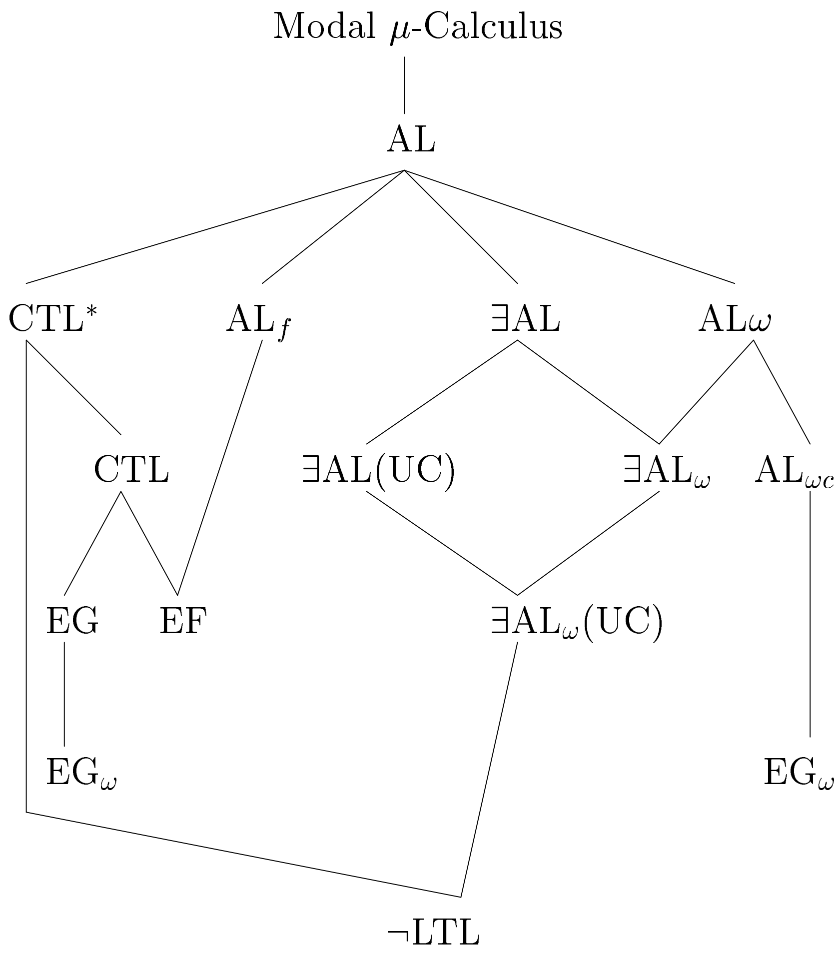

Figure 1: The relative expressive power of several logics.

product of the initial states in $\mathcal{S}$ and $\mathcal{A}_f$. A state in the finite control of $\mathcal{S}'$ is final iff the part of it in $\mathcal{A}_f$ is final. Similarly, each state in the finite control of $\mathcal{S}'$ is assigned the constraint π_i that is assigned to the $\mathcal{A}_f$-part of it. Every constraint π_i is SC definable. Thus the set of final configurations of $\mathcal{S}'$ can be written as a disjunction of canonical products of the form $\langle q_i, \vec{l_i} \leq \vec{x} \leq \vec{u}_i \rangle$, where q_i is a final state of $\mathcal{S}'$.

For every $\langle q_i, \vec{l_i} \leq \vec{x} \leq \vec{u}_i \rangle$ we compute a tree whose nodes are labeled with canonical products with $\langle q_i, \vec{l_i} \leq \vec{x} \leq \vec{u}_i \rangle$ at the root in the following way: For every node p with canonical product P we compute $pre(P)$ with respect to $\mathcal{S}'$. Since $\mathcal{S}'$ is lossy, this is a disjunction of upward closed canonical products.

For each of these we compute the intersection with the π_i assigned to the current state of the finite control of $\mathcal{S}'$. Thus we get a disjunction of canonical products of the form $\langle q, \vec{l} \leq \vec{x} \leq \vec{u}\rangle$ where the upper bound $\vec{u}$ is in a finite set of upper bounds $UB(\pi_1, \ldots, \pi_m)$ that occur in some canonical product in $\pi_1, \ldots, \pi_m$. Every child-node of p is labeled with one of the canonical products in this disjunction. We stop the construction of a branch of the tree when we encounter a node marked with a canonical product s.t. there is a previous node in the branch whose canonical product describes a larger set of states. The set of states of the finite control of $\mathcal{S}'$ is finite. Also all upper bounds used in canonical products in the tree are from the finite set $UB(\pi_1, \ldots, \pi_m)$. Thus by Dickson's Lemma the tree must be finite, because we eventually encounter larger lower bounds than previously in the branch.

Then we take the union of all canonical products in all these trees where the associated control state of $\mathcal{S}'$ has the form (q, q') where q' is an initial state in $\mathcal{A}_f$. We compute the projection on the first component q of the control state and thus get the description of a set of states of $\mathcal{S}$. This is $[\![\varphi]\!]_{\mathcal{S}}$. ■

Theorem 6.3 *The global model checking problem for lossy VASS and the logic AL_f is decidable.*

Proof By induction on the nesting-depth of operators $\exists\mathcal{A}_f$ in the formula. The result follows from boolean operations and Lemma 6.2. ■

The following results even hold for non-lossy VASS. The aim is to show decidability of the global model checking problem for VASS and the logic $\exists AL_\omega(UC)$. We define a generalized notion of configurations of VASS which includes the symbol ω. This symbol denotes arbitrarily high numbers of tokens on a place. It is used as an abbreviation in the following way:

$$\begin{array}{l}\langle q, (\omega, \omega, \ldots, \omega, x_{k+1}, \ldots, x_n)\rangle \models \varphi \;\Leftrightarrow \\ \exists n_1, \ldots, n_k \in I\!N.\ \langle q, (n_1, n_2, \ldots, n_k, x_{k+1}, \ldots, x_n)\rangle \models \varphi\end{array}$$

(Of course the ω can occur at any position, e.g. $\langle q, (x_1, x_2, \omega, x_4, \omega, x_6)\rangle$.)

Lemma 6.4 *Let $\mathcal{S}$ be a VASS and φ a formula of the form $\exists\mathcal{A}_\omega(\pi_1, \ldots, \pi_m)$ where all the π_i are in UC. Let s be a generalized configuration of $\mathcal{S}$ (i.e. it can contain ω). It is decidable if $s \models \varphi$.*

Proof We compute the product of $\mathcal{S}$ and $\mathcal{A}_\omega$ and get a new VASS $\mathcal{S}'$. The states of the finite control of $\mathcal{S}'$ have the form (q, q') where q is a state of the finite control of $\mathcal{S}$ and q' is a state of $\mathcal{A}_\omega$. (q, q') is repeating iff q' is repeating in $\mathcal{A}_\omega$. It is assigned the constraint π_i that is assigned to q'. The initial state s' of $\mathcal{S}'$ is the product of s with the initial state q_0 of $\mathcal{A}_\omega$.

We search for an infinite run of $\mathcal{S}'$ that satisfies the assigned constraint π_i at every state and visits some repeating state infinitely often. The chances for finding such a run are always bigger if the initial state is larger, because all π_i are upward closed.

First we compute the Karp-Miller coverability graph [KM69] of $\mathcal{S}'$ with root s'. Only those nodes are considered that satisfy the assigned constraint π_i. By Dickson's Lemma this graph is finite. If there exists an infinite accepting run then there must be an infinite cyclic accepting run that starts at a Büchi-repeating node in this graph. This run corresponds to a cyclic path in the coverability graph that starts and ends at a Büchi-repeating node and has an overall positive effect of all fired transitions and can thus be repeated infinitely often.

For every Büchi-repeating node n in the coverability graph we compute a finite-state automaton A_n on finite sequences such that its labeled transition graph is the largest strongly connected subgraph containing n, and its initial state and its only final state is n. We label every arc in A_n with a unique symbol λ_i. To every λ_i we assign an effect-vector $\Delta_i \in \mathbb{Z}^n$ that describes the effect of the transition that was fired in the step from one node to the other. Let k be the number of states in A_n.

The aim is to find a cyclic path in A_n from node n back to n where the sum of all effect-vectors of all traversed arcs is $\geq \vec{0}$. (Note that the effect-vector of an arc that is traversed j times counts j times, i.e. it is multiplied by j.) Such a cyclic path with positive overall effect corresponds to a possible infinite accepting run of the system $\mathcal{S}'$.

Since n is the initial state and the only final state in A_n, every word in $L(A_n)$ corresponds to a cyclic path from n to n. For any word w, let $|w|_{\lambda_i}$ be the number of occurrences of λ_i in w. The question is now if there is a word $w \in L(A_n)$ s.t.

$$\sum_{1 \leq i \leq k} |w|_{\lambda_i} \Delta_i \geq \vec{0}$$

This is decidable, since the set $\{(|w|_{\lambda_1}, \ldots, |w|_{\lambda_k}) \mid w \in L(A_n)\}$ is semilinear by Parikh's Theorem [Par66]. We check this condition for every A_n, and we have $s \models \varphi$ if and only if a positive linear combination is found. ■

Lemma 6.5 *Let $\mathcal{S}$ be a VASS and φ a formula of the form $\exists \mathcal{A}_\omega(\pi_1, \ldots, \pi_m)$ where all the π_i are in UC. The set $[\![\varphi]\!]_{\mathcal{S}}$ is UC definable and effectively constructible.*

Proof The set $[\![\varphi]\!]_{\mathcal{S}}$ is upward closed, because all π_i are upward closed. Thus, this set is characterized by the finite set of its minimal elements (see

Lemma 3.2 and 3.3). To find the minimal elements, we use a construction that was described by Valk and Jantzen in [VJ85]. The important point here is that we can use Lemma 6.4 to check the existence of configurations that satisfy φ. For example, if $\langle q, (\omega, x_2, x_3)\rangle \models \varphi$ then we can check if $\langle q, (n_1, x_2, x_3)\rangle \models \varphi$ for $n_1 = 0$, $n_1 = 1$, $n_1 = 2$, ... until we find the minimal n_1 s.t. $\langle q, (n_1, x_2, x_3)\rangle \models \varphi$. ■

Theorem 6.6 *The global model checking problem is decidable for VASS and the logic* $\exists AL_\omega(UC)$.

Proof By induction on the nesting-depth of the formula and Lemma 6.5. ■

Theorem 6.7 *The global model checking problem is decidable for lossy VASS and the logic* $\exists AL(UC)$.

Proof By induction on the nesting depth of the formula and Theorem 6.3 and Theorem 6.6. ■

Theorem 6.8 *The model checking problem for lossy VASS and the logic* $AL_{\omega c}$ *is decidable.*

Proof Let $\mathcal{S}$ be a lossy VASS and φ a $AL_{\omega c}$ formula. Let s be some state of $\mathcal{S}$. We show decidability of the question $s \models \varphi$ by induction on the nesting-depth of the operator $\exists\mathcal{A}_{\omega c}$ in the formula. The base case where φ contains no operator $\exists\mathcal{A}_{\omega c}$ is trivial. In the general case we use boolean operations to reduce the problem to problems of the form $s \models \exists\mathcal{A}_{\omega c}(\varphi_1, \ldots, \varphi_n)$ where the φ_i have a smaller nesting-depth. We compute the product of $\mathcal{S}$ and $\mathcal{A}_{\omega c}$ and construct the tree of all possible successors of s that satisfy the assigned φ_i. (By induction hypothesis we can check if $s \models \varphi_i$ for any state s.) The construction of a branch stops if one of the following conditions is satisfied: (1) There is no successor that satisfies the assigned φ_i. (2) We reach a node with the same state of the finite control, but a larger marking than a previous node with a state s'.

By Dickson's Lemma every branch has finite length. Since the tree is finitely branching, it is finite and can be effectively constructed. Since $\mathcal{S}$ is lossy and all states in $\mathcal{A}_{\omega c}$ are repeating states, we know that $s \models \exists\mathcal{A}_{\omega c}(\varphi_1, \ldots, \varphi_n)$ iff some branch of the tree terminates by condition 2. ■

Theorem 6.9 *Model checking lossy VASS with the logic EG is decidable.*

Proof The proof is very similar to the proof of Theorem 6.8. The only differences are that one also has to consider finite runs that end in a deadlock and the one-step next operator. This can easily be added to the construction in Theorem 6.8. ■

Theorems 6.8 and 6.9 say that the model checking problem is decidable for a lossy VASS and an EG-formula/ $AL_{\omega c}$-formula φ. However, in both cases the set $[\![\varphi]\!]_{\mathcal{S}}$ is not effectively constructible (although it is SC definable). If it were constructible then Lemma 6.2 could be used to decide model checking lossy VASS with formulae of the form $EFEG_\omega\pi$, where π is a constraint in SC. However, this problem has very recently been shown to be undecidable.

Proposition 6.10 *Model checking lossy VASS with formulae of the form $EFEG_\omega\pi$, where π is a constraint in SC is undecidable.*

Proof This is a corollary of a more general undecidability result for lossy BPP (Basic Parallel Processes), which follows (not immediately) from the result on lossy counter machines in Proposition 7.2 (see [May98]). ∎

Remark 6.11 *This undecidability result also implies undecidability of model checking lossy VASS with the logic $\exists AL_\omega$. One can encode properties of the form $EFEG_\omega\pi$ in $\exists AL_\omega$ in the following way: Let $\mathcal{A}_\omega$ be an automaton with states q, q', and transitions $q \to q$, $q \to q'$ and $q' \to q'$ which are labeled with any action. The predicate true is assigned to q and the predicate π is assigned to q'. q is the initial state and q' is the only repeating state. Let $\mathcal{A}'_\omega$ be an automaton with only one state q which is the initial state and repeating and a transition $q \to q$ with any action. The predicate π is assigned to q. Then for any lossy VASS s we have*

$$s \models EFEG_\omega\pi \iff s \models \mathcal{A}_\omega(true, \pi) \vee \mathcal{A}'_\omega(\pi)$$

7 Lossy VASS with Inhibitor Arcs

Lossy VASS can be extended with inhibitor arcs. This means introducing transitions that can only fire if some defined places are empty (i.e. they can test for zero). Thus lossy VASS with inhibitor arcs are equivalent to lossy counter machines. Normal VASS with inhibitor arcs are Turing-powerful, but lossy VASS with inhibitor arcs are not.

Theorem 7.1 *For lossy VASS with inhibitor arcs*

1. *the global model checking problem is decidable for the logic AL_f.*
2. *model checking is decidable for the logics $AL_{\omega c}$ and EG.*

Proof The constructions in Lemma 6.2 and Theorem 6.8 carry over directly to lossy VASS with inhibitor arcs. ∎

Inhibitor arcs can never keep a transition from firing, because one can just loose the tokens on the places that inhibit it. However, after such a transition has fired, the number of tokens on the inhibiting places is fixed and known exactly. Such a guarantee is impossible to achieve in lossy VASS without inhibitor arcs. Thus not all results for lossy VASS carry over to lossy VASS with inhibitor arcs.

Proposition 7.2 *Let $\mathcal{S}$ be a lossy VASS with inhibitor arcs. It is undecidable if there exists an initial configuration s s.t. there is an infinite run of the system $(s, \mathcal{S})$.*

Proof This is a corollary of a more general undecidability result for lossy counter machines in [May98]. The main idea is that one can enforce that lossiness occurs only finitely often in the infinite run.

Theorem 7.3 *Model checking lossy VASS with inhibitor arcs with the logic LTL is undecidable.*

Proof We reduce the problem of Proposition 7.2 to the model checking problem. We construct a lossy VASS with inhibitor arcs $\mathcal{S}'$ that does the following: First it guesses an arbitrary configuration s of $\mathcal{S}$ doing only the atomic action a. Then it simulates $\mathcal{S}$ on s doing only the atomic action b. Let $\mathcal{A}_\omega$ be a Büchi-automaton with initial state q and repeating state q' and transitions $q \xrightarrow{a} q$, $q \xrightarrow{b} q'$ and $q' \xrightarrow{b} q'$. Let s' be the initial state of $\mathcal{S}'$. We have reduced the question of Proposition 7.2 to the question if $(s', \mathcal{S}') \models \exists \mathcal{A}_\omega(true, true)$. This question can be expressed in LTL. ■

It follows immediately that model checking lossy VASS with inhibitor arcs with $AL_\omega(UC)$ is undecidable. It is interesting to compare this result with Proposition 6.10. It shows that for undecidability of the model checking problem it suffices to have either inhibitor arcs in the system or downward closed constraints in the logic. One can be encoded in the other and vice versa.

The set $post^*(s)$ is DC definable since it is downward closed. However, it is not constructible for lossy VASS with inhibitor arcs (unlike for lossy VASS, see Theorem 4.3).

Theorem 7.4 *The set $post^*(s)$ is not constructible for lossy VASS with inhibitor arcs.*

Proof Boundedness is undecidable for reset Petri nets [DFS98]. This result carries over to lossy reset Petri nets. Lossy VASS with inhibitor arcs can simulate lossy reset Petri nets. It follows that boundedness is undecidable for lossy VASS with inhibitor arcs and thus $post^*(s)$ is not constructible. ■

8 Conclusion

We have addressed verification problems for VASS with lossy variables. These systems can model communicating systems with unbounded lossy unordered buffers. They can be seen as abstractions of (lossy) fifo-channel systems. Considering lossy VASS instead of lossy fifo-channel systems has several advantages: First, the set of reachable configurations backward and forward is effectively constructible, while for lossy channel systems only the backward reachability set is computable. This is interesting, since in practice forward reachability analysis can be more efficient than backward analysis, and it can be combined with efficient graph exploration techniques (like on-the-fly, partial-order based procedures) in order to avoid state-explosion. Another advantage of considering lossy VASS is that in addition to safety properties (Theorem 6.3), also liveness properties can be checked and fairness constraints can be taken into account (Theorem 6.7), while it has been shown that this is impossible for lossy fifo-channel systems [AJ96].

In our work, we have also established results for normal VASS and lossy VASS with inhibitor arcs (lossy counter machines). Interestingly, it turns out that these two models are incomparable. Moreover, all the positive/negative results we obtained for lossy VASS with inhibitor arcs are the same as for lossy fifo-channel systems. So, an interesting open question is whether these two models are comparable.

The following table summarizes the results on the decidability of model checking for VASS, lossy VASS with test for zero, lossy VASS and lossy fifo-channel systems. By '++' we denote the fact that for any formula φ the set $[\![\varphi]\!]$ is SC definable and effectively constructible (global model checking), while '+' means that only model checking is decidable. We denote by — that model checking is undecidable. The symbol '?' denotes an open problem.

Logic	**VASS**	**Lossy VASS+0**	**Lossy VASS**	**Lossy fifo**
AL_f/EF	— [Esp97]	++	++ [AJ93]	++ [AJ93]
$\exists AL_\omega(UC)/LTL$	++ /+[Esp94]	—	++	— [AJ96]
$\exists AL(UC)$	?	—	++	— [AJ96]
$AL_{\omega c}$/EG	— [EK95]	+	+ [AJ93]	+ [AJ93]
$\exists AL_\omega$/CTL	— [EK95]	—	—	— [AJ96]

The results in this table are new, except where references are given. For normal VASS and LTL, decidability of the model checking problem was known [Esp94], but the construction of the set $[\![\varphi]\!]$ is new. The results in [AJ93] are just about EF and EG formulas without nesting, not for the full logics AL_f and $AL_{\omega c}$.

As for the open problem in this table, note that we can decide for a VASS

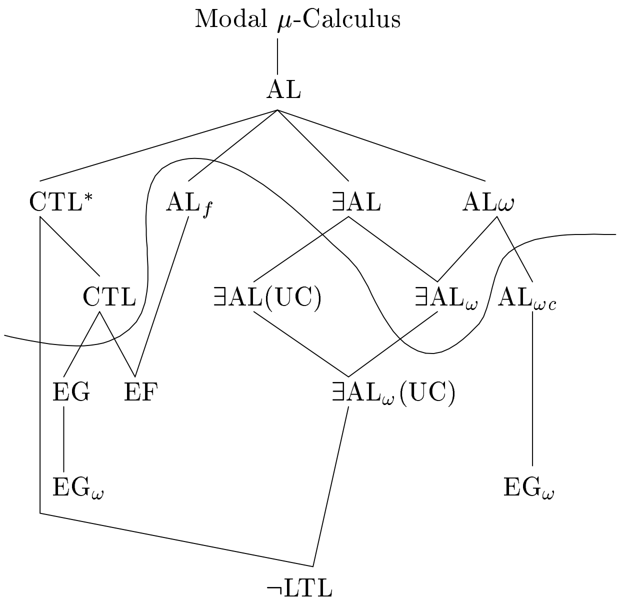

Figure 2: Decidability of the model checking problem for lossy VASS.

and any ∃AL(UC) formula, whether the set $[\![\varphi]\!]$ is SC definable, and in this case we can effectively construct it and decide the model checking question. However, if this set is not SC definable (in general this set could be even not semilinear), we cannot answer the model checking question.

The main new result in this paper is that some liveness properties, as described by $\exists AL_\omega(UC)$, are decidable for lossy VASS, unlike for lossy FIFO-channel systems. Also these liveness properties are undecidable for lossy VASS with test for zero. The general conclusion is that safety properties are always decidable for lossy systems, while for liveness properties this depends on the particular model and on the atomic propositions used in the logic.

The Figures 2 and 3 show the limits of the decidability of the model checking problem for lossy VASS and lossy VASS with inhibitor arcs. Model checking is decidable for the less expressive logics below the border and undecidable for those above it.

Acknowledgment: We thank Peter Habermehl for interesting discussions.

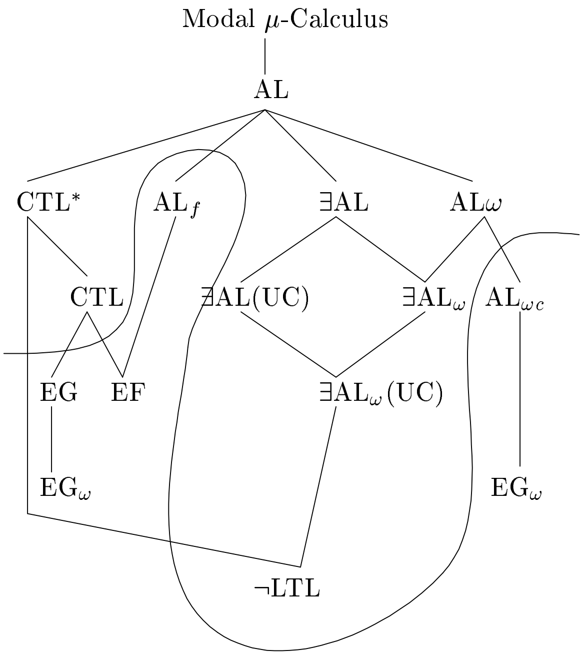

Figure 3: Decidability of the model checking problem for lossy VASS with inhibitor arcs.

References

[ABJ98] P. Abdulla, A. Bouajjani, and B. Jonsson. On-the-fly Analysis of Systems with Unbounded, Lossy Fifo Channels. In *10th Intern. Conf. on Computer Aided Verification (CAV'98)*. LNCS 1427, June 1998.

[ACJT96] P. Abdulla, K. Cerans, B. Jonsson, and Y-K. Tsay. General Decidability Theorems for Infinite-state Systems. In *LICS'96*. IEEE, 1996.

[AJ93] P. Abdulla and B. Jonsson. Verifying Programs with Unreliable Channels. In *LICS'93*. IEEE, 1993.

[AJ96] P. Abdulla and B. Jonsson. Undecidable verification problems for programs with unreliable channels. *Information and Computation*, 130(1):71–90, 1996.

[BGWW97] B. Boigelot, P. Godefroid, B. Willems, and P. Wolper. The power of QDDs. In *SAS'97*. LNCS 1302, 1997.

[BH97] A. Bouajjani and P. Habermehl. Symbolic Reachability Analysis of FIFO-Channel Systems with Nonregular Sets of Configurations. In *24th International Colloquium on Automata, Languages and Programming (ICALP'97)*. LNCS 1256, July 1997.

[CFI96] Gérard Cécé, Alain Finkel, and S. Purushothaman Iyer. Unreliable Channels Are Easier to Verify Than Perfect Channels. *Information and Computation*, 124(1):20–31, 1996.

[CGK87] E. Clarke, O. Grümberg, and R. Kurshan. A Synthesis of two Approaches for Verifying Finite State Concurrent Systems. In *manuscript, Carnegie Mellon University*, 1987.

[DFS98] C. Dufourd, A. Finkel, and Ph. Schnoebelen. Reset nets between decidability and undecidability. In *Proc. of ICALP'98*, volume 1443 of *LNCS*. Springer Verlag, 1998.

[EK95] J. Esparza and A. Kiehn. On the model checking problem for branching time logics and Basic Parallel Processes. In *CAV'95*, volume 939 of *LNCS*, pages 353–366. Springer Verlag, 1995.

[Esp94] J. Esparza. On the decidability of model checking for several μ-calculi and Petri nets. In *Trees in Algebra and Programming – CAAP'94*, volume 787 of *LNCS*. Springer Verlag, 1994.

[Esp97] J. Esparza. Decidability of model checking for infinite-state concurrent systems. *Acta Informatica*, 34:85–107, 1997.

[GL94] O. Grumberg and D. Long. Model Checking and Modular Verification. *ACM Transactions on Programming Languages and Systems*, 16, 1994.

[KM69] R. Karp and R. Miller. Parallel program schemata. *JCSS*, 3, 1969.

[May98] R. Mayr. Lossy counter machines. Technical Report TUM-I9827, TU-München, October 1998. wwwbrauer.informatik.tu-muenchen.de/$\sim$mayrri.

[Par66] R.J. Parikh. On Context-Free Languages. *JACM*, 13, 1966.

[Tho89] W. Thomas. Computation Tree Logic and Regular ω-Languages. LNCS 354, 1989.

[Tho90] W. Thomas. Automata on Infinite Objects. In *Handbook of Theo. Comp. Sci.* Elsevier Sci. Pub., 1990.

[VJ85] R. Valk and M. Jantzen. The Residue of Vector Sets with Applications to Decidability Problems in Petri Nets. *Acta Informatica*, 21, 1985.

[VW86] M.Y. Vardi and P. Wolper. An Automata-Theoretic Approach to Automatic Program Verification. In *LICS'86*. IEEE, 1986.

[Wol83] P. Wolper. Temporal logic can be more expreessive. *Information and Control*, 56, 1983.