Vérification des systèmes à compteurs par accélération: présentation de l'outil FAST

Jérôme Leroux

Postdoctorant avec Pierre McKenzie, DIRO

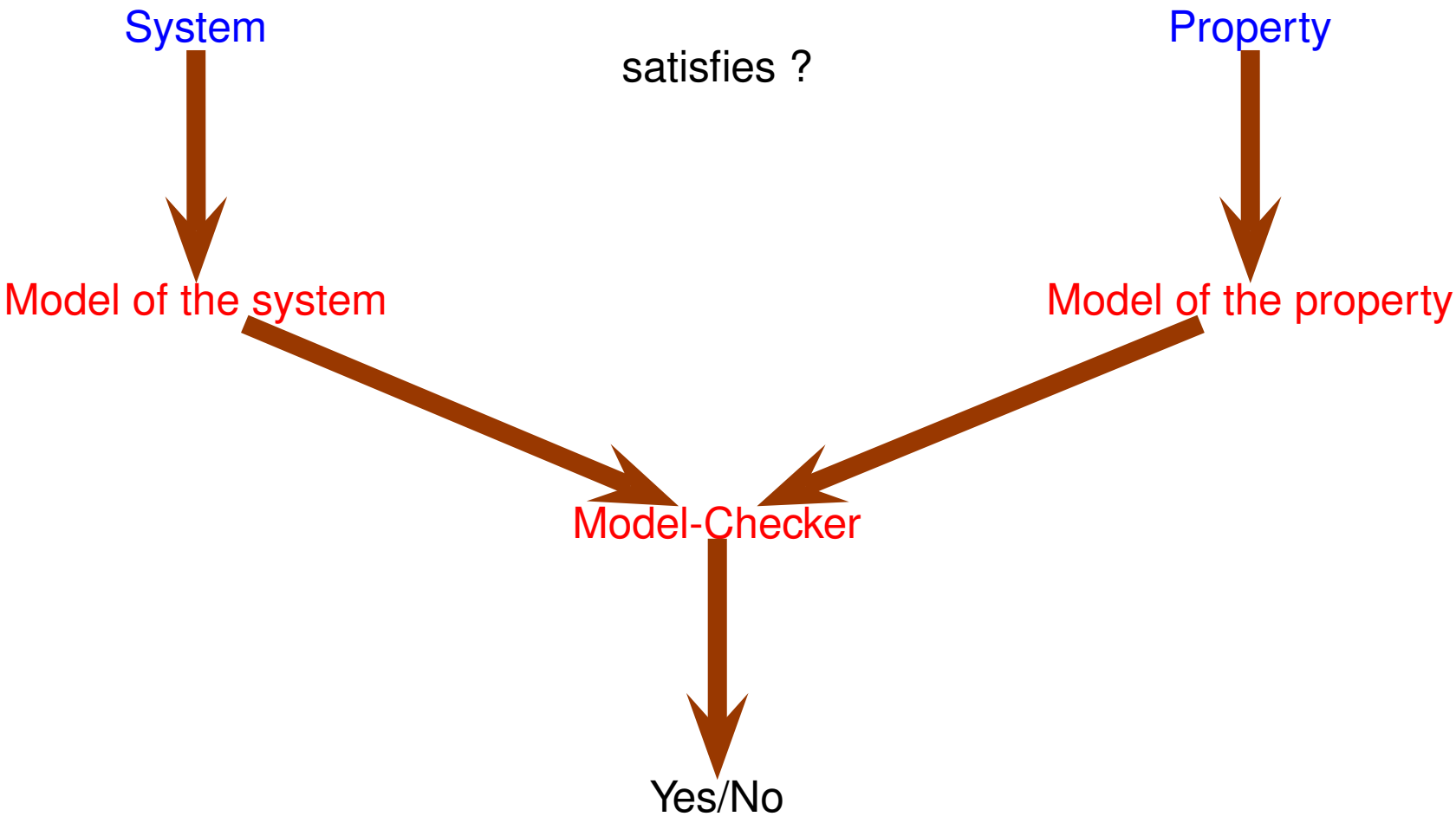
Why verification ?

- Critical systems are everywhere: Aircraft autopilot, ABS, GPS, HUB, TCP/IP, and so on.
- Systems are even more complex.
  - They are not designed from scratch,
  - They interact with other reactive systems.

Formal methods are needed.

- Test.
- Theorem Prover.
- Model-Checking,...

Model-checking



Model of the system

Systems with integer variables are everywhere:

- Parametrized systems: Lift elevator.
- Abstraction of systems:
  - Embedded systems: TTP [Bouajjani, Merceron]...
  - Cache coherence protocols: Dragon, MESI, MOESI, ... [Delzanno]..
  - Communication protocols: TCP/IP, FireWire ...
  - Multithreaded programs: java programs [Delzanno, Raskin, Van Begin]....

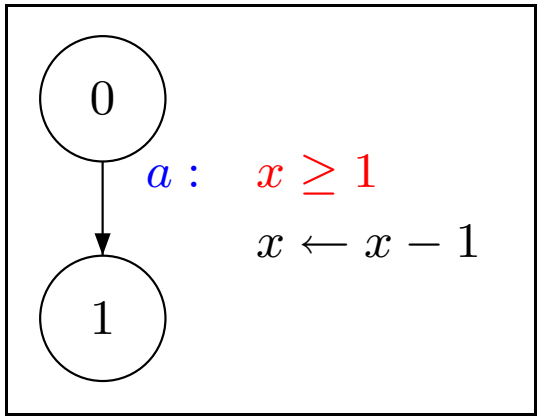
Models:

- Petri Nets, Extended Petri Nets [Ciardo],[Dufourd, Finkel, Schnoebelen]...,
- Broadcast Protocols [Emerson, Namjoshi],[Esparza, Finkel, Mayr], [Finkel, L.]...
- Counter Automata.

Def: A counters system S is a tuple $S = (\Sigma, f_\Sigma)$ where:

- Σ is a finite set of actions,
- $f_\Sigma = \{f_a;\ a \in \Sigma\}$ is a set of functions $f_a : D_a \rightarrow \mathbb{N}^m$.

A simple example



$\Sigma = \{a\}$

$D_a = \{(0, x) \in \mathbb{N}^2;\ x \geq 1\}$

$f_a(0, x) = (1, x - 1)$

Model of the property

Def: The reachability problem for counters systems:

Input: $\begin{cases} S = (\Sigma, f_\Sigma) \text{ a counter system} \\ X_0 \subseteq \mathbb{N}^m \text{ a set of initial states} \\ X_{bad} \subseteq \mathbb{N}^m \text{ a set of bad states} \end{cases}$

Question:Does there exist $\sigma \in \Sigma^*$, $x_0 \in X_0$ and $x_{bad} \in X_{bad}$ such that $f_\sigma(x_0) = x_{bad}$?

Two ways:

- Compute the reachability set $\mathrm{post}^*_S(X_0) = \bigcup_{\sigma \in \Sigma^*} f_\sigma(X_0)$.
  Check $\mathrm{post}^*_S(X_0) \cap X_{bad} = \emptyset$.
- Compute the co-reachability set $\mathrm{pre}^*_S(X_{bad}) = \bigcup_{\sigma \in \Sigma^*} f_\sigma^{-1}(X_{bad})$.
  Check $\mathrm{pre}^*_S(X_{bad}) \cap X_0 = \emptyset$.

However, the reachability problem is undecidable even for 2 counters automata.

How to compute post_S^* *or* pre_S^* *?*

Symbolic reachability:

$X \leftarrow X_0$.

While there exists $a \in \Sigma$ such that $f_a(X) \not\subseteq X$ do

Select(a). // $a \in \Sigma$

$X \leftarrow X \cup f_a(X)$.

/* *We have* $X = \mathrm{post}_S^*(X_0)$ */.

Pb: This algorithm does not terminate in general.

Solutions:

- Consider restricted subclasses of counters systems,
- Use approximate computation,
- Use acceleration.

What is acceleration ?

Acceleration [Boigelot, Wolper]
Let $\sigma \in \Sigma^*$ and $X \subseteq \mathbb{N}^m$.

$$f_\sigma^*(X) = \bigcup_{k \geq 0} f_\sigma^k(X)$$

Accelerated symbolic reachability:

$X \leftarrow X_0$.

While there exists $a \in \Sigma$ such that $f_a(X) \not\subseteq X$ do

Select(σ). // $\sigma \in \Sigma^*$

$X \leftarrow f_\sigma^*(X)$.

/* *We have* $X = \mathrm{post}_S^*(X_0)$ */.

To automate this algorithm, we need:

- To effectively compute $f_\sigma^*(X)$ from a representation of X.
- To find/reduce "the good sequences" σ.

Outline

- Saturated Digit Automata.
- Symbolic reachability by acceleration.
- The tool FAST.

Saturated Digit Automata (SDA)

(NDD) Number Decision Diagrams [Boigelot, Wolper], [Boudet, Comon]...

$$\boxed{\gamma_b : x \longrightarrow 2x + b}$$

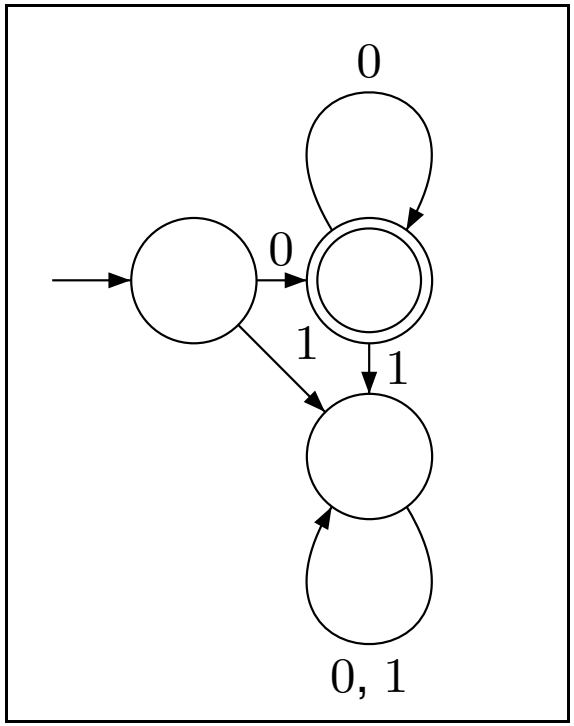
	7	=	1		1	1	0^*
	2	=		0		1	0^*
SDA	(7,2)	=	1	0	1	1	1	0^*

$$\boxed{\gamma_b : (x_1, x_2) \longrightarrow (2x_2 + b, x_1)}$$

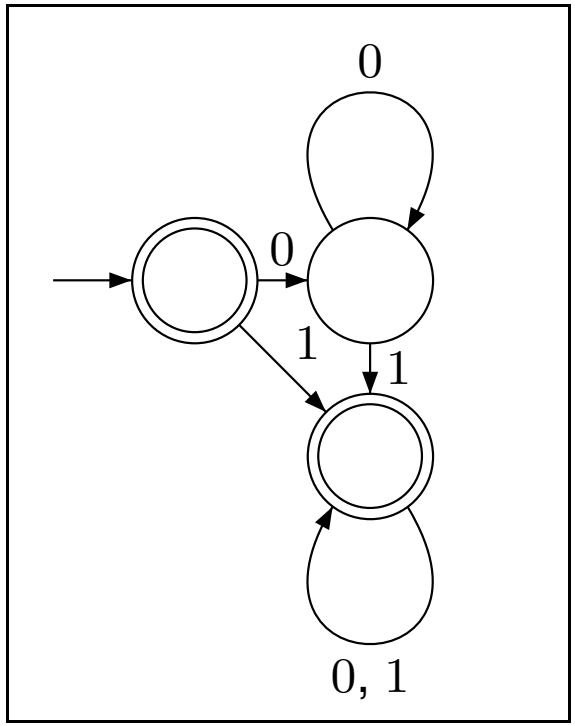
The vector represented by a word $\sigma = b_1 \dots b_k$ is

$$\rho_m(\sigma) = \gamma_{b_1} \circ \cdots \circ \gamma_{b_k}(0, \dots, 0)$$

"Saturated", what does it mean ?



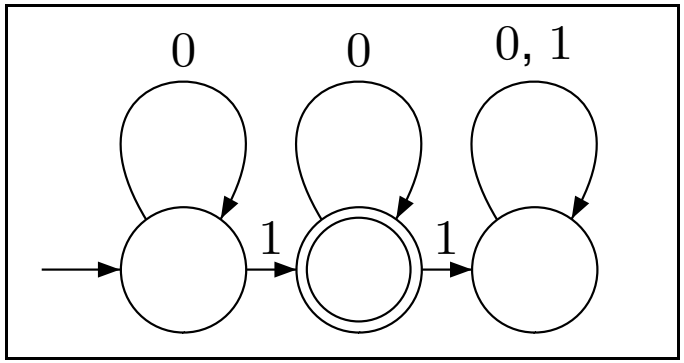
$\rho_1(\mathcal{L}) = \{0\}$



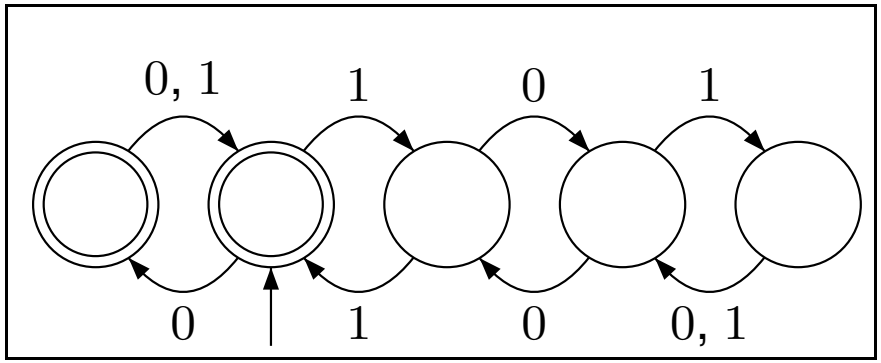
$\rho_1(\mathcal{L}^c) = \mathbb{N}$

Def: A SDA that represents a set $X \subseteq \mathbb{N}^m$ is a deterministic and complete automaton that recognizes the language $\rho_m^{-1}(X) \subseteq \{0,1\}^*$.

SDA-definable sets



$X = \{2^n;\ n \geq 0\}$



$X = \{(x_1, x_2) \in \mathbb{N}^2;\ x_1 \leq x_2\}$

Theo [Bryère,Hansel, Michaux, Villemaire]: A set $X \subseteq \mathbb{N}^m$ is SDA-definable if and only if X is defined in $< \mathbb{N}, \leq, +, V_2 >$.

$\exists y \ \ (x = 3y \wedge \neg(z = x))$

$x = y.z$

Operations over an SDA

Boolean operations $\wedge$, $\vee$, $\neg$	PTIME(n^2)
Complexity of $\subseteq$	PTIME(n^2)
Quantifications $\exists x$, $\forall x$	EXPTIME
Canonical representation	PTIME($n.\ln(n)$)

Outline

- Saturated Digit Automata.
- Symbolic reachability by acceleration.
- The tool FAST.

How to compute $f^*(X)$?

Monoïd multiplicatively generated by M:
$\langle M \rangle = \{I, M, M^2, M^3, \ldots\}$

Theo:[Boigelot] Let $f(x) = M.x + v$ be an affine function such that:

- definition domain D is a polyhedra, and
- monoïd $\langle M \rangle$ is finite.

For any subset X represented by an SDA,
$f^*(X)$ can be effectivelly represented by a SDA.

Theo: Let $f(x) = M.x + v$ be an affine function such that:

- definition domain D is represented by a SDA, and
- monoïd $\langle M \rangle$ is finite.

For any subset X represented by a SDA,
$f^*(X)$ can be effectivelly represented by a SDA.

Proof idea

Assume $f(x) = x + v$ defined over $D \subseteq \mathbb{N}^m$.

We have $f^k(x) = x + k.v$.

$$f^*(X) = \left\{x' \in \mathbb{N}^m;\ \exists x \in X\ \exists k \left(x' = x + k.v \wedge \forall i < k\ \ x + i.v \in D\right)\right\}$$

Reduction technique

Reduction technique.
$f_1(x) = M.x + v$ defined over D_{f_1}
$f_2(x) = M.x + v$ defined over D_{f_2}

We just consider $g(x) = M.x + v$ defined over $D_g = D_{f_1} \cup D_{f_2}$.

Def: The reduced set $[F]$ of a set of affine functions F is such that
$g(x) = M.x + v$ is in $[F]$ iff

$$\emptyset \neq D_g = \bigcup_{(f(x)=M.x+v)\in F} D_f$$

How to find out the good acceleration

Def: A counters system $S = (\Sigma, f_\Sigma)$ has a finite monoïd if the monoïd multiplicatively generated by the matrices M_a is finite (Recall that $f_a(x) = M_a.x + v_a$).

Theo: Let S be a counters system with a finite monoïd and assume that any D_a is represented by an SDA. We have:

- The reduced set $G_k = [\{f_\sigma;\ \sigma \in \Sigma^{\leq k}\}]$ is computable in PTIME in k.
- $g^*(X)$ is effectivelly representable by an SDA, for any $g \in G_k$ and any X represented by an SDA.

The reduced algorithm

Reduced-accelerated symbolic reachability:

$X \leftarrow X_0$.

Let $k \geq 1$.

Compute $G = G_k$.

While there exists $a \in \Sigma$ such that $f_a(X) \not\subseteq X$ do

Select(g). //$g \in G$

$X \leftarrow g^*(X)$.

/* *We have* $X = \mathrm{post}^*_S(X_0)$ */.

Outline

- Saturated Digit Automata.
- Symbolic reachability by acceleration.
- The tool FAST.

The tool FAST

FAST is a tool :

- with a powerful model
  - finite monoïd, and
  - definition domains D_a represented by Presburger formula.
- that automatically computes the set of reachable states in most practical cases,
- with a powerful strategy language specification (Hytech-like) [Bardin].
- easy to use thanks to a GUI interface [Worobel].

The model of FAST generalizes:

- Petri Nets,
- Extended Petri Nets [Ciardo],[Dufourd, Finkel, Schnoebelen]...,
- Broadcast Protocols [Emerson, Namjoshi],[Esparza, Finkel, Mayr], [Finkel, L.]...
- Counter Automata.

The kernel of FAST

The kernel of FAST:

$X \leftarrow X_0$.

Let $k \geq 1$.

Compute $G = G_k$.

While there exists $a \in \Sigma$ such that $f_a(X) \not\subseteq X$ do

RandomSelect(g). //$g \in G$

$X \leftarrow g^*(X)$.

While there exists $g \in G$ such that

$$\begin{cases} g(X) \not\subseteq X \\ |\mathcal{A}(g^*(X))| \leq |\mathcal{A}(X)| \end{cases} \quad \text{do } X \leftarrow g^*(X)$$

/* *We have* $X = \mathrm{post}^*_S(X_0)$ */.

FAST *in practice 1/3*

80% of 40 counters systems (mainly taken from [ALV,BABYLON,TREX]) have been automatically analysed.

Bounded Petri Nets: Producer/Consumer, Lamport ME, Dekker ME, RTP, Peterson ME, Reader/Writer.

Petri Nets: CSM - N, FMS, Multipoll, Kanban, Mesh2x2, Mesh3x2, Manufacturing system, Manufacturing system (check deadlock freedom), PNCSA, extended ReaderWriter, SWIMMING POOL.

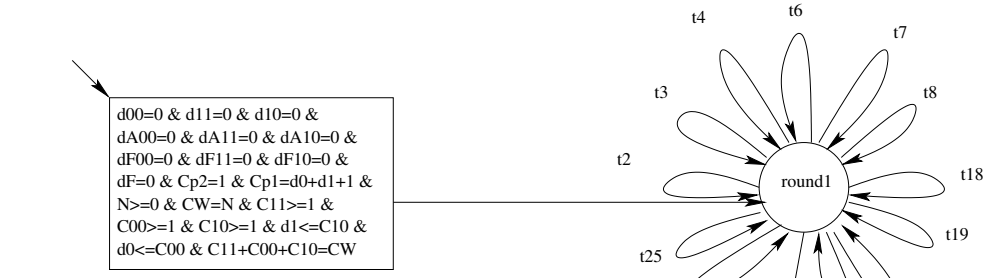
Petri Nets with Transfer Arcs: Last-in First-served, Esparza-Finkel-Mayr.

Broadcast Protocols: Inc/Dec, Producer/Consumer with Java threads - 2, Producer/Consumer with Java threads - N, 2-Producer/2-Consumer with Java threads, Central Server system, Consistency Protocol, M.E.S.I. Cache Coherence Protocol, M.O.E.S.I. Cache Coherence Protocol, Synapse Cache Coherence Protocol, Illinois Cache Coherence Protocol, Berkeley Cache Coherence Protocol, Firefly Cache Coherence Protocol, Dragon Cache Coherence Protocol, Futurebus+ Cache Coherence Protocol.

Others: lift controller - N, bakery4, barber4, ticket 2i, ticket 3i, TTP-1fault.

FAST *in practice 2/3*

FAST automatically verifies the TTP protocol with 2-faults [Bouajjani,Merceron].



t2 : Cp1<N & d11<C11 & CW−2d0−2d00−2d10>0/
d11++,Cp1++,Cp2++
t3: Cp1<N & d10<C10−d1 & CW −2d0 −2d00 −2d11>0/
d10++,Cp1++,Cp2++
t4 : Cp1<N & d00<C00−d0 & CW−2d1−2d10−2d11>0/
d00++,Cp1++,Cp2++
t6 : Cp1<N & d11<C11−d1 & CW−2d0−2d00−2d10<=0/
dF++,Cp1++,Cp2++,C11−−
t7 : Cp1<N & d10<C10 & CW−2d0−2d00−2d11<=0/
dF++,Cp1++,Cp2++,C10−−
t8 : Cp1<N &d00<C00−d0 & CW−2d1−2d10−2d11<=0/
dF++,Cp1++,Cp2++,C00−−
t18 : Cp1>=N & Cp2<N & Pred1/ d11++,Cp1++,Cp2++,dA11++
t19 : Cp1>=N & Cp2<N & Pred2/ d10++,Cp1++,Cp2++,dA10++
t21 : Cp1>=N & Cp2<N & Pred3/d00++,Cp1++,Cp2++,dA00++
t22 : Cp1>=N & Cp2<N & !Pred1/dF++,dF11++,Cp1++,Cp2++,C11−−
t23 : Cp1>=N & Cp2<N & !Pred2/dF++,dF10++,Cp1++,Cp2++,C10−−
t25 : Cp1>=N & Cp2<N & !Pred3/dF++, dF00++,Cp1++,Cp2++,C00−−

t26 : Cp2=N / dF=0,d11=0,d10=0,d00=0,Cp2=0

t27 : Cp2<N & d11<C11 & C11−C10−C00>0 / d11++,Cp2++
t28 : Cp2<N & d10<C10 & C10−C11−C00>0 / d10++,Cp2++
t30 : Cp2<N & d00<C00 & C00−C10−C11>0 / d00++, Cp2++
t31 : Cp2<N & d11<C11 & C11−C10−C00<=0 /
C11−−,Cp2++,dF++,CF++
t32 : Cp2<N & d10<C10 & C10−C11−C00<=0 /
C10−−,Cp2++,CF++,dF++
t33 : Cp2<N & d00<C00 & C00−C10−C11<=0 /
C00−−,Cp2++,CF++,dF++
t34 : Cp2<N & dF<CF / Cp2++,dF++

Pred1 :
d1+d11−dA11−dF11−dA10−dF10−d0−d10−d00+dA00+dF00>0
Pred2 :
d1+d10−dA10−dF10−dA11−dF11−d0−d11−d00+dA00+dF00>0
Pred3 :
d0+d00−dA00−dF00−d1−d11−d10+dA11+dA10+dF11+dF10>0

FAST *in practice 3/3*

$|\Sigma| = 34$ actions
$m = 19$ counters

$(Cp1 \geq N) \wedge (Cp2 < N) \wedge (d11 < C11)$
$\wedge(-d1 - d11 + dA11 + dF11 + dA10 + dF10 + d0 + d10 + d00 - dA00 - dF00 \geq 0)$

$dF' = dF + 1$
$Cp1' = Cp1 + 1$
$Cp2' = Cp2 + 1$
$dF11' = dF11 + 1$
$C11' = C11 - 1$

Tools with acceleration and counters

	guards	actions	acceleration	auto. cycle search
FAST [Bardin, Darlot, Finkel, L., Petrucci, Van-Begin, Worobel]	$\langle \mathbb{N}, +, \leq \rangle$	$x' = M.x + v$	yes	yes
LASH [Boigelot, Bontemps, Cécé, François, Jodogne, Latour, Rassart, Wolper]	convex sets	$x' = M.x + v$	yes	no
TREX [Asarin, Bouajjani, Collomb-Annichini, Lakhnech, Sighireanu]	$\bigwedge \begin{cases} x_i \leq x_j + c \\ x_i \leq c \\ x_i \geq c \end{cases}$	$\bigwedge \begin{cases} x_i = x_j + c \\ x_i = c \end{cases}$	yes	yes

Conclusion

We have presented:

- The Saturated Digit Automata.
- The acceleration technics for integer systems.
- The tool FAST.

Future Works

- Efficient computation of $f^*(X)$ for particular f [Bardin, Finkel, L.:TACAS'04].
- Structure of the SDA:
  - structure/size of $\mathrm{pre}_S^{\leq k}(X)$ [Finkel, L.: SPIN'04] [Finkel, L.: CAV'04].
  - Structure of an SDA of $< \mathbb{N}, \leq, + >$ [L.: Submitted to CIAA'04] Work in progress with McKenzie.
- Implement an efficient SDA library Work in progress [Fast Team].