Verification of Dynamic Register Automata

Parosh Aziz Abdulla
Mohammed Faouzi Atig
Ahmet Kara
Othmane Rezine
Verification of a class of protocols:

- **Dynamic creation** of processes
  - **Unique** ID for each process
- Finite number of **registers** per process
  - Used to store the ID of others
- **Point-to-point** communication
- **Rendez-vous** communication
Dynamic Register Automata

[Bollig et al. 2010, 2013]
Application of Dynamic Register Automata

- Leader Election Protocol
- Peer To Peer Protocol
- Ad-Hoc Networks
Dynamic Register Automata

Verification of Dynamic Register Automata

- Applications
- Formal Model
Dynamic Register Automata

Verification of Dynamic Register Automata

Formal Model

Process Model

\[ x \leftarrow \text{create}(q_3, y) \]

q1 \[\rightarrow\] q2

q3 \[\rightarrow\] q4

y?\langle x \rangle

y!\langle m \rangle

x?\langle m \rangle

z!\langle x \rangle

reset\langle x \rangle
Dynamic Register Automata

Finite number of States

Formal Model

Process Model

Verification of Dynamic Register Automata

Finite number of States

$q_1 \leftarrow \text{create}(q_3, y)$

$q_1 \rightarrow y!<m>$

$q_2 \rightarrow y!<x>$

$q_3 \rightarrow z?<x>$

$q_3 \leftarrow x?<m>$

$q_4 \rightarrow \text{reset}<x>$
Dynamic Register Automata

Formal Model

Process Model

Finite number of States

Finite number of Registers
Dynamic Register Automata

Finite number of States

Finite number of Registers

Finite message alphabet

Formal Model

Process Model

$q_1 \xleftarrow{\text{create}(q_3, y)} q_3$

$q_1 \xrightarrow{y!<m>} q_2$

$q_2 \xrightarrow{y!<x>} q_1$

$q_3 \xrightarrow{x?<m>} q_4$

$q_3 \xrightarrow{\text{reset}<x>} q_4$

$q_4 \xrightarrow{z?<x>} q_2$

$q_4 \xrightarrow{\text{reset}<x>} q_3$
Dynamic Register Automata

**Formal Model**
- Finite number of States
- Finite number of Registers
- Finite message alphabet

**Transitions:**
- Create Process

**Process Model**

- $x \leftarrow \text{create}(q_3, y)$
Dynamic Register Automata

Finite number of States
Finite number of Registers
Finite message alphabet

Transitions:
Create Process
Send Message

Formal Model

Process Model

\[ x \leftarrow \text{create}(q_3, y) \]

\[ y!<m> \]

\[ y!<x> \]

\[ x?<m> \]

\[ z?<x> \]

\[ \text{reset}<x> \]
Dynamic Register Automata

Formal Model
- Finite number of States
- Finite number of Registers
- Finite message alphabet

Transitions:
- Create Process
- Send Message
- Receive Message

Process Model

Verification of Dynamic Register Automata

Finite number of States
Finite number of Registers
Finite message alphabet
Dynamic Register Automata

Finite number of States
Finite number of Registers
Finite message alphabet

Transitions:
Create Process
Send ID

Formal Model

Process Model

\[
q_1 \xrightarrow{x \leftarrow \text{create}(q_3, y)} q_2 \\
q_2 \xrightarrow{y!<m>} q_3 \\
q_3 \xrightarrow{z?<x>} q_4 \\
q_4 \xrightarrow{\text{reset}<x>} q_1 \\
\]

\[
q_1 \xrightarrow{y!<x>} q_2 \\
q_2 \xrightarrow{x?<m>} q_3 \\
q_3 \xrightarrow{z?<x>} q_4 \\
q_4 \xrightarrow{\text{reset}<x>} q_1 \\
\]
**Dynamic Register Automata**

**Formal Model**
- Finite number of States
- Finite number of Registers
- Finite message alphabet

**Process Model**

Transitions:
- Create Process
- Send ID
- Receive ID

```
x ← create(q3, y)

y!<m>

y!<x>

z?<x>

x?<m>

reset<x>
```
Dynamic Register Automata

Formal Model

Finite number of States
Finite number of Registers
Finite message alphabet

Transitions:
Create Process
Send Msg / ID
Receive Msg / ID
Register Reset

Process Model

 Verification of Dynamic Register Automata

Finite message alphabet
Formal Model
Dynamic Register Automata

**Formal Model**
- Finite number of States
- Finite number of Registers
- Finite message alphabet

**Transitions:**
- Create Process
- Send Msg / ID
- Receive Msg / ID
- Register Reset

**Process Model**
- States: q1, q2, q3, q4
- Transitions:
  - $x \leftarrow \text{create}(q_3, y)$
  - $y!<m>$
  - $y!<x>$
  - $z?<x>$
  - $x?<m>$
  - $\text{reset}<x>$
Dynamic Register Automata

Formal Model

Process Model

\[ x \leftarrow \text{create}(q_3, y) \]
Dynamic Register Automata

Verification of Dynamic Register Automata

Formal Model

Process Model

Finite State System

\[ x \leftrightarrow \text{create}(q_3, y) \]

\[ q_1 \quad q_2 \quad q_3 \quad q_4 \]

\[ y!<m> \quad y!<x> \quad x?<m> \quad z?<x> \quad \text{reset}<x> \]
Dynamic Register Automata

Verification of Dynamic Register Automata

Formal Model

Configuration
Network of Processes

Process Model
Finite State System

$q_1 \xleftrightarrow{\text{create}(q_3, y)} q_2$

$q_1 \xrightarrow{y!<m>} q_2$
$q_2 \xrightarrow{z?<x>} q_4$
$q_4 \xrightarrow{\text{reset}<x>} q_3$
$q_3 \xrightarrow{x!<m>} q_1$
Dynamic Register Automata

Formal Model

Configuration
Network of Processes

Process Model
Finite State System

\[
\begin{align*}
x & \leftrightarrow \text{create}(q_3, y) \\
q_1 & \quad y!langle m \rangle \\
y!langle x \rangle & \quad x?langle m \rangle \\
z?langle x \rangle & \quad \text{reset}langle x \rangle \\
q_2 & \\
q_3 & \quad x?langle m \rangle \\
q_4 & \\
\end{align*}
\]
**Dynamic Register Automata**

**Formal Model**

**Configuration**

- **q3**

**Control States**

**Process Model**

- **q1**
- **q2**
- **q3**
- **q4**

- \( x \leftarrow \text{create}(q_3, y) \)
- \( y! < m > \)
- \( y! < x > \)
- \( x? < m > \)
- \( z? < x > \)
- \( \text{reset} < x > \)
Dynamic Register Automata

Verification of Dynamic Register Automata

Formal Model

Configuration

Process Model

Unique IDs

$\text{create}(q_3, y)$

$q_1 \xleftarrow{x} q_2$

$q_3 \xrightarrow{y!<m>} q_2$

$q_3 \xrightarrow{y!<x>} q_2$

$q_3 \xrightarrow{x?<m>} q_2$

$q_3 \xleftarrow{z?<x>} q_2$

$q_3 \xrightarrow{\text{reset}<x>} q_2$
Dynamic Register Automata

Verification of Dynamic Register Automata

Formal Model

Configuration

Process Model

Registers

q3

x: 7
y: 0
z: ⊥

q1

x: ⊥
y: ⊥
z: 7

q2

y!<m>
x?<m>
z?<x>

q3

y!<x>

q4

reset<x>

x ↔ create(q3, y)
Dynamic Register Automata

Formal Model

Verification of Dynamic Register Automata

Configuration

Process Model

Registers

\[ q_1 \]

\[ q_3 \]

\[ q_3 \]

\[ q_3 \]

\[ y \leftarrow \text{create} (q_3, y) \]

\[ y! < m > \]

\[ y! < x > \]

\[ x? < m > \]

\[ z? < x > \]

\[ \text{reset} < x > \]
Dynamic Register Automata

Formal Model

Configuration

Process Model

Verification of Dynamic Register Automata
Dynamic Register Automata

Formal Model

Verification of Dynamic Register Automata

Configuration

Communication graph

Process Model

Formal Model

- **Process Model**
  - Configuration
  - Create(q3, y)
  - q1
  - q2
  - q3
  - q4
  - y!<m>
  - y!<x>
  - x?<m>
  - z?<x>
  - q1
  - q2
  - q3
  - q4
  - reset<x>

- **Communication graph**
  - x ↔ create(q3, y)
Dynamic Register Automata

Formal Model

Configuration Communication graph

Process Model

Verication of Dynamic Register Automata

\( q_1 \)

\( x \leftarrow \text{create}(q_3, y) \)

\( q_2 \)

\( y! <m> \)

\( q_3 \)

\( y! <x> \)

\( x? <m> \)

\( z? <x> \)

\( q_4 \)

\( \text{reset}<x> \)
Dynamic Register Automata

Formal Model

Configuration

Communication graph

Process Model

Verification of Dynamic Register Automata

**Process Model**

- **q1**
- **q2**
- **q3**
- **q4**

**Communication Graph**

- **x** \(\leftrightarrow\) **create** \((q3, y)\)
- **y!\langle m\rangle**
- **y!\langle x\rangle**
- **x?\langle m\rangle**
- **z?\langle x\rangle**
- **reset\langle x\rangle**
Dynamic Register Automata

Formal Model

Configuration
Communication graph

Process Model

Verification of Dynamic Register Automata

 Formal Model

Configuration
Communication graph

Process Model

Verification of Dynamic Register Automata
Dynamic Register Automata

### Formal Model

#### Configuration
- Communication graph

#### Process Model
- $q_1 \leftrightarrow \text{create}(q_3, y)$
- $y! < m$
- $x? < m$
- $z? < x$
- $x \leftrightarrow \text{reset}(x)$

### 30th Anniversary of Dynamic Register Automata
Dynamic Register Automata

Formal Model

Verification of Dynamic Register Automata

Configuration

Communication graph

Process Model

Formal Model
Dynamic Register Automata

Formal Model

Configuration
Communication graph

Process Model

Verification of Dynamic Register Automata
Dynamic Register Automata

Formal Model

Configuration

Process Model

Verification of Dynamic Register Automata

Communication graph
Dynamic Register Automata

Formal Model

Configuration

Transition Relation

Process Model

$q_1$ $q_3$

$z$

$y$

$x$ $create(q_3, y)$ $y!\langle m \rangle$

$x?\langle m \rangle$

$z?\langle x \rangle$

reset$\langle x \rangle$

$q_2$

$q_3$

$q_1$

$q_4$
Dynamic Register Automata

Verification of Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Process Creation

\[ \text{create}(q_3, y) \]
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Process Creation

Creating Process

\[ \text{create}(q_3, y) \]
Dynamic Register Automata

Formal Model

Verification of Dynamic Register Automata

Configuration

Transition Relation

Process Model

Process Creation

Creating Process

- Transition Relation
- Configuration
- Process Model
- Process Creation
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Process Creation

Transition Relation

Verification of Dynamic Register Automata

Created Process

Creating Process

x ↔ create(q3, y)

q1

y!<m>

y!<x>

x?<m>

z?<x>

reset<x>

q2

q3

q4
Transition Relation
Configuration
Transition Relation

Created Process

Formal Model

Process Model

Process Creation

Creating Process

Created Process

Dynamic Register Automata
Dynamic Register Automata

Formal Model

Verification of Dynamic Register Automata

Configuration
Transition Relation

Process Model

Process Creation

Creating Process

Created Process

Creating Process

x → create(q3, y)

q1
y!<m>
x!<x>
q2
y!<x>
z?<x>
q4
reset<x>

q3
x?<m>

q3
y
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Process Creation

\[ C_1 \]

- \( q_1 \) \( \rightarrow \) \( y \) \( \rightarrow \) \( q_3 \)

\[ C_2 \]

- \( q_2 \) \( \rightarrow \) \( x \) \( \rightarrow \) \( q_3 \)

\[ \xleftarrow{x} \text{create}(q_3, y) \]

\[ q_1 \leftarrow q_2 \]

- \( y!\langle m \rangle \rightarrow q_2 \)
- \( y!\langle x \rangle \rightarrow q_3 \)
- \( x?\langle m \rangle \rightarrow q_3 \)
- \( z?\langle x \rangle \rightarrow q_4 \)
- \( \text{reset}\langle x \rangle \rightarrow q_4 \)
Dynamic Register Automata

Formal Model

Configuration

Process Model

Transition Relation

Process Creation

Verification of Dynamic Register Automata

Create (q3, y)

x ← create (q3, y)

q1 → q2
y!<m>
y!<x>
x?<m>
z?<x>

q3 → q4
reset<x>
Dynamic Register Automata

Formal Model

Configuration

Process Model

Transition Relation

Process Creation

Send & Receive ID

- $\text{create}(q_3, y)$

Diagram:

- States: $q_1, q_2, q_3, q_4$
- Transitions:
  - $q_1 \xrightarrow{y!>m} q_2$
  - $q_2 \xrightarrow{x?>m} q_3$
  - $q_3 \xrightarrow{\text{reset}>x} q_4$
  - $q_4 \xrightarrow{z?>x} q_1$

Verification of Dynamic Register Automata
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Send & Receive ID

Verification of Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Send & Receive ID

Process Model

\[ C_2 \]

\[ q_3 \]

\[ y \]

\[ z \]

\[ q_3 \]

\[ x \]

\[ q_2 \]

\[ C_2 \]

\[ q_3 \]

\[ y \]

\[ z \]

\[ q_3 \]

\[ x \]

\[ q_2 \]

\[ x \leftarrow \text{create}(q_3, y) \]

\[ q_1 \]

\[ y!<m> \]

\[ q_2 \]

\[ y?<x> \]

\[ x?<m> \]

\[ q_3 \]

\[ z!<x> \]

\[ q_4 \]

\[ \text{reset}<x> \]
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Send & Receive ID

Process Model
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Send & Receive ID

Process Model

Verification of Dynamic Register Automata

$C_3$

$q_3$ $y$ $q_2$ $x$ $y$

Sender

$q_1$ $y!<$m$>$ $q_2$
$x?$<$x$>$

$q_3$ $x?$<$m$>$ $q_4$
$reset<$x$>$

To

$x$ $\leftarrow$ create($q_3$, $y$)

$q_2$ $z!<$x$>$
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Send & Receive ID

Verification of Dynamic Register Automata

\[ C_3 \]

\[
\begin{align*}
q_3 & \xrightarrow{y} q_2 \\
q_2 & \xrightarrow{x} q_3 \\
q_3 & \xrightarrow{y} q_2 \\
q_2 & \xrightarrow{y} q_3
\end{align*}
\]

To
Receiver

\[ x \leftarrow \text{create}(q_3, y) \]

Sender

\[
\begin{align*}
q_1 & \xrightarrow{y} q_2 \\
q_2 & \xrightarrow{x} q_3 \\
q_3 & \xrightarrow{z} q_4 \\
q_4 & \xrightarrow{\text{reset}} q_1
\end{align*}
\]

\[ y! <m> \]

\[ x? <m> \]

\[ z! <x> \]

\[ \text{reset} <x> \]
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Send & Receive ID

Process Model

Verification of Dynamic Register Automata

C_3

q3

y

z

q3

q2

x

Sender

To

Receiver

x ← create(q3, y)

q1

y!<m>

q2

y?<x>

x?<m>

z!<x>

q3

Reset<x>

q4
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Send & Receive ID

Process Model

Verification of Dynamic Register Automata
Dynamic Register Automata

Verification of Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Send & Receive ID

C

q3

y

z

q3

y

q3

x

q2

q2

Sender

To

Receiver

From

x \leftarrow \text{create}(q3, y)

y!<m>

reset<x>

x?<m>

z!<x>

y?<x>

q1

q2

q3

q4
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Send & Receive ID

Point-To-Point, Rendez-Vous communication

Sender

Receiver

To

From

$x \leftarrow \text{create}(q_3, y)$

$q_1$

$q_2$

$q_3$

$q_4$

$y!\langle m \rangle$

$x?\langle m \rangle$

$z!\langle x \rangle$

reset$\langle x \rangle$
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Send & Receive ID

What
Sender

$q_3$

$y$

$z$

$x$

$q_2$

$q_1$

$q_3$

$q_4$

$create(q_3, y)$

$y!<$m$>

$x?<$m$>

$reset<$x$>

$z!<$x$>
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Send & Receive ID

Where to?
Receiver

What
Sender

\[ C_3 \]

\[ q_3 \]

\[ x \]

\[ y \]

\[ z \]

\[ q_2 \]

\[ q_3 \]

\[ x \]

\[ y \]

\[ z \]

\[ q_3 \]

\[ q_4 \]

\[ q_1 \]

\[ q_2 \]

\[ q_3 \]

\[ q_4 \]

\[ x \leftarrow \text{create}(q_3, y) \]

\[ y!\langle m \rangle \]

\[ x?\langle m \rangle \]

\[ z!\langle x \rangle \]

\[ \text{reset}\langle x \rangle \]
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Send & Receive ID

\[ C_3 \]

- \( q_3 \)
- \( q_4 \)

Transition Relation:

- \( x \rightarrow \text{create}(q_3, y) \)
- \( y \rightarrow y! <m> \rightarrow q_2 \)
- \( x \rightarrow x? <m> \rightarrow q_3 \)
- \( z \rightarrow z! <x> \rightarrow q_4 \)
- \( y \rightarrow y? <x> \rightarrow q_1 \)
- \( q_1 \rightarrow q_2 \)
- \( q_2 \rightarrow q_3 \)
- \( q_3 \rightarrow q_4 \)
- \( q_4 \rightarrow q_3 \)
- \( q_4 \rightarrow \text{reset}<x> \rightarrow q_3 \)
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Send & Receive ID

Process Model

Verification of Dynamic Register Automata

C3

C4

q3

y

z

q3

q1

q2

x

y

z

q4

q3

y

z

q3

q1

x

y

z

q4

q3

y

z

q3

q4

x

y

z

q4

x

create(q3, y)

y!<m>

reset<x>

z!<x>

x?<m>

x?

y?<x>

q3

q2

q1
Dynamic Register Automata

Formal Model

Configuration

Process Model

Transition Relation

Process Creation

Send & Receive ID

\[
\text{create}(q_3, y)
\]

\[
\begin{align*}
q_1 & \xrightarrow{y?x} q_2 \\
q_2 & \xrightarrow{x?m} q_3 \\
q_3 & \xrightarrow{z!x} q_4 \\
q_4 & \xrightarrow{\text{reset}x} q_1
\end{align*}
\]
Dynamic Register Automata

Formal Model

Configuration

Process Model

Transition Relation

Process Creation

Send & Receive ID

\[
\text{create}(q_3, y)
\]
Verification of Dynamic Register Automata

Formal Model

Configuration

Process Model

Transition Relation

Process Creation

Send & Receive ID

Selective Receive

\[
\text{create}(q_3, y)
\]

\[
\begin{align*}
x & \leftarrow y!<m> \\
y?<x> & \leftarrow \text{reset}<x> \\
x?<m> & \leftarrow z!<x>
\end{align*}
\]
Dynamic Register Automata

Formal Model

Configuration

Process Model

Transition Relation

Process Creation

Send & Receive ID

Non Selective Receive

Verification of Dynamic Register Automata

Configuration

Process Creation

Send & Receive ID

Non Selective Receive

Graphical representation:

- **q1**
- **q2**
- **q3**
- **q4**

- Transition labeled with `create(q3, y)`
- Transition labeled with `y!<m>`
- Transition labeled with `z!<x>`
- Transition labeled with `reset<x>`
Dynamic Register Automata

Formal Model

Verification of Dynamic Register Automata

Configuration

Process Model

Transition Relation

Process Creation

Send & Receive ID

$q1 \leftarrow \text{create}(q3, y)$

$q2$

$q3$

$q4$

$x \leftarrow y!\langle m \rangle$

$y?\langle x \rangle$

$x?\langle m \rangle$

$z!\langle x \rangle$

reset$\langle x \rangle$
Dynamic Register Automata

Process Model

Formal Model

Configuration

Transition Relation

Process Creation

Send & Receive ID

Register Reset

Verification of Dynamic Register Automata

x ← create(q3, y)

q1

y!<m>

q2

y!<x>

x?<m>

q3

z?<x>

reset<x>

q4
Dynamic Register Automata

Formal Model

Configuration

Transition Relation

Process Model

Register Reset

\[ \text{create}(q_3, y) \]

\[ x \leftarrow \text{reset}<x> \]
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Register Reset

Verification of Dynamic Register Automata
Dynamic Register Automata

Verification of Dynamic Register Automata

Formal Model

Configuration
Process Model
Transition Relation

Register Reset

Which register
Dynamic Register Automata

Formal Model

Configuration
Transition Relation

Process Model

Register Reset

Verification of Dynamic Register Automata

*create*(q₃, y)
Dynamic Register Automata

Formal Model

Configuration

Transition Relation

C_3

C_4

Process Model

Register Reset

\( q_3 \xrightarrow{x} q_4 \)

\( q_4 \xrightarrow{y} q_1 \)

\( x \xleftarrow{} \text{create}(q_3, y) \)

\( y! \langle m \rangle \)

\( x? \langle m \rangle \)

\( z! \langle x \rangle \)

\( \text{reset} \langle x \rangle \)

\( q_1 \)

\( q_2 \)

\( q_3 \)

\( q_4 \)
Dynamic Register Automata

Verification of Dynamic Register Automata

Formal Model

Configuration

Process Model

Transition Relation

Process Creation

Send & Receive ID

Register Reset

![Diagram of Dynamic Register Automata]

- $x \leftarrow \text{create}(q_3, y)$
- $y!\langle m \rangle$
- $x?\langle m \rangle$
- $z?\langle x \rangle$
- $\text{reset}\langle x \rangle$
Dynamic Register Automata

Verification of Dynamic Register Automata

Applications

Formal Model

Process Model

Configuration

Transition Relation
Verification of Dynamic Register Automata

Applications

Formal Model

State Reachability
Dynamic Register Automata

State Reachability

Verification of Dynamic Register Automata

Initial Configuration

$C_0$

$q_1$
Dynamic Register Automata

State Reachability

Initial Configuration

Verification of Dynamic Register Automata

Initial Configuration: $C_0$

State Reachability: $q_1$

Run: $C_0$
Dynamic Register Automata

State Reachability
Initial Configuration

Run: $C_0$, $C_1$
Dynamic Register Automata

State Reachability

Initial Configuration

C₀

C₁

C₂

Run: C₀, C₁, C₂
Dynamic Register Automata

State Reachability
Initial Configuration

Verification of Dynamic Register Automata

Initial Configuration

$C_0$

Run: $C_0$, $C_1$, $C_2$, ...
Dynamic Register Automata

State Reachability

Initial Configuration

Verification of Dynamic Register Automata

Run: $C_0, C_1, C_2, \ldots, C_n$
Dynamic Register Automata

Verification of Dynamic Register Automata

State Reachability

Initial Configuration

Run: $C_0, C_1, C_2, \ldots, C_n$

State Reachability Problem:
Dynamic Register Automata

State Reachability

Initial Configuration

Run: \(C_0, C_1, C_2, \ldots, C_n\)

State Reachability Problem:

Given a control state \(q_{BAD}\), is there a run starting from the initial configuration that reaches a configuration where \(q_{BAD}\) occurs.
State Reachability

Initial Configuration
Run: $C_0, C_1, C_2, \ldots, C_n$

State Reachability Problem:
Given a control state $q_{BAD}$, is there a run starting from the initial configuration that reaches a configuration where $q_{BAD}$ occurs.
Dynamic Register Automata

**State Reachability**

**Verification of Dynamic Register Automata**

**Initial Configuration**

**Run:** $C_0, C_1, C_2, \ldots, C_n$

**State Reachability Problem:**

Given a control state $q_{BAD}$, is there a run starting from the initial configuration that reaches a configuration where $q_{BAD}$ occurs.

**Undecidable**
Dynamic Register Automata

Applications

Formal Model

State Reachability

Undecidable
Contribution

Verification of Dynamic Register Automata

State Reachability

Study of the decidability and complexity for different sub-classes of the problem
Contribution

State Reachability

Study of the decidability and complexity for different sub-classes of the problem

Bounded DRA

Undecidable
Contribution

State Reachability

Study of the decidability and complexity for different sub-classes of the problem

Bounded DRA

Undecidable

Strongly Bounded DRA

Undecidable
Contribution

Verification of Dynamic Register Automata

State Reachability

Study of the decidability and complexity for different sub-classes of the problem

Bounded DRA: Undecidable

Strongly Bounded DRA: Undecidable

Strongly Safe DRA: Decidable

Non-Primitive Recursive
Contribution

Verification of Dynamic Register Automata

State Reachability

Study of the decidability and complexity for different sub-classes of the problem

Bounded DRA: Undecidable

Strongly Bounded DRA: Undecidable

Strongly Safe DRA: Decidable

Non-Primitive Recursive
Bound the Simple Path Length of the configuration graph
Bounded DRA

Bound the Simple Path Length of the configuration graph
Bounded DRA

Bound the Simple Path Length of the configuration graph

k = 3

q3 → x → q1 → y → q2 → x → q3
Bounded DRA

Bound the Simple Path Length of the configuration graph
Bounded DRA

Bound the Simple Path Length of the configuration graph

$k = 1$
Bounded DRA

Bound the Simple Path Length of the configuration graph
Bounded DRA

Bound the Simple Path Length of the configuration graph
Bounded DRA

Verification of Dynamic Register Automata

Bound the Simple Path Length of the configuration graph

Using Rendez-Vous Communication
Bounded DRA

Verification of Dynamic Register Automata

Bound the Simple Path Length of the configuration graph

Using Rendez-Vous Communication
Bounded DRA

Verification of Dynamic Register Automata

Bound the Simple Path Length of the configuration graph

Using Rendez-Vous Communication
Bounded DRA

Bound the Simple Path Length of the configuration graph

Undecidable

k = 1
Verification of Dynamic Register Automata

Contribution

State Reachability

- Bounded DRA: Undecidable
- Strongly Bounded DRA: Undecidable
- Strongly Safe DRA: Decidable, Non-Primitive Recursive
Verification of Dynamic Register Automata

State Reachability

Bounded DRA: Undecidable

Strongly Bounded DRA: Undecidable

Strongly Safe DRA: Decidable, Non-Primitive Recursive
Bound the Simple Path Length of the underlying undirected configuration graph.
Strongly Bounded DRA

Bound the Simple Path Length of the underlying undirected configuration graph

$q_2 \rightarrow q_1 \rightarrow q_1 \rightarrow q_1 \rightarrow q_2 \rightarrow \cdots$

$k = 1$
Strongly Bounded DRA

Bound the Simple Path Length of the underlying undirected configuration graph
Strongly Bounded DRA

Verification of Dynamic Register Automata

Bound the Simple Path Length of the *underlying* undirected configuration graph

**q1**

**q2**

**k = 6**
Strongly Bounded DRA

Bound the Simple Path Length of the underlying undirected configuration graph

Still: Unbounded Number of Processes
Strongly Bounded DRA

Bound the Simple Path Length of the underlying undirected configuration graph

Undecidable
Strongly Bounded DRA

Bound the Simple Path Length of the underlying undirected configuration graph Undecidable

Two counters Minsky Machine
Strongly Bounded DRA

Bound the Simple Path Length of the underlying undirected configuration graph Undecidable

Two counters Minsky Machine

Strongly-Bounded Reachability
Strongly Bounded DRA

Bound the Simple Path Length of the underlying undirected configuration graph Undecidable

Two counters Minsky Machine

k = 4 Strongly-Bounded Reachability
Strongly Bounded DRA

Verification of Dynamic Register Automata

Bound the Simple Path Length of the underlying undirected configuration graph Undecidable

Two counters Minsky Machine
Strongly Bounded DRA

Bound the Simple Path Length of the underlying undirected configuration graph **Undecidable**

Two counters Minsky Machine
Strongly Bounded DRA

Bound the Simple Path Length of the underlying undirected configuration graph Undecidable

Two counters Minsky Machine
Strongly Bounded DRA

Verification of Dynamic Register Automata

Bound the Simple Path Length of the underlying undirected configuration graph

Undecidable

Two counters Minsky Machine

\text{zerotest}(c_2)
Strongly Bounded DRA

Bound the Simple Path Length of the underlying undirected configuration graph Undecidable

Two counters Minsky Machine
Strongly Bounded DRA

Bound the Simple Path Length of the underlying undirected configuration graph Undecidable

Two counters Minsky Machine
Strongly Bounded DRA

Bound the Simple Path Length of the underlying undirected configuration graph **Undecidable**

Two counters Minsky Machine
Verification of Dynamic Register Automata

Contribution

State Reachability

Bounded DRA

Strongly Bounded DRA

Strongly Safe DRA

Undecidable

Undecidable

Decidable

Non-Primitive Recursive
Contribution

Verification of Dynamic Register Automata

State Reachability

Bounded DRA

Strongly Bounded DRA

Strongly Safe DRA

Undecidable

Undecidable

Decidable

Non-Primitive Recursive
Contribution

Verification of Dynamic Register Automata

State Reachability

Bounded DRA  Undecidable

Strongly Bounded DRA  Undecidable

Strongly Safe DRA  Decidable

Degenerative DRA  Non-Primitive Recursive
Degenerative DRA

Dynamic Register Automata
Degenerative DRA

Verification of Dynamic Register Automata

Degenerative counter part \( \text{Deg}(D) \)

\[
\begin{align*}
\text{reset}<x,y,z> & \quad \text{create}(q_3, y) \\
q_1 & \xrightarrow{x} q_2 \\
q_2 & \xrightarrow{y!<m>} q_3 \\
q_3 & \xrightarrow{y!<x>} q_1 \\
q_3 & \xrightarrow{z?<x>} q_4 \\
q_4 & \xrightarrow{x?<m>} q_3 \\
\end{align*}
\]
Contribution

Verification of Dynamic Register Automata

State Reachability

Bounded DRA

Undecidable

Strongly Bounded DRA

Undecidable

Strongly Safe DRA

Decidable

Degenerative DRA

Non-Primitive Recursive
Strongly Safe DRA

Verification of Dynamic Register Automata

DRA is degenerative

Bound the Simple Path Length of the underlying undirected configuration graph
Strongly Safe DRA

Verification of Dynamic Register Automata

DRA is degenerative

Bound the Simple Path Length of the underlying undirected configuration graph

Well-Structured Transition Systems

[Abdulla et al. 1996], [Finkel et al. 2001]

Symbolic representation of infinite set of configurations
Well-Structured Transition Systems

- Define a **Well-Quasi Order** on configurations
- Prove **Monotonicity** of Transition Relation
- Provide an algorithm to compute the **Pre** of an upward closed set
Well-Structured Transition Systems

- Define a **Well-Quasi Order** on configurations
- Prove **Monotonicity** of Transition Relation
- Provide an algorithm to compute the **Pre** of an upward closed set
Strongly Safe DRA

Well-Structured Transition Systems

- Define a **Well-Quasi Order** on configurations

**Ordering: subgraph relation**
Strongly Safe DRA

Verification of Dynamic Register Automata

Well-Structured Transition Systems

- Define a **Well-Quasi Order** on configurations

**Ordering:** subgraph relation

\[ q_3 \xrightarrow{x} q_1 \xrightarrow{y} q_2 \subseteq \]

\[ q_3 \xrightarrow{x} q_1 \xrightarrow{y} q_2 \xrightarrow{z} q_2 \]
Well-Structured Transition Systems

› Define a **Well-Quasi Order** on configurations

**Ordering**: subgraph relation

[Ding, 1992] Subgraphs and Well-Quasi Ordering

(Induced) subgraph relation is a WQO on Strongly-Bounded graphs
Strongly Safe DRA

Well-Structured Transition Systems

- Define a Well-Quasi Order on configurations

Ordering: subgraph relation
Well-Structured Transition Systems

Define a **Well-Quasi Order** on configurations

Ordering: subgraph relation
Well-Structured Transition Systems

- Define a **Well-Quasi Order** on configurations

- Prove **Monotonicity** of Transition Relation

- Provide an algorithm to compute the **Pre** of an upward closed set
Well-Structured Transition Systems

- Define a **Well-Quasi Order** on configurations
- Prove **Monotonicity** of Transition Relation
- Provide an algorithm to compute the **Pre** of an upward closed set
Strongly Safe DRA

Well-Structured Transition Systems

- Prove **Monotonicity** of Transition Relation

\[ C_1 \quad C_2 \quad C_3 \]
Strongly Safe DRA

Well-Structured Transition Systems

- Prove Monotonicity of Transition Relation

\[ C_2 \sqsubseteq C_3 \sqsubseteq C_1 \]
Strongly Safe DRA

Verification of Dynamic Register Automata

Well-Structured Transition Systems

-Prove **Monotonicity** of Transition Relation

C_2 ⊑ C_3

C_1 → C_3
Well-Structured Transition Systems

- Prove Monotonicity of Transition Relation

\[
\begin{array}{c}
C_2 \\
\square \\
C_1 \\
\rightarrow \\
C_3 \\
C_4
\end{array}
\]
Well-Structured Transition Systems

- Prove **Monotonicity** of Transition Relation

\[
\begin{align*}
\mathbf{C}_2 & \xrightarrow{} \mathbf{C}_4 \\
\square & \\
\mathbf{C}_1 & \xrightarrow{} \mathbf{C}_3
\end{align*}
\]
Well-Structured Transition Systems

Prove **Monotonicity** of Transition Relation

\[ C_2 \rightarrow C_4 \]

\[ C_1 \rightarrow C_3 \]
Strongly Safe DRA

Well-Structured Transition Systems

- Prove **Monotonicity** of Transition Relation
Strongly Safe DRA

Well-Structured Transition Systems

- Prove **Monotonicity** of Transition Relation

Degenerative
Strongly Safe DRA

Verification of Dynamic Register Automata

Well-Structured Transition Systems

- Prove **Monotonicity** of Transition Relation

Degenerative

C₁

C₂
Strongly Safe DRA

Well-Structured Transition Systems

› Prove Monotonicity of Transition Relation
Strongly Safe DRA

Well-Structured Transition Systems

› Prove **Monotonicity** of Transition Relation

Subgraph relation: Register Mapping & States preserved
Strongly **Safe** DRA

Well-Structured Transition Systems

› Prove **Monotonicity** of Transition Relation

Subgraph relation: Register Mapping & States preserved
Strongly Safe DRA

Well-Structured Transition Systems

- Prove **Monotonicity** of Transition Relation

Subgraph relation: Register Mapping & States preserved
Strongly Safe DRA

Verification of Dynamic Register Automata

Well-Structured Transition Systems

- Define a **Well-Quasi Order** on configurations
- Prove **Monotonicity** of Transition Relation
- Provide an algorithm to compute the **Pre** of an upward closed set
Well-Structured Transition Systems

- Define a **Well-Quasi Order** on configurations
- Prove **Monotonicity** of Transition Relation
- Provide an algorithm to compute the Pre of an upward closed set
Contribution

Verification of Dynamic Register Automata

State Reachability

Bounded DRA

Strongly Bounded DRA

Strongly Safe DRA

Degenerative DRA

Undecidable

Undecidable

Decidable

Non-Primitive Recursive
**Conclusion**

**Dynamic Register Automata**
- Dynamic Creation of Processes
- Register Mapping
- Point-to-Point Comm.
- Rendez-Vous Comm.

**State Reachability**
- Bounded DRA: Undecidable
- Strongly Bounded DRA: Undecidable
- Strongly Safe DRA: Decidable