An SMT-based Approach to Fair Termination Analysis

Javier Esparza, Philipp J. Meyer

Technische Universität München
Fair Termination Analysis

- Fair termination: No non-fair infinite execution sequence $\sigma$.
- PSPACE-complete for boolean programs.
Fair Termination Analysis

- Fair termination: No non-fair infinite execution sequence $\sigma$.
- PSPACE-complete for boolean programs.

SMT-Based Approach

- Incomplete method based on reduction to feasibility of linear arithmetic constraints.
- Strengthened with refinement cycle which adds mixed linear and boolean constraints.
- Similar method previously applied for safety properties (An SMT-based Approach to Coverability Analysis, CAV14).
Lamport’s 1-bit Algorithm for Mutual Exclusion

```plaintext
procedure Process 1
begin
    b_1 := 0
    while true do
        b_1 := 1
        while b_2 = 1 do skip od
        (⋆ critical section ⋆)
        b_1 := 0
    od
end

procedure Process 2
begin
    b_2 := 0
    while true do
        b_2 := 1
        if b_1 = 1 then
            b_2 := 0
            while b_1 = 1 do skip od
            goto q_1
        fi
        q_5: (⋆ critical section ⋆)
        b_2 := 0
    od
end
```
Communicating Automata Model

Property: If both processes are executed infinitely often, then the first process should enter the critical section ($p_3$) infinitely often.
Property: If both processes are executed infinitely often, then the first process should enter the critical section ($p_3$) infinitely often.
Property: If both processes are executed infinitely often, then the first process should enter the critical section \((p_3)\) infinitely often.
Property: For every infinite transition sequence $\sigma$, we have
\[
\varphi(\sigma) = \bigvee_{i=1}^{4} (s_i \in \text{inf}(\sigma)) \land \bigvee_{i=1}^{7} (t_i \in \text{inf}(\sigma)) \implies s_2 \in \text{inf}(\sigma).
\]
Loop Sequences

\[
\{p_1, nb_1, nb_2, q_1\} \xrightarrow{t_1 t_6 t_7 s_1 t_1 t_2 t_3 s_2 t_5 s_3 t_4} \{p_1, nb_1, nb_2, q_1\}
\]
Loop Sequences

\[
\{p_1, nb_1, nb_2, q_1\} \xrightarrow{t_1 t_6 t_7 s_1 t_1 t_2 t_3 s_2 t_5 s_3 t_4} \{p_1, nb_1, nb_2, q_1\}
\]

\[
\#\sigma = (\quad)
\]
Loop Sequences

\[
\{p_1, nb_1, nb_2, q_1\} \xrightarrow{t_1t_6t_7s_1t_1t_2t_3s_2t_5s_3t_4} \{p_1, nb_1, nb_2, q_1\}
\]

\[
\#\sigma = \begin{pmatrix}
\#t_1 & \#t_2 & \#t_3 & \#t_4 & \#t_5 & \#t_6 & \#t_7 & \#s_1 & \#s_2 & \#s_3 & \#s_4 \\
2 & & & & & & & & & & \\
\end{pmatrix}
\]
Loop Sequences

\[
\{p_1, nb_1, nb_2, q_1\} \xrightarrow{t_1t_6t_7s_1t_1t_2t_3s_2t_5s_3t_4} \{p_1, nb_1, nb_2, q_1\}
\]

\[
\#\sigma = \begin{pmatrix}
2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 
\end{pmatrix}
\]
Loop Sequences

\[ \{p_1, nb_1, nb_2, q_1\} \xrightarrow{t_1t_6t_7s_1t_1t_2t_3s_2t_5s_3t_4} \{p_1, nb_1, nb_2, q_1\} \]

\[ \#\sigma = (2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0) \]
Necessary Condition for Loops

\[ X = \begin{pmatrix} \#t_1 & \#t_2 & \#t_3 & \#t_4 & \#t_5 & \#t_6 & \#t_7 & \#s_1 & \#s_2 & \#s_3 & \#s_4 \\ t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & s_1 & s_2 & s_3 & s_4 \end{pmatrix} \]
Necessary Condition for Loops

\[ X = \begin{pmatrix} \#t_1 & \#t_2 & \#t_3 & \#t_4 & \#t_5 & \#t_6 & \#t_7 & \#s_1 & \#s_2 & \#s_3 & \#s_4 \\ t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & s_1 & s_2 & s_3 & s_4 \end{pmatrix} \]
Necessary Condition for Loops

\[ X = (\begin{array}{c}
\#t_1 & \#t_2 & \#t_3 & \#t_4 & \#t_5 & \#t_6 & \#t_7 & \#s_1 & \#s_2 & \#s_3 & \#s_4
\end{array}) \]

\[ q_1 : \quad t_4 + t_7 = t_1 \]
Necessary Condition for Loops

\[ X = \begin{pmatrix} \#t_1 & \#t_2 & \#t_3 & \#t_4 & \#t_5 & \#t_6 & \#t_7 & \#s_1 & \#s_2 & \#s_3 & \#s_4 \\ t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & s_1 & s_2 & s_3 & s_4 \end{pmatrix} \]

\[ q_1 : \quad t_4 + t_7 = t_1 \]
Necessary Condition for Loops

\[ X = \begin{pmatrix} \#t_1 & \#t_2 & \#t_3 & \#t_4 & \#t_5 & \#t_6 & \#t_7 & \#s_1 & \#s_2 & \#s_3 & \#s_4 \\ t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & s_1 & s_2 & s_3 & s_4 \end{pmatrix} \]

\[ q_1 : \quad t_4 + t_7 = t_1 \]

\[ q_2 : \quad t_1 = t_2 + t_6 \]
Necessary Condition for Loops

\[ X = \begin{pmatrix} #t_1 & #t_2 & #t_3 & #t_4 & #t_5 & #t_6 & #t_7 & #s_1 & #s_2 & #s_3 & #s_4 \\ t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & s_1 & s_2 & s_3 & s_4 \end{pmatrix} \]

\[ q_1 : \quad t_4 + t_7 = t_1 \]
\[ q_2 : \quad t_1 = t_2 + t_6 \]
\[ q_3 : \quad t_2 = t_3 \]
\[ q_4 : \quad t_3 = t_4 \]
\[ q_5 : \quad t_6 = t_7 \]
Necessary Condition for Loops

\[ p_1 : s_3 = s_1 \]
\[ p_2 : s_1 = s_2 \]
\[ p_3 : s_2 = s_3 \]

\[ b_2 : t_1 = t_3 + t_7 \]
\[ nb_2 : t_3 + t_7 = s_1 \]

\[ b_1 : s_1 = s_3 \]
\[ nb_1 : s_3 = s_1 \]

\[ q_1 : t_4 + t_7 = t_1 \]
\[ q_2 : t_1 = t_2 + t_6 \]
\[ q_3 : t_2 = t_3 \]
\[ q_4 : t_3 = t_4 \]
\[ q_5 : t_6 = t_7 \]
Termination Constraints

- Accumulate constraints in matrix form as $C \cdot X = 0$.
- If there is an infinite transition sequence $\sigma$, then the following constraints have a solution $X$:

$$C \::\:: \begin{cases} C \cdot X = 0 \\ X \geq 0 \\ X \neq 0 \end{cases}$$

- If the constraints have no solution, then the program is terminating.
- A solution $X$ is realizable if there is a sequence $\sigma$ with $\#\sigma = X$. 
Fair Termination Constraints

- Fairness condition given by boolean formula $\varphi$ over $t \in \inf(\sigma)$.
- If the program is not fairly terminating, then there is an infinite transition sequence $\sigma$ satisfying $\sigma \models \neg \varphi$.
- Add constraint $\neg \varphi(X)$ to $C$ for fair termination constraints.

Fairness for Lamport’s Algorithm

$$\varphi(\sigma) = \bigvee_{i=1}^{4} (s_i \in \inf(\sigma)) \land \bigvee_{i=1}^{7} (t_i \in \inf(\sigma)) \implies s_2 \in \inf(\sigma)$$

$$\neg \varphi(X) = (s_1 + s_2 + s_3 + s_4 > 0) \land (t_1 + t_3 + t_4 + t_5 + t_6 + t_7 > 0) \land (s_2 = 0)$$
Fair Termination Constraints

\[
\begin{align*}
    s_3 &= s_1 & t_4 + t_7 &= t_1 & s_1 &\geq 0 & t_1 &\geq 0 \\
    s_1 &= s_2 & t_1 &= t_2 + t_6 & s_2 &\geq 0 & t_2 &\geq 0 \\
    s_2 &= s_3 & t_2 &= t_3 & s_3 &\geq 0 & t_3 &\geq 0 \\
    & & t_3 &= t_4 & & t_4 &\geq 0 \\
    & & t_6 &= t_7 & & t_5 &\geq 0 \\
    s_1 &= s_3 & t_1 &= t_3 + t_7 & & t_6 &\geq 0 \\
    s_3 &= s_1 & t_3 + t_7 &= s_1 & & t_7 &\geq 0 \\

    s_1 + s_2 + s_3 + s_4 + t_1 + t_3 + t_4 + t_5 + t_6 + t_7 &> 0 \\

    (s_1 + s_2 + s_3 + s_4 > 0) &\land \\
    (t_1 + t_3 + t_4 + t_5 + t_6 + t_7 > 0) &\land \\
    (s_2 = 0)
\end{align*}
\]
Fair Termination Constraints: Solution

\[
X = \begin{pmatrix} t_1 & t_2 & t_3 & t_4 & t_5 & t_6 & t_7 & s_1 & s_2 & s_3 & s_4 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}
\]

\[
s_3 = s_1 \quad t_4 + t_7 = t_1 \quad s_1 \geq 0 \quad t_1 \geq 0
\]

\[
s_1 = s_2 \quad t_1 = t_2 + t_6 \quad s_2 \geq 0 \quad t_2 \geq 0
\]

\[
s_2 = s_3 \quad t_2 = t_3 \quad s_3 \geq 0 \quad t_3 \geq 0
\]

\[
t_3 = t_4
\]

\[
t_6 = t_7
\]

\[
s_1 = s_3 \quad t_1 = t_3 + t_7
\]

\[
s_3 = s_1 \quad t_3 + t_7 = s_1 \quad t_7 \geq 0
\]

\[
s_1 + s_2 + s_3 + s_4 + t_1 + t_3 + t_4 + t_5 + t_6 + t_7 > 0
\]

\[
(s_1 + s_2 + s_3 + s_4 > 0) \land
\]

\[
(t_1 + t_3 + t_4 + t_5 + t_6 + t_7 > 0) \land
\]

\[
(s_2 = 0)
\]
Fair Termination Constraints: Solution

\[
X = \begin{pmatrix}
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\]
Solution realizable?

$X$ realized by $\sigma$ with $\text{inf}(\sigma) = \{s_4, t_5\}$. 
Solution realizable?

$X$ realized by $\sigma$ with $\text{inf}(\sigma) = \{s_4, t_5\}$. 
Refinement Component

$q_1$, $q_4$ and $b_2$ are in mutual exclusion.
Refinement Component

$q_1$, $q_4$ and $b_2$ are in mutual exclusion.
Refinement Component

$q_1, q_4$ and $b_2$ are in mutual exclusion.
Refinement Component

$q_1, q_4$ and $b_2$ are in mutual exclusion.
Refinement Constraint

$X$ realized by $\sigma$ with $\inf(\sigma) = \{s_4, t_5\}$. 

\[
\begin{align*}
&D:
\end{align*}
\]
Refinement Constraint

\( X \) not realizable \( \Rightarrow \) Generate refinement constraint \( \delta \).
Refinement Constraint

$$\delta = (s_4 = 0) \lor (t_5 = 0) \lor (t_1 + t_3 + t_4 + t_7 > 0)$$
Refinement Loop

\[ C \text{ sat?} \]
Refinement Loop

$C$ sat? \(\rightarrow\) unsat \(\rightarrow\) terminating
Refinement Loop

\[ \mathcal{C} \text{ sat?} \]

- unsat \[ \rightarrow \text{terminating} \]

- sat \[ \rightarrow \text{Obtain solution } X. \]
Refinement Loop

\[ \mathcal{C} \text{ sat?} \quad \text{unsat} \quad \rightarrow \quad \text{terminating} \]

Obtain solution \( X \).

Refinement component to discard \( X \)?
Refinement Loop

\[ C \text{ sat?} \quad \text{unsat} \quad \text{terminating} \]

Obtain solution \( X \).

inconclusive \[ \text{no} \quad \text{Refinement component to discard } X? \]

Refinement Loop
Refinement Loop

Obtain solution $X$.

$C$ sat? 

- unsat -> terminating
- sat

Refinement component to discard $X$?

- yes
- inconclusive
- no

Generate refinement constraint $\delta$. 

- yes
- no
Refinement Loop

\( \mathcal{C} := \mathcal{C} \cup \{\delta\} \)

\( \mathcal{C} \) sat?

Obtain solution \( X \).

Generate refinement constraint \( \delta \).

Refinement component to discard \( X \)?

yes

inconclusive

no
Experimental Evaluation

Benchmarks

- IBM/SAP — Workflow nets from business process models
  - 1976 examples
  - 1836 terminating

- Erlang — Models from the verification of Erlang programs
  - 50 examples, up to 66950 places and 213626 transitions
  - 33 terminating

- Literature — Selected examples from the literature
  - 5 examples, with unbounded variables
  - All terminating

- Classical — Classic asynchronous programs for mutual exclusion and distributed algorithms
  - 5 examples, scalable in number of processes
  - All fairly terminating
Rate of Success

terminating

IBM
SAP
Erlang
Literature
Classical
Total

1264
572
33
5
5
1879
Rate of Success

<table>
<thead>
<tr>
<th>Category</th>
<th>Terminating</th>
<th>Without Refinement</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM</td>
<td>1263</td>
<td>1264</td>
</tr>
<tr>
<td>SAP</td>
<td>571</td>
<td>572</td>
</tr>
<tr>
<td>Erlang</td>
<td>27</td>
<td>33</td>
</tr>
<tr>
<td>Literature</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Classical</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>Total</td>
<td>1861</td>
<td>1879</td>
</tr>
</tbody>
</table>
Rate of Success

- **terminating**
- **w/o refinement**
- **with refinement**

<table>
<thead>
<tr>
<th></th>
<th>IBM</th>
<th>SAP</th>
<th>Erlang</th>
<th>Literature</th>
<th>Classical</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1264</td>
<td>572</td>
<td>33</td>
<td>5</td>
<td>5</td>
<td>1879</td>
</tr>
<tr>
<td></td>
<td>1263</td>
<td>571</td>
<td>27</td>
<td>0</td>
<td>0</td>
<td>1873</td>
</tr>
<tr>
<td></td>
<td>27</td>
<td>27</td>
<td>0</td>
<td>0</td>
<td></td>
<td>1861</td>
</tr>
</tbody>
</table>

IBM, SAP, Erlang, Literature, Classical, Total
Performance on Positive Examples

![Graph showing performance on positive examples with execution time (s) on the y-axis and number of places on the x-axis. The graph includes a time limit marker.](image-url)
Performance on Positive Examples

![Graph showing execution time vs. number of places]

- Execution time (s)
- Number of places

- Time limit: 8 minutes
- 11 seconds
- 3 seconds

Logarithmic scale from $10^{-2}$ to $10^5$
Performance on Negative Examples

![Graph showing execution time (s) vs. number of places. The x-axis represents the number of places on a logarithmic scale from $10^0$ to $10^5$, and the y-axis represents execution time (s) on a logarithmic scale from $10^{-2}$ to $10^4$. The time limit is marked as a horizontal line at $10^4$. The data points are scattered across the graph, indicating a trend of increasing execution time with the number of places.](image-url)
Performance on Negative Examples

![Graph showing execution time (s) vs. number of places](image)

- Execution time (s) on a log-log scale.
- Number of places also on a log-log scale.
- Red line indicating a time limit of 5 seconds.
- Data points indicating 1000 places.
Refinement Steps

![Graph showing the relationship between Refinement steps and Number of places. The x-axis represents the Number of places on a logarithmic scale from $10^0$ to $10^5$, and the y-axis represents Refinement steps on a logarithmic scale from $1$ to $100$. The graph includes data points at $4$, $27$, and $320$.](image-url)
Comparison with SPIN on Scaled Classical Suite

![Graph showing comparison between SPIN and Petrinizer](image)

- **SPIN (s)** vs **Petrinizer (s)**
- **Leader Election**
- **Snapshot**
- **Lamport**
- **Peterson**
- **Szymanski**

*Note: Some data points exceed the time limit or run out of memory.*
Summary

- Fast and effective technique for proving fair termination
- Incomplete, but high degree of completeness
- Large instances can be handled
- Constraints can be used as a certificate of fair termination