On Tools for Coverability

Michael Blondin, Christoph Haase, Grégoire Sutre
Part I: QCover

Michael Blondin

Joint work with Alain Finkel, Christoph Haase and Serge Haddad
Verifying safety with Petri nets

Process 1

Process 2

Lamport mutual exclusion "1-bit algorithm"
Verifying safety with Petri nets

Process 1

while True:
    x = True
    while y:
        pass
    # critical section
    x = False

Process 2

while True:
    y = True
    if x:
        y = False
    while x:
        pass
    goto # critical section

Lamport mutual exclusion "1-bit algorithm"
Verifying safety with Petri nets

```
while True:
    x = True
while y: pass
# critical section
x = False
```

```
while True:
    y = True
    if x then:
        y = False
    while x: pass
    goto # critical section
    y = False
```
Verifying safety with Petri nets

while True:
    x = True
while y: pass
    # critical section
    x = False

while True:
    y = True
    if x then:
        y = False
    while x: pass
goto ♠
    # critical section
    y = False
Verifying safety with Petri nets

Process 1

\[
\text{while True:} \\
x = \text{True} \\
\text{while } y: \text{ pass} \\
\# \text{ critical section} \\
x = \text{False}
\]

Process 2

\[
\text{while True:} \\
\star \quad y = \text{True} \\
\quad \text{if } x \text{ then:} \\
\quad \quad y = \text{False} \\
\quad \text{while } x: \text{ pass} \\
\quad \text{goto } \star \\
\# \text{ critical section} \\
y = \text{False}
\]
Verifying safety with Petri nets

```
while True:
    x = True
    while y: pass
    # critical section
    x = False
```

```
while True:
    y = True
    if x then:
        y = False
    while x: pass
    goto y
    # critical section
    y = False
```
Verifying safety with Petri nets

while True:
    x = True
    while y: pass
    # critical section
    x = False

while True:
    y = True
    if x then:
        y = False
    while x: pass
goto *

# critical section
y = False
while True:
    x = True
while y: pass
    # critical section
    x = False

while True:
    y = True
    if x then:
        y = False
while x: pass
    goto ★
    # critical section
    y = False
while True:
    x = True
while y: pass
    # critical section
    x = False

while True:
    y = True
if x then:
    y = False
while x: pass
    goto ⭐
    # critical section
    y = False
Verifying safety with Petri nets

```
while True:
    x = True
    while y:
        pass
    # critical section
    x = False
```

```
while True:
    y = True
    if x:
        y = False
    while x:
        pass
goto
```

Processes at both critical sections each ≥ 1
Verifying safety with Petri nets

Processes at both critical sections

Each

\[ \geq 1 \]

\[ \geq 0 \]
Verifying safety with Petri nets

Coverability problem

Processes at both critical sections

each $\geq 1$

$\geq 0$
Coverability problem

Problem

Input: Petri net $\mathcal{N}$, initial marking $m_0$, target marking $m$

Question: Is some $m' \geq m$ reachable from $m_0$ in $\mathcal{N}$?
**Problem**

<table>
<thead>
<tr>
<th>Input:</th>
<th>Petri net $\mathcal{N}$, initial marking $m_0$, target marking $m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question:</td>
<td>Is some $m' \geq m$ reachable from $m_0$ in $\mathcal{N}$?</td>
</tr>
</tbody>
</table>

**How to solve it?**

- **Forward:** build reachability tree from initial marking
- **Backward:** find predecessors of markings covering target
- **EXPSPACE-complete**
### Problem

**Input:** Petri net $\mathcal{N}$, initial marking $m_0$, target marking $m$

**Question:** Is some $m' \geq m$ reachable from $m_0$ in $\mathcal{N}$?

### How to solve it?

**Karp & Miller '69**

- **Forward:** build reachability tree from initial marking
- **Backward:** find predecessors of markings covering target
- EXPSPACE-complete
Coverability problem

**Problem**

Input: Petri net $\mathcal{N}$, initial marking $m_0$, target marking $m$

Question: Is some $m' \geq m$ reachable from $m_0$ in $\mathcal{N}$?

**How to solve it?**

Arnold & Latteux '78, Abdulla et al. '96

- **Forward**: build reachability tree from initial marking
- **Backward**: find predecessors of markings covering target
- EXPSPACE-complete
# Coverability problem

## Problem

**Input:** Petri net $\mathcal{N}$, initial marking $m_0$, target marking $m$

**Question:** Is some $m' \geq m$ reachable from $m_0$ in $\mathcal{N}$?

## How to solve it?

- **Lipton '76, Rackoff '78**

  - **Forward:** build reachability tree from initial marking
  - **Backward:** find predecessors of markings covering target
  - **EXPSPACE-complete**
### Coverability problem

#### Problem

**Input:** Petri net $\mathcal{N}$, initial marking $m_0$, target marking $m$

**Question:** Is some $m' \geq m$ reachable from $m_0$ in $\mathcal{N}$?

#### How to solve it?

- **Forward:** build reachability tree from initial marking
- **Backward:** find predecessors of markings covering target
- EXPSPACE-complete
Backward algorithm
Backward algorithm

What initial markings may cover (0, 2)?
Backward algorithm

2
Backward algorithm

Cannot cover target marking.
Backward algorithm
Backward algorithm

Cannot cover target marking.
Backward algorithm
Backward algorithm
Backward algorithm

Cannot cover target marking
Backward algorithm

Basis size may become doubly exponential

(Bozzelli & Ganty '11)
Backward algorithm

We only care about some initial marking...
Backward algorithm

We only care about some initial marking...

Speedup by pruning basis!
(Discrete) Petri nets
(Discrete) Petri nets
(Discrete) Petri nets

Continuous

\[ 2 \times 1 = 2 \times 4 = 2^n \times 4/10 \]
Petri nets
Petri nets

\[ 2 \rightarrow 1/2^n \rightarrow \]
Continuous

\[ \frac{\text{Discrete}}{\text{Petri nets}} \]
Continuity to over-approximate coverability

\( m \) is coverable from \( m_0 \)

\[ \downarrow \]

\( m \) is \( \mathbb{Q} \)-coverable from \( m_0 \)
Continuity to over-approximate coverability

\( m \) is coverable from \( m_0 \)

\( \downarrow \)

\( m \) is \( \mathbb{Q} \)-coverable from \( m_0 \)

\( \downarrow \) \( \uparrow \)

\( m_0 \) and \( m \) satisfy conditions of

Esparza, Ledesma-Garza, Majumdar, Meyer & Niksic '14

\( \text{NP} / \text{EXPTIME} \)
Continuity to over-approximate coverability

$m$ is \textbf{not} coverable from $m_0$

$m$ is \textbf{not} $\mathbb{Q}$-coverable from $m_0$
Coverability in continuous Petri nets

Fix some continuous Petri net \((P, T, \text{Pre, Post})\)

\[
m \text{ is } \mathbb{Q}\text{-coverable from } m_0 \text{ iff...}
\]

Fracq & Haddad '13

\[
m \text{ is } \mathbb{Q}\text{-coverable from } m_0 \text{ iff...}
\]

Fracq & Haddad '13
Fix some continuous Petri net \((P, T, \text{Pre}, \text{Post})\)

\[
m \text{ is } \mathbb{Q}-\text{coverable from } m_0 \text{ iff...} \quad \text{Fraca & Haddad '13}
\]

there exist \(m' \geq m\) and \(v \in \mathbb{Q}_{\geq 0}^T\) such that

- \(m' = m_0 + (\text{Post} - \text{Pre}) \cdot v\)
Fix some continuous Petri net \((P, T, \text{Pre}, \text{Post})\)

\
\begin{align*}
\text{m is } \mathbb{Q}\text{-coverable from } m_0 \text{ iff...} & \quad \text{Fraca & Haddad '13} \\
\text{there exist } m' & \geq m \quad \text{and} \quad v \in \mathbb{Q}_{\geq 0}^T \quad \text{such that} \\
\quad \cdot \quad m' & = m_0 + (\text{Post} - \text{Pre}) \cdot v \\
\quad \cdot \quad \text{some execution from } m_0 \text{ fires exactly } \{t \in T : v_t > 0\}
\end{align*}
\n
Fix some continuous Petri net \((P, T, \text{Pre}, \text{Post})\)

\[ m \text{ is } \mathbb{Q}\text{-coverable from } m_0 \text{ iff...} \] 

- \( m' \geq m \) and \( v \in \mathbb{Q}^T_{\geq 0} \) such that
  - \( m' = m_0 + (\text{Post} - \text{Pre}) \cdot v \)
  - some execution from \( m_0 \) fires exactly \( \{ t \in T : v_t > 0 \} \)
  - some execution to \( m' \) fires exactly \( \{ t \in T : v_t > 0 \} \)

Fraca & Haddad '13
Coverability in continuous Petri nets

Consider the net with transitions labeled $a$ and $b$, and a place $2$.

- $m_0 = (2, 0)$
- $m = (0, 2)$

$m$ is $\mathbb{Q}$-coverable from $m_0$ iff...

Fraca & Haddad '13

- There exist $m' \geq m$ and $v_a, v_b \in \mathbb{Q}_{\geq 0}$ such that
  - $m' = m_0 + (\text{Post} - \text{Pre}) \cdot v$
  - Some execution from $m_0$ fires exactly $\{t \in \{a, b\} : v_t > 0\}$
  - Some execution to $m'$ fires exactly $\{t \in \{a, b\} : v_t > 0\}$

Polynomial time!
Coverability in continuous Petri nets

![Petri net diagram]

\[ m_0 = (2, 0) \]
\[ m = (0, 2) \]

\[ m \text{ is } \mathbb{Q}\text{-coverable from } m_0 \text{ iff...} \]

<table>
<thead>
<tr>
<th>Fraca &amp; Haddad '13</th>
</tr>
</thead>
<tbody>
<tr>
<td>there exist ( m' \geq m ) and ( v_a, v_b \in \mathbb{Q}_{\geq 0} ) such that</td>
</tr>
<tr>
<td>- ( 0 \leq v_b + v_a \leq 2 )</td>
</tr>
<tr>
<td>- ( 2 \leq v_b )</td>
</tr>
<tr>
<td>- some execution from ( m_0 ) fires exactly ( { t \in {a, b} : v_t &gt; 0 } )</td>
</tr>
<tr>
<td>- some execution to ( m' ) fires exactly ( { t \in {a, b} : v_t &gt; 0 } )</td>
</tr>
</tbody>
</table>
Coverability in continuous Petri nets

\[ m_0 = (2, 0) \]
\[ m = (0, 2) \]

\[ m \text{ is } \mathbb{Q}-\text{coverable from } m_0 \text{ iff...} \]

Fraca & Haddad '13

there exist \( m' \geq m \) and \( v_a, v_b \in \mathbb{Q}_{\geq 0} \) such that

1. \( 0 \leq v_b + v_a \leq 2 \) \( \implies v_a = 0, v_b = 2, m' = m \)
2. \( 2 \leq v_b \)

- some execution from \( m_0 \) fires exactly \( \{ t \in \{a, b\} : v_t > 0 \} \)
- some execution to \( m' \) fires exactly \( \{ t \in \{a, b\} : v_t > 0 \} \)
Coverability in continuous Petri nets

\[
m_0 = (2, 0) \\
m = (0, 2)
\]

\[\begin{array}{l}
\text{there exist } m' \geq m \text{ and } v_a, v_b \in \mathbb{Q}_{\geq 0} \text{ such that} \\
\quad 0 \leq v_b + v_a \leq 2 \quad \implies \quad v_a = 0, \ v_b = 2, \ m' = m \\
\quad 2 \leq v_b \\
\quad \text{some execution from } m_0 \text{ fires exactly } \{t \in \{a, b\} : v_t > 0\} \\
\quad \text{some execution to } m' \text{ fires exactly } \{t \in \{a, b\} : v_t > 0\}
\end{array}\]

Fraca & Haddad '13

Straightforward

More subtle

Even better approximation

Polynomial time!
Coverability in continuous Petri nets

\[ m_0 = (2, 0) \]
\[ m = (0, 2) \]

**m is \( \mathbb{Q} \)-coverable from \( m_0 \) iff...**

Fraca & Haddad '13

there exist \( m' \geq m \) and \( v_a, v_b \in \mathbb{Q}_{\geq 0} \) such that

- \( 0 \leq v_b + v_a \leq 2 \implies v_a = 0, v_b = 2, m' = m \)
- \( 2 \leq v_b \)
- some execution from \( m_0 \) fires exactly \( \{ t \in \{a, b\} : v_t > 0 \} \)
- some execution to \( m' \) fires exactly \( \{ t \in \{a, b\} : v_t > 0 \} \)
Coverability in continuous Petri nets

$m$ is $\mathbb{Q}$-coverable from $m_0$ iff...

there exist $m' \geq m$ and $v_a, v_b \in \mathbb{Q}_{\geq 0}$ such that

- $0 \leq v_b + v_a \leq 2 \quad \implies \quad v_a = 0, \; v_b = 2, \; m' = m$
- $2 \leq v_b$

• some execution from $m_0$ fires exactly $\{b\}$
• some execution to $m'$ fires exactly $\{b\}$

$m_0 = (2, 0)$
$m = (0, 2)$
Coverability in continuous Petri nets

$m_0 = (2, 0)$
$m = (0, 2)$

$m$ is $\mathbb{Q}$-coverable from $m_0$ iff...

Fraca & Haddad '13

there exist $m' \geq m$ and $v_a, v_b \in \mathbb{Q}_{\geq 0}$ such that

- $0 \leq v_b + v_a \leq 2 \implies v_a = 0, v_b = 2, m' = m$  
- $2 \leq v_b$

- some execution from $m_0$ fires exactly \{b\}
- some execution to $m'$ fires exactly \{b\}
Coverability in continuous Petri nets

$m$ is $\mathbb{Q}$-coverable from $m_0$ iff...

there exist $m' \geq m$ and $v_a, v_b \in \mathbb{Q}_{\geq 0}$ such that

- $0 \leq v_b + v_a \leq 2 \implies v_a = 0, v_b = 2, m' = m$
- $2 \leq v_b$
- some execution from $m_0$ fires exactly $\{b\}$
- some execution to $m'$ fires exactly $\{b\}$
**Coverability in continuous Petri nets**

$m$ is $\mathbb{Q}$-coverable from $m_0$ iff...

- there exist $m' \geq m$ and $v_a, v_b \in \mathbb{Q}_{\geq 0}$ such that
  - $0 \leq v_b + v_a \leq 2 \implies v_a = 0, v_b = 2, m' = m$
  - $2 \leq v_b$
- some execution from $m_0$ fires exactly $\{b\}$
- some execution to $m'$ fires exactly $\{b\}$

$m_0 = (2, 0)$

$m = (0, 2)$

Fracca & Haddad '13

$m$ is $\mathbb{Q}$-coverable from $m_0$ iff...

\[ m_0 = (2, 0) \]

\[ m = (0, 2) \]
Coverability in continuous Petri nets

$m_0 = (2, 0)$
$m = (0, 2)$

$m$ is $\mathbb{Q}$-coverable from $m_0$ iff...

Fraca & Haddad '13

there exist $m' \geq m$ and $v_a, v_b \in \mathbb{Q}_{\geq 0}$ such that

1. $0 \leq v_b + v_a \leq 2 \implies v_a = 0, v_b = 2, m' = m$

2. $2 \leq v_b$

- some execution from $m_0$ fires exactly $\{b\}$

- some execution to $m'$ fires exactly $\{b\}$
Coverability in continuous Petri nets

There exist \( m' \geq m \) and \( v_a, v_b \in \mathbb{Q}_{\geq 0} \) such that:

- \( 0 \leq v_b + v_a \leq 2 \) \( \implies v_a = 0, v_b = 2, m' = m \)
- \( 2 \leq v_b \)
- Some execution from \( m_0 \) fires exactly \( \{b\} \)
- Some execution to \( m' \) fires exactly \( \{b\} \)

\( m_0 = (2, 0) \)
\( m = (0, 2) \)
## Coverability in continuous Petri nets

### Diagram
![Diagram of two continuous Petri nets connected by a transition labeled 2. The places are labeled a and b.]

### Text

**m is $\mathbb{Q}$-coverable from $m_0$ iff...**

<table>
<thead>
<tr>
<th>Condition</th>
<th>Description</th>
</tr>
</thead>
</table>
| $m' \geq m$ and $v_a, v_b \in \mathbb{Q}_{\geq 0}$ | such that
| $0 \leq v_b + v_a \leq 2$ | $\implies$ $v_a = 0, v_b = 2, m' = m$ ✓
| $2 \leq v_b$ | ✓
| some execution from $m_0$ fires exactly $\{b\}$ | ✓
| some execution to $m'$ fires exactly $\{b\}$ | x

**Fraca & Haddad '13**

$m_0 = (2, 0)$

$m = (0, 2)$

**Not $\mathbb{Q}$-coverable from**
$m$ is $\mathbb{Q}$-coverable from $m_0$ iff...

Fraca & Haddad '13

there exist $m' \geq m$ and $v \in \mathbb{Q}_{\geq 0}^T$ such that

- $m' = m_0 + (\text{Post} - \text{Pre}) \cdot v$

- some execution from $m_0$ fires exactly $\{t \in T : v_t > 0\}$

- some execution to $m'$ fires exactly $\{t \in T : v_t > 0\}$
Logical characterization

$\mathbb{Q}$-coverability can be encoded in a linear size formula of existential $\text{FO}(\mathbb{Q}_{\geq 0}, +, <)$

$m$ is $\mathbb{Q}$-coverable from $m_0$ iff...

there exist $m' \geq m$ and $v \in \mathbb{Q}_{\geq 0}$ such that

- $m' = m_0 + (\text{Post} - \text{Pre}) \cdot v$
- some execution from $m_0$ fires exactly $\{t \in T : v_t > 0\}$
- some execution to $m'$ fires exactly $\{t \in T : v_t > 0\}$
\textbf{Logical characterization} \hspace{1cm} \textbf{TACAS'16}

$\mathbb{Q}$-coverability can be encoded in a linear size formula of

existential $\text{FO}(\mathbb{N}, +, <)$

\textbf{m is $\mathbb{Q}$-coverable from $m_0$ iff...} \hspace{1cm} \textbf{Fraca & Haddad '13}

there exist $m' \geq m$ and $v \in \mathbb{Q}_{\geq 0}^T$ such that

- $m' = m_0 + (\text{Post} - \text{Pre}) \cdot v$
- some execution from $m_0$ fires exactly $\{t \in T : v_t > 0\}$
- some execution to $m'$ fires exactly $\{t \in T : v_t > 0\}$

\text{Even better approximation}
### Logical characterization

Q-coverability can be encoded in a linear size formula of existential FO(\(\mathbb{Q}_{\geq 0}, +, <\))

<table>
<thead>
<tr>
<th>Logical characterization</th>
<th>TACAS'16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q-coverability can be encoded in a linear size formula of existential FO((\mathbb{Q}_{\geq 0}, +, &lt;))</td>
<td></td>
</tr>
</tbody>
</table>

### \(m\) is Q-coverable from \(m_0\) iff...

Fraca & Haddad '13

<table>
<thead>
<tr>
<th>(m) is Q-coverable from (m_0) iff...</th>
<th>Fraca &amp; Haddad '13</th>
</tr>
</thead>
<tbody>
<tr>
<td>there exist (m' \geq m) and (v \in \mathbb{Q}_{\geq 0}^T) such that</td>
<td></td>
</tr>
<tr>
<td>• (m' = m_0 + (\text{Post} - \text{Pre}) \cdot v)</td>
<td>Straightforward</td>
</tr>
<tr>
<td>• some execution from (m_0) fires exactly ({t \in T : v_t &gt; 0})</td>
<td></td>
</tr>
<tr>
<td>• some execution to (m') fires exactly ({t \in T : v_t &gt; 0})</td>
<td></td>
</tr>
</tbody>
</table>
Coverability in continuous Petri nets

<table>
<thead>
<tr>
<th>Logical characterization</th>
<th>TACAS'16</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathbb{Q} )-coverability can be encoded in a linear size formula of existential FO(( \mathbb{Q}_{\geq 0}, +, &lt; ))</td>
<td></td>
</tr>
</tbody>
</table>

\[ m \text{ is } \mathbb{Q}-\text{coverable from } m_0 \text{ iff...} \]

\[ \text{there exist } m' \geq m \text{ and } v \in \mathbb{Q}_{\geq 0}^T \text{ such that} \]

- \[ m' = m_0 + (\text{Post} - \text{Pre}) \cdot v \]

- \[ \text{some execution from } m_0 \text{ fires exactly } \{ t \in T : v_t > 0 \} \]

- \[ \text{some execution to } m' \text{ fires exactly } \{ t \in T : v_t > 0 \} \]

F cara & Haddad '13

More subtle

Polynomial time!
Encoding the firing set conditions

Testing whether some transitions can be fired from initial marking
Encoding the firing set conditions

Testing whether some transitions can be fired from initial marking
Encoding the firing set conditions

Testing whether some transitions can be fired from initial marking
Simulate a "breadth-first" transitions firing
Simulate a "breadth-first" transitions firing by numbering places/transitions

(Verma, Seidl & Schwentick '05)
Encoding the firing set conditions

Simulate a "breadth-first" transitions firing by numbering places/ transitions
(Verma, Seidl & Schwentick '05)
Simulate a "breadth-first" transitions firing by numbering places/ transitions
(Verma, Seidl & Schwentick '05)
Simulate a "breadth-first" transitions firing by numbering places/transitions (Verma, Seidl & Schwentick '05)
Simulate a "breadth-first" transitions firing by numbering places/transitions
(Verma, Seidl & Schwentick '05)
Simulate a "breadth-first" transitions firing by numbering places/ transitions 
(Verma, Seidl & Schwentick '05)
Simulate a "breadth-first" transitions firing by numbering places/.transitions
(Verma, Seidl & Schwentick '05)
Simulate a "breadth-first" transitions firing by numbering places/ transitions (Verma, Seidl & Schwentick '05)
Simulate a "breadth-first" transitions firing by numbering places/transitions

(Verma, Seidl & Schwentick '05)
Encoding the firing set conditions

Simulate a "breadth-first" transitions firing by numbering places/transitions
(Verma, Seidl & Schwentick '05)
Simulate a "breadth-first" transitions firing by numbering places/transitions
(Verma, Seidl & Schwentick '05)
Encoding the firing set conditions

\[ \varphi(x) = \exists y : \bigwedge_{p \in P} y(p) > 0 \rightarrow \bigwedge_{t \in \cdot p} y(t) < y(p) \cdots \]
if target marking $m$ is not $\mathbb{Q}$-coverable:
    return False

Polynomial time
if target marking $m$ is not $\mathbb{Q}$-coverable:
    return False

$X = \{\text{target marking } m\}$

while (initial marking $m_0$ not covered by $X$):
    $B = \text{markings obtained from } X \text{ one step backward}$
    $B = B \setminus \{b \in B : \neg \varphi(b)\}$
    if $B = \emptyset$: return False
    $\varphi(x) = \varphi(x) \land \bigwedge_{\text{pruned } b} x \not\geq b$
    $X = X \cup B$

return True
if target marking \( m \) is not \( Q \)-coverable:

return False

\( X = \{ \text{target marking } m \} \)

while (initial marking \( m_0 \) not covered by \( X \)):

\( B = \) markings obtained from \( X \) one step backward

\( B = B \setminus \{ b \in B : \neg \varphi(b) \} \)

if \( B = \emptyset \):
    return False

\( \varphi(x) = \varphi(x) \land \bigwedge_{\text{pruned } b} x \not\equiv b \)

\( X = X \cup B \)

return True
if target marking $m$ is not $\mathbb{Q}$-coverable:
    return False

$X = \{\text{target marking } m\}$

while (initial marking $m_0$ not covered by $X$):
    $B = \text{markings obtained from } X \text{ one step backward}$
    $B = B \setminus \{b \in B : \neg \varphi(b)\}$
    if $B = \emptyset$: return False
    $\varphi(x) = \varphi(x) \land \bigwedge_{\text{pruned}} b \not\leq b$
    $X = X \cup B$

return True
if target marking $m$ is not $Q$-coverable:
    return False

$X = \{\text{target marking } m\}$

while (initial marking $m_0$ not covered by $X$):
    $B = \text{markings obtained from } X \text{ one step backward}$
    $B = B \setminus \{b \in B : \neg \varphi(b)\}$
    if $B = \emptyset$: return False
    $\varphi(x) = \varphi(x) \land \bigwedge_{\text{pruned}} b \ x \not\supseteq b$
    $X = X \cup B$

return True
if target marking $m$ is not $\mathcal{Q}$-coverable:
    return False

$X = \{\text{target marking } m\}$

while (initial marking $m_0$ not covered by $X$):
    $B = \text{markings obtained from } X \text{ one step backward}$
    $B = B \setminus \{b \in B : \neg \varphi(b)\}$
    Q-coverability pruning
    if $B = \emptyset$: return False
    (better than poly. time)
    $\varphi(x) = \varphi(x) \land \bigwedge_{\text{pruned } b} x \not\leq b$
    $X = X \cup B$

return True
if target marking $m$ is not $\mathbb{Q}$-coverable:
    return False

$X = \{\text{target marking } m\}$

while (initial marking $m_0$ not covered by $X$):
    $B = \text{markings obtained from } X \text{ one step backward}$
    $B = B \setminus \{b \in B : \neg \varphi(b)\}$
    if $B = \emptyset$: return False

$\varphi(x) = \varphi(x) \land \bigwedge_{\text{pruned }} b \ x \not\succ b$

$X = X \cup B$

return True
if target marking $m$ is not $\mathbb{Q}$-coverable:
    return False

$X = \{\text{target marking } m\}$

while (initial marking $m_0$ not covered by $X$):
    $B = \text{markings obtained from } X \text{ one step backward}$
    
    $B = B \setminus \{b \in B : \neg \varphi(b)\}$

    if $B = \emptyset$: return False

    $\varphi(x) = \varphi(x) \land \bigwedge_{\text{pruned } b} x \not\succeq b$

    $X = X \cup B$

return True
if target marking $m$ is not $Q$-coverable:
    return False

$X = \{\text{target marking } m\}$

while (initial marking $m_0$ not covered by $X$):
    $B = \text{markings obtained from } X \text{ one step backward}$
    $B = B \setminus \{b \in B : \neg \varphi(b)\}$
    if $B = \emptyset$: return False
    $\varphi(x) = \varphi(x) \land \bigwedge_{\text{pruned } b} x \not\geq b$
    $X = X \cup B$

return True
if target marking \( m \) is not \( Q \)-coverable:
   return False

\( X = \{ \text{target marking } m \} \)

while (initial marking \( m_0 \) not covered by \( X \)):
   \( B = \) markings obtained from \( X \) one step backward
   \( B = B \setminus \{ b \in B : \neg \varphi(b) \} \)
   if \( B = \emptyset \): return False
   \( \varphi(x) = \varphi(x) \land \bigwedge_{\text{pruned}} b \not\preceq b \)
   \( X = X \cup B \)

return True
An implementation: QCover

+ SMT solver Z3 (Microsoft Research)

https://github.com/blondimi/qcover

Tested on...

• 176 Petri nets (avg. 1054 places, 8458 transitions)
• C/Erlang programs with threads
• Mutual exclusion protocols, communication protocols, etc.
• Message analysis of a medical and a bug tracking system
An implementation: QCover

Instances proven safe

- QCover
- Petrinizer
- BFC
- MIST

Running time in seconds

- 105/115
- 95
- 63
- 35

Largest nets proved safe:
- 21143 places
- 7150 trans.
- 42 secs.

- 6690 places
- 11934 trans.
- 21 secs.

- 754 places
- 27370 trans.
- 3 secs.

9/10
An implementation: QCover

Instances proven safe:

- QCover
- Petrinizer
- BFC
- MIST

Largest nets proved safe:

- 2,1143 places, 7,150 transitions, 42 seconds
- 6,690 places, 11,934 transitions, 21 seconds
- 754 places, 27,370 transitions, 3 seconds
An implementation: QCover

Instances proven safe

Instances proven safe or unsafe

QCover
Petrinizer
bfc
mist

Largest nets proved safe:
21143 places
7150 trans.
42 secs.
6690 places
11934 trans.
21 secs.
754 places
27370 trans.
3 secs.

9/10
An implementation: QCover

Markings pruning efficiency across all iterations

Largest nets proved safe:
- 21143 places, 7150 trans., 42 secs.
- 6690 places, 11934 trans., 21 secs.
- 754 places, 27370 trans., 3 secs.

9/10
Possible extensions

• Combine our approach with a forward algorithm to better handle unsafe instances
Possible extensions

- Combine our approach with a forward algorithm to better handle unsafe instances.
Possible extensions

- Combine our approach with a forward algorithm to better handle unsafe instances

- **Use more efficient data structures, e.g. sharing trees**
  
  (Delzanno, Raskin & Van Begin '04)
• Combine our approach with a forward algorithm to better handle unsafe instances

• Use more efficient data structures, e.g. sharing trees
  
  (Delzanno, Raskin & Van Begin '04)

• Support Petri nets extensions
Part II: ICover

Grégoire Sutre

Joint work with Thomas Geffroy and Jérôme Leroux
Verifying Systems with Petri Nets

C code
Property

Erlang code
Property

...
Verifying Systems with Petri Nets

- C code
- Erlang code
- ...
Coverability in Petri nets

$$\text{init} \xrightarrow{*} m \geq \text{target} ?$$

**Decidability - Complexity**

- Decidable (Karp and Miller - 1969)
- \text{EXPSPACE}-complete (Lipton - 1976, Rackoff - 1978)
Coverability in Petri nets

\[ \text{init} \rightarrow^* m \geq \text{target} ? \]

Tools

Mist (Ganty, Geeraerts, Raskin, Van Begin, ...)
- interval sharing trees
- backward search + place invariants
- abstraction refinement

BFC (Kaiser, Kroening, Wahl)
Target set widening + forward Karp-Miller

Petrinizer (Esparza, Ledesma-Garza, Majumdar, Meyer, Niksic)
SMT, state equation + traps

QCover (Blondin, Finkel, Haase, Haddad)
SMT, continuous reachability + backward search
ICover: Generalisation of QCover with Invariants

Assumption:

1. $I$ is an invariant ($I$ contains all reachable markings)
2. $I$ is a downward closed set

$U_0 := \uparrow(target \cap I)$
ICover: Generalisation of QCover with Invariants

Assumption:

1. $I$ is an invariant ($I$ contains all reachable markings)
2. $I$ is a downward closed set

$U_0 := \emptyset$ : Safe !
Assumption:

1. $I$ is an invariant ($I$ contains all reachable markings)
2. $I$ is a downward closed set

$U_0 := \uparrow(target \cap I)$

$U_1 := U_0 \cup \uparrow(pre(U_0) \cap I)$
**ICover: Generalisation of QCover with Invariants**

**Assumption:**

1. $I$ is an invariant ($I$ contains all reachable markings)
2. $I$ is a downward closed set

\[
U_0 := \uparrow(target \cap I)
\]

\[
U_1 := U_0 \cup \uparrow(pre(U_0) \cap I)
\]

\[
U_2 := U_1 \cup \uparrow(pre(U_1) \cap I)
\]

Always terminates (Dickson’s lemma)
ICover: Generalisation of QCover with Invariants

Assumption:
1. \( I \) is an invariant (\( I \) contains all reachable markings)
2. \( I \) is a downward closed set

\[
U_0 := \uparrow (\text{target} \cap I)
\]

\[
U_1 := U_0 \cup \uparrow (\text{pre}(U_0) \cap I)
\]

\[
U_2 := U_1 \cup \uparrow (\text{pre}(U_1) \cap I)
\]

\[
\ldots
\]

\[
U_{k+1} := U_k \cup \uparrow (\text{pre}(U_k) \cap I)
\]

Always terminates
(Dickson’s lemma)
if target ∈ I then
    \[ B \leftarrow \{ target \} \];
else
    return False;
end

while \( m_{\text{init}} \notin \uparrow B \) do
    \( N \leftarrow \min(\text{pre}(\uparrow B)) \setminus \uparrow B \)
    \( P \leftarrow N \cap I \)
    if \( P = \emptyset \) then
        return False;
    end
    \( B \leftarrow \min(B \cup P) \);
end
return True;

- \( I \) is an invariant
- \( I \) is a downward closed set
Invariant: Sign Analysis

\[ p_a \geq 0 \]

\[ p_a = 0 \land p_r = 0 \land p_s = 0 \]
Invariant: Sign Analysis

\( pq = 0 \)  
\( \geq 0 \)  
\( = 0 \)

Can’t be fired

Fireable
Invariant: Sign Analysis

\[ p_q = 0 \] and \[ p_r \geq 0 \] can’t be fired

\[ p_s = 0 \] fireable
Invariant: Sign Analysis

\[ pq = 0 \land pr = 0 \land ps = 0 \]

can’t be fired

fireable
Invariant: Sign Analysis

Invariant: $p_q = 0 \land p_r = 0 \land p_s = 0$
Invariant: State Inequation

\[ \begin{align*}
\mathbf{x}_i &\geq 0 \\
m(p_1) &\leq 1 - x_1 \\
m(p_2) &\leq x_1 - x_2 + 2x_3 \\
m(p_3) &\leq 2x_2 - x_3
\end{align*} \]

\[
p_1 \xrightarrow{t_1} p_2 \xrightarrow{t_2} p_3 \xrightarrow{t_3} \ldots \xrightarrow{t_2} p_3 \xrightarrow{t_3} r
\]

\[
\Delta(t_1) = p_2 - p_1 \\
\Delta(t_2) = 2p_3 - p_2 \\
\Delta(t_3) = 2p_2 - p_3
\]
Invariant: State Inequation

\[
\begin{align*}
p_1 & \xrightarrow{t_1} p_2 \\
p_2 & \xrightarrow{t_2} p_3 \\
p_3 & \xrightarrow{t_3} p_1
\end{align*}
\]

\[
p_1 \xrightarrow{t_1} \xrightarrow{t_2} \xrightarrow{t_3} \ldots \xrightarrow{t_2} \xrightarrow{t_3} r
\]

\[
\Delta(t_1) = p_2 - p_1 \\
\Delta(t_2) = 2p_3 - p_2 \\
\Delta(t_3) = 2p_2 - p_3
\]

\[x_i: \text{number of occurrences of } t_i\]

\[r = \text{init} + x_1 \Delta(t_1) + x_2 \Delta(t_2) + x_3 \Delta(t_3)\]
Invariant: State Inequation

$p_1 \rightarrow t_1 \rightarrow p_2 \rightarrow t_2 \rightarrow p_3 \rightarrow t_3 \rightarrow \ldots \rightarrow t_2 \rightarrow t_3 \rightarrow r \geq m$

$\Delta(t_1) = p_2 - p_1$
$\Delta(t_2) = 2p_3 - p_2$
$\Delta(t_3) = 2p_2 - p_3$

$x_i$: number of occurrences of $t_i$

$m \leq r = init + x_1\Delta(t_1) + x_2\Delta(t_2) + x_3\Delta(t_3)$
Invariant: State Inequation

\[
\begin{align*}
\Delta(t_1) &= p_2 - p_1 \\
\Delta(t_2) &= 2p_3 - p_2 \\
\Delta(t_3) &= 2p_2 - p_3 \\
x_i &\text{: number of occurrences of } t_i \\
m &\leq r = init + x_1\Delta(t_1) + x_2\Delta(t_2) + x_3\Delta(t_3)
\end{align*}
\]

Invariant

\[
I := \{m \mid \exists x, init + \sum_{t \in T} x(t)\Delta(t) \geq m\}
\]
Invariant: State Inequation

\[\begin{align*}
\Delta(t_1) &= p_2 - p_1 \\
\Delta(t_2) &= 2p_3 - p_2 \\
\Delta(t_3) &= 2p_2 - p_3 \\
x_i: \text{ number of occurrences of } t_i
\end{align*}\]

\[m \leq r = \text{init} + x_1 \Delta(t_1) + x_2 \Delta(t_2) + x_3 \Delta(t_3)\]

\[\left\{ \begin{array}{l}
x_1, x_2, x_3 \geq 0 \\
m(p_1) \leq 1 - x_1 \\
m(p_2) \leq x_1 - x_2 + 2x_3 \\
m(p_3) \leq 2x_2 - x_3
\end{array} \right.\]
Invariant: State Inequation

\[ p_1 \leq 1 \]

[Diagram with places and transitions labeled: \( p_1 \rightarrow t_1 \rightarrow p_2 \rightarrow t_2 \rightarrow p_3 \rightarrow t_3 \rightarrow \ldots \rightarrow t_2 \rightarrow t_3 \rightarrow r \geq m \)]

\[ \Delta(t_1) = p_2 - p_1 \]
\[ \Delta(t_2) = 2p_3 - p_2 \]
\[ \Delta(t_3) = 2p_2 - p_3 \]

\[ x_i: \text{number of occurrences of } t_i \]
\[ m \leq r = \text{init} + x_1 \Delta(t_1) + x_2 \Delta(t_2) + x_3 \Delta(t_3) \]

Invariant

\[ I := \{ m \mid \exists x, \text{init} + \sum_{t \in T} x(t) \Delta(t) \geq m \} \]
Experimentations

New Tool: $ICover$

- Based on $QCover$ written in Python (~900 lines of codes)
- Both use the SMT-Solver $z3$ (Bjorner et al. - 2007)
- $ICover$ available as a patch of $QCover$ (~400 lines of codes)
- dept-info.labri.u-bordeaux.fr/~tgeffroy/icover

Results

- Benchmarks (176 instances) used by $QCover$ and others
- $QCover$ solved 106 / 115 safe instances (2000 seconds per instance)
- $QCover$ solved 37 / 61 unsafe instances (idem)
- $ICover$ solved as much safe instances and one more unsafe
- It works ! 10 000 seconds ($QCover$) to 5 000 seconds ($ICover$)
Experimentations: Sign Analysis As A Pre-processing

\[ t_1 \rightarrow t_2 \rightarrow t_3 \rightarrow t_4 \rightarrow t_5 \rightarrow t_7 \rightarrow t_6 \]

\[ t_3 \rightarrow t_4 \]

\[ t_4 \rightarrow t_3 \]

\[ t_2 \rightarrow t_5 \]

\[ t_1 \rightarrow t_7 \]

\[ t_7 \rightarrow t_6 \]

\[ t_6 \rightarrow t_1 \]

\[ t_5 \rightarrow t_2 \]
Experimentations: Sign Analysis As A Pre-processing

![Diagram]

- $t_1$, $t_2$, $t_3$, $t_4$, $t_5$, $t_6$, $t_7$
- $t_1$ to $t_2$
- $t_2$ to $t_3$
- $t_3$ to $t_4$
- $t_4$ to $t_5$
- $t_5$ to $t_6$
- $t_6$ to $t_7$
- $t_7$ to $t_1$

Conditions:
- $t_4 = 0$
- $t_4 \geq 0$

- Can't be fired
- Fireable

Note: The diagram illustrates the process and conditions for sign analysis pre-processing.
Experimentations: Sign Analysis As A Pre-processing

\[ t_1 \rightarrow t_2 \rightarrow t_7 \rightarrow t_6 \]

- \( t_1 \): Fireable
- \( t_7 \): Can't be fired
- \( t_2 \): Can't be fired
- \( t_6 \): Fireable

\[ t = 0 \geq 0 \]
Experimentations: Sign Analysis As A Pre-processing

- $t_1$  
- $t_2$  
- $t_7$  
- $t_6$

- $= 0$  
- $\geq 0$

- can't be fired
- fireable
Experimentations: Sign Analysis As A Pre-processing

![Diagram]

- $t_1$ to $t_2$
- $t_7$ to $t_6$
- $t_6$ to $t_7$
- $t_1$ to $t_6$

$t_7 \geq 0$ cannot be fired.
Results of Pre-processing

% of places left

% of transitions left

places in the original Petri net

transitions in the original Petri net
Experimental results: Pruning with State Inequation vs $\succ$.

- Time for $Pre + QCover$ (s)
- Efficiency
- % also pruned in $ICover$
- # markings pruned in $QCover$
State Inequation More Precise with Pre-Processing

- Can’t cover $p_1 + p_2 + p_3$ from $p_1$
- State inequation: $p_1 \leq 1$ not precise enough
- State inequation: $p_1 + p_2 + p_3 \leq 1$ precise enough
State Inequation More Precise with Pre-Processing

- Can’t cover $p_1 + p_2 + p_3$ from $p_1$
- State inequation: $p_1 \leq 1$ not precise enough
- State inequation: $p_1 + p_2 + p_3 \leq 1$ precise enough
Theorem (Recalde, Teruel and Silva - 1999)

In a pre-processed Petri net, \( m \) satisfies the state inequation iff there exists \( m' \geq m \) and a sequence \( m_0, m_1, \ldots \) such that \( \text{init} \xrightarrow{-\*} m_k \) for every \( k \) and such that \( m_0, m_1, \ldots \) converges toward \( m' \).

\[ p_1 \leq 1 \]

- \( p_1 + p_2 \) not coverable from \( p_1 \)
- \( p_1 + p_2 \) satisfy the state inequation: \( p_1 \leq 1 \)
State Inequation vs \( \rightarrow \rightarrow \)

\[
\begin{align*}
& p_1 \\
& 1 - \varepsilon \rightarrow t_1 \rightarrow \varepsilon \rightarrow t_2 \\
& p_2
\end{align*}
\]

- \( p_1 + p_2 \) not coverable from \( p_1 \) with \( \rightarrow \rightarrow \)
- \( p_1 + p_2 \) satisfy the state inequation: \( p_1 \leq 1 \)

Theorem (Recalde, Teruel and Silva - 1999)

In a pre-processed Petri net, \( m \) satisfies the state inequation iff there exists \( m' \geq m \) and a sequence \( m_0, m_1, ... \) such that init \( \rightarrow \rightarrow m_k \) for every \( k \) and such that \( m_0, m_1, ... \) converges toward \( m' \).
State Inequation vs \(\rightarrow\)

- \(p_1 + p_2\) not coverable from \(p_1\) with \(\rightarrow\)
- \(p_1 + p_2\) satisfy the state inequation: \(p_1 \leq 1\)

Theorem (Recalde, Teruel and Silva - 1999)

In a pre-processed Petri net, \(m\) satisfies the state inequation iff there exists \(m' \geq m\) and a sequence \(m_0, m_1, \ldots\) such that \(\text{init} \rightarrow m_k\) for every \(k\) and such that \(m_0, m_1, \ldots\) converges toward \(m'\).
State Inequation vs $\rightarrow$

\[
p_1 \leq 1 - \varepsilon \quad \text{and} \quad p_2 \geq \varepsilon
\]

- $p_1 + p_2$ not coverable from $p_1$
- $p_1 + p_2$ satisfy the state inequation: $p_1 \leq 1$

Theorem (Recalde, Teruel and Silva - 1999)

In a pre-processed Petri net, $m$ satisfies the state inequation iff there exists $m' \geq m$ and a sequence $m_0, m_1, \ldots$ such that $\text{init} \rightarrow^* m_k$ for every $k$ and such that $m_0, m_1, \ldots$ converges toward $m'$. 

State Inequation vs \( \rightarrow \rightarrow \)

- \( p_1 + p_2 \) not coverable from \( p_1 \) with \( \rightarrow \rightarrow \)
- \( p_1 + p_2 \) satisfy the state inequation: \( p_1 \leq 1 \)

**Theorem (Recalde, Teruel and Silva - 1999)**

In a pre-processed Petri net, \( m \) satisfies the state inequation iff there exists \( m' \geq m \) and a sequence \( m_0, m_1, \ldots \) such that \( \text{init} \rightarrow m_k \) for every \( k \) and such that \( m_0, m_1, \ldots \) converges toward \( m' \).
**Theorem (Recalde, Teruel and Silva - 1999)**

In a pre-processed Petri net, $m$ satisfies the state inequation iff there exists $m' \geq m$ and a sequence $m_0, m_1, ...$ such that $\text{init} \rightarrow^* m_k$ for every $k$ and such that $m_0, m_1, ...$ converges toward $m'$.
New

- Backward coverability algorithm with invariant-based pruning
- Pre-processing is a cheap way to accelerate verification
- In practice, in a pre-processed Petri net, state inequation is almost as good as $\rightarrow$ coverability

Future work

- Find other cheap pre-processings and invariants
- Apply to other classes of well-structured transition systems
Part III: Best practices

Christoph Haase
General remarks

Tools...

• increase visibility outside your peer group
• help understanding what is relevant to other people
• generate feedback for theoretical work
• can convince reviewers
• attract students
Before you start

• Choice of language
  • interpreted vs. compiled
  • statically vs. dynamically typed

• Bindings for SMT solver

• Performance of memory operations
Software engineering aspects

• Object oriented programming
• Unit tests
• Documentation
• Use profilers to find bottlenecks
Benchmarks

• One of the most important aspects

• Use other people's benchmarks

• Contact authors if necessary

• Pitfalls:
  • Parsing can entail large costs
  • Avoid unfair treatment of competitors
  • Choose evaluation metrics wisely
Availability

• Obtain institutional clearance $\in F_\omega$

• Choose license: BSD preferred by industry

• Use public code repositories, e.g. GitHub
Future

- Identify relevant Petri net subclasses and extensions, e.g.
  - business processes
  - process mining
  - population protocols
  - thread transition systems

- Submit to and integrate into existing software competitions
The SMT solver is always faster than you!
Thank you! Diolch!