A survey on WSTS

Alain Finkel

LSV, ENS Paris-Saclay (ex ENS Cachan)

IIT Mumbai, India

5th March 2018

Based on joint works with Michael Blondin, Jean Goubault-Larrecq & Pierre McKenzie.
Exercise 1

- $T(w) = \text{length of a longest computation starting from } w \in \Sigma^*.$
- $T(w) \in \mathbb{N}_\omega.$
- $w \leq_T w'$ if $T(w) \leq T(w').$
Exercise 1

- \( T(w) = \) 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 2

\( y \) is not coverable from \( x \) iff \( y \not\in \downarrow \text{Post}^*(x) \).

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

**procedure 2: non coverability certificate of \( y \) from \( x \)**

```plaintext
while \( \neg (\downarrow \text{Post} (S_i) \subseteq S_i \text{ and } x \in S_i \text{ and } y \not\in S_i) \) do
    \( i \leftarrow i + 1 \)
return false
```

**procedure 2: non coverability certificate of \( y \) from \( x \)**

```plaintext
while \( \neg (\text{Pre} (\uparrow F_i) \subseteq \uparrow F_i \text{ and } x \not\in \uparrow F_i \text{ and } y \in \uparrow F_i) \) do
    \( i \leftarrow i + 1 \)
return false
```
- Find a picture for representing $Pre^*$-coverability semi-algorithm.

- Find a picture for representing $Post^*$-coverability semi-algorithm.
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-ω extensions, lossy fifo systems, broadcast protocols, CFG, graph rewriting, depth bounded processes, fragments of the π-calculus,...)

Example of WSTS: Petri nets

![Petri net diagram]
Well Structured Transition Systems (WSTS) encompass a large number of infinite state systems (PN and reset-transfer-affine-ω extensions, lossy fifo systems, broadcast protocols, CFG, graph rewriting, depth bounded processes, fragments of the π-calculus,....)

Example of WSTS: Petri nets
Multiple decidability results are known for (finitely branching) WSTS.

Example of WSTS: Petri nets

\[
\text{Post}(\bullet \quad \bullet \quad \bullet \quad \bullet) = \quad \bullet \quad \bullet \quad \bullet \quad \bullet
\]
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: ω–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)

$$\text{Post}(\bullet \circ \circ \circ) = \circ \circ \circ \circ, \circ \circ \circ \circ, \circ \circ \circ \circ, \ldots$$
Well structured transition system (F, ICALP’87)

\[ S = (X, \rightarrow, \leq) \] where

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

\[
\forall x_0, x_1, \ldots \exists i < j \text{ s.t. } x_i \leq x_j.
\]

\[
\forall x \rightarrow y \geq x' y' \exists.
\]
Well structured transition system (F, ICALP’87)

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

- \( \mathbb{N}^3 \),
- \( \rightarrow \subseteq X \times X \),
- monotony,
- well-quasi-ordered.
Well structured transition system (F, ICALP’87)

\[ S = (X, \rightarrow, \leq) \] where

- \( X \) set,
- \( \rightarrow \subseteq \mathbb{N}^3 \times \mathbb{N}^3 \),
- 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.
Well structured transition system \((F, \text{ICALP'87})\)

\[ S = (X, \rightarrow, \leq) \text{ where} \]
- \(X\) set,
- \(\rightarrow \subseteq X \times X\),
- \text{monotony},
- \text{well-quasi-ordered}.
Well structured transition system (F, ICALP’87)

\[ S = (X, \rightarrow, \leq) \text{ where} \]

- \( X \) set,
- \( \rightarrow \subseteq X \times X \),
- \text{monotony},
- \text{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.
Well structured transition system \((F, \text{ICALP’87})\)

\[ S = (X, \rightarrow, \leq) \text{ where} \]

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

\[
\begin{align*}
\forall x & \rightarrow y \\
\land & x' \rightarrow^+ y'
\end{align*}
\]
Well structured transition system \((F, ICALP'87)\)

\[ S = (\mathcal{X}, \rightarrow, \leq) \]

- \(\mathcal{X}\) set,
- \(\rightarrow \subseteq \mathcal{X} \times \mathcal{X}\),
- strong monotony,
- well-quasi-ordered.
Well structured transition system \([F, ICALP’87]\)

\[ S = (X, \rightarrow, \leq) \text{ where} \]

- \(X\) set,
- \(\rightarrow \subseteq X \times X\),
- monotony,
- well-quasi-ordered:
  \[ \forall x_0, x_1, \ldots \exists i < j \text{ s.t. } x_i \leq x_j. \]
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 \textit{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:** \((X, \rightarrow, \leq)\) a WSTS, \(x_0 \in X\).

**Question:** \(\exists x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \ldots\)?
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 $C = \{S_i | i \geq 0\}$. $S_i$ 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.
## The survey for termination

<table>
<thead>
<tr>
<th>Post-effective</th>
<th>Finitely branching</th>
<th>Transitive</th>
<th>Decidability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Decidable [F87]</td>
</tr>
<tr>
<td>non effective</td>
<td>Yes</td>
<td>Yes + strict-strong</td>
<td>Undecidable [FS01]</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>NO</td>
<td>Undecidable [BFM16]</td>
</tr>
<tr>
<td>Yes</td>
<td>NO</td>
<td>Yes + strict-strong</td>
<td>Undecidable [BFM14]</td>
</tr>
</tbody>
</table>
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 monotonity (F, ICALP’87)

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

- **Undecidable** for post-effective finitely branching WSTS (with wpo) with strong monotonity (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 $\leq_T$. 
<table>
<thead>
<tr>
<th>Post-effective</th>
<th>Finitely branching</th>
<th>Strict monotony</th>
<th>wpo</th>
<th>Decidability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>D [F87]</td>
</tr>
<tr>
<td>non effective</td>
<td>Yes</td>
<td>Yes + strong</td>
<td>Yes</td>
<td>U [FS01]</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>NO but strong</td>
<td>Yes</td>
<td>U [ICALP’98]</td>
</tr>
<tr>
<td>Yes</td>
<td>NO</td>
<td>Yes</td>
<td>Yes</td>
<td>D [BFM’16]</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>wqo</td>
<td>???</td>
</tr>
<tr>
<td>Yes</td>
<td>NO</td>
<td>Yes</td>
<td>wqo</td>
<td>???</td>
</tr>
</tbody>
</table>

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
5th March 2018

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

$$\forall x_0, x_1, \ldots \exists i < j \text{ s.t. } x_i \leq x_j \text{ implies } \forall x_0, x_1, \ldots \exists i_1 < i_2 < \ldots < i_n < \ldots \text{ s.t. } x_{i_1} \leq x_{i_2} \leq \ldots \leq x_{i_n} \leq .$$
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 beginning 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

\[
\forall x_0, x_1, \ldots \exists i < j \text{ s.t. } x_i \leq x_j \text{ implies } \forall x_0, x_1, \ldots \exists i_1 < i_2 < \ldots < i_n < \ldots \text{ s.t. } x_{i_1} \leq x_{i_2} \leq \ldots \leq x_{i_n} \leq .
\]

- PROOF: Define the set \( A = \{ i \mid \forall j > i; x_i \not< x_j \} \). \( A \) 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 \textit{coverable} from \( x \) if

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

For monotone transition systems, \( y \) is \textit{coverable} from \( x \) if

\begin{itemize}
  \item \( \exists x' \mid x \xrightarrow{*} x' \geq y \) (this is the definition!) iff
  \item \( x \in \text{Pre}^*(\uparrow y) \) (this could be the definition!) iff
  \item \( y \in \downarrow \text{Post}^*(x) \) (this could be the definition!).
\end{itemize}

Remark

\begin{itemize}
  \item \( \text{Pre}^*(\uparrow y) = \uparrow \text{Pre}^*(\uparrow y) \)
\end{itemize}
Coverability

For monotone transition systems, \( y \) is **coverable** from \( x \) if

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

Remark

- \( \text{Pre}^*(\uparrow y) = \uparrow \text{Pre}^*(\uparrow y) \)
- \( \downarrow \text{Post}^*(x) = \downarrow \text{Post}^*(\downarrow x) \).
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 $\exists x' \geq y$, $x \rightarrow x'$, 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 $\exists x' \geq y, \ x \rightarrow x'$, one finally will find $x'$.

- Non-coverability is also semi-decidable:
  - $\neg (\exists x' \geq y, \ x \rightarrow x')$ iff $x \not\in Pre^*(\uparrow y) = \uparrow J_m$ 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 \( x \not\in Pre^*(\uparrow y) = \uparrow J_m \) 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 \( x \not\in \uparrow J \), 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 $\exists x' \geq y$, $x \rightarrow^* x'$, one finally will find $x'$.

- Non-coverability is also semi-decidable:
  - $\neg(\exists x' \geq y, x \rightarrow^* x')$ 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. $\uparrow J_m = Pre^*(\uparrow y) \subset \uparrow J_p$ 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 $\text{Pre}(↑x)$ is computable (Abdulla, Cerans, Jonsson, Tsay, LICS’96)

- 1998: decidability of coverability for WSTS assuming $↑\text{Pre}(↑x)$ is computable (F., Schnoebelen LATIN’98)
Remarks on the backward coverability algorithm

- It computes $\text{Pre}^*(\uparrow y)$ 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 $y \not\in \downarrow \text{Post}^*(x)$.

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

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

```
i ← 0;
while \lnot(\downarrow \text{Post}(D_i) \subseteq D_i \text{ and } x \in D_i \text{ and } y \not\in D_i) \text{ do }
  i ← i + 1
return false
```

**Effective hypotheses**

- dcs are recursive.
- Union of dcs is computable
- $\downarrow \text{Post}(D)$ 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 2nd 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!

≤ 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:** \(x \overset{*}{\rightarrow} x' \geq y\)?
Coverability

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

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

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

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

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

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

**Forward method**

**Coverability:**

- Enumerate executions \(\downarrow x \overset{*}{\rightarrow} D\),
- Accept if \(y \in D\).
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
Coverability

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

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

---

**Forward method**

**Coverability:**
- Enumerate executions \(\downarrow x \rightarrow^* D\),
- 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: \((X, \rightarrow, \leq)\) 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 \(D = I_1 \cup \ldots \cup I_k\)
- Reject if \(y \notin D\).
Coverability

**Input:** $(X, \to, \leq)$ 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
- Reject if $y \notin D$. 

---

An effective forward coverability algorithm
The survey/story of coverability
A survey/story of KM algorithm

Exercises
Preambule
Introduction
A (partial) survey
News on coverability
Still coverability
Conclusion
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, \(x \in D\)
- Reject if \(y \notin D\).
Coverability

Input: \((X, \to, \leq)\) 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, \(\downarrow x \subseteq I_1 \cup \ldots \cup I_k\)
- Reject if \(y \notin D\).
Coverability

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

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

---

**Forward method**

**Coverability:**

- Enumerate executions \(\downarrow x \xrightarrow{\ast} D\),
- Accept if \(y \in D\).

**Non coverability:**

- Enumerate \(D \subseteq X\) downward closed, \(\exists j\) s.t. \(\downarrow x \subseteq l_j\)
- Reject if \(y \notin D\).
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 \(y \notin D\).
The survey/story of coverability for WSTS

<table>
<thead>
<tr>
<th>Year</th>
<th>Authors</th>
<th>Mathematical hyp.</th>
<th>Effectivity hyp.</th>
<th>back/forward</th>
</tr>
</thead>
<tbody>
<tr>
<td>1978</td>
<td>Arnold &amp; Latteux</td>
<td>reset VAS</td>
<td>YES</td>
<td>backward</td>
</tr>
<tr>
<td>1987</td>
<td>F.</td>
<td>very WSTS (strong+strict, $\omega^2$-wqo,...)</td>
<td>effective very WSTS</td>
<td>forward</td>
</tr>
<tr>
<td>1996</td>
<td>Abdulla &amp; CJT</td>
<td>strong monotony</td>
<td>$\text{Pre}_S(\uparrow x)$ comp.</td>
<td>backward</td>
</tr>
<tr>
<td>1998</td>
<td>F. Schnoebelen</td>
<td>monotony</td>
<td>$\uparrow \text{Pre}_S(\uparrow x)$ comp.</td>
<td>backward</td>
</tr>
<tr>
<td>2004</td>
<td>Geeraerts &amp; RV</td>
<td>strong monotony, ADL</td>
<td>effective ADL</td>
<td>forward</td>
</tr>
<tr>
<td>2006</td>
<td>Geeraerts &amp; RV</td>
<td>monotony, ADL</td>
<td>effective ADL</td>
<td>forward</td>
</tr>
<tr>
<td>2009</td>
<td>F. &amp; Goubault-Larrecq</td>
<td>strong monotony, weak ADL, flattable</td>
<td>effective WADL</td>
<td>forward</td>
</tr>
<tr>
<td>2009</td>
<td>F. &amp; Goubault-Larrecq</td>
<td>strong monotony, fltable</td>
<td>ideally effective</td>
<td>forward</td>
</tr>
<tr>
<td>2014</td>
<td>Blondin &amp; FM</td>
<td>monotony,</td>
<td>ideally effective</td>
<td>forward</td>
</tr>
<tr>
<td>2016</td>
<td>Blondin &amp; FM</td>
<td>monotony, no wqo but FAC</td>
<td>ideally effective</td>
<td>forward</td>
</tr>
<tr>
<td>2017</td>
<td>Trivial</td>
<td>no monotony, wqo (Minsky machines)</td>
<td>ideally effective</td>
<td>Undec.</td>
</tr>
<tr>
<td>2017</td>
<td>Sutre</td>
<td>monotony, no wqo but WF</td>
<td>ideally effective</td>
<td>Undec.</td>
</tr>
</tbody>
</table>
# A survey (to complete) of KM algorithms for WSTS

<table>
<thead>
<tr>
<th>Year</th>
<th>Authors</th>
<th>Model</th>
<th>Termination</th>
</tr>
</thead>
<tbody>
<tr>
<td>1969</td>
<td>Karp &amp; Miller</td>
<td>VASS</td>
<td>YES</td>
</tr>
<tr>
<td>1978</td>
<td>Valk</td>
<td>post self-modifying PN</td>
<td>YES</td>
</tr>
<tr>
<td>1978</td>
<td>Valk</td>
<td>self-modifying PN</td>
<td>NO</td>
</tr>
<tr>
<td>1994</td>
<td>Abdulla &amp; Jonsson</td>
<td>LCS</td>
<td>NO</td>
</tr>
<tr>
<td>1998</td>
<td>Dufourd &amp; F. &amp; Schnoebelen</td>
<td>3-dim reset/transfer VASS</td>
<td>NO</td>
</tr>
<tr>
<td>1998</td>
<td>Emerson &amp; Namjoshi</td>
<td>WSTS model checking</td>
<td>NO</td>
</tr>
<tr>
<td>1999</td>
<td>Esparza &amp; F. &amp; Mayr</td>
<td>broadcast protocols &amp; transfer PN</td>
<td>NO</td>
</tr>
<tr>
<td>2000</td>
<td>F. &amp; Sutre</td>
<td>2-dim reset/transfer VASS</td>
<td>YES</td>
</tr>
<tr>
<td>2004</td>
<td>F. &amp; McKenzie &amp; Picaronny</td>
<td>strongly increasing ω-resursive nets</td>
<td>YES</td>
</tr>
<tr>
<td>2004</td>
<td>Raskin &amp; Van Begin</td>
<td>PN+NBA</td>
<td>NO</td>
</tr>
<tr>
<td>2005</td>
<td>Goubault-Larrecq &amp; Verma</td>
<td>BVASS</td>
<td>YES</td>
</tr>
<tr>
<td>2009</td>
<td>F. &amp; Goubault-Larrecq</td>
<td>ω²-WSTS, cover-flattable</td>
<td>YES</td>
</tr>
<tr>
<td>2010</td>
<td>F. &amp; Sangnier</td>
<td>PN+0-test</td>
<td>YES</td>
</tr>
<tr>
<td>2011</td>
<td>Acciai, Boreale, Henzinger, Meyer,...</td>
<td>depth-bounded processes, ν-PN</td>
<td>NO</td>
</tr>
<tr>
<td>2011</td>
<td>Chambard &amp; F. &amp; Schmitz</td>
<td>trace-bounded ω²-WSTS</td>
<td>YES</td>
</tr>
<tr>
<td>2013</td>
<td>Geeraerts &amp; Heußner &amp; Praveen &amp; Raskin</td>
<td>ω-PN</td>
<td>YES</td>
</tr>
<tr>
<td>2013</td>
<td>Hüchting &amp; Majumdar &amp; Meyer</td>
<td>name-bounded π-calculus processes</td>
<td>YES</td>
</tr>
<tr>
<td>2016</td>
<td>Hofman &amp; Lasota &amp; Lazic &amp; Leroux &amp; ST</td>
<td>unordered PN</td>
<td>YES</td>
</tr>
</tbody>
</table>
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)
  - 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, Lazic, 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)
  - Lazic 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

Explore more in details WBTS and find applications of WBTS (like 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.

Developing the WSTS theory and a prototype for finding bugs in web services and choreographies.

Make the first efficient prototype for reachability for Petri nets.
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.
- Developing 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.
- Developing 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!