Abstraction-based Incremental Inductive Coverability for Petri nets

Jiawen Kang        YunJun Bai        Li Jiao

State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China

June 2021
Abstraction

- Check the coverability problem of Petri nets

- Combine IC3 with place-merge abstraction (IC3+PMA)
Outline

① Preliminaries

② IC3 algorithm for PN

③ Place-merge abstraction (PMA)

④ IC3+PMA algorithm

⑤ Experiments
Definition

A Petri net is a tuple $N = (P, T, W, m_0)$ where:
- $P$ is a finite set of places
- $T$ is a finite set of transitions such that $P \cap T = \emptyset$
- $W$ is an arc function: $(P \times T) \cup (T \times P) \to \mathbb{N}$ describing the relationship between places and transitions
- $m_0$ is the initial marking. A marking $m \in \mathbb{N}^{|P|}$ is a vector specifying a number $m(p)$ of tokens for each place $p \in P$.

For vector $m_1, m_2 \in \mathbb{N}^{|P|}$

$m_1 \preceq m_2$ iff for every $p \in P: m_1(p) \leq m_2(p)$
Definition

Let $N$ be a Petri net.
- $pre(m) = \{m' | \exists t \in T: m' \rightarrow m\}$
- $Reach_i$ contains all reachable markings from $m_0$ within $i$ steps.
- $Reach = \bigcup_{i \geq 0} Reach_i$ contains all reachable markings from $m_0$. 
Coverability problem

Let $N$ be a Petri net, $m_t$ the target marking.
- The coverability problem is to prove whether there exists a reachable marking $m_r \in \text{Reach}$ such that $m_t \preceq m_r$.  

Coverability problem

Let $N$ be a Petri net, $m_t$ the target marking.
- The coverability problem is to prove whether there exists a reachable marking $m_r \in \text{Reach}$ such that $m_t \preceq m_r$.
- The coverable set of $N$ within $i$ steps is $\text{Cover}_i = \text{Reach}_i$
- The coverable set of $N$ is $\text{Cover} = \text{Reach}$
IC3 is a state-of-art of model checking

Efficient implementation of IC3 to check the coverability problem of Petri nets without using SMT solvers
IC3 algorithm for Petri nets

IC3 maintains a sequence $F_0, F_1 \ldots F_k$

where $F_i$ is a downward-closed set called frame that over-approximates the coverable set within $i$ steps.

The algorithm generally proceeds by alternating two phases: the blocking phase and the propagation phase.
IC3 algorithm for Petri nets

Blocking phase: $block(a, i)$
IC3 algorithm for Petri nets

Blocking phase: $\text{block}(a, i)$

try to prove $a^\uparrow$ is unreachable within $i$ steps
IC3 algorithm for Petri nets

Blocking phase: $\text{block}(a, i)$

$$F_0 = m_0^\downarrow$$

$$F_1$$

$$\ldots$$

$$F_{i-1}$$

$$F_i$$

$$\ldots$$
IC3 algorithm for Petri nets

Blocking phase: \( \text{block}(a, i) \)

\[
\begin{align*}
\text{Cover}_0 & \quad \text{in} \quad F_0 = m_0' \\
\text{Cover}_1 & \quad \text{in} \quad F_1 \\
\text{...} & \quad \text{in} \quad F_{i-1} \\
\text{Cover}_i & \quad \text{in} \quad F_i \\
\end{align*}
\]

given a pair \((a, i)\)
IC3 algorithm for Petri nets

Blocking phase: $block(a, i)$

$Cover_0$ \[ F_0 = m_0 \]

$Cover_1$ \[ F_1 \]

$...$

$Cover_{i-1}$ \[ F_{i-1} \]

$Cover_i$ \[ F_i \]

Given a pair $(a, i)$

Try to prove $a^\uparrow$ is unreachable within $i$ steps
IC3 algorithm for Petri nets

Blocking phase: \( \text{block}(a, i) \)

\[
\begin{align*}
F_0 &= m_0^\downarrow \\
F_1 &
\end{align*}
\]

\[
\begin{align*}
\text{Cover}_0 &
\end{align*}
\]

\[
\begin{align*}
\text{Cover}_1 &
\end{align*}
\]

\[
\begin{align*}
\text{Cover}_{i-1} &
\end{align*}
\]

\[
\begin{align*}
\text{Cover}_i &
\end{align*}
\]

\[
\begin{align*}
a^\uparrow &
\end{align*}
\]
IC3 algorithm for Petri nets

Blocking phase: $\text{block}(a, i)$

\[
\begin{align*}
\text{Cover}_0 & \quad \text{in} & \quad F_0 = m_0^\downarrow \\
\text{Cover}_1 & \quad \text{in} & \quad F_1 \\
\vdots & & \vdots \\
\text{Cover}_{i-1} & \quad \text{in} & \quad F_{i-1} \\
\text{Cover}_i & \quad \text{in} & \quad F_i \\
\text{pre}(a^\uparrow) \cap F_{i-1} & \div a^\uparrow
\end{align*}
\]
IC3 algorithm for Petri nets

Blocking phase: $\text{block}(a, i)$

$Cover_0 \subseteq F_0 = m_0^\downarrow$

$Cover_1 \subseteq F_1$

$\ldots$

$Cover_{i-1} \subseteq F_{i-1}$

$Cover_i \subseteq F_i \subseteq \ldots$

$pre(a^\uparrow) \cap F_{i-1} / a^\uparrow \neq \emptyset$
IC3 algorithm for Petri nets

Blocking phase: $block(a, i)$

\[
\begin{align*}
Cover_0 \quad & F_0 = m_0^\downarrow \\
Cover_1 \quad & F_1 \\
\vdots \quad & \vdots \\
Cover_{i-1} \quad & F_{i-1} \\
Cover_i \quad & F_i \quad \vdots \\
\end{align*}
\]

$pre(a^\uparrow) \cap F_{i-1} /a^\uparrow \neq \emptyset$

extract an unselected marking $b$
from $pre(a^\uparrow) \cap F_{i-1} /a^\uparrow$
IC3 algorithm for Petri nets

Blocking phase: \( \text{block}(a, i) \)

\[
\begin{align*}
\text{Cover}_0 & \subseteq m_0^\uparrow \\
\text{Cover}_1 & \subseteq F_1 \\
\vdots & \quad \vdots \\
\text{Cover}_{i-1} & \subseteq F_{i-1} \\
\text{Cover}_i & \subseteq F_i \\
\end{align*}
\]

\[b^\uparrow \rightarrow a^\uparrow\]

\[
\text{pre}(a^\uparrow) \cap F_{i-1} / a^\uparrow \neq \emptyset
\]

extract an unselected marking \( b \)
from \( \text{pre}(a^\uparrow) \cap F_{i-1} / a^\uparrow \)
Blocking phase: \( \text{block}(a, i) \)

\[
\begin{align*}
\text{Cover}_0 & \quad \text{IN} \quad F_0 = m_0^{\uparrow} \\
\text{Cover}_1 & \quad \text{IN} \quad F_1 \\
\ldots & \\
\text{Cover}_{i-1} & \quad \text{IN} \quad F_{i-1} \\
\text{Cover}_i & \quad \text{IN} \quad F_i \quad \ldots \\
\end{align*}
\]

\[b^{\uparrow} \rightarrow a^{\uparrow}\]

\[\text{pre}(a^{\uparrow}) \cap F_{i-1} / a^{\uparrow} \neq \emptyset\]

extract an unselected marking \( b \)

from \(\text{pre}(a^{\uparrow}) \cap F_{i-1} / a^{\uparrow}\)

generate a new pair \((b, i - 1)\)

\(\text{block}(b, i - 1)\)
IC3 algorithm for Petri nets

Blocking phase: *block*(*a*, *i*)

\[
\text{Cover}_0 \quad \text{Cover}_1 \quad \ldots \\
F_0 = \downarrow m_0 \\
F_1 \quad \ldots \\
F_{i-1} \\
F_i \quad \ldots
\]

\[\uparrow b \quad \rightarrow \quad \uparrow a\]

\[\text{pre}(\uparrow a) \cap F_{i-1} / \uparrow a \neq \emptyset\]

extract an unselected marking \(b\)

from \(\text{pre}(\uparrow a) \cap F_{i-1} / \uparrow a\)

generate a new pair \((b, i - 1)\)

*block*(*b*, *i* − 1)

try to prove \(\uparrow b\) is unreachable

within \(i - 1\) steps
IC3 algorithm for Petri nets

Blocking phase: \( \text{block}(a, i) \)

\[
\begin{align*}
Cover_0 & \quad \text{in} & F_0 &= m_0^\uparrow \\
Cover_1 & \quad \text{in} & F_1 & \quad \cdots \\
Cover_{i-1} & \quad \text{in} & F_{i-1} & \quad \cdots \\
Cover_i & \quad \text{in} & F_i & \quad \cdots \\
\end{align*}
\]

\[
d^\uparrow \quad \rightarrow \quad c^\uparrow \quad \rightarrow \quad \cdots \quad \rightarrow \quad b^\uparrow \quad \rightarrow \quad a^\uparrow
\]

\[
\text{pre}(a^\uparrow) \cap F_{i-1} / a^\uparrow \neq \emptyset
\]

extract an unselected marking \( b \)

from \( \text{pre}(a^\uparrow) \cap F_{i-1} / a^\uparrow \)

generate a new pair \((b, i - 1)\)

\( \text{block}(b, i - 1) \)

try to prove \( b^\uparrow \) is unreachable
within \( i - 1 \) steps

Kang, Bai, Jiao

June 24, 2021
IC3 algorithm for Petri nets

Blocking phase: $\text{block}(a, i)$

$Cover_0 \in F_0 = m_0^\uparrow$

$Cover_1 \in F_1$

$\ldots$

$Cover_{i-1} \in F_{i-1}$

$Cover_i \in F_i \ldots$

$d^\uparrow \rightarrow c^\uparrow \rightarrow \ldots \rightarrow b^\uparrow \rightarrow a^\uparrow$

finally generate a new pair $(d, 0)$

$pre(a^\uparrow) \cap F_{i-1} / a^\uparrow \neq \emptyset$

extract an unselected marking $b$

from $pre(a^\uparrow) \cap F_{i-1} / a^\uparrow$

generate a new pair $(b, i - 1)$

$\text{block}(b, i - 1)$

try to prove $b^\uparrow$ is unreachable within $i - 1$ steps
IC3 algorithm for Petri nets

Blocking phase: \( \text{block}(a, i) \)

\[
\begin{align*}
Cover_0 & \quad \text{IN} \\
F_0 & = m_0^{\downarrow} \\
d^{\uparrow} \\
\end{align*}
\begin{align*}
Cover_1 & \quad \text{IN} \\
F_1 & \\
c^{\uparrow} \\
\end{align*}
\begin{align*}
\ldots & \quad \text{IN} \\
F_{i-1} & \\
b^{\uparrow} \\
\end{align*}
\begin{align*}
Cover_i & \quad \text{IN} \\
F_i & \\
a^{\uparrow} \\
\end{align*}
\]

finally generate a new pair \((d, 0)\)

find a path from \(m_0^{\downarrow}\) to \(a^{\uparrow}\)

\[
\begin{align*}
\text{pre}(a^{\uparrow}) \cap F_{i-1} / a^{\uparrow} \neq \emptyset
\end{align*}
\]

extract an unselected marking \(b\)

from \(\text{pre}(a^{\uparrow}) \cap F_{i-1} / a^{\uparrow}\)

generate a new pair \((b, i - 1)\)

\(\text{block}(b, i - 1)\)

try to prove \(b^{\uparrow}\) is unreachable within \(i - 1\) steps
Blocking phase: \( \text{block}(a, i) \)

\[
\begin{align*}
\text{Cover}_0 &\quad \text{Cover}_1 & \quad \ldots & \quad \text{Cover}_{i-1} & \quad \text{Cover}_i \\
F_0 = m_0 &\quad F_1 & \quad \ldots & \quad F_{i-1} & \quad F_i \\
\uparrow &\quad \uparrow & \quad \ldots & \quad \uparrow & \quad \uparrow \\
d &\quad c & \quad \ldots & \quad b & \quad a \\
\end{align*}
\]

finally generate a new pair \((d, 0)\)

find a path from \(m_0\) to \(a\)

fail to block \(a\) at \(F_i\)
i.e. \(a\) is coverable

\[
\text{pre}(a) \cap F_{i-1} / a \neq \emptyset
\]

extract an unselected marking \(b\)
from \(\text{pre}(a) \cap F_{i-1} / a\)

generate a new pair \((b, i - 1)\)

\(\text{block}(b, i - 1)\)

try to prove \(b\) is unreachable
within \(i - 1\) steps
Blocking phase: $\text{block}(a, i)$

$Cover_0 \subseteq Cover_1 \subseteq \ldots \subseteq Cover_{i-1} \subseteq Cover_i$

$F_0 = m_0^\downarrow$

$F_1$

$\ldots$

$F_{i-1}$

$F_i$

$\pre(a^\uparrow) \cap F_{i-1} / a^\uparrow$
IC3 algorithm for Petri nets

Blocking phase: \( \text{block}(a, i) \)

\[
\begin{align*}
F_0 &= m_0^\downarrow \\
F_1 &\subseteq \cdots \\
F_i \\
\end{align*}
\]

\[
\text{pre}(a^\uparrow) \cap F_{i-1} / a^\uparrow = \emptyset
\]
IC3 algorithm for Petri nets

### Blocking phase: $block(a, i)$

<table>
<thead>
<tr>
<th>$Cover_0$</th>
<th>$Cover_1$</th>
<th>$Cover_{i-1}$</th>
<th>$Cover_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_0 = m_0^\downarrow$</td>
<td>$F_1$</td>
<td>...</td>
<td>$F_{i-1}$</td>
</tr>
</tbody>
</table>

$pre(a^\uparrow) \cap F_{i-1} / a^\uparrow = \emptyset$

$a^\uparrow$ cannot be reachable in 1 step from $Cover_{i-1}$
IC3 algorithm for Petri nets

Blocking phase: \( \text{block} (a, i) \)

\[
\begin{align*}
\text{Cover}_0 & \quad \text{Cover}_1 \\
F_0 &= m_0^\downarrow & F_1 \\
\text{...} & \quad \text{...} \\
\text{Cover}_{i-1} & \quad \text{Cover}_i \\
\emptyset & \quad a^\uparrow \\
\end{align*}
\]

\[
\text{pre}(a^\uparrow) \cap F_{i-1} / a^\uparrow = \emptyset
\]

\( a^\uparrow \) cannot be reachable in 1 step from \( \text{Cover}_{i-1} \)
Blocking phase: $\text{block}(a, i)$

$\text{Cover}_0 \subseteq \text{Cover}_1 \subseteq \ldots \subseteq \text{Cover}_{i-1} \subseteq \text{Cover}_i$

$F_0 = m_0 \downarrow$

$F_1 \rightarrow \ldots \rightarrow F_{i-1} \rightarrow F_i \rightarrow \ldots$

$\emptyset \rightarrow a^\uparrow$

$\text{pre}(a^\uparrow) \cap F_{i-1} / a^\uparrow = \emptyset$

$a^\uparrow$ cannot be reachable in 1 step from $\text{Cover}_{i-1}$

$a$ is uncoverable within $i$ steps
IC3 algorithm for Petri nets

Blocking phase: \( \text{block}(a, i) \)

\[
\begin{align*}
F_0 = m_0^\uparrow & \\
F_1 & \\
\ldots & \\
F_{i-1} & \\
F_i & \\
\ldots & \\
\emptyset & \rightarrow a^\uparrow \\
\end{align*}
\]

\[
\text{pre}(a^\uparrow) \cap F_{i-1} / a^\uparrow = \emptyset
\]

\( a^\uparrow \) cannot be reachable in 1 step from \( \text{Cover}_{i-1} \)

\( a \) is uncoverable within \( i \) steps

\( a^\uparrow \) can be removed from the coverable set \( F_i \)
IC3 algorithm for Petri nets

Blocking phase: $\text{block}(a, i)$

$\text{Cover}_0 \in F_0 = m_0^\uparrow$

$\text{Cover}_1 \in F_1$

$\ldots$

$\text{Cover}_{i-1} \in F_{i-1}$

$\text{Cover}_i \in F_i \setminus a^\uparrow$

$\phi \rightarrow a^\uparrow$

$\text{pre}(a^\uparrow) \cap F_{i-1} / a^\uparrow = \phi$

$a^\uparrow$ cannot be reachable in 1 step from $\text{Cover}_{i-1}$

$a$ is uncoverable within $i$ steps

$a^\uparrow$ can be removed from the coverable set $F_i$
IC3 algorithm for Petri nets

\[
\text{input } N = \langle P, T, W, m_0 \rangle \text{ and } m_t \\
\text{initialize } F_0 = m_0^\downarrow, F_1 = \mathbb{N}^{|P|}, k = 1
\]
IC3 algorithm for Petri nets

input $N = \langle P, T, W, m_0 \rangle$ and $m_t$
initialize $F_0 = m_0 \downarrow, F_1 = \mathbb{N}^{|P|}, k = 1$

generate a pair $(m_t, k)$
IC3 algorithm for Petri nets

input \( N = \langle P, T, W, m_0 \rangle \) and \( m_t \)
initialize \( F_0 = m_0 \downarrow, F_1 = \mathbb{N}^{|P|}, k = 1 \)

generate a pair \((m_t, k)\)

try to block \( m_t \) at \( F_k \)
IC3 algorithm for Petri nets

- Input $N = \langle P, T, W, m_0 \rangle$ and $m_t$
- Initialize $F_0 = m_0^\downarrow$, $F_1 = \mathbb{N}^{|P|}$, $k = 1$
- Generate a pair $(m_t, k)$
- Try to block $m_t$ at $F_k$
IC3 algorithm for Petri nets

input $N = \langle P, T, W, m_0 \rangle$ and $m_t$
initialize $F_0 = m_0^\downarrow, F_1 = \mathbb{N}^{|P|}, k = 1$

generate a pair $(m_t, k)$

try to block $m_t$ at $F_k$

failed

a path from $m_0$ to $m_t^\uparrow$ is found
IC3 algorithm for Petri nets

1. **Input**: $N = \langle P, T, W, m_0 \rangle$ and $m_t$
2. **Initialize**: $F_0 = m_0^\downarrow, F_1 = \mathbb{N}^{\mid P\mid}, k = 1$
3. **Generate a pair**: $(m_t, k)$
4. **Try to block**: $m_t$ at $F_k$
5. **Result**:
   - **Failed**: a path from $m_0$ to $m_t^\uparrow$ is found
   - **Coverable**: $m_t$ is coverable in $k$-steps
IC3 algorithm for Petri nets

input $N = \langle P, T, W, m_0 \rangle$ and $m_t$
initialize $F_0 = m_0 \downarrow, F_1 = \mathbb{N}^{|P|}, k = 1$

generate a pair $(m_t, k)$

try to block $m_t$ at $F_k$

failed

a path from $m_0$ to $m_t \uparrow$ is found

$m_t$ is coverable in k-steps

End
IC3 algorithm for Petri nets

input $N = \langle P, T, W, m_0 \rangle$ and $m_t$
initialize $F_0 = m_0^\downarrow, F_1 = \mathbb{N}^{|P|}, k = 1$

generate a pair $(m_t, k)$

try to block $m_t$ at $F_k$

failed

a path from $m_0$ to $m_t^\uparrow$ is found

$m_t$ is coverable in k-steps

End
IC3 algorithm for Petri nets

input \( N = \langle P, T, W, m_0 \rangle \) and \( m_t \)
initialize \( F_0 = m_0^\downarrow, F_1 = \mathbb{N}^{\mid P \mid}, k = 1 \)

generate a pair \((m_t, k)\)

try to block \( m_t \) at \( F_k \)

\[ k = k + 1 \]
\[ F_k = \mathbb{N}^{\mid P \mid} \]

a path from \( m_0 \) to \( m_t^\uparrow \) is found

\( m_t \) is coverable in \( k \)-steps

End
IC3 algorithm for Petri nets

input $N = \{P, T, W, m_0\}$ and $m_t$
initialize $F_0 = m_0^\downarrow, F_1 = \mathbb{N}^{\mid P \mid} k = 1$

generate a pair $(m_t, k)$

generate a pair $(m_t, k)$

try to block $m_t$ at $F_k$

failed

a path from $m_0$ to $m_t^\uparrow$ is found

successfully

$k = k + 1$

$F_k = \mathbb{N}^{\mid P \mid}$

$m_t$ is coverable in k-steps

$F_i = F_{i+1}$

End
IC3 algorithm for Petri nets

input $N = \langle P, T, W, m_0 \rangle$ and $m_t$
initialize $F_0 = m_0^\downarrow, F_1 = \mathbb{N}^{|P|}, k = 1$

generate a pair $(m_t, k)$

try to block $m_t$ at $F_k$

failed

a path from $m_0$ to $m_t^\uparrow$ is found

successfully

$k = k + 1$
$F_k = \mathbb{N}^{|P|}$

$m_t$ is coverable in $k$-steps

Yes

$F_i = F_{i+1}$

End

Kang, Bai, Jiao

June 24, 2021
IC3 algorithm for Petri nets

input $N = \langle P, T, W, m_0 \rangle$ and $m_t$
initialize $F_0 = m_0^\downarrow, F_1 = \mathbb{N}^{|P|}, k = 1$

generate a pair $(m_t, k)$

try to block $m_t$ at $F_k$

failed

a path from $m_0$ to $m_t^\uparrow$ is found

$m_t$ is coverable in k-steps

Yes

Yes

post($F_i$) $\subseteq F_{i+1}$

invariant found

$m_t$ is uncoverable

End

invariant found

$k = k + 1$

$F_k = \mathbb{N}^{|P|}$

End
IC3 algorithm for Petri nets

input $N = \langle P, T, W, m_0 \rangle$ and $m_t$
initialize $F_0 = m_0^\downarrow, F_1 = \mathbb{N}^{|P|}, k = 1$

generate a pair $(m_t, k)$

try to block $m_t$ at $F_k$

$\begin{align*}
&k = k + 1 \\
&F_k = \mathbb{N}^{|P|}
\end{align*}$

success\(\text{fully}

\text{post}(F_i) \subseteq F_{i+1}
invariant found

$m_t$ is uncoverable

Yes

$F_i = F_{i+1}$

End

failed

a path from $m_0$ to $m_t^\uparrow$ is found

$m_t$ is coverable in $k$-steps
IC3 algorithm for Petri nets

input $N = \langle P, T, W, m_0 \rangle$ and $m_t$
initialize $F_0 = m_0^\downarrow, F_1 = \mathbb{N}^{|P|}, k = 1$

generate a pair $(m_t, k)$

try to block $m_t$ at $F_k$

$k = k + 1$
$F_k = \mathbb{N}^{|P|}$

Yes: $post(F_i) \subseteq F_{i+1}$
invariant found
$m_t$ is uncoverable

End

Yes: $F_i = F_{i+1}$

failed

a path from $m_0$ to $m_t^\uparrow$ is found

$m_t$ is coverable in k-steps
IC3 algorithm for Petri nets

input $N = \langle P, T, W, m_0 \rangle$ and $m_t$
initialize $F_0 = m_0^{\downarrow}, F_1 = \mathbb{N}^{|P|}, k = 1$

generate a pair $(m_t, k)$

try to block $m_t$ at $F_k$

- successfully
  
  $k = k + 1$
  $F_k = \mathbb{N}^{|P|}$

- failed
  
  a path from $m_0$ to $m_t^{\uparrow}$ is found

$\text{post}(F_i) \subseteq F_{i+1}$

invariant found

$m_t$ is uncoverable

$m_t$ is coverable in $k$-steps

End

Kang, Bai, Jiao
IC3 algorithm for Petri nets

Input $N = (P, T, W, m_0)$ and $m_t$
Initialize $F_0 = m_0 \downarrow, F_1 = \mathbb{N}^{|P|}, k = 1$

Generate a pair $(m_t, k)$

Try to block $m_t$ at $F_k$

If successfully

$k = k + 1$

$F_k = \mathbb{N}^{|P|}$

Post$(F_i) \subseteq F_{i+1}$

Invariant found

$m_t$ is coverable in $k$-steps

End

If failed

$m_t$ is uncoverable

Try to block $m_t$ at $F_k$

If successfully

$a$ path from $m_0$ to $m_t \uparrow$ is found

End

If No

$m_t$ is uncoverable

End
IC3 algorithm for Petri nets

- **Input:** $N = \langle P, T, W, m_0 \rangle$ and $m_t$
- **Initialize:** $F_0 = m_0^\downarrow, F_1 = \mathbb{N}^{|P|}, k = 1$

**Generate a pair** $(m_t, k)$

**Try to block** $m_t$ at $F_k$

- **Successfully:** $k = k + 1$, $F_k = \mathbb{N}^{|P|}$
- **Failed:**
  - **Path found:**
    - **Post:** $\text{post}(F_i) \subseteq F_{i+1}$
    - $m_t$ is coverable in $k$-steps
  - **Invariant found:**
    - $F_i = F_{i+1}$
    - $m_t$ is uncoverable

**End**
IC3 algorithm for Petri nets

input $N = \langle P, T, W, m_0 \rangle$ and $m_t$
initialize $F_0 = m_0^\dagger$, $F_1 = \mathbb{N}^{\{P\}}$, $k = 1$

generate a pair $(m_t, k)$

IC3 works on Petri nets with high dimensionality directly

try to block $m_t$ at $F_k$

$k = k + 1$
$F_k = \mathbb{N}^{\{P\}}$

Yes

post($F_i$) $\subseteq F_{i+1}$
invariant found
$m_t$ is uncoverable

No

$F_i = F_{i+1}$

$a$ path from $m_0$ to $m_t^\dagger$ is found
$m_t$ is coverable in $k$-steps

End
IC3 algorithm for Petri nets

input $N = (P, T, W, m_0)$ and $m_t$
initialize $F_0 = m_0^\downarrow, F_1 = N^{\mid P\mid}, k = 1$

generate a pair $(m_t, k)$

IC3 works on Petri nets with high dimensionality directly

try to block $m_t$ at $F_k$

Can IC3 perform better on Petri nets?

Failed

a path from $m_0$ to $m_t^\uparrow$ is found

$F_k = N^{\mid P\mid}$

$m_t$ is coverable in $k$-steps

Yes

$F_i = F_{i+1}$

post($F_i$) $\subseteq F_{i+1}$ invariant found
$m_t$ is uncoverable

No

End
Place-merge abstraction

Merge some places of original Petri net into a single abstract place, get an abstract Petri net with lower dimensionality.
Place-merge abstraction

Merge some places of original Petri net into a single abstract place, get an abstract Petri net with lower dimensionality.

**Definition**

Given a Petri net $N = \langle P, T, W, m_0 \rangle$, where $P = \{p_1, p_1 \ldots p_k\}$
- The abstraction function is a surjective function $\alpha: P \to \hat{P}$, where $\hat{P} = \{\hat{p}_1, \hat{p}_2 \ldots \hat{p}_{\hat{k}}\}$ and $\hat{k} \leq k$. 
Place-merge abstraction

Merge some places of original Petri net into a single abstract place, get an abstract Petri net with lower dimensionality.
Merge some places of original Petri net into a single abstract place, get an abstract Petri net with lower dimensionality.

\[
\begin{align*}
\alpha(p_0) &= \alpha(p_1) = q_0 \\
\alpha(p_2) &= \alpha(p_3) = \alpha(p_4) = q_1
\end{align*}
\]
Place-merge abstraction

Merge some places of original Petri net into a single abstract place, get an abstract Petri net with lower dimensionality.

\[ \alpha(p_0) = \alpha(p_1) = q_0 \]
\[ \alpha(p_2) = \alpha(p_3) = \alpha(p_4) = q_1 \]

All weights of arcs are equal to 1 except for \( W(q_1, t_2) = 2 \).
Proposition

Given a Petri net $N = \langle P, T, W, m_0 \rangle$ and one of its abstractions $\hat{N} = \langle \hat{P}, T, \hat{W}, \hat{m}_0 \rangle$, $m_t$ and its abstract version $\hat{m}_t$

- If $m_t$ is coverable in $N$, then its abstract version $\hat{m}_t$ is coverable in $\hat{N}$. But the converse does not hold.
Place-merge abstraction

**Proposition**

Given a Petri net $N = \langle P, T, W, m_0 \rangle$ and one of its abstractions $\hat{N} = \langle \hat{P}, T, \hat{W}, \hat{m}_0 \rangle$, $m_t$ and its abstract version $\hat{m}_t$

- If $m_t$ is coverable in $N$, then its abstract version $\hat{m}_t$ is coverable in $\hat{N}$. But the converse does not hold.

$\hat{m}_t$ is uncoverable in $\hat{N}$
### Proposition

Given a Petri net $N = \langle P, T, W, m_0 \rangle$ and one of its abstractions $\hat{N} = \langle \hat{P}, T, \hat{W}, \hat{m}_0 \rangle$, $m_t$ and its abstract version $\hat{m}_t$

- If $m_t$ is coverable in $N$, then its abstract version $\hat{m}_t$ is coverable in $\hat{N}$. But the converse does not hold.

\[
\hat{m}_t \text{ is uncoverable in } \hat{N} \quad \rightarrow \quad m_t \text{ is uncoverable in } N
\]
Place-merge abstraction

Proposition

Given a Petri net $N = \langle P, T, W, m_0 \rangle$ and one of its abstractions $\hat{N} = \langle \hat{P}, T, \hat{W}, \hat{m}_0 \rangle$, $m_t$ and its abstract version $\hat{m}_t$
- If $m_t$ is coverable in $N$, then its abstract version $\hat{m}_t$ is coverable in $\hat{N}$. But the converse does not hold.

$\hat{m}_t$ is uncoverable in $\hat{N} \quad \rightarrow \quad m_t$ is uncoverable in $N$

$\hat{m}_t$ is coverable in $\hat{N}$
Proposition

Given a Petri net \( N = \langle P, T, W, m_0 \rangle \) and one of its abstractions \( \hat{N} = \langle \hat{P}, \hat{T}, \hat{W}, \hat{m}_0 \rangle \), \( m_t \) and its abstract version \( \hat{m}_t \)

- If \( m_t \) is coverable in \( N \), then its abstract version \( \hat{m}_t \) is coverable in \( \hat{N} \). But the converse does not hold.

\[
\begin{align*}
\hat{m}_t \text{ is uncoverable in } \hat{N} & \quad \rightarrow \quad m_t \text{ is uncoverable in } N \\
m_t \text{ is coverable in } N & \quad \leftrightarrow \quad m_t \text{ is coverable in } N
\end{align*}
\]
Place-merge abstraction

Spurious counterexample
Place-merge abstraction

Spurious counterexample
Place-merge abstraction

Spurious counterexample

Abstract PN

\[ (0, 3) \xrightarrow{t_0} (1, 2) \xrightarrow{t_0} (2, 1) \]
Place-merge abstraction

Spurious counterexample

Abstract PN

\begin{align*}
(0, 3) \xrightarrow{t_0} (1, 2) \xrightarrow{t_0} (2, 1)
\end{align*}

Original PN

\begin{align*}
(0, 0, 1, 1, 1) & \xrightarrow{t_0} (1, 0, 1, 1, 0) & \xrightarrow{t_0} (1, 1, 0, 1, 0) \\
(0, 0, 1, 2, 0) & \xrightarrow{t_0} (1, 0, 0, 1, 1) & \xrightarrow{t_0} (1, 1, 1, 0, 0) \\
(0, 0, 2, 1, 0) & \xrightarrow{t_0} (0, 1, 1, 1, 0) & \xrightarrow{t_0} (2, 0, 0, 1, 0) \\
(0, 0, 3, 0, 0) & \xrightarrow{t_0} (0, 1, 0, 1, 1) & \xrightarrow{t_0} (2, 0, 1, 0, 0)
\end{align*}
Place-merge abstraction

Spurious counterexample

Abstract PN

Original PN

$t_0$ is not enabled here
Place-merge abstraction

When a counterexample is spurious
When a counterexample is spurious

Counter-example $\pi = t_0 t_1 \ldots t_{k-1}$ is not spurious iff

$m_0 \xrightarrow{t_0} m_1 \xrightarrow{t_1} m_2 \xrightarrow{t_2} \ldots \xrightarrow{t_{k-1}} m_k \wedge m_t \preceq m_k$
Place-merge abstraction

When a counterexample is spurious

Counter-example $\pi = t_0 t_1 \ldots t_{k-1}$ is not spurious iff

$$m_0 \xrightarrow{t_0} m_1 \xrightarrow{t_1} m_2 \xrightarrow{t_2} \ldots \xrightarrow{t_{k-1}} m_k \land m_t \preceq m_k$$

The path $\pi$ is spurious:
① $t_i$ is not enabled at $m_i$ ($0 \leq i < k$), or
② $t_i$ is enabled at $m_i$ ($0 \leq i < k$), but $m_t \nleq m_k$
Place-merge abstraction

How to refine an abstraction?
Place-merge abstraction

How to refine an abstraction?

$t_i$ is not enabled at $m_i$ ($0 \leq i < k$)

- extract places satisfying $m_i(p) < W(p, t_i)$
- merge these places into a new abstract place
Place-merge abstraction

How to refine an abstraction?

- $t_i$ is not enabled at $m_i$ ($0 \leq i < k$)
  - extract places satisfying $m_i(p) < W(p, t_i)$
  - merge these places into a new abstract place

- $t_i$ is enabled at $m_i$ ($0 \leq i < k$), but $m_t \not\leq m_k$
  - extract places satisfying $m_t(p) > m_k(p)$
  - merge these places into a new abstract place
How to refine an abstraction?

- Place-merge abstraction

**Abstract PN**

\[
\begin{array}{c}
(0, 3) \xrightarrow{t_0} (1, 2) \xrightarrow{t_0} (2, 1)
\end{array}
\]

**Original PN**

\[
\begin{array}{c}
(0, 0, 1, 1, 1) \xrightarrow{t_0} (1, 0, 0, 1, 1) \xrightarrow{t_0} (1, 1, 0, 1, 0) \\
(0, 0, 2, 1, 0) \xrightarrow{t_0} (0, 1, 1, 1, 0) \\
(0, 0, 3, 0, 0) \xrightarrow{t_0} (0, 1, 0, 1, 1) \\
\end{array}
\]

\[
\begin{array}{c}
(1, 1, 0, 1, 0) \\
(1, 1, 1, 0, 0) \\
(2, 0, 0, 1, 0) \\
(2, 0, 1, 0, 0) \\
\end{array}
\]

\[t_0 \text{ is not enabled here}\]
Place-merge abstraction

Abstraction refinement

Abstract PN

Original PN

$t_0$ is not enabled here
Place-merge abstraction

Abstraction refinement

Abstract PN

\[ (0, 3) \xrightarrow{t_0} (1, 2) \xrightarrow{t_0} (2, 1) \]

Original PN

\[
\begin{array}{c}
(0,0,1,1,1) \\
(0,0,1,2,0) \\
(0,0,2,1,0) \\
(0,0,3,0,0) \\
\hline
(1,0,0,0,1,1,1) \\
(0,1,0,0,1,1,1,0) \\
(0,0,1,1,1,0) \\
(0,0,1,1,1,0) \\
\end{array}
\]

\[ t_0 \text{ is not enabled here} \]
**Place-merge abstraction**

**Abstraction refinement**

extract \( p_2 \) from \( q_1 \)!

\[
\begin{align*}
\alpha(p_0) &= \alpha(p_1) = q_0 \\
\alpha(p_2) &= \alpha(p_3) = \alpha(p_4) = q_1
\end{align*}
\]

\( t_0 \) is not enabled here
Abstraction refinement

extract $p_2$ from $q_1$!

$\alpha(p_0) = \alpha(p_1) = q_0$
$\alpha(p_3) = \alpha(p_4) = q_1$
$\alpha(p_2) = q_2$

$t_0$ is not enabled here
Abstraction refinement

extract $p_2$ from $q_1$!

<table>
<thead>
<tr>
<th>Abstraction</th>
<th>Original PN</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha(p_0) = \alpha(p_1) = q_0$</td>
<td></td>
</tr>
<tr>
<td>$\alpha(p_3) = \alpha(p_4) = q_1$</td>
<td></td>
</tr>
<tr>
<td>$\alpha(p_2) = q_2$</td>
<td></td>
</tr>
</tbody>
</table>

$t_0$ is not enabled here
Place-merge abstraction

Abstraction refinement

extract $p_2$ from $q_1$!

$\alpha(p_0) = \alpha(p_1) = q_0$

$\alpha(p_3) = \alpha(p_4) = q_1$

$\alpha(p_2) = q_2$

$t_0$ is not enabled here
IC3+PMA algorithm

- Try to improve the outperformance of IC3
- IC3 is the core of IC3+PMA
- Place-merge abstraction reduces the dimensionality of PN
- IC3 works on the abstract PN with lower dimensionality
merge all places into a single place
IC3+PMA algorithm

merge all places into a single place

check abstraction
IC3+PMA algorithm

- Merge all places into a single place
- Check abstraction
- Uncoverable
- Get an inductive invariant of abstraction
IC3+PMA algorithm

merge all places into a single place

check abstraction

uncoverable

get an inductive invariant of abstraction

$m_t$ is uncoverable in original model
IC3+PMA algorithm

merge all places into a single place

check abstraction

uncoverable

get an inductive invariant of abstraction

$m_t$ is uncoverable in original model

End
IC3+PMA algorithm

1. Merge all places into a single place.
2. Check abstraction.
   - If uncoverable, get an inductive invariant of abstraction.
   - If $m_t$ is uncoverable in the original model, end.

End
IC3+PMA algorithm

merge all places into a single place

coverable

check abstraction

get a counter-example $\pi$

uncoverable

get an inductive invariant of abstraction

$m_t$ is uncoverable in original model

End
IC3+PMA algorithm

- Merge all places into a single place

  - Check abstraction

    - Coverable: Get a counter-example $\pi$

      - Is $\pi$ spurious

    - Uncoverable: Get an inductive invariant of abstraction

      - $m_t$ is uncoverable in original model

End
IC3+PMA algorithm

merge all places into a single place

check abstraction

coverable

get a counter-example $\pi$

is $\pi$ spurious

No

$m_t$ is coverable in original model

End

uncoverable

get an inductive invariant of abstraction

$m_t$ is uncoverable in original model

June 24, 2021
IC3+PMA algorithm

merge all places into a single place

check abstraction

get a counter-example $\pi$

is $\pi$ spurious

No

$m_t$ is coverable in original model

End

coverable

uncoverable

get an inductive invariant of abstraction

$m_t$ is uncoverable in original model

Kang, Bai, Jiao
**IC3+PMA algorithm**

1. Merge all places into a single place
2. Check abstraction:
   - If coverable, get a counter-example $\pi$
     - If $\pi$ is spurious, go back.
     - If $m_t$ is coverable in the original model, end.
   - If uncoverable, get an inductive invariant of abstraction.
     - If $m_t$ is uncoverable in the original model, end.

End.
IC3+PMA algorithm

merge all places into a single place

check abstraction

coverable

get a counter-example $\pi$

Yes

is $\pi$ spurious

No

$m_t$ is coverable in original model

uncoverable

get an inductive invariant of abstraction

$m_t$ is uncoverable in original model

End
IC3+PMA algorithm

get a new abstraction

refinement

get a counter-example $\pi$

Yes

is $\pi$ spurious

No

$\mathcal{m}_t$ is coverable in original model

merge all places into a single place

check abstraction

coverable

uncoverable

get an inductive invariant of abstraction

$\mathcal{m}_t$ is uncoverable in original model

End
IC3+PMA algorithm

merge all places into a single place

coverable

check abstraction

uncoverable

generate a new abstraction

refinement

generate a counter-example $\pi$

Yes

is $\pi$ spurious

No

$m_t$ is coverable in the original model

$m_t$ is uncoverable in the original model

End
IC3+PMA algorithm

1. Get a new abstraction
2. Refinement
3. Get a counter-example $\pi$
4. Is $\pi$ spurious?
   - Yes: $m_t$ is coverable in original model
   - No: $m_t$ is uncoverable in original model
5. Check abstraction
   - Coverable: Merge all places into a single place
   - Uncoverable: Get an inductive invariant of abstraction
6. End
Experiments

- total 80 benchmarks
- compare running time between IC3 and IC3+PMA
- IC3+PMA outperforms IC3 on 53.75% of benchmarks
- dimensionality has decreased by 63.34% on average
Experiments

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Places</th>
<th>IC3+PMA AbsPlaces</th>
<th>IC3+PMA Ref</th>
<th>IC3+PMA time(s)</th>
<th>IC3 time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncoverable instances</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>newrrt</td>
<td>9</td>
<td>1</td>
<td>0</td>
<td>&lt;0.01</td>
<td>0.06</td>
</tr>
<tr>
<td>kanban (bounded)</td>
<td>16</td>
<td>1</td>
<td>0</td>
<td>&lt;0.01</td>
<td>1.22</td>
</tr>
<tr>
<td>manufacturing</td>
<td>13</td>
<td>1</td>
<td>0</td>
<td>&lt;0.01</td>
<td>0.16</td>
</tr>
<tr>
<td>fms</td>
<td>22</td>
<td>4</td>
<td>3</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>fms_attic</td>
<td>22</td>
<td>4</td>
<td>3</td>
<td>0.01</td>
<td>0.04</td>
</tr>
<tr>
<td>mesh2x2</td>
<td>32</td>
<td>5</td>
<td>4</td>
<td>0.01</td>
<td>0.03</td>
</tr>
<tr>
<td>mesh3x2</td>
<td>52</td>
<td>5</td>
<td>4</td>
<td>0.02</td>
<td>0.08</td>
</tr>
<tr>
<td>pingpong</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>RandCAS 2</td>
<td>110</td>
<td>8</td>
<td>7</td>
<td>0.08</td>
<td>0.44</td>
</tr>
<tr>
<td>Conditionals 2</td>
<td>214</td>
<td>26</td>
<td>25</td>
<td>1.39</td>
<td>5.79</td>
</tr>
<tr>
<td>Coverable instances</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>leabasicapproach</td>
<td>16</td>
<td>5</td>
<td>4</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>Dekker 1</td>
<td>41</td>
<td>27</td>
<td>25</td>
<td>2.08</td>
<td>3.23</td>
</tr>
<tr>
<td>DoubleLock1 1</td>
<td>64</td>
<td>35</td>
<td>32</td>
<td>11.26</td>
<td>13.31</td>
</tr>
<tr>
<td>Pthread5 1</td>
<td>80</td>
<td>47</td>
<td>44</td>
<td>97.28</td>
<td>Timeout</td>
</tr>
<tr>
<td>RandLock0 2</td>
<td>110</td>
<td>48</td>
<td>46</td>
<td>21.40</td>
<td>24.89</td>
</tr>
<tr>
<td>Spin2003 2</td>
<td>56</td>
<td>38</td>
<td>35</td>
<td>67.35</td>
<td>Timeout</td>
</tr>
<tr>
<td>Szymanski 1</td>
<td>61</td>
<td>46</td>
<td>44</td>
<td>19.62</td>
<td>32.69</td>
</tr>
<tr>
<td>Constants 1</td>
<td>26</td>
<td>14</td>
<td>13</td>
<td>0.03</td>
<td>0.03</td>
</tr>
<tr>
<td>FuncPtr3 1</td>
<td>40</td>
<td>16</td>
<td>13</td>
<td>0.19</td>
<td>0.33</td>
</tr>
</tbody>
</table>

IC3+PMA performs better
## Experiments

IC3+PMA performs better

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Places</th>
<th>IC3+PMA AbsPlaces</th>
<th>IC3+PMA Ref</th>
<th>IC3+PMA time(s)</th>
<th>IC3 time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncoverable instances</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>newrtp</td>
<td>9</td>
<td>1</td>
<td>0</td>
<td>&lt;0.01</td>
<td>0.06</td>
</tr>
<tr>
<td>kanban (bounded)</td>
<td>16</td>
<td>1</td>
<td>0</td>
<td>&lt;0.01</td>
<td>1.22</td>
</tr>
<tr>
<td>manufacturing</td>
<td>13</td>
<td>1</td>
<td>0</td>
<td>&lt;0.01</td>
<td>0.16</td>
</tr>
<tr>
<td>fms</td>
<td>22</td>
<td>4</td>
<td>3</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>fms_attic</td>
<td>22</td>
<td>4</td>
<td>3</td>
<td>0.01</td>
<td>0.04</td>
</tr>
<tr>
<td>mesh2x2</td>
<td>32</td>
<td>5</td>
<td>4</td>
<td>0.01</td>
<td>0.03</td>
</tr>
<tr>
<td>mesh3x2</td>
<td>52</td>
<td>5</td>
<td>4</td>
<td>0.02</td>
<td>0.08</td>
</tr>
<tr>
<td>pingpong</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>RandCAS 2</td>
<td>110</td>
<td>8</td>
<td>7</td>
<td>0.08</td>
<td>0.44</td>
</tr>
<tr>
<td>Conditionals 2</td>
<td>214</td>
<td>26</td>
<td>25</td>
<td>1.39</td>
<td>5.79</td>
</tr>
<tr>
<td>Coverable instances</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>leabasicapproach</td>
<td>16</td>
<td>5</td>
<td>4</td>
<td>&lt;0.01</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>Dekker 1</td>
<td>41</td>
<td>27</td>
<td>25</td>
<td>2.08</td>
<td>3.23</td>
</tr>
<tr>
<td>DoubleLock1 1</td>
<td>64</td>
<td>35</td>
<td>32</td>
<td>11.26</td>
<td>13.31</td>
</tr>
<tr>
<td>Pthread5 1</td>
<td>80</td>
<td>47</td>
<td>44</td>
<td>97.28</td>
<td>Timeout</td>
</tr>
<tr>
<td>RandLock0 2</td>
<td>110</td>
<td>48</td>
<td>46</td>
<td>21.40</td>
<td>24.89</td>
</tr>
<tr>
<td>Spin2003 2</td>
<td>56</td>
<td>38</td>
<td>35</td>
<td>67.35</td>
<td>Timeout</td>
</tr>
<tr>
<td>Szymanski 1</td>
<td>61</td>
<td>46</td>
<td>44</td>
<td>19.62</td>
<td>32.69</td>
</tr>
<tr>
<td>Constants 1</td>
<td>26</td>
<td>14</td>
<td>13</td>
<td>0.03</td>
<td>0.03</td>
</tr>
<tr>
<td>FuncPtr3 1</td>
<td>40</td>
<td>16</td>
<td>13</td>
<td>0.19</td>
<td>0.33</td>
</tr>
<tr>
<td>Benchmark</td>
<td>Places</td>
<td>IC3+PMA AbsPlaces</td>
<td>IC3+PMA Ref</td>
<td>IC3+PMA time(s)</td>
<td>IC3 time(s)</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>--------</td>
<td>-------------------</td>
<td>-------------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>Uncoverable instances</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peterson</td>
<td>14</td>
<td>10</td>
<td>8</td>
<td>0.35</td>
<td>0.13</td>
</tr>
<tr>
<td>Lamport</td>
<td>11</td>
<td>7</td>
<td>6</td>
<td>0.06</td>
<td>0.02</td>
</tr>
<tr>
<td>Ext. ReadWrite (small consts)</td>
<td>24</td>
<td>14</td>
<td>13</td>
<td>1.23</td>
<td>0.28</td>
</tr>
<tr>
<td>x0_AA_q1</td>
<td>312</td>
<td>#</td>
<td>#</td>
<td>Timeout</td>
<td>70.28</td>
</tr>
<tr>
<td>csm</td>
<td>14</td>
<td>9</td>
<td>8</td>
<td>0.19</td>
<td>0.02</td>
</tr>
<tr>
<td><strong>Coverable instances</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RandCAS 1</td>
<td>48</td>
<td>34</td>
<td>33</td>
<td>0.85</td>
<td>0.67</td>
</tr>
<tr>
<td>StackCAS0 1</td>
<td>41</td>
<td>30</td>
<td>29</td>
<td>3.72</td>
<td>2.14</td>
</tr>
<tr>
<td>StackLock0 1</td>
<td>37</td>
<td>26</td>
<td>25</td>
<td>2.33</td>
<td>1.06</td>
</tr>
<tr>
<td>Lu-fig2 1</td>
<td>39</td>
<td>20</td>
<td>19</td>
<td>0.22</td>
<td>0.12</td>
</tr>
<tr>
<td>Lu-fig2 2</td>
<td>61</td>
<td>35</td>
<td>32</td>
<td>43.06</td>
<td>9.05</td>
</tr>
</tbody>
</table>

IC3+PMA performs worse
Experiments

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Places</th>
<th>IC3+PMA AbsPlaces</th>
<th>IC3+PMA Ref</th>
<th>IC3+PMA time(s)</th>
<th>IC3 time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncoverable instances</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peterson</td>
<td>14</td>
<td>10</td>
<td>8</td>
<td>0.35</td>
<td>0.13</td>
</tr>
<tr>
<td>Lamport</td>
<td>11</td>
<td>7</td>
<td>6</td>
<td>0.06</td>
<td>0.02</td>
</tr>
<tr>
<td>Ext. ReadWrite (small configs)</td>
<td>24</td>
<td>14</td>
<td>13</td>
<td>1.23</td>
<td>0.28</td>
</tr>
<tr>
<td>x0_AA_q1</td>
<td>312</td>
<td>#</td>
<td>#</td>
<td>Timeout</td>
<td>70.28</td>
</tr>
<tr>
<td>csm</td>
<td>14</td>
<td>9</td>
<td>8</td>
<td>0.19</td>
<td>0.02</td>
</tr>
<tr>
<td>Coverable instances</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RandCAS 1</td>
<td>48</td>
<td>34</td>
<td>33</td>
<td>0.85</td>
<td>0.67</td>
</tr>
<tr>
<td>StackCAS0 1</td>
<td>41</td>
<td>30</td>
<td>29</td>
<td>3.72</td>
<td>2.14</td>
</tr>
<tr>
<td>StackLock0 1</td>
<td>37</td>
<td>26</td>
<td>25</td>
<td>2.33</td>
<td>1.06</td>
</tr>
<tr>
<td>Lu-fig2 1</td>
<td>39</td>
<td>20</td>
<td>19</td>
<td>0.22</td>
<td>0.12</td>
</tr>
<tr>
<td>Lu-fig2 2</td>
<td>61</td>
<td>35</td>
<td>32</td>
<td>43.06</td>
<td>9.05</td>
</tr>
</tbody>
</table>

- the efficiency of refinement method is not so high
- the way to deal with frames after refinement is not efficient
future work

- optimize the implementation to achieve better results
- apply the approach to analyze more properties and models
Thank You For Your Attention