Well-Abstracted Transition Systems (extended abstract)

A. Finkel[1]⋆, S. P. Iyer[2]⋆⋆, and G. Sutre[1]

[1] LSV, ENS Cachan & CNRS UMR 8643, 61, Av. du Pdt. Wilson, Cachan, France
[2] Dept of Computer Science, NC State University, Raleigh, NC 27615, USA

Abstract. Formal methods based on symbolic representations have been found to be very effective. In the case of infinite state systems, there has been a great deal of interest in *accelerations* – a technique for characterizing the result of iterating an execution sequence an arbitrary number of times, in a sound, but not necessarily complete, way. We propose the use of abstractions as a general framework to design accelerations. We investigate SemiLinear Regular Expressions (SLREs) as symbolic representations for FIFO automata. In particular, we show that SLREs are *easy to manipulate* (inclusion between two SLREs is in NP $\cap$ coNP), they form the *core* of known FIFO symbolic representations ($SLREs = QDDs \cap CQDDs$), and they are usually *sufficient* since for FIFO automata with one channel, an arbitrary iteration of a loop is LRE representable.

1 Introduction

Formal methods are now routinely applied in design and implementation of finite state systems, such as those that occur in VLSI circuits. It has also been applied fairly regularly in the design and implementation of network protocols. Based on the success of formal methods in reasoning about finite state systems [BCM+92] there has been a great deal of interest in reasoning about infinite state systems. Given that programs, as well as network protocols, are infinite state in nature there is a need for automatic techniques to extend the reach of formal methods to a much larger class.

Infinite state systems could, in general, be Türing-powerful. Consequently, in reasoning about any non-trivial property of such systems we will have to contend with incompleteness. At least two approaches have been considered in the literature: (a) semi-computation of the set of reachable states [BGWW97, BH97, AAB99], and (b) computation of a superset of reachability set [CC77, PP91]. A requirement common to both approaches is that an infinite set of reachable states (from some given initial state) be finitely described. Clearly, the finite description should be such that it admits questions of membership and emptiness to be answered effectively. However, given that the reachability set is

⋆ Supported in part by FORMA, a project funded by DGA, CNRS and MENSER.
⋆⋆ Supported in part by ARO under grant DAAG55-98-1-03093.

explored in an iterative fashion, an even more important question is "how does one infer the existence of an infinite set of states in the reachability set? And how does one calculate it?" Techniques called *accelerations* or *meta*-transitions have been discussed in the literature [Céc98, BGWW97, BH97, AAB99, CJ98, FS00]. We focus in this paper on symbolic representations for the computation of the reachability set of FIFO automata — a finite control with multiple unbounded FIFO channels. To the best of our knowledge, Pachl uses for the first time regular expressions to represent infinite sets of channel contents [Pac87]. In [FM96], linear regular expressions have been defined and used. Boigelot et al. chosed a deterministic finite automata based representation [BGWW97] and afterwards Bouajjani et al. added Pressburger formulae [BH97]. Simple regular expressions have been introduced for lossy FIFO automata [AAB99].

We propose to address the issue "what are symbolic representations?" In this paper, we show how symbolic representations and accelerations can be couched in terms of abstract interpretation [CC77], a powerful semantics-based technique for explaining data flow analysis. We illustrate the usefulness of this approach by exploring Linear and SemiLinear Regular Expressions (LREs and SLREs) as symbolic representations for FIFO automata. In particular, we show the following about SLREs:

- SLREs are *easy to manipulate*: indeed, SLREs are exactly regular languages of *polynomial density*. This class enjoys good complexity properties. In particular, we prove that inclusion between two SLREs is in NP $\cap$ coNP.
- they form the *core* of known FIFO symbolic representations: more formally, a set of queue contents is SLRE representable iff it is both CQDD representable and QDD representable ($SLREs = QDDs \cap CQDDs$),
- and they are usually *sufficient* since for FIFO automata with one channel, an arbitrary iteration of a loop is LRE representable. This result was used, but not proved, in [FM96]. The proof is technical and it can be found in the full paper [FIS00]. Several examples in the literature have a SLRE representable reachability set: the Alternating Bit Protocol [Pac87], the bounded retransmission protocol of Philips [AAB99], the producer/consumer described in [BGWW97] and the connection/deconnection protocol [JJ93].

The road map for the paper is as follows: in Section 2 we introduce the notion of abstractions, in Section 3 we discuss FIFO automaton and symbolic representations based on SLREs. In Section 4 we introduce our notion of symbolic representations as abstractions. In Section 5 we introduce the notion of acceleration in our setting, and also discuss accelerations based on LREs. Finally, in Section 6 we compare QDDs, CQDDs and SLREs.

2 Labelled transition systems

We write 2^S (resp. $\mathcal{P}_f(S)$) to denote the set of subsets (resp. finite subsets) of a set S. Let S^I be the set of I-indexed (I finite) vectors of elements of S. Given $\mathsf{w} \in S^I$, $i \in I$ and $s \in S$, define $\mathsf{w}[i := s]$ to be the vector w' that differs from w

at the index i ($\mathsf{w}'[i] = s$), but is the same elsewhere. Any element $s \in S$ induces the vector $\mathsf{s} \in S^I$ defined by $\mathsf{s}[i] = s$ for all $i \in I$. An I-indexed vector $\mathsf{w} \in S^I$ may also be denoted anonymously by $< i \mapsto \mathsf{w}[i] >$.

A *quasi-ordered set* is a pair $(S, \preceq)$ where S is a set and $\preceq$ is a reflexive and transitive relation over S ; we denote by $\prec$ the relation over S defined by $a \prec b$ if $a \preceq b$ and $b \not\preceq a$. A quasi-ordered set $(S, \preceq)$ satisfies the *ascending chain condition* if there does not exist an infinite strictly increasing sequence $s_0 \prec s_1 \prec s_2 \cdots s_k \prec s_{k+1} \cdots$ in S.

Let Σ be an alphabet (a finite, non empty set). We write Σ^* (resp. Σ^ω) for the set of all *finite words* (resp. *infinite words*), and ε denotes the empty word. We denote by Σ^+ the set $\Sigma^* \setminus \{\varepsilon\}$. For two words $x, y \in \Sigma^*$, $x \cdot y$ (shortly written xy) is their *concatenation*. A word y is a *left factor* of a word x, written $y \leq x$, if $x = y \cdot z$ for some word z and moreover we write $z = y^{-1}x$. The pair $(\Sigma^*, \leq)$ is a quasi-ordered set.

Definition 2.1. *A* labelled transition system *LTS is a triple* $LTS = (\mathsf{S}, \Sigma, \rightarrow)$ *where* S *is a set of states,* Σ *is a finite set a labels and* $\rightarrow \subseteq \mathsf{S} \times \Sigma \times \mathsf{S}$ *is a labelled transition relation.*

For any state $s \in \mathsf{S}$, we write $s \xrightarrow{\varepsilon} s$. If $\sigma = l_1 l_2 \cdots l_k$ is a finite word in Σ^+ and $s, s' \in \mathsf{S}$, we write $s \xrightarrow{\sigma} s'$ when there exists a finite sequence $s_0, s_1, \ldots, s_k \in \mathsf{S}$ such that $s = s_0$, $s' = s_k$ and $s_0 \xrightarrow{l_1} s_1 \xrightarrow{l_2} s_2 \cdots s_{k-1} \xrightarrow{l_k} s_k$.

Note that the labelled transition relation $\rightarrow$ captures the system behavior as it moves from one state to another. Since the central problem we wish to discuss is based on set of states reachable from a given state s by the $\rightarrow$ relation we associate with every labelled transition system LTS, two total functions $post : 2^{\mathsf{S}} \times \Sigma^* \rightarrow 2^{\mathsf{S}}$ and $post^* : 2^{\mathsf{S}} \rightarrow 2^{\mathsf{S}}$ defined by:

$$post(X, \sigma) = \{s' \in \mathsf{S} \mid \exists\ s \in X, s \xrightarrow{\sigma} s'\}, \text{ and, } post^*(X) = \bigcup_{\sigma \in \Sigma^*} post(X, \sigma)$$

The main aim of our paper is a fast computation of $post^*(s_0)$, for a given $s_0 \in \mathsf{S}$. However, in general, $post^*(s_0)$ could be infinite (and even recursively enumerable). To deal with this problem symbolic representations have been used [BGWW97, BH97, CJ98]. Typically, assumptions are made about what properties these symbolic representations should satisfy. In section 4 we will discuss a minimal set of assumption on these abstractions (*aka* symbolic representations) and how they relate to transition systems.

3 FIFO symbolic representations to compute *post**

Let us present the very well-known Alternating Bit Protocol. It is generally modelled by a system of 2 extended automata which communicate through 2 one directional FIFO channels. It is clear that we may always compute the cartesian product of the 2 extended automata, and this yields a single extended automaton with 2 bi-directional FIFO channels, namely a *FIFO automaton.*

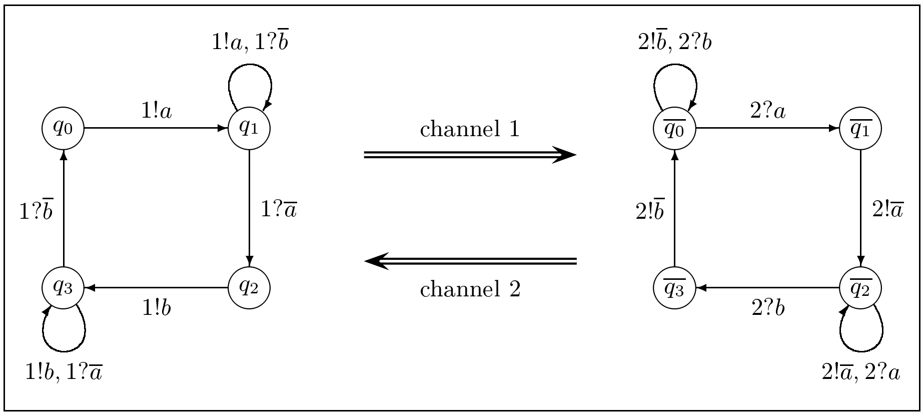

Fig. 1. A FIFO automaton ABP modeling the Alternating Bit Protocol

FIFO automata – a finite control with a set of FIFO channels – is a turing powerful [BZ83] mathematical model for the protocol specification languages Estelle and SDL. Even one channel FIFO automata can simulate Turing Machines and hence we cannot expect to find an algorithm computing exactly $post^*$. However, we will define a general semi-algorithm using symbolic representations, which computes exactly $post^*$ (or an over-approximation).

3.1 FIFO automata

A FIFO automaton is a 4-tuple $F = (Q, C, M, \delta)$ where Q is a finite set of control states, C is a finite set of channel names, and, M is a finite set of message types, and, $\delta \subseteq Q \times (C \times \{?, !\} \times M) \times Q$ is a finite set of control transitions.

Transitions of the form $(q, c?m, q')$ denote receive actions and transitions of the form $(q, c!m, q')$ denote send actions. A *loop* in a FIFO automaton is any non-empty word $\sigma = (q_0, l_0, q'_0)(q_1, l_1, q'_1) \cdots (q_k, l_k, q'_k) \in \delta^*$ such that $q'_k = q_0$ and for all $0 \leq i \leq k-1$, $q'_i = q_{i+1}$. Moreover, we say that σ is a loop *on* q_0.

Example 3.1. For ABP, we have e.g. that $\sigma_1 = (q_1, \overline{q_0}) \xrightarrow{1!a} (q_1, \overline{q_0})$ and $\sigma_2 = (q_0, \overline{q_0}) \xrightarrow{1!a} (q_1, \overline{q_0}) \xrightarrow{1?\overline{a}} (q_2, \overline{q_0}) \xrightarrow{2!\overline{b}} (q_2, \overline{q_0}) \xrightarrow{1!b} (q_3, \overline{q_0}) \xrightarrow{1?\overline{b}} (q_0, \overline{q_0})$ are loops.

We assume that the channels are perfect and that messages sent by a sender are received by a receiver in the same order they are sent. The operational semantics of a FIFO automaton is given by the following labelled transition system.

Definition 3.2. *A FIFO automaton $F = (Q, C, M, \delta)$ defines the labelled transition system $LTS(F) = (\mathsf{S}, \Sigma, \rightarrow)$ as follows:*

- $\mathsf{S} = Q \times (M^*)^C$*, the set of states containing a control state q and a C-indexed vector of words* w *denoting the channel contents.*

- $\Sigma = \delta$.
- $(q, \mathsf{w}) \xrightarrow{(p,c?m,p')} (q', \mathsf{w}')$ *provided* $q = p$, $q' = p'$ *and* $\mathsf{w} = \mathsf{w}'[c := m\mathsf{w}'[c]]$.
- $(q, \mathsf{w}) \xrightarrow{(p,c!m,p')} (q', \mathsf{w}')$ *provided* $q = p$, $q' = p'$ *and* $\mathsf{w}' = \mathsf{w}[c := \mathsf{w}[c]m]$.

For convenience, we write $(q, \mathsf{w}) \xrightarrow{c?m} (q', \mathsf{w}')$ (resp. $(q, \mathsf{w}) \xrightarrow{c!m} (q', \mathsf{w}')$) when we have $(q, \mathsf{w}) \xrightarrow{(q,c?m,q')} (q', \mathsf{w}')$ (resp. $(q, \mathsf{w}) \xrightarrow{(q,c!m,q')} (q', \mathsf{w}')$).

Example 3.3. For ABP, we have for instance: $(q_0, \overline{q_0}; \varepsilon, \varepsilon) \xrightarrow{1!a} (q_1, \overline{q_0}; a, \varepsilon) \xrightarrow{1!a} (q_1, \overline{q_0}; aa, \varepsilon) \xrightarrow{2?a} (q_1, \overline{q_1}; a, \varepsilon) \xrightarrow{2!\overline{a}} (q_1, \overline{q_2}; a, \overline{a}) \xrightarrow{2!\overline{a}} (q_1, \overline{q_2}; a, \overline{a}\overline{a}) \xrightarrow{1?\overline{a}} (q_2, \overline{q_2}; a, \overline{a})$.

3.2 Linear regular expressions

Since *post** may be infinitely large for FIFO automata, we need to finitely represent infinite sets of states in order to compute *post**. Moreover, the unboundedness of *post** arises from the unbounded channels (the set of control states is finite). Thus, a natural way to proceed is to finitely represent infinite sets of channel contents, i.e. vectors of words.

Example 3.4. For ABP, we have:

$$\begin{array}{rlll} post^*(q_0, \overline{q_0}; \varepsilon, \varepsilon) = & (q_0, \overline{q_0}) \times (b^*, \overline{b}^*) & \cup & (q_1, \overline{q_0}) \times (b^* a\, a^*, \overline{b}^*) \\ \cup & (q_1, \overline{q_1}) \times (a^*, \overline{b}^*) & \cup & (q_1, \overline{q_2}) \times (a^*, \overline{b}^* \overline{a}\, \overline{a}^*) \\ \cup & (q_2, \overline{q_2}) \times (a^*, \overline{a}^*) & \cup & (q_3, \overline{q_2}) \times (a^* b\, b^*, \overline{a}^*) \\ \cup & (q_3, \overline{q_3}) \times (b^*, \overline{a}^*) & \cup & (q_3, \overline{q_0}) \times (b^*, \overline{a}^* \overline{b}\, \overline{b}^*) \end{array}$$

In the rest of the paper, we write M for a finite set of message types and C for a finite set of channel names. Following Pachl, we will use regular expressions to represent infinite sets of FIFO channel contents. We write $REG(M)$ for the set of *regular expressions (REGs)* over M. We denote by $[\![\rho]\!]$ the language associated to a regular expression ρ, and we write $\emptyset$ for the regular expression denoting an empty set of words.

To compactly represent the result of receiving a message from the front of a channel we will use the notion of derivatives [Brz64]. We will thus write, for a regular expression ρ and a message $m \in M$, $m^{-1}(\rho)$ to denote a regular expression with meaning $[\![m^{-1}(\rho)]\!] = \{y \mid my \in [\![\rho]\!]\}$.

Let us define the two following subclasses of regular expressions:

- A *linear regular expression (written LRE)* is any regular expression of the form $x_0 {y_0}^* x_1 {y_1}^* \cdots x_{n-1} {y_{n-1}}^* x_n$ with $x_i \in M^*$ and $y_j \in M^+$.
- A *semilinear regular expression (written SLRE)* is any finite sum of LREs (possibly the empty sum $\emptyset$).

We write $LRE(M)$ (resp. $SLRE(M)$) for the set of linear (resp. semilinear) regular expressions over M.

Example 3.5. Note that $post^*$ for the Alternating Bit Protocol is composed of 16 LREs. There are other examples in the literature where $post^*$ can be described by SLREs, namely the bounded retransmission protocol of Philips [AAB99], the producer/consumer described in [BGWW97] and the connection/deconnection protocol [JJ93].

We define the *size* of LREs and SLREs as follows:

- the size $|\rho|$ of a LRE $\rho = x_0 {y_0}^* x_1 {y_1}^* \cdots x_{n-1} {y_{n-1}}^* x_n$ is given by $|\rho| = |x_n| + \Sigma_{i=0}^{n-1}(|x_i| + |y_i| + 1)$, and,
- the size $|\rho|$ of a SLRE $\rho = \rho_0 + \rho_1 \cdots + \rho_m$ is given by $|\rho| = \Sigma_{i=0}^{m}|\rho_i|$.

This definition will allow us to express a complexity result about SLREs in Section 6. The following proposition expresses a strong property of SLREs and it is used to prove Theorem 5.5 and Theorem 6.6. The proof is achieved by induction on the size of a SLRE.

Proposition 3.6. *The class of SLRE languages is closed under intersection with regular languages.*

3.3 FIFO symbolic representations

In order to handle different kinds of symbolic representations for FIFO automata (based on QDDs [BGWW97], CQDDs [BH97] or SLREs), we now define formally what we mean by FIFO symbolic representation. We essentially require a FIFO symbolic representation to be closed under *symbolic reception* (written ??) and under *symbolic emission* (written !!), in order to symbolically compute $post^*$.

Definition 3.7. *A* (C, M)-FIFO symbolic representation *(shortly a* FIFO symbolic representation*) is a* 6*-tuple* $\mathcal{F}_{C,M}$ *(shortly* $\mathcal{F}$*) such that* $\mathcal{F}_{C,M} = (\mathsf{F}, \bot, !!, ??, [\![\cdot]\!], +)$ *where* F *is a set of symbolic channel contents,* $\bot \in \mathsf{F}$*,* $?? : C \times M \times \mathsf{F} \rightarrow \mathsf{F}$*,* $!! : C \times M \times \mathsf{F} \rightarrow \mathsf{F}$*,* $[\![\cdot]\!] : \mathsf{F} \rightarrow 2^{(M^*)^C}$ *and* $+ : \mathsf{F} \times \mathsf{F} \rightarrow \mathsf{F}$ *are four total functions satisfying for every* $f, f' \in \mathsf{F}$*,* $c \in C$ *and* $m \in M$*:*

$$[\![\bot]\!] = \emptyset \tag{1}$$

$$[\![??(c, m, f)]\!] = \{\mathsf{w}[c := m^{-1}\mathsf{w}[c]] \mid \mathsf{w} \in [\![f]\!], m \leq \mathsf{w}[c]\} \tag{2}$$

$$[\![!!(c, m, f)]\!] = \{\mathsf{w}[c := \mathsf{w}[c]m] \mid \mathsf{w} \in [\![f]\!]\} \tag{3}$$

$$[\![f + f']\!] = [\![f]\!] \cup [\![f']\!] \tag{4}$$

Note that we suppose that FIFO symbolic representations are closed under union and contain the empty set, but these assumptions essentially allow us to simplify notations. We are now ready to show how a FIFO symbolic representation can be defined based on LREs.

Proposition 3.8. *Given* $m \in M$ *and a linear regular expression* ρ*, we have that* $m^{-1}(\rho)$ *is a semilinear regular expression over* M*.*

For any LRE ρ, we also denote by $m^{-1}(\rho)$ the set of LREs $\{\rho_1, \rho_2, \ldots, \rho_n\}$ such that $m^{-1}(\rho) = \rho_1 + \rho_2 + \cdots + \rho_n$.

Definition 3.9. *Let* $\mathcal{LRE} = (\mathcal{P}_f(LRE(M)^C), \emptyset, !!, ??, [\![\cdot]\!], \cup)$*, where* $??$*,* $!!$ *and* $[\![\cdot]\!]$ *are three total functions defined as follows:*

$$??(c, m, \chi) = \{\mathsf{r}[c := \rho] \mid \mathsf{r} \in \chi, \rho \in m^{-1}(\mathsf{r}[c])\}$$
$$!!(c, m, \chi) = \{\mathsf{r}[c := \mathsf{r}[c] \cdot m] \mid \mathsf{r} \in \chi\}$$
$$[\![\chi]\!] = \bigcup_{\mathsf{r}\in\chi} \prod_{c \in C} [\![\mathsf{r}[c]]\!]$$

Proposition 3.10. $\mathcal{LRE}$ *is a FIFO symbolic representation.*

Now, starting from a finite set of states, how to reach a "symbolic state" denoting an infinite set of states ? A natural way is to compute the effect of a loop, for instance iterating the loop σ_1 of ABP gives us: $(q_1, \overline{q_0}; \varepsilon, \varepsilon) \xrightarrow{(!a)^*} (q_1, \overline{q_0}; a^*, \varepsilon)$. We present in the next section our general framework of abstraction and acceleration formalizing these intuitive strategy. This general setting is given for labelled transition system, and is then applied to FIFO automata.

4 Abstraction and acceleration of labelled transition systems

We now introduce the notion of abstraction of labelled transition systems, a fairly general setting for our discussions. "What an abstraction is" is based on a minimal set of assumptions which allows us to present and compare several symbolic representations in an uniform setting. Formally,

Definition 4.1. *An* abstraction $\mathcal{A}_{LTS}$ *of a labelled transition system* $LTS = (\mathsf{S}, \Sigma, \rightarrow)$ *is a* 5*-tuple* $\mathcal{A}_{LTS} = (\mathsf{A}, post_{\mathsf{A}}, \gamma, \sqcup, \Sigma)$ *where* A *is a set of (abstract) states,* $post_{\mathsf{A}} : \mathsf{A} \times \Sigma \rightarrow \mathsf{A}$*,* $\gamma : \mathsf{A} \rightarrow 2^{\mathsf{S}}$ *and* $\sqcup : \mathsf{A} \times \mathsf{A} \rightarrow \mathsf{A}$ *are three total functions satisfying for every* $a, b \in \mathsf{A}$ *and* $l \in \Sigma$*:*

$$\gamma(a \sqcup b) = \gamma(a) \cup \gamma(b), \text{ and,} \tag{5}$$
$$post(\gamma(a), l) \subseteq \gamma(post_{\mathsf{A}}(a, l)) \subseteq post^*(\gamma(a)) \tag{6}$$

An abstract state a denotes a potentially infinite set $\gamma(a)$ of (concrete) states. Condition 5 enforces the set of abstract states A to be closed under (abstract) union $\sqcup$. Condition 6 ensures that an abstract exact computation of $post^*$ can be performed. Note that a labelled transition system may have several abstractions. In the rest of the paper, we write $\mathcal{A}_{LTS}$ to denote that $\mathcal{A}_{LTS}$ is an abstraction of a labelled transition system LTS.

An abstraction carries with it a natural quasi-ordering $\sqsubseteq$ induced by the function γ and defined by $a \sqsubseteq b$ if $\gamma(a) \subseteq \gamma(b)$. Furthermore, this quasi-ordering induces a natural equivalence relation $\cong$ defined by $a \cong b$ if $a \sqsubseteq b$ and $b \sqsubseteq a$. Note that two equivalent abstract states denote the same set of states.

The importance of the $\cong$ relation is that the $\sqcup$ operation is commutative and associative with respect to the equivalence. Hence for every finite subset X of

A, $\sqcup X$ is well defined with respect to $\cong$. This allows us to compute $post^*$ in an iterative manner, where we take the unions (i.e., $\sqcup$) as we go along, and thus have to maintain only one element of the symbolic representation at any time in computing the reachability set. Note that on the contrary to the usual abstract interpretation framework [CC77], we are not assuming that the abstraction domain A is closed under arbitrary union (i.e., lubs) but only finite ones.

An abstraction also implicitly defines a deterministic LTS $(\mathsf{A}, \Sigma, post_{\mathcal{A}})$. A simple consequence of our definitions is that the notion of abstraction immediately gives us a simulation:

Proposition 4.2. *Let $\mathcal{A}_{LTS} = (\mathsf{A}, post_{\mathcal{A}}, \gamma, \sqcup, \Sigma)$ be an abstraction of a labelled transition system $LTS = (\mathsf{S}, \Sigma, \rightarrow)$. For every $a_0 \in \mathsf{A}$, the labelled transition system $(\mathsf{A}, \Sigma, post_{\mathcal{A}})$ with initial state a_0 simulates LTS with the set of initial states $\gamma(a_0)$.*

Abstraction of FIFO automata A FIFO symbolic representation $\mathcal{F} = (\mathsf{F}, \bot, !!, ??, [\![\cdot]\!], +)$ allows us to associate a canonical abstraction to any FIFO automaton F, as follows.

Definition 4.3. *Let $\mathcal{A}^{\mathcal{F}}_{LTS(F)} = (\mathsf{A}, post_{\mathcal{A}}, \gamma, \sqcup, \Sigma)$, where $post_{\mathcal{A}} : \mathsf{A} \times \Sigma \rightarrow \mathsf{A}$, $\gamma : \mathsf{A} \rightarrow 2^{\mathsf{S}}$ and $\sqcup : \mathsf{A} \times \mathsf{A} \rightarrow \mathsf{A}$, be defined by:*

$$
\begin{aligned}
\mathsf{A} &= \mathsf{F}^{Q} \\
post_{\mathcal{A}}(\mathsf{a}, (q, c?m, q')) &= \bot[q' := !!(c, m, \mathsf{a}[q])] \\
post_{\mathcal{A}}(\mathsf{a}, (q, c!m, q')) &= \bot[q' := ??(c, m, \mathsf{a}[q])] \\
\gamma(\mathsf{a}) &= \bigcup_{q \in Q} \{q\} \times [\![\mathsf{a}[q]]\!] \\
\sqcup(\mathsf{a}, \mathsf{b}) &= < q \mapsto \mathsf{a}[q] + \mathsf{b}[q] >
\end{aligned}
$$

It is readily seen that for any FIFO symbolic representation $\mathcal{F}$ and for any FIFO automaton F, $\mathcal{A}^{\mathcal{F}}_{LTS(F)}$ is an abstraction of $LTS(F)$. Thus, we obtain that $\mathcal{A}^{\mathcal{LRE}}_{LTS(F)}$ is an abstraction.

We will now discuss how several strategies alternately called acceleration or meta-transitions can be captured uniformly by our definitions. Without acceleration, the abstract state obtained at each step of the computation of $post^*(s_0)$ represents only a finite portion of $post^*(s_0)$. Consequently, the advantage of using a symbolic representation might never be realized.

The way out of this dilemma is based on the observation that if a sequence of transitions σ can be iterated infinitely often from a state s, and if $\bigcup_{i \in \mathbb{N}} post(s, \sigma^i)$ can be calculated in a symbolic manner, then a portion of the reachability set containing an *infinite* set of states can be captured in a *single* step. The motivation is exactly what is behind the notion of *widening* in abstract interpretation [CC77], where in iterative data flow analysis of programs the effect of executing a loop (in a program) an arbitrary number of times is captured as a widening operator. In the following we adapt Cousot and Cousot's definition for our situation.

Definition 4.4. *A total function* $\nabla : \mathsf{A} \times \Sigma^+ \to \mathsf{A}$ *is an* acceleration *for an abstraction* $\mathcal{A}_{LTS} = (\mathsf{A}, post_{\mathcal{A}}, \gamma, \sqcup, \Sigma)$ *provided it satisfies the following condition for every* $a \in \mathsf{A}$ *and* $\sigma \in \Sigma^+$*:*

$$\nabla(a, \sigma) \sqsupseteq a \tag{7}$$

An acceleration ∇ *is* sound *if it satisfies the following condition for every* $a \in \mathsf{A}$ *and* $\sigma \in \Sigma^+$*:*

$$\gamma(\nabla(a, \sigma)) \subseteq post^*(\gamma(a)) \tag{8}$$

An acceleration ∇ *is* complete *if it satisfies the following condition for every* $a \in \mathsf{A}$ *and* $\sigma \in \Sigma^+$*:*

$$\gamma(\nabla(a, \sigma)) \supseteq \bigcup_{i \in \mathbb{N}} post(\gamma(a), \sigma^i) \tag{9}$$

The three conditions in the definition above deserve some explanation. Condition (7) — an inclusion condition — ensures that the abstract state obtained after an acceleration is bigger. The definition of a *sound* acceleration, Condition (8), enforces the constraint that all states computed by an acceleration are included in $post^*$. Similarly, the definition of a *complete* acceleration, Condition (9) enforces the constraint that all states resulting from an arbitrary number of executions of σ are included in result of the acceleration.

Given an acceleration we suggest developing a reachability tree in an uniform fashion according to the algorithm below. Most of the details of this algorithm

Algorithm 1 `SymbolicTree`$(\mathcal{A}_{LTS}, a_0, \nabla)$

Input: an abstraction $\mathcal{A}_{LTS} = (\mathsf{A}, post_{\mathcal{A}}, \gamma, \sqcup, \Sigma)$, an initial abstract state $a_0 \in \mathsf{A}$ and an acceleration ∇ for $\mathcal{A}_{LTS}$.

create root `r` labelled with a_0
while there are unmarked nodes **do**
  pick an unmarked node `n`$:a$
  if $a \sqsubseteq \bigsqcup\{b \mid$ `m`$:b$ is a marked node$\}$ **then**
    skip {the node `n` is covered}
  else
    $b \leftarrow a$
    for each ancestor `m` of `n` **do**
      $b \leftarrow \nabla(b, \sigma)$ where $\mathtt{m} \xrightarrow{\sigma} \mathtt{n}$
    for each label $l \in \Sigma$ **do**
      $a' \leftarrow post_{\mathcal{A}}(b, l)$
      construct a son `n'`$:a'$ of `n` and label the arc $\mathtt{n} \xrightarrow{l} \mathtt{n}'$
  mark `n`

should be clear. The acceleration gets applied, if at all possible, to obtain a potentially infinite set of reachable states. We may show that this symbolic tree, T, captures the reachable states, as required $post^*(\gamma(a_0)) \subseteq \bigcup_{\mathtt{n}:a \in T} \gamma(a)$. If moreover ∇ is a sound acceleration then we have $post^*(\gamma(a_0)) = \bigcup_{\mathtt{n}:a \in T} \gamma(a)$.

Remark 4.5. If $(\mathsf{A}, \sqsubseteq)$ satisfies the ascending chain condition then the `Symbolic Tree` algorithm terminates.

5 Acceleration of FIFO automata using linear regular expressions

For any loop σ on some control state q, we define two total functions (also written) σ and σ^* from $2^{(M^*)^C}$ to $2^{(M^*)^C}$ as follows:

$$\sigma(X) = \{\mathsf{w} \in (M^*)^C \mid (q, \mathsf{w}) \in post(\{q\} \times X, \sigma)\}, \text{ and, } \sigma^*(X) = \bigcup_{i \in \mathbb{N}} \sigma^i(X)$$

We will use the notion of $\mathcal{F}$*-representable* loops to define a canonical acceleration to any abstraction.

Definition 5.1. *A loop σ is* $\mathcal{F}$-representable *if for every $f \in \mathsf{F}$, there exists $g \in \mathsf{F}$ such that $[\![g]\!] = \sigma^*([\![f]\!])$.*

When σ is $\mathcal{F}$-representable we write $g = \sigma^*(f)$ even if there is not a unique g such that $[\![g]\!] = \sigma^*([\![f]\!])$. Notice that $\sigma^*(f)$ is well defined with respect to $\cong$.

Definition 5.2. *Let $\nabla_{\mathcal{F}} : \mathsf{A} \times \Sigma^+ \to \mathsf{A}$ be defined by:*

$$\nabla_{\mathcal{F}}(\mathsf{a}, \sigma) = \begin{cases} \mathsf{a}[q := \sigma^*(\mathsf{a}[q])] & \textit{if } \sigma \textit{ is a } \mathcal{F}\textit{-representable loop on } q \\ \mathsf{a} & \textit{otherwise} \end{cases}$$

We now show that $\nabla_{\mathcal{F}}$ is indeed an acceleration.

Proposition 5.3. *The two following statements hold:*

i) the total function $\nabla_{\mathcal{F}}$ is a sound acceleration for $\mathcal{A}^{\mathcal{F}}_{LTS(F)}$, and,
ii) If every loop of F is $\mathcal{F}$-representable then $\nabla_{\mathcal{F}}$ is complete.

In the rest of this section, we focus on accelerations for $\mathcal{LRE}$ based abstractions. We first define a sound and complete abstraction for FIFO automata with one channel. We then define a sound acceleration and a complete acceleration for FIFO automata with multiple channels.

5.1 Acceleration for one channel

The generic tree construction algorithm, from Section 4, suggests that we should accelerate a sequence σ if we can show that σ can be repeated infinitely often. To that end, consider that a FIFO automaton moves from (q, x) to (q, y) on a execution sequence σ. If it was possible for σ to be repeated infinitely often, then the only messages being removed from, or being added to the channel, come from σ. Clearly, the order in which messages are received, or sent, is preserved. The following theorem was used, but not proved, in [FM96]. The proof is technical and it can be found in the complete paper [FIS00].

Theorem 5.4. *Every loop of a FIFO automaton with one channel is $\mathcal{LRE}$-representable.*

Note that from [BH97], we know that every loop of a FIFO automaton is $\mathcal{CQDD}$-representable (see Theorem 6.5). However, their proof extensively use Pressburger formulae (even for one channel) and hence our theorem cannot be deduced from [BH97]. A straightforward consequence of Theorem 5.4 is that for any FIFO automaton with one channel F, $\nabla_{\mathcal{LRE}}$ is a sound and complete acceleration for $\mathcal{A}^{\mathcal{LRE}}_{LTS(F)}$.

5.2 Acceleration for multiple channels

To characterize $\mathcal{LRE}$-representable loops, we use a condition on loops defined in [BGWW97]. We say that a loop σ is *non counting* if one of the following conditions is satisfied:

i) $|M| > 1$ and there is a channel c such that every send transition in σ operates on c,
ii) $|M| = 1$ and there is at most one channel which is growing with σ.

Theorem 5.5. *A loop σ is $\mathcal{LRE}$-representable iff it is non counting.*

A straightforward consequence of the theorem is that for any FIFO automaton F, $\nabla_{\mathcal{LRE}}$ is a sound acceleration for $\mathcal{A}^{\mathcal{LRE}}_{LTS(F)}$.

We now show how a complete acceleration can be defined for $\mathcal{A}^{\mathcal{LRE}}_{LTS(F)}$. For every loop σ and for every channel $c \in C$, the *projection of σ on c*, denoted by $\sigma_{|_c}$, is obtained from σ by replacing every operation on c' with $c' \neq c$ by an ε — the model of FIFO automata can be easily extended with ε transitions and the results we presented so far are easily extended to this more general model. We define $\overline{\sigma^*} : \mathcal{P}_f(LRE(M)^C) \to \mathcal{P}_f(LRE(M)^C)$ by:

$$\overline{\sigma^*}(\chi) = \bigcup_{\mathsf{r} \in \chi} \prod_{c \in C} (\sigma_{|_c})^*(\mathsf{r}[c])$$

Lemma 5.6. *For every loop σ and for every $\mathcal{LRE}$ symbolic channel content $\chi \in \mathcal{P}_f(LRE(M)^C)$, we have $\sigma^*(\llbracket\chi\rrbracket) \subseteq \llbracket\overline{\sigma^*}(\chi)\rrbracket$.*

Notation. Given a control state q, an $\mathcal{LRE}$ symbolic channel content χ and a loop σ on q, we say that it is possible to execute σ^ω from (q, χ), written $(q, \chi) \xrightarrow{\sigma^\omega}$, if for every $n \in \mathbb{N}$ there exists $\mathsf{w} \in \llbracket\chi\rrbracket$ and $\mathsf{w}' \in (M^*)^C$ such that $(q, \mathsf{w}) \xrightarrow{\sigma^n} (q, \mathsf{w}')$.

We define the total function $\overline{\nabla}_{\mathcal{LRE}} : (\mathcal{P}_f(LRE(M)^C))^Q \times \Sigma^+ \to (\mathcal{P}_f(LRE(M)^C))^Q$ by:

$$\overline{\nabla}_{\mathcal{LRE}}(\mathsf{a}, \sigma) = \begin{cases} \mathsf{a}[q := \overline{\sigma}^*(\mathsf{a}[q])] & \text{if } \sigma \text{ is a loop on } q \text{ such that } (q, (\mathsf{a}[q])) \xrightarrow{\sigma^\omega} \\ \mathsf{a}[q := \bigsqcup_{i=0}^{k} \sigma^i(\mathsf{a}[q])] & \text{if } \sigma \text{ is a loop on } q \text{ such that } \sigma^k(\mathsf{a}[q]) = \emptyset \\ \mathsf{a} & \text{otherwise} \end{cases}$$

Proposition 5.7. *For any FIFO automaton $F = (Q, C, M, \delta)$ the total function $\overline{\nabla}_{\mathcal{LRE}}$ is a complete acceleration for $\mathcal{A}^{\mathcal{LRE}}_{LTS(F)}$.*

Notice that when $|M| = 1$, a FIFO automaton can be viewed as a Petri Net (or equivalently a Vector Addition System with States) and in this case $\overline{\nabla}_{\mathcal{LRE}}$ is similar to Karp and Miller's strategy to compute a coverability set. Hence, we obtain that for every initial abstract state a_0, $\texttt{SymbolicTree}(\mathcal{A}^{\mathcal{LRE}}_{LTS(F)}, a_0, \overline{\nabla}_{\mathcal{LRE}})$ terminates and computes an abstract state which is a tight over-approximation of $post^*(\gamma(a_0))$.

6 Comparison of LREs, QDDs and CQDDs

We now show how our notion of abstraction and acceleration is applicable to current work on meta-transitions (QDDs and CQDDs).

6.1 QDDs as an abstraction

In this subsection we present a slightly different abstraction, namely QDDs due to Boigelot and Godfroid [BGWW97]. A QDD D for a FIFO automaton $F = (Q, C, M, \delta)$, where $C = \{c_1, c_2, \cdots c_n\}$ is a deterministic finite automaton operating on the alphabet $C \times M$ and satisfying the following condition: for every accepted word $w \in L(D)$, the projection of w on C is in ${c_1}^* {c_2}^* \cdots {c_n}^*$. Intuitively, a channel content $\mathsf{w} \in (M^*)^C$ is in the set of channel contents represented by D iff there exists $u_1 u_2 \cdots u_n \in L(D)$ such that for every $1 \leq i \leq n$, u_i may be written as $u_i = (c_i, m_1)(c_i, m_2) \cdots (c_i, m_k)$ with $\mathsf{w}[c_i] = m_1 m_2 \cdots m_k$. In [BGWW97], is is proved that sets of channel contents representable by QDDs are exactly those that are recognizable sets. Hence, QDDs correspond to the FIFO symbolic representation based on regular expressions. More formally,

Definition 6.1. *Let $\mathcal{QDD} = (\mathcal{P}_f(REG(M)^C), \emptyset, !!, ??, [\![\cdot]\!], \cup)$, where $??$, $!!$ and $[\![\cdot]\!]$ are three total functions defined as follows:*

$$??(c, m, \chi) = \{\mathsf{r}[c := m^{-1}(\mathsf{r}[c])] \mid \mathsf{r} \in \chi\}$$
$$!!(c, m, \chi) = \{\mathsf{r}[c := \mathsf{r}[c] \cdot m] \mid \mathsf{r} \in \chi\}$$
$$[\![\chi]\!] = \bigcup_{\mathsf{r} \in \chi} \prod_{c \in C} [\![\mathsf{r}[c]]\!]$$

Note that $\mathcal{QDD}$ is clearly a FIFO symbolic representation and hence $\mathcal{A}^{\mathcal{QDD}}_{LTS(F)}$ is an abstraction of $LTS(F)$.

Theorem 6.2 ([BGWW97]). *Let $F = (Q, C, M, \delta)$ be a FIFO automaton. A loop σ is $\mathcal{QDD}$-representable iff it is non counting.*

Let us remark that a loop is $\mathcal{QDD}$-representable iff it is $\mathcal{LRE}$-representable. A straightforward consequence of the theorem is that for any FIFO automaton F, $\nabla_{\mathcal{QDD}}$ is a sound acceleration for $\mathcal{A}^{\mathcal{QDD}}_{LTS(F)}$.

6.2 CQDDs as an abstraction

A generalization of QDDs called Constrained Queue Decision Diagrams (CQDDs) was considered in [BH97]. Let us retranslate in our formalism the original definition of CQDDs.

Definition 6.3 ([BH97]). *An* accepting component *is a pair* (r, φ) *such that:*

i) $\mathsf{r} \in LRE(M)^C$*, and,*
ii) for every $c \in C$*, the LRE* $\mathsf{r}[c]$ *contains none of the following factors:* $(au)^* a$*,* $u^* v^*$ *with* $u, v \in M^*$ *and* $a \in M$*, and,*
iii) φ *is a Presburger arithmetic formula with a set of free variables* $X = \{x_c^i \mid c \in C, 1 \leq i \leq n_c\}$ *where* n_c *is the number of occurences of '$*$' in* $\mathsf{r}[c]$.

A Constrained Queue Decision Diagram (CQDD) *is a finite set of accepting components and we write* $CQDD(C, M)$ *for the set of CQDDs over* C *and* M*.*

This definition is more simple than the original definition (stated in [BH97]) given in terms of deterministic restricted simple automata.

We order the occurences of '$*$' in a LRE ρ by reading ρ from left to right. To a LRE ρ with n occurences of '$*$', we associate a function $f_\rho : \mathbb{N}^n \to \Sigma^*$ where $f_\rho(k_1, k_2, \ldots, k_n)$ is the word obtained from ρ by replacing the i^{th} occurence of '$*$' by k_i for every i, $1 \leq i \leq n$. The set of channel contents $[\![\chi]\!]$ associated with a CQDD χ is given by $[\![\chi]\!] = \cup_{(\mathsf{r},\varphi)\in\chi} [\![(\mathsf{r}, \varphi)]\!]$ where $[\![(\mathsf{r}, \varphi)]\!]$ is the set of channel contents $\mathsf{w} \in (M^*)^C$ such that for some valuation $v : X \to \mathbb{N}$ *satisfying* φ, we have $\mathsf{w}[c] = f_{\mathsf{r}[c]}(v(x_c^1), v(x_c^2), \ldots, v(x_c^{n_c}))$, for every $c \in C$. Notice that if φ is *valid* (i.e. satisfied by all valuations) then $[\![(\mathsf{r}, \varphi)]\!] = [\![r]\!]$ is a $\mathcal{LRE}$ set of channel contents.

CQDDs inspite of their power are closed under the operations we need:

Proposition 6.4 ([BH97]). *Given a CQDD* χ*, a channel* c *and a message* m*, there exists two CQDDs* χ' *and* χ'' *such that:*

$$[\![\chi']\!] = \{\mathsf{w}[c := m^{-1}\mathsf{w}[c]] \mid \mathsf{w} \in [\![\chi]\!], m \leq \mathsf{w}[c]\}$$
$$[\![\chi'']\!] = \{\mathsf{w}[c := \mathsf{w}[c]m] \mid \mathsf{w} \in [\![\chi]\!]\}$$

and we write $\chi' = ??(c, m, \chi)$ *and* $\chi'' = !!(c, m, \chi)$*.*

We are now ready to define the FIFO symbolic representation based on CQDDs. Defining $\mathcal{CQDD} = (CQDD(C, M), \emptyset, !!, ??, [\![\cdot]\!], \cup)$, we get that $\mathcal{CQDD}$ is a FIFO symbolic representation and hence $\mathcal{A}_{LTS(F)}^{\mathcal{CQDD}}$ is an abstraction of $LTS(F)$.

Theorem 6.5 ([BH97]). *Every loop of a FIFO automaton is* $\mathcal{CQDD}$*-representable.*

A straightforward consequence of the theorem is that for any FIFO automaton F, $\nabla_{\mathcal{CQDD}}$ is a sound and complete acceleration for $\mathcal{A}_{LTS(F)}^{\mathcal{CQDD}}$.

6.3 Comparison

The following theorem shows that $\mathcal{LRE}$ expressible sets of channel contents are exactly the intersection between $\mathcal{QDD}$ expressible sets of channel contents and $\mathcal{CQDD}$ expressible sets of channel contents.

Theorem 6.6. *Let L be a subset of $(M^*)^C$. The two following assertions are equivalent:*

i) there exists an $\mathcal{LRE}$ symbolic channel content χ such that $[\![\chi]\!] = L$,
ii) there exists a $\mathcal{QDD}$ symbolic channel content χ and a $\mathcal{CQDD}$ symbolic channel content χ' such that $[\![\chi]\!] = [\![\chi']\!] = L$

	Acceleration	
	one channel	multiple channels
$\mathcal{LRE}$	$\nabla_{\mathcal{LRE}}$: sound, complete	$\nabla_{\mathcal{LRE}}$: sound, non-complete $\overline{\nabla}_{\mathcal{LRE}}$: non-sound, complete
$\mathcal{QDD}$	$\nabla_{\mathcal{QDD}}$: sound, complete	$\nabla_{\mathcal{QDD}}$: sound, non-complete
$\mathcal{CQDD}$	$\nabla_{\mathcal{CQDD}}$: sound, complete	$\nabla_{\mathcal{CQDD}}$: sound, complete

Fig. 2. Comparison of the accelerations

It is readily seen that the $post_{\mathcal{A}_{\mathcal{LRE}}}$ and $\sqcup$ operations can be computed in *linear* time for the $\mathcal{LRE}$ based abstraction. Moreover, we have

Theorem 6.7. *The inclusion and the equivalence problems between SLREs are in* NP $\cap$ coNP.

No precise complexity results are known for $\mathcal{QDD}$ and $\mathcal{CQDD}$, but inclusion for these FIFO symbolic representations requires at least an exponential time.

Compared to QDDs and CQDDs, SLREs seem to have more practical relevance, since they are usually enough to represent sets of FIFO contents and to iterate loops (see Figure 2), and they have a better complexity. Starting from a single initial state of FIFO automaton with one channel, the generic `SymbolicTree` algorithm applied to these three abstractions/accelerations will give the same result. For FIFO automata with multiple channels, $\nabla_{\mathcal{LRE}}$ and $\nabla_{\mathcal{QDD}}$ applied to an $\mathcal{LRE}$ abstract state produce the same regular languages.

Acknowledgements

We thank the referees for their help to improve the paper.

References

[AAB99] P. Abdulla, A. Annichini, and A. Bouajjani. Symbolic verification of lossy channel systems: Application to the bounded retransmission protocol. *Lecture Notes in Computer Science*, 1579:208–222, 1999.

[BCM+92] J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, and L. J. Hwang. Symbolic model checking: 10^{20} states and beyond. *Information and Computation*, 98(2):142–170, June 1992.

[BGWW97] B. Boigelot, P. Godefroid, B. Willems, and P. Wolper. The power of QDDs. In *Proc. 4th Int. Symp. Static Analysis, Paris, France, September 1997*, volume 1302, pages 172–186. Springer-Verlag, 1997.

[BH97] A. Bouajjani and P. Habermehl. Symbolic reachability analysis of FIFO channel systems with nonregular sets of configurations. In *Proc. 24th Int. Coll. Automata, Languages, and Programming (ICALP'97), Bologna, Italy, July 1997*, volume 1256 of *Lecture Notes in Computer Science*, pages 560–570. Springer-Verlag, 1997.

[Brz64] Janusz A. Brzozowski. Derivatives of regular expressions. *Journal of the ACM*, 11(4):481–494, October 1964.

[BZ83] D. Brand and P. Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 30(2):323–342, April 1983.

[CC77] Patrick Cousot and Radhia Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the Fourth Annual Symposium on Principles of Programming Languages*, pages 238–252. ACM SIGACT and SIGPLAN, ACM Press, 1977.

[Céc98] G. Cécé. Vérification, analyse et approximations symboliques des automates communicants. Thèse ENS de Cachan, January 1998.

[CJ98] H. Comon and Y. Jurski. Multiple counters automata, safety analysis and Presburger arithmetic. 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 268–279. Springer, 1998.

[FIS00] A. Finkel, S. P. Iyer, and G. Sutre. Well-abstracted transition systems: application to fifo automata. Research Report LSV-2000-6, Lab. Specification and Verification, ENS de Cachan, Cachan, France, June 2000.

[FM96] A. Finkel and O. Marcé. A minimal symbolic coverability graph for infinite-state comunicating automata. Technical report, LIFAC, 1996.

[FS00] A. Finkel and G. Sutre. Decidability of reachability problems for classes of two counters automata. In *Proc. 17th Ann. Symp. Theoretical Aspects of Computer Science (STACS'2000), Lille, France, Feb. 2000*, volume 1770 of *Lecture Notes in Computer Science*, pages 346–357. Springer, 2000.

[JJ93] Thierry Jéron and Claude Jard. Testing for unboundedness of fifo channels. *Theoretical Computer Science*, 113(1):93–117, 1993.

[Pac87] J. K. Pachl. Protocol description and analysis based on a state transition model with channel expressions. In *Proc. of Protocol Specification, Testing and Verification, VII*, 1987.

[PP91] Wuxu Peng and S. Purushothaman. Data flow analysis of communicating finite state machines. *ACM Transactions on Programming Languages and Systems*, 13(3):399–442, July 1991.