# A survey on WSTS

Alain Finkel

LSV, ENS Paris-Saclay (ex ENS Cachan)

IIT Mumbai, India
5*th* March 2018

- Based on joint works with Michael Blondin, Jean Goubault-Larrecq & Pierre McKenzie.

# Exercise 1

- Sir,
  What exactly is the definition of downward closed sets- is it the complement of upward closed sets or is it the intuitive notion?
- How do we define its basis?
- Other questions ?

# Exercise 2

- Find a picture for representing *Pre*$^*$-coverability semi-algorithm.
- Find a picture for representing *Post*$^*$-coverability semi-algorithm.

# Exercise 3

- $T(w) =$ length of a longest computation starting from $w \in \Sigma^*$.
- $T(w) \in \mathbb{N}_\omega$.
- $w \leq_T w'$ if $T(w) \leq T(w')$.

# Exercise 3

- $T(w)$ = length of a longest computation starting from $w \in \Sigma^*$.
- $T(w) \in \mathbb{N}_\omega$.
- $w \leq_T w'$ if $T(w) \leq T(w')$.

Prove the following theorem

**Theorem**

*Turing machines are WSTS with strict and strong monotony wrt $\leq_T$.*

# Exercise 4

$y$ is not coverable from $x$ iff $y \notin \downarrow \mathrm{Post}^*(x)$.

Let $(S_i)_i$ be an enumeration of finite sets of ideals, $\downarrow \mathrm{Post}^*(x) = S_m$, for some $m$ and $(F_i)_i$ an enumeration of finite sets $F_i \subseteq X$.

**procedure 2: non coverability certificate of $y$ from $x$**

**while** $\neg(\downarrow \mathrm{Post}(S_i) \subseteq S_i$ and $x \in S_i$ and $y \notin S_i)$ **do**
  $i \leftarrow i + 1$
**return** *false*

**procedure 2: non coverability certificate of $y$ from $x$**

**while** $\neg(\mathrm{Pre}(\uparrow F_i) \subseteq \uparrow F_i)$ and $x \notin \uparrow F_i$ and $y \in \uparrow F_i)$ **do**
  $i \leftarrow i + 1$
**return** *false*

# Exercises 5

- Find a direct proof of Erdös Tarski Theorem avoiding wqo.

# Exercises 5

- Find a direct proof of Erdös Tarski Theorem avoiding wqo.
- For wpo, we define $x < y$ if $x \leq y$ and $x \neq y$.
  Define $x < y$ when $\leq$ is a wqo.

# Exercises 5

- Find a direct proof of Erdös Tarski Theorem avoiding wqo.
- For wpo, we define $x < y$ if $x \leq y$ and $x \neq y$.
  Define $x < y$ when $\leq$ is a wqo.
- Give a definition of $Min(X)$.

# Exercises 5

- Find a direct proof of Erdös Tarski Theorem avoiding wqo.
- For wpo, we define $x < y$ if $x \leq y$ and $x \neq y$.
  Define $x < y$ when $\leq$ is a wqo.
- Give a definition of $Min(X)$.
  $Min(X) = \{x \mid \forall y, y \leq x \implies x \leq y \not\leq x\}$

# Exercises 5

- Find a direct proof of Erdös Tarski Theorem avoiding wqo.
- For wpo, we define $x < y$ if $x \leq y$ and $x \neq y$.
  Define $x < y$ when $\leq$ is a wqo.
- Give a definition of $Min(X)$.
  $Min(X) = \{x \mid \forall y, y \leq x \implies x \leq y \not\leq x\}$
- Prove that if $(X, \leq)$ is WF then for all $x$ there is a $m \in Min(X)$ s.t. $x \geq m$.

# Exercises 5

- Find a direct proof of Erdös Tarski Theorem avoiding wqo.
- For wpo, we define $x < y$ if $x \leq y$ and $x \neq y$.
  Define $x < y$ when $\leq$ is a wqo.
- Give a definition of $Min(X)$.
  $Min(X) = \{x \mid \forall y, y \leq x \implies x \leq y \not\leq x\}$
- Prove that if $(X, \leq)$ is WF then for all $x$ there is a $m \in Min(X)$ s.t. $x \geq m$.
- For $U = \uparrow U$, prove that $Min(U)$ is a (infinite) basis of $U$ when $\leq$ is WF. Why it is not the case if $\leq$ is not WF ?

# Exercises 5

- Find a direct proof of Erdös Tarski Theorem avoiding wqo.
- For wpo, we define $x < y$ if $x \leq y$ and $x \neq y$.
  Define $x < y$ when $\leq$ is a wqo.
- Give a definition of $Min(X)$.
  $Min(X) = \{x \mid \forall y, y \leq x \implies x \leq y \not\leq x\}$
- Prove that if $(X, \leq)$ is WF then for all $x$ there is a $m \in Min(X)$ s.t. $x \geq m$.
- For $U = \uparrow U$, prove that $Min(U)$ is a (infinite) basis of $U$ when $\leq$ is WF. Why it is not the case if $\leq$ is not WF ?
- For $U = \uparrow U$, prove that $Min(U)$ is finite ($\neq \emptyset$) when $\leq$ is a WF+FAC wpo.

# Exercises 5

- Find a direct proof of Erdös Tarski Theorem avoiding wqo.
- For wpo, we define $x < y$ if $x \leq y$ and $x \neq y$.
  Define $x < y$ when $\leq$ is a wqo.
- Give a definition of $Min(X)$.
  $Min(X) = \{x \mid \forall y, y \leq x \implies x \leq y \not\leq x\}$
- Prove that if $(X, \leq)$ is WF then for all $x$ there is a $m \in Min(X)$ s.t. $x \geq m$.
- For $U = \uparrow U$, prove that $Min(U)$ is a (infinite) basis of $U$ when $\leq$ is WF. Why it is not the case if $\leq$ is not WF ?
- For $U = \uparrow U$, prove that $Min(U)$ is finite ($\neq \emptyset$) when $\leq$ is a WF+FAC wpo.
- For $U = \uparrow U$, prove that $Min(U)/\equiv$ is finite ($\neq \emptyset$) when $\leq$ is WF+FAC wqo.

# Exercises 5

- Find a direct proof of Erdös Tarski Theorem avoiding wqo.
- For wpo, we define $x < y$ if $x \leq y$ and $x \neq y$.
  Define $x < y$ when $\leq$ is a wqo.
- Give a definition of $Min(X)$.
  $Min(X) = \{x \mid \forall y, y \leq x \implies x \leq y \not\leq x\}$
- Prove that if $(X, \leq)$ is WF then for all $x$ there is a $m \in Min(X)$ s.t. $x \geq m$.
- For $U = \uparrow U$, prove that $Min(U)$ is a (infinite) basis of $U$ when $\leq$ is WF. Why it is not the case if $\leq$ is not WF ?
- For $U = \uparrow U$, prove that $Min(U)$ is finite ($\neq \emptyset$) when $\leq$ is a WF+FAC wpo.
- For $U = \uparrow U$, prove that $Min(U)/\equiv$ is finite ($\neq \emptyset$) when $\leq$ is WF+FAC wqo.
- Conclude that $\leq$ is wqo iff $\leq$ is WF + FAC.

# Exercises 6

- The language $L(M) \subseteq \Sigma^*$ of a Turing machine $M$ is the set of words $w \in \Sigma^*$ that are on the tape when $M$ reaches a terminal control state.
- A Turing machine $M$ is *regular* if $L(M)$ is regular.

# Exercises 6

- The language $L(M) \subseteq \Sigma^*$ of a Turing machine $M$ is the set of words $w \in \Sigma^*$ that are on the tape when $M$ reaches a terminal control state.
- A Turing machine $M$ is *regular* if $L(M)$ is regular.
- Prove that regular Turing machines are recursive.

# Exercises 6

- The language $L(M) \subseteq \Sigma^*$ of a Turing machine $M$ is the set of words $w \in \Sigma^*$ that are on the tape when $M$ reaches a terminal control state.
- A Turing machine $M$ is *regular* if $L(M)$ is regular.
- Prove that regular Turing machines are recursive. Can you deduce an algorithm for deciding whether $w \in L(M)$ ?

# Exercises 6

- The language $L(M) \subseteq \Sigma^*$ of a Turing machine $M$ is the set of words $w \in \Sigma^*$ that are on the tape when $M$ reaches a terminal control state.
- A Turing machine $M$ is *regular* if $L(M)$ is regular.
- Prove that regular Turing machines are recursive. Can you deduce an algorithm for deciding whether $w \in L(M)$ ?
- Give an algorithm to decide $w \in L(M)$ for regular and context-free Turing machines.

# Exercises 6

- The language $L(M) \subseteq \Sigma^*$ of a Turing machine $M$ is the set of words $w \in \Sigma^*$ that are on the tape when $M$ reaches a terminal control state.
- A Turing machine $M$ is *regular* if $L(M)$ is regular.
- Prove that regular Turing machines are recursive. Can you deduce an algorithm for deciding whether $w \in L(M)$ ?
- Give an algorithm to decide $w \in L(M)$ for regular and context-free Turing machines.
- Give an algorithm for deciding reachability for Petri nets having (unknown) semilinear/Presburger reachability sets.

# Exercises 6

- The language $L(M) \subseteq \Sigma^*$ of a Turing machine $M$ is the set of words $w \in \Sigma^*$ that are on the tape when $M$ reaches a terminal control state.
- A Turing machine $M$ is *regular* if $L(M)$ is regular.
- Prove that regular Turing machines are recursive. Can you deduce an algorithm for deciding whether $w \in L(M)$ ?
- Give an algorithm to decide $w \in L(M)$ for regular and context-free Turing machines.
- Give an algorithm for deciding reachability for Petri nets having (unknown) semilinear/Presburger reachability sets.
- Jan K. Pachl: Protocol Description and Analysis Based on a State Transition Model with Channel Expressions. PSTV 1987: 207-219.

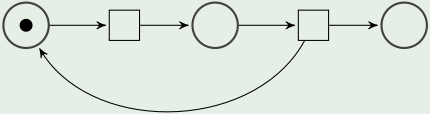
# Motivation

Verification of infinite-state models

- counter machines with reset-transfer-affine-$\omega$ extensions
- Lossy fifo systems and variants with time, data and priority
- Parameterized broadcast protocols and other
- CFG, graph rewriting
- Systems with pointers, graph memory (Well-Structured Graph Transformation Systems (CONCUR 2014))
- Fragments of the $\pi$-calculus, depth bounded processes

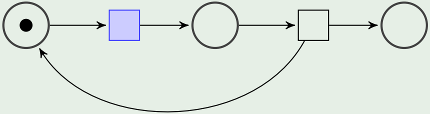
Well Structured Transition Systems (WSTS) encompass a large number of infinite state systems (PN and reset-transfer-affine-$\omega$ extensions, lossy fifo systems, broadcast protocols, CFG, graph rewriting, depth bounded processes, fragments of the $\pi$-calculus,....)

### Example of WSTS: Petri nets



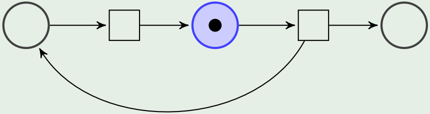
Well Structured Transition Systems (WSTS) encompass a large number of infinite state systems (PN and reset-transfer-affine-$\omega$ extensions, lossy fifo systems, broadcast protocols, CFG, graph rewriting, depth bounded processes, fragments of the $\pi$-calculus,....)

## Example of WSTS: Petri nets



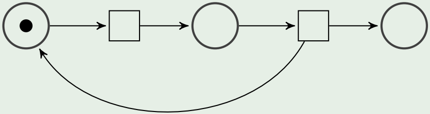
Well Structured Transition Systems (WSTS) encompass a large number of infinite state systems (PN and reset-transfer-affine-$\omega$ extensions, lossy fifo systems, broadcast protocols, CFG, graph rewriting, depth bounded processes, fragments of the $\pi$-calculus,....)

## Example of WSTS: Petri nets



Multiple decidability results are known for (finitely branching) WSTS.

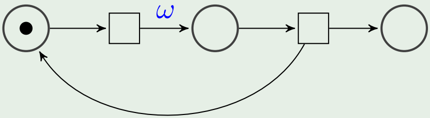
**Example of WSTS: Petri nets**



$$\mathrm{Post}(\odot\ \bigcirc\ \bigcirc) = \bigcirc\ \odot\ \bigcirc$$

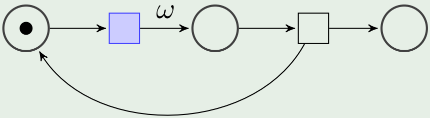
And also for (infinitely branching) WSTS such as systems with infinitely many initial states and parametric systems

**Example of WSTS: $\omega$–Petri nets** (Geeraerts, Heußner, Praveen & Raskin PN'13)



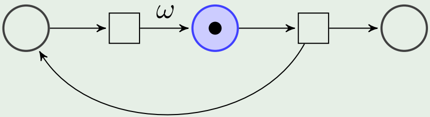
And also for (infinitely branching) WSTS such as systems with infinitely many initial states and parametric systems

**Example of WSTS: $\omega$–Petri nets** (Geeraerts, Heußner, Praveen & Raskin PN'13)



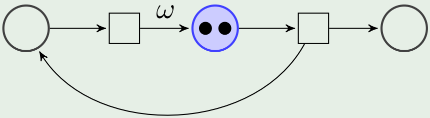
And also for (infinitely branching) WSTS such as systems with infinitely many initial states and parametric systems

## Example of WSTS: $\omega$–Petri nets (Geeraerts, Heußner, Praveen & Raskin PN'13)



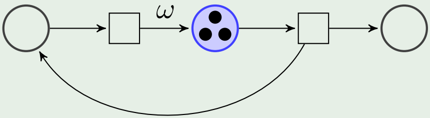
And also for (infinitely branching) WSTS such as systems with infinitely many initial states and parametric systems

**Example of WSTS: $\omega$–Petri nets** (Geeraerts, Heußner, Praveen & Raskin PN'13)



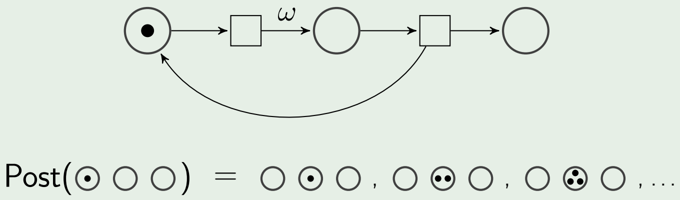
And also for (infinitely branching) WSTS such as systems with infinitely many initial states and parametric systems

**Example of WSTS: $\omega$–Petri nets** (Geeraerts, Heußner, Praveen & Raskin PN'13)



And also for (infinitely branching) WSTS such as systems with infinitely many initial states and parametric systems

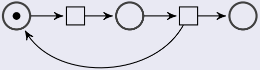
**Example of WSTS: $\omega$–Petri nets** (Geeraerts, Heußner, Praveen & Raskin PN'13)



## Well structured transition system (F, ICALP'87)

$S = (X, \to, \leq)$ where

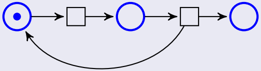
- $X$ set,
- $\to \subseteq X \times X$,
- monotony,
- well-quasi-ordered.



## Well structured transition system (F, ICALP'87)

$S = (X, \to, \leq)$ where

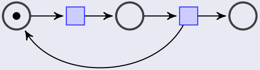
- $\mathbb{N}^3$,
- $\to \subseteq X \times X$,
- monotony,
- well-quasi-ordered.



## Well structured transition system (F, ICALP'87)

$S = (X, \to, \leq)$ where

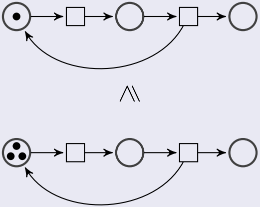
- $X$ set,
- $\to \subseteq \mathbb{N}^3 \times \mathbb{N}^3$,
- monotony,
- well-quasi-ordered.



## Well structured transition system (F, ICALP'87)

$S = (X, \to, \leq)$ where

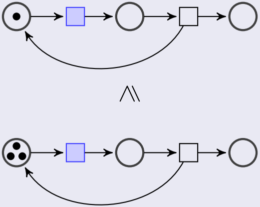
- $X$ set,
- $\to \subseteq X \times X$,
- monotony,
- well-quasi-ordered.



## Well structured transition system (F, ICALP'87)

$S = (X, \to, \leq)$ where

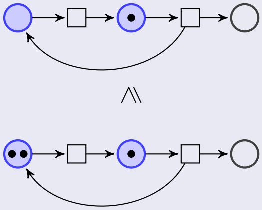
- $X$ set,
- $\to \subseteq X \times X$,
- monotony,
- well-quasi-ordered.



## Well structured transition system (F, ICALP'87)

$S = (X, \to, \leq)$ where

- $X$ set,
- $\to \subseteq X \times X$,
- monotony,
- well-quasi-ordered.



## Well structured transition system (F, ICALP'87)

$S = (X, \rightarrow, \leq)$ where

- $X$ set,
- $\rightarrow \subseteq X \times X$,
- monotony,
- well-quasi-ordered.

$$\begin{array}{ccccc} \forall & x & \rightarrow & y & \\ & \rotatebox{90}{$\leq$} & & \rotatebox{90}{$\leq$} & \\ & x' & \xrightarrow{*} & y' & \exists \end{array}$$

## Well structured transition system (F, ICALP'87)

$S = (X, \rightarrow, \leq)$ where

- $X$ set,
- $\rightarrow \subseteq X \times X$,
- transitive monotony,
- well-quasi-ordered.

$$\begin{array}{ccccc} \forall & x & \rightarrow & y & \\ & \rotatebox{90}{$\leq$} & & \rotatebox{90}{$\leq$} & \\ & x' & \boxed{\xrightarrow{+} \quad y'} & & \exists \end{array}$$

## Well structured transition system (F, ICALP'87)

$S = (X, \rightarrow, \leq)$ where

- $X$ set,
- $\rightarrow \subseteq X \times X$,
- strong monotony,
- well-quasi-ordered.

$$\begin{array}{ccccc} \forall & x & \rightarrow & y & \\ & \rotatebox{90}{$\leq$} & & \rotatebox{90}{$\leq$} & \\ & x' & \boxed{\rightarrow \quad y'} & & \exists \end{array}$$

## Well structured transition system (F, ICALP'87)

$S = (X, \to, \leq)$ where

- $X$ set,
- $\to \subseteq X \times X$,
- monotony,
- well-quasi-ordered:
  $\forall x_0, x_1, \ldots \; \exists i < j \;$ s.t. $\; x_i \leq x_j$.

### The magical theorem of wqo

$(X, \leq)$ is a wqo if and only if every upward closed set $U = \uparrow U \subseteq X$ has a finite basis, i.e., it is equal to a finite union of elements $\uparrow u_i$ with $u_i \in U$.

### Many caracterisations of wqo

$\leq$ is a wqo if and only if $\leq$ is FAC $+$ WF.

# WSTS Everywhere! (F, Schnoebelen LATIN'98, TCS'01)

- $T(w)$ = length of a longest computation starting from $w \in \Sigma^*$.
- $T(w) \in \mathbb{N}_\omega$.
- $w \leq_T w'$ if $T(w) \leq T(w')$.
- $\leq_T$ is a wqo on $\Sigma^*$.

# WSTS Everywhere! (F, Schnoebelen LATIN'98, TCS'01)

- $T(w)$ = length of a longest computation starting from $w \in \Sigma^*$.
- $T(w) \in \mathbb{N}_\omega$.
- $w \leq_T w'$ if $T(w) \leq T(w')$.
- $\leq_T$ is a wqo on $\Sigma^*$.

**Theorem**

*Turing machines are WSTS with strict and strong monotony wrt $\leq_T$.*

# WSTS Everywhere!

- $\leq_T$ is not decidable.
- Hence TM are non-effective WSTS.
- This also proves that there is no (non-trivial) decidability result for non-effective WSTS (not surprising !).

## Objective

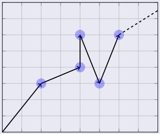
We want to study the usual reachability problems, e.g.,

- Reachability...but it is undecidable for general WSTS :((

## Objective

We want to study the usual reachability problems, e.g.,

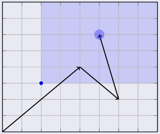
- Reachability...but it is undecidable for general WSTS :((
- Termination



## Objective

We want to study the usual reachability problems, e.g.,

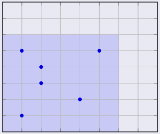
- Reachability...but it is undecidable for general WSTS :((
- Termination
- Coverability (the most used property)



## Objective

We want to study the usual reachability problems, e.g.,

- Reachability...but it is undecidable for general WSTS :((
- Termination
- Coverability (the most used property)
- Boundedness



## Objective

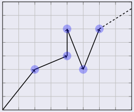
We want to study the usual reachability problems, e.g.,

- Reachability...but it is undecidable for general WSTS :((
- Termination
- Coverability (the most used property)
- Boundedness
- And other properties like eventuality, simulation by finite automaton...

## Termination

*Input*: $(X, \to, \leq)$ a WSTS, $x_0 \in X$.

*Question:* $\exists x_0 \to x_1 \to x_2 \to \ldots ?$



# Termination

- Decidable for post-effective finitely branching WSTS with transitive monotony (F, ICALP'87)

# Termination

- Decidable for post-effective finitely branching WSTS with transitive monotony (F, ICALP'87)
- Undecidable for post-effective finitely branching WSTS with non-transitive monotony (Blondin-F-McKenzie, 2016).

# Termination

- Decidable for post-effective finitely branching WSTS with transitive monotony (F, ICALP'87)
- Undecidable for post-effective finitely branching WSTS with non-transitive monotony (Blondin-F-McKenzie, 2016).
- Undecidable for post-effective infinitely branching WSTS with strict and strong monotony (deduced from Dufourd, Jančar & Schnoebelen, ICALP'99).

# Termination

- Decidable for post-effective finitely branching WSTS with transitive monotony (F, ICALP'87)
- Undecidable for post-effective finitely branching WSTS with non-transitive monotony (Blondin-F-McKenzie, 2016).
- Undecidable for post-effective infinitely branching WSTS with strict and strong monotony (deduced from Dufourd, Jančar & Schnoebelen, ICALP'99).
- Undecidable for non-effective finitely branching WSTS with strict and strong monotony (F-Schnoebelen, TCS'01), since every TM is a WSTS for $\leq_T$.

## Proposition (2016)

Termination is undecidable for post-effective finitely branching WSTS with non-transitive monotony.

## Proposition (2016)

Termination is undecidable for post-effective finitely branching WSTS with non-transitive monotony.

## Proof

We give a reduction from the halting problem.
Let $M_i$ be a TM, and let $S_i = (\mathbb{N}, \rightarrow_i, \leq)$ defined by:
$x \rightarrow_i x + 1$ if $M_i$ does not halt in $\leq x$ steps. Let $C = \{S_i \mid i \geq 0\}$.
$S_i$ is finitely branching, post-effective, monotone but not transitive and $\leq$ is a wpo.

## Proposition (2016)

Termination is undecidable for post-effective finitely branching WSTS with non-transitive monotony.

## Proof

We give a reduction from the halting problem.
Let $M_i$ be a TM, and let $S_i = (\mathbb{N}, \rightarrow_i, \leq)$ defined by:
$x \rightarrow_i x + 1$ if $M_i$ does not halt in $\leq x$ steps. Let $C = \{S_i \mid i \geq 0\}$.
$S_i$ is finitely branching, post-effective, monotone but not transitive and $\leq$ is a wpo.

Now, $\exists$ infinite run $x_0 = 0 \rightarrow_i x_1 \rightarrow_i \ldots$ iff $M_i$ does not halt.
Hence termination for $C$ is undecidable. $\square$

## The survey for termination

| Post-effective | Finitely branching | Transitive | Decidability |
|---|---|---|---|
| Yes | Yes | Yes | Decidable [F87] |
| non effective | Yes | Yes + strict-strong | Undecidable [FS01] |
| Yes | Yes | NO | Undecidable [BFM16] |
| Yes | NO | Yes + strict-strong | Undecidable [BFM14] |

# Boundeness

- Decidable for post-effective finitely branching WSTS (with wpo) with strict transitive monotony (F, ICALP'87)

# Boundeness

- Decidable for post-effective finitely branching WSTS (with wpo) with strict transitive monotony (F, ICALP'87)
- Decidable for post-effective infinitely branching WSTS (with wpo) with strict non-transitive monotony (Blondin-F-McKenzie, 2016).

# Boundeness

- Decidable for post-effective finitely branching WSTS (with wpo) with strict transitive monotony (F, ICALP'87)

- Decidable for post-effective infinitely branching WSTS (with wpo) with strict non-transitive monotony (Blondin-F-McKenzie, 2016).

- Undecidable for post-effective finitely branching WSTS (with wpo) with strong monotony (deduced from Dufourd, Jančar & Schnoebelen, ICALP'99).

# Boundeness

- Decidable for post-effective finitely branching WSTS (with wpo) with strict transitive monotony (F, ICALP'87)
- Decidable for post-effective infinitely branching WSTS (with wpo) with strict non-transitive monotony (Blondin-F-McKenzie, 2016).
- Undecidable for post-effective finitely branching WSTS (with wpo) with strong monotony (deduced from Dufourd, Jančar & Schnoebelen, ICALP'99).
- Undecidable for non-effective finitely branching WSTS (with wpo) with strict and strong monotony (F-Schnoebelen, TCS'01), since every TM is a WSTS for $\leq_T$.

## The survey for boundedness

| **Post-effective** | **Finitely branching** | **Strict monotony** | **wpo** | **Decidability** |
|---|---|---|---|---|
| Yes | Yes | Yes | Yes | D [F87] |
| non effective | Yes | Yes + strong | Yes | U [FS01] |
| Yes | Yes | NO but strong | Yes | U [ICALP'98] |
| Yes | NO | Yes | Yes | D [BFM'16] |
| Yes | Yes | Yes | wqo | ??? |
| Yes | NO | Yes | wqo | ??? |

Exercise: Is the boundedness problem decidable for WSTS with strict monotony ?

# A survey on WSTS

Alain Finkel

LSV, ENS Paris-Saclay (ex ENS Cachan)

IIT Mumbai, India
5*th* March 2018

- Based on joint works with Michael Blondin, Jean Goubault-Larrecq & Pierre McKenzie.

# Coming back with exercises

- Say that a sequence $x_0, x_1, \ldots$ is bad if there are no $i, j$ s.t. $i < j$ and $x_i \leq x_j$

# Coming back with exercises

- Say that a sequence $x_0, x_1, \ldots$ is bad if there are no $i, j$ s.t. $i < j$ and $x_i \leq x_j$
- What is the maximal length of bad sequences begining with $n$ in $(\mathbb{N}, \leq)$ with $(n, n)$ in $(\mathbb{N}^2, \leq)$, and with $(n, n, n)$ in $(\mathbb{N}^3, \leq)$ ?

# Coming back with exercises

- Say that a sequence $x_0, x_1, \ldots$ is bad if there are no $i, j$ s.t. $i < j$ and $x_i \leq x_j$
- What is the maximal length of bad sequences begining with $n$ in $(\mathbb{N}, \leq)$ with $(n, n)$ in $(\mathbb{N}^2, \leq)$, and with $(n, n, n)$ in $(\mathbb{N}^3, \leq)$ ?
- Let us prove that
  $\forall x_0, x_1, \ldots \; \exists i < j \;$ s.t. $\; x_i \leq x_j$ implies

# Coming back with exercises

- Say that a sequence $x_0, x_1, \ldots$ is bad if there are no $i, j$ s.t. $i < j$ and $x_i \leq x_j$
- What is the maximal length of bad sequences begining with $n$ in $(\mathbb{N}, \leq)$ with $(n, n)$ in $(\mathbb{N}^2, \leq)$, and with $(n, n, n)$ in $(\mathbb{N}^3, \leq)$ ?
- Let us prove that $\forall x_0, x_1, \ldots \; \exists i < j$ s.t. $x_i \leq x_j$ implies $\forall x_0, x_1, \ldots \; \exists i_1 < i_2 < \ldots < i_n < ..$ s.t. $x_{i_1} \leq x_{i_2} \leq \ldots \leq x_{i_n} \leq .$

# Coming back with exercises

- Say that a sequence $x_0, x_1, \ldots$ is bad if there are no $i, j$ s.t. $i < j$ and $x_i \leq x_j$
- What is the maximal length of bad sequences begining with $n$ in $(\mathbb{N}, \leq)$ with $(n, n)$ in $(\mathbb{N}^2, \leq)$, and with $(n, n, n)$ in $(\mathbb{N}^3, \leq)$ ?
- Let us prove that $\forall x_0, x_1, \ldots \ \exists i < j$ s.t. $x_i \leq x_j$ implies $\forall x_0, x_1, \ldots \ \exists i_1 < i_2 < \ldots < i_n < ..$ s.t. $x_{i_1} \leq x_{i_2} \leq \ldots \leq x_{i_n} \leq .$
- PROOF: Define the set $A = \{i \mid \forall j > i; x_i \not\leq x_j\}$. $A$ is finite else contradiction; let $k$ the largest index of $x_k$ in $A$, hence for all $i > k$, one may construct an infinite non-decreasing sequence from $x_i$.

# A quick story of coverability in WSTS

## Coverability

For monotone transition systems, $y$ is coverable from $x$ if

- $\exists x' \mid x \xrightarrow{*} x' \geq y$ ( this is the definition !) iff

## Coverability

For monotone transition systems, $y$ is coverable from $x$ if

- $\exists x' \mid x \xrightarrow{*} x' \geq y$ ( this is the definition !) iff
- $x \in \mathrm{Pre}^*(\uparrow y)$ ( this could be the definition !) iff

## Coverability

For monotone transition systems, $y$ is coverable from $x$ if

- $\exists x' \mid x \xrightarrow{*} x' \geq y$ ( this is the definition !) iff
- $x \in \mathrm{Pre}^*(\uparrow y)$ ( this could be the definition !) iff
- $y \in \downarrow \mathrm{Post}^*(x)$ ( this could be the definition !).

## Remark

- $\mathrm{Pre}^*(\uparrow y) = \uparrow \mathrm{Pre}^*(\uparrow y)$

## Coverability

For monotone transition systems, $y$ is coverable from $x$ if

- $\exists x' \mid x \xrightarrow{*} x' \geq y$ ( this is the definition !) iff
- $x \in \mathrm{Pre}^*(\uparrow y)$ ( this could be the definition !) iff
- $y \in \downarrow \mathrm{Post}^*(x)$ ( this could be the definition !).

## Remark

- $\mathrm{Pre}^*(\uparrow y) = \uparrow \mathrm{Pre}^*(\uparrow y)$
- $\downarrow \mathrm{Post}^*(x) = \downarrow \mathrm{Post}^*(\downarrow x)$.

## A conceptual coverability algorithm, not the original

Execute two procedures in parallel, one looking for a coverability certificate and one looking for a non coverability certificate.

## A conceptual coverability algorithm, not the original

Execute two procedures in parallel, one looking for a coverability certificate and one looking for a non coverability certificate.

- Coverability is semi-decidable:
  - if $\exists x' \geq y,\ x \xrightarrow{*} x'$, one finally will find $x'$.

Enumeration of upward closed sets by their finite basis is a consequence of $(X, \leq)$ is WQO.

## A conceptual coverability algorithm, not the original

Execute two procedures in parallel, one looking for a coverability certificate and one looking for a non coverability certificate.

- Coverability is semi-decidable:
  - if $\exists x' \geq y,\ x \xrightarrow{*} x'$, one finally will find $x'$.
- Non-coverability is also semi-decidable:
  - $\neg(\exists x' \geq y,\ x \xrightarrow{*} x')$ iff $x \notin Pre^*(\uparrow y) = \uparrow J_m$ for some $m$.

Enumeration of upward closed sets by their finite basis is a consequence of $(X, \leq)$ is WQO.

## A conceptual coverability algorithm, not the original

Execute two procedures in parallel, one looking for a coverability certificate and one looking for a non coverability certificate.

- Coverability is semi-decidable:
  - if $\exists x' \geq y$, $x \xrightarrow{*} x'$, one finally will find $x'$.
- Non-coverability is also semi-decidable:
  - $\neg(\exists x' \geq y, x \xrightarrow{*} x')$ iff $x \notin Pre^*(\uparrow y) = \uparrow J_m$ for some $m$.
  - One enumerates all the finite sets (*) $J \subseteq X$ such that $y \in \uparrow J$ and $Pre(\uparrow J) \subseteq \uparrow J$ (hence $Pre^*(\uparrow J) = \uparrow J$) and $x \notin \uparrow J$, hence $Pre^*(\uparrow y) = \uparrow J_m \subseteq \uparrow J = Pre^*(\uparrow J)$.

Enumeration of upward closed sets by their finite basis is a consequence of $(X, \leq)$ is WQO.

## A conceptual coverability algorithm, not the original

Execute two procedures in parallel, one looking for a coverability certificate and one looking for a non coverability certificate.

- Coverability is semi-decidable:
  - if $\exists x' \geq y$, $x \xrightarrow{*} x'$, one finally will find $x'$.

- Non-coverability is also semi-decidable:
  - $\neg(\exists x' \geq y,\ x \xrightarrow{*} x')$ iff $x \notin Pre^*(\uparrow y) = \uparrow J_m$ for some $m$.
  - One enumerates all the finite sets (*) $J \subseteq X$ such that $y \in \uparrow J$ and $Pre(\uparrow J) \subseteq \uparrow J$ (hence $Pre^*(\uparrow J) = \uparrow J$) and $x \notin \uparrow J$, hence $Pre^*(\uparrow y) = \uparrow J_m \subseteq \uparrow J = Pre^*(\uparrow J)$.
  - Since we are sure that at least one $J$ exists ($J_m$ !), one finally will find one. May be we find a large $J_p$ s.t. $\uparrow J_m = Pre^*(\uparrow y) \subsetneq \uparrow J_p$ but $x \notin \uparrow J_p \implies x \notin Pre^*(\uparrow y)$.

    Enumeration of upward closed sets by their finite basis is a consequence of $(X, \leq)$ is WQO.

## The story of the backward coverability algorithm

- 1978: coverability for reset VAS is decidable (Arnold and Latteux published in French in CALCOLO'78). Their algorithm is an instance of the backward algorithm (LICS'96).
- 1993: decidability of coverability for LCS (Abdulla, Cerans, Jonsson, Tsay, LICS'93)
- 1996: decidability of coverability for strong WSTS assuming Pre($\uparrow x$) is computable (Abdulla, Cerans, Jonsson, Tsay, LICS'96)
- 1998: decidability of coverability for WSTS assuming $\uparrow$Pre($\uparrow x$) is computable (F., Schnoebelen LATIN'98)

## Remarks on the backward coverability algorithm

- It computes $\mathrm{Pre}^*(\uparrow y)$ that is more than solving coverability.
- It is often but not always computable, ex: depth-bounded processes (Wies, Zufferey, Henzinger, FOSSACS'10)
- Backward algorithms are often less efficient than forward algorithms.

## The downward approach for coverability

- Initially presented by Geeraerts, Raskin, and Van Begin (FSTTCS'04) for strongly monotone WSTS with Adequate Domain of Limits (ADL).

## The downward approach for coverability

- Initially presented by Geeraerts, Raskin, and Van Begin (FSTTCS'04) for strongly monotone WSTS with Adequate Domain of Limits (ADL).
- Simplified and extended with Goubault-Larrecq (STACS'09): ADL is not an hypothesis, it always exists.

## The downward approach for coverability

- Initially presented by Geeraerts, Raskin, and Van Begin (FSTTCS'04) for strongly monotone WSTS with Adequate Domain of Limits (ADL).
- Simplified and extended with Goubault-Larrecq (STACS'09): ADL is not an hypothesis, it always exists.
- Still simplified and extended with Blondin, McKenzie (ICALP'14): ideal completion for infinitely branching.

## The downward approach for coverability

- Initially presented by Geeraerts, Raskin, and Van Begin (FSTTCS'04) for strongly monotone WSTS with Adequate Domain of Limits (ADL).
- Simplified and extended with Goubault-Larrecq (STACS'09): ADL is not an hypothesis, it always exists.
- Still simplified and extended with Blondin, McKenzie (ICALP'14): ideal completion for infinitely branching.
- Still simplified and extended with Blondin, McKenzie: WQO is not necessary. Decidable for more than WSTS. (arxiv, august 2016, in LMCS'2017).

$y$ is not coverable from $x$ iff $y \notin \downarrow\mathrm{Post}^*(x)$.

Let $(D_i)_i$ be an enumeration of dcs, hence $\downarrow\mathrm{Post}^*(x) = D_m$, for some $m$.

**procedure 2: enumerates dcs to find non coverability certificate of $y$ from $x$**

$i \leftarrow 0$;
**while** $\neg(\downarrow\mathrm{Post}(D_i) \subseteq D_i$ and $x \in D_i$ and $y \notin D_i)$ **do**
    $i \leftarrow i + 1$
**return** *false*

## Effective hypotheses

- dcs are recursive.
- Union of dcs is computable
- $\downarrow\mathrm{Post}(D)$ is computable.
- Inclusion between dcs is decidable.
- Works for post effective infinitely branching systems.

## Theorem

Let $S = (X, \rightarrow, \leq)$ be a monotone transition system + there exists an enumeration of downward closed sets of $X$, and let $x, y \in X$.

1. $y$ is coverable from $x$ iff Procedure 1 terminates.
2. $y$ is not coverable from $x$ iff Procedure 2 terminates.

This theorem does not provide an algorithm.

## Remark

WSTS, hence WQO implies possible enumeration of downward closed sets (by minimal elements of upward closed sets) but the converse is false: $(\mathbb{Z}, \leq)$ is not WQO but one may enumerate the $D_i$ as follows: $D_i = \downarrow x_i$ for $x_i \in \mathbb{Z}$ or $D_i = \mathbb{Z}$.

**Question**

How to enumerate downward closed sets ?

**Answer**

By enumerating ideals ! (come to the next seminar tomorrow)

### With the $2^{nd}$ magical theorem of wqo

If $\leq$ is a wqo then every downward closed set $D = \downarrow D$ has a finite basis, i.e., it is equal to a finite union of ideals.
(ideal $=$ downward closed set $+$ directed).

### Remark

It is an if then but not an if and only if.

### We will see a more magical theorem of FAC $=$ "half wqo"

Come tomorrow !

**We are tomorrow !**

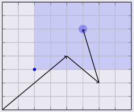
$\leq$ is FAC if and only if every downward closed set $D = \downarrow D$ has a finite basis, i.e., it is equal to a finite union of ideals.

The proof is in the paper WBTS in LMCS'2017.

## Coverability

*Input*: $(X, \rightarrow, \leq)$ a WSTS, $x, y \in X$.

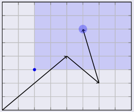
*Question:* $x \xrightarrow{*} x' \geq y$?



## Coverability

*Input*: $(X, \to, \leq)$ a WSTS, $x, y \in X$.

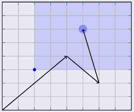
*Question:* $y \in \downarrow \mathrm{Post}^*(x)$?



## Coverability

*Input*: $(X, \to, \leq)$ a WSTS, $x, y \in X$.

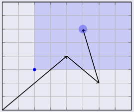
*Question:* $y \in \downarrow \mathrm{Post}^*(x)$?



## Coverability

*Input*: $(X, \rightarrow, \leq)$ a WSTS, $x, y \in X$.

*Question:* $y \in \downarrow \mathrm{Post}^*(x)$?



## *Forward* method

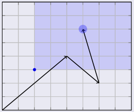
Coverability:

- Enumerate executions $\downarrow x \xrightarrow{*} D$,
- Accept if $y \in D$.

## Coverability

*Input*: $(X, \rightarrow, \leq)$ a WSTS, $x, y \in X$.

*Question:* $y \in \downarrow \mathrm{Post}^*(x)$?



## *Forward* method

Coverability:

- Enumerate executions $\downarrow x \xrightarrow{*} D$,
- Accept if $y \in D$.

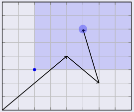
Non coverability:

- Enumerate

## Coverability

*Input*: $(X, \rightarrow, \leq)$ a WSTS, $x, y \in X$.

*Question:* $y \in \downarrow\text{Post}^*(x)$?



## *Forward* method

Coverability:

- Enumerate executions $\downarrow x \xrightarrow{*} D$,
- Accept if $y \in D$.

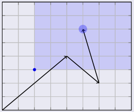
Non coverability:

- Enumerate $D \subseteq X$ downward closed, $x \in D$ and $\downarrow\text{Post}(D) \subseteq D$
- Reject if $y \notin D$.

## Coverability

*Input*: $(X, \to, \leq)$ a WSTS, $x, y \in X$.

*Question*: $y \in \downarrow \mathrm{Post}^*(x)$?



## *Forward* method

Coverability:

- Enumerate executions $\downarrow x \xrightarrow{*} D$,
- Accept if $y \in D$.

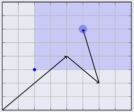
Non coverability:

- Enumerate $D = I_1 \cup \ldots \cup I_k$
- Reject if $y \notin D$.

## Coverability

*Input*: $(X, \rightarrow, \leq)$ a WSTS, $x, y \in X$.

*Question:* $y \in \downarrow\mathrm{Post}^*(x)$?



## *Forward* method

Coverability:

- Enumerate executions $\downarrow x \xrightarrow{*} D$,
- Accept if $y \in D$.

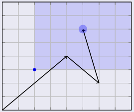
Non coverability:

- Enumerate $D \subseteq X$ downward closed
- Reject if $y \notin D$.

## Coverability

*Input*: $(X, \to, \leq)$ a WSTS, $x, y \in X$.

*Question:* $y \in \downarrow \mathrm{Post}^*(x)$?



## *Forward* method

Coverability:

- Enumerate executions $\downarrow x \xrightarrow{*} D$,
- Accept if $y \in D$.

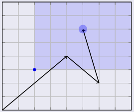
Non coverability:

- Enumerate $D \subseteq X$ downward closed, $x \in D$
- Reject if $y \notin D$.

## Coverability

*Input*: $(X, \to, \leq)$ a WSTS, $x, y \in X$.

*Question:* $y \in \downarrow \mathrm{Post}^*(x)$?



## *Forward* method

Coverability:

- Enumerate executions $\downarrow x \xrightarrow{*} D$,
- Accept if $y \in D$.

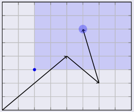
Non coverability:

- Enumerate $D \subseteq X$ downward closed, $\downarrow x \subseteq I_1 \cup \ldots \cup I_k$
- Reject if $y \notin D$.

## Coverability

*Input*: $(X, \rightarrow, \leq)$ a WSTS, $x, y \in X$.

*Question:* $y \in \downarrow \mathrm{Post}^*(x)$?



## *Forward* method

Coverability:

- Enumerate executions $\downarrow x \xrightarrow{*} D$,
- Accept if $y \in D$.

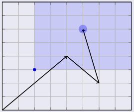
Non coverability:

- Enumerate $D \subseteq X$ downward closed, $\exists j$ s.t. $\downarrow x \subseteq I_j$
- Reject if $y \notin D$.

## Coverability

*Input*: $(X, \rightarrow, \leq)$ a WSTS, $x, y \in X$.

*Question:* $y \in \downarrow \mathrm{Post}^*(x)$?



## *Forward* method

Coverability:

- Enumerate executions $\downarrow x \xrightarrow{*} D$,
- Accept if $y \in D$.

Non coverability:

- Enumerate $D \subseteq X$ downward closed, $x \in D$ and $\downarrow \mathrm{Post}(D) \subseteq D$
- Reject if $y \notin D$.

## The survey/story of coverability for WSTS

| **Year** | **Authors** | **Mathematical hyp.** | **Effectivity hyp.** | **back/forward** |
|---|---|---|---|---|
| 1978 | Arnold & Latteux | reset VAS | YES | backward |
| 1987 | F. | very WSTS (strong+strict, $\omega^2$-wqo,...) | effective very WSTS | forward |
| 1996 | Abdulla & CJT | strong monotony | $\mathrm{Pre}_S(\uparrow x)$ comp. | backward |
| 1998 | F. Schnoebelen | monotony | $\uparrow \mathrm{Pre}_S(\uparrow x)$ comp. | backward |
| 2004 | Geeraerts & RV | strong monotony, ADL | effective ADL | forward |
| 2006 | Geeraerts & RV | monotony, ADL | effective ADL | forward |
| 2009 | F. & Goubault-Larrecq | strong monotony, weak ADL, flattable | effective WADL | forward |
| 2009 | F. & Goubault-Larrecq | strong monotony, flattable | ideally effective | forward |
| 2014 | Blondin & FM | monotony, | ideally effective | forward |
| **2016** | Blondin & FM | monotony, no wqo but FAC | ideally effective | forward |
| **2017** | Trivial | no monotony, wqo (Minsky machines) | ideally effective | Undec. |
| **2017** | Sutre | monotony, no wqo but WF | ideally effective | Undec |

## A survey (to complete) of KM algorithms for WSTS

| Year | Authors | Model | Termination |
|---|---|---|---|
| 1969 | Karp & Miller | VASS | YES |
| 1978 | Valk | post self-modifying PN | YES |
| 1978 | Valk | self-modifying PN | NO |
| 1994 | Abdulla & Jonsson | LCS | NO |
| 1998 | Dufourd & F. & Schnoebelen | 3-dim reset/transfer VASS | NO |
| 1998 | Emerson & Namjoshi | WSTS model checking | NO |
| 1999 | Esparza & F. & Mayr | broadcast protocols & transfer PN | NO |
| 2000 | F. & Sutre | 2-dim reset/transfer VASS | YES |
| 2004 | F. & McKenzie & Picaronny | strongly increasing $\omega$-resursive nets | YES |
| 2004 | Raskin & Van Begin | PN+NBA | NO |
| 2005 | Goubault-Larrecq & Verma | BVASS | YES |
| 2009 | F. & Goubault-Larrecq | $\omega^2$-WSTS, cover-flattable | YES |
| 2010 | F. & Sangnier | PN+0-test | YES |
| 2011 | Acciai, Boreale, Henzinger, Meyer,... | depth-bounded processes, $\nu$-PN | NO |
| 2011 | Chambard & F. & Schmitz | trace-bounded $\omega^2$-WSTS | YES |
| 2013 | Geeraerts & Heußner & Praveen & Raskin | $\omega$-PN | YES |
| 2013 | Hüchting & Majumdar & Meyer | name-bounded $\pi$-calculus processes | YES |
| 2016 | Hofman & Lasota & Lazic & Leroux & ST | unordered PN | YES |

- ICALP'87 (F)
  - WSTS definitions
  - decidability of termination
  - decidability of boundedness
  - computation of the coverability set hence decidability of coverability (under stronger hyp.)

- ICALP'87 (F)
  - WSTS definitions
  - decidability of termination
  - decidability of boundedness
  - computation of the coverability set hence decidability of coverability (under stronger hyp.)
- LICS'96 (Abdulla, Cerans, Jonsson, Tsay)
  - decidability of coverability with a backward algorithm
  - decidability of simulation with finite-state systems
  - undecidability of repeated control-state (for LCS).

- ICALP'87 (F)
  - WSTS definitions
  - decidability of termination
  - decidability of boundedness
  - computation of the coverability set hence decidability of coverability (under stronger hyp.)
- LICS'96 (Abdulla, Cerans, Jonsson, Tsay)
  - decidability of coverability with a backward algorithm
  - decidability of simulation with finite-state systems
  - undecidability of repeated control-state (for LCS).
- LICS'98 (Emerson, Namjoshi), LICS'99 (Esparza, F, Mayr)
  - broadcast protocols are WSTS
  - model checking of WSTS (with procedures)
- WSTS everywhere, TCS'01 (F, Schnoebelen)

- FSTTCS'04 (Geeraerts, Raskin and Van Begin):
  - The first forward coverability algorithm for WSTS (with ADL).
- STACS'09, ICALP'09 (F, Goubault-Larrecq), ICALP'14 (Blondin, F, McKenzie)
  - ADL is not an hypothesis.
  - Ideal completion of any WSTS
  - Computation of the clover for flattable WSTS
  - $\omega^2$-WSTS are completable and robust....

- FSTTCS'04 (Geeraerts, Raskin and Van Begin):
  - The first forward coverability algorithm for WSTS (with ADL).
- STACS'09, ICALP'09 (F, Goubault-Larrecq), ICALP'14 (Blondin, F, McKenzie)
  - ADL is not an hypothesis.
  - Ideal completion of any WSTS
  - Computation of the clover for flattable WSTS
  - $\omega^2$-WSTS are completable and robust....
- 2015-2016: Use of ideals decomposition in:
  - RP'15: The Ideal View on Rackoff's Coverability Technique (Lazić, Schmitz)
  - LICS'15: Demystifying Reachability in Vector Addition Systems (Leroux, Schmitz).
  - FOSSACS'16: Coverability Trees for Petri Nets with Unordered Data (Schmitz and a lot of authors...)
  - LICS'16: $\nu$-Petri nets (Lazić, Schmitz).
  - ...

# WSTS Everywhere!

- $S = (\mathbb{N}^k, \leq)$.
  - Petri nets: WSTS with strict and strong monotony.
  - Positive Affine nets, Reset/Transfer Petri nets: WSTS with strong (but not strict) monotony.

# WSTS Everywhere!

- $S = (\mathbb{N}^k, \leq)$.
  - Petri nets: WSTS with strict and strong monotony.
  - Positive Affine nets, Reset/Transfer Petri nets: WSTS with strong (but not strict) monotony.
- $S = (Q \times \Sigma^{*k}, = \times \sqsubseteq^k)$.
  - LCS: WSTS with non-strict monotony.

# WSTS still verywhere!

- Data nets: $S = (Q \times \mathbb{N}^k)^*$
  - Lazic, Newcomb, Ouaknine, Roscoe, Worrell (PN'07)
  - Hofman, Lasota, Lazić, Leroux, Schmitz, Totzke (FOSSACS'16).
  - Lasota (PN'16)
- $\nu$-Petri nets: $S = (Q \times \mathbb{N}^k)^\oplus$.
  - Rosa-Velardo, de Frutos-Escrig (PN'07)
  - Lazić and Schmitz (LICS'16).
- Pi-calculus: Depth-Bounded Processes (trees).
  - Wies, Zufferey, Henzinger (FOSSACS'10, VMCAI'12).
- Timed Petri nets: $Regions = ((Q \times \mathbb{N}^k)^\oplus)^*$
  - Bonnet, F, Haddad, Rosa-Velardo (FOSSACS'10)
  - Haddad, Schmitz, Schnoebelen (LICS'12).
- Process algebra (BPP,...).

## Further work

## Further work

- Explore more in details WBTS and find applications of WBTS (comme tomorrow).

## Further work

- Explore more in details WBTS and find applications of WBTS (comme tomorrow).
- Computing efficiently with ideals (no brut force enumeration).

## Further work

- Explore more in details WBTS and find applications of WBTS (comme tomorrow).
- Computing efficiently with ideals (no brut force enumeration).
- Design Karp-Miller algorithm for $\omega^2$-WSTS (FSTTCS'2017).

## Further work

- Explore more in details WBTS and find applications of WBTS (comme tomorrow).
- Computing efficiently with ideals (no brut force enumeration).
- Design Karp-Miller algorithm for $\omega^2$-WSTS (FSTTCS'2017).
- Go to model checking.

Interships available: ENS Paris-Saclay, CSA, MSR,...many levels: Bachelor, Master, PhD, post-PhD

## Further work

- Explore more in details WBTS and find applications of WBTS (comme tomorrow).
- Computing efficiently with ideals (no brut force enumeration).
- Design Karp-Miller algorithm for $\omega^2$-WSTS (FSTTCS'2017).
- Go to model checking.

## Interships available: ENS Paris-Saclay, CSA, MSR,...many levels: Bachelor, Master, PhD, post-PhD

- Different topics: theoretical and/or applied subjects.

## Further work

- Explore more in details WBTS and find applications of WBTS (comme tomorrow).
- Computing efficiently with ideals (no brut force enumeration).
- Design Karp-Miller algorithm for $\omega^2$-WSTS (FSTTCS'2017).
- Go to model checking.

## Interships available: ENS Paris-Saclay, CSA, MSR,...many levels: Bachelor, Master, PhD, post-PhD

- Different topics: theoretical and/or applied subjects.
- Developping the WSTS theory and a prototype for finding bugs in web services and choreographies.

## Further work

- Explore more in details WBTS and find applications of WBTS (comme tomorrow).
- Computing efficiently with ideals (no brut force enumeration).
- Design Karp-Miller algorithm for $\omega^2$-WSTS (FSTTCS'2017).
- Go to model checking.

## Interships available: ENS Paris-Saclay, CSA, MSR,...many levels: Bachelor, Master, PhD, post-PhD

- Different topics: theoretical and/or applied subjects.
- Developping the WSTS theory and a prototype for finding bugs in web services and choreographies.
- Make the first efficient prototype for reachability for Petri nets.

# Thank you!