A Brief Introduction to Model Checking
Model Checking

- A technique for verifying finite state concurrent systems;
  - a benefit on this restriction: largely automatic;
  - a problem to fight: state space explosion;
- A logic pointview: the system as a semantical model $M$, and a property as a logical formula $\varphi$;
  - to check whether $M \vDash \varphi$ (by exhaustive search);
  - possible approaches: model checking chooses to work on models directly;
- Reasonable efficiency, giving answers in seconds/minutes;
- Counter-examples provide insight to understand the failures.
Model Checking

- Pioneers’ work: West and Zafiropulo (1977, 1978), Clarke and Emerson (1981), and Quielle and Sifakis (1981);

- Applications: electric circuits, communication protocols, digital controllers, system designs, ..., widely accepted in industry;


- Model checkers: FDR, Spin, Morφ, νSMV, CADP, Uppaal, PRISM, HyTech, COSPAN, STeP, Kronos ...
Model Checking

To apply it, we need the follows:

- **Modeling languages**: describe the systems, e.g. a process algebraic language $\mu$CRL;
  - semantics of the languages, e.g. LTS, Kripke structures, automata;

- **Specification languages**: formulate properties, e.g. LTL, CTL, regular alternation-free $\mu$-calculus;
  - safety and liveness properties;
  - $[T^* \cdot \text{error}] F$;
  - $[ T^* \cdot \text{send} \cdot (\neg \text{receive})^* ] \langle (\neg \text{receive})^* \cdot \text{receive} \rangle T$;

- **Algorithms**: verify properties.
Model Checking and Leader Election Algorithms

The process:

- Systems
- Formal description
- Semantical model
- Logical formulas
- Model checker
- Desired properties

- yes
- no

© Jun Pang
Research issues:

- Approaches to fight state space explosion;
- Expressiveness of logics;
- Efficiency of algorithms.
Partial order reduction:

- Idea: fix a particular order of interleaving behaviors, while preserving properties of interest;

- CWI: $\tau$-confluence reduction (preserving branching bisimulation).
Symmetry reduction (similar to data independence):

- Idea: construct a quotient structure, by exploiting automorphisms of the system’s state space;

- CWI: symmetry reduction for LTSs.
Abstraction (on data):

- Idea: replace a semantical model by an abstract (simpler, finite) model, the abstraction needs to be safe;

- CWI: abstract interpretation for $\mu$CRL; patterns for uniform parallel processes, abstraction for liveness properties.
On-the-fly:

- Idea: not generate unnecessary state space, especially when a formula is false;
- CWI: interface with the model checker CADP.
Symbolic model checking (OBDDs):

- Idea: avoid explicit enumeration of set, by expressing set as a propositional formula;
  OBDDs as data structures to represent the state space;
- CWI: a checker for modal formulas for processes with data.

\[
S = [0, 2, 4, 5, 6, 7] \Rightarrow \\
S = [000, 010, 100, 101, 110, 111] \Rightarrow \\
S = \{s | s_3 = 0 \lor s_1 = 1\}, (s = s_1 s_2 s_3)
\]
Distributed and parallel model checking:

- Problem: the state space does not completely fit into the main memory of a computer;

- Idea: increase the computational power by building a cluster of stations;

- CWI: distributed state space generation and reduction w.r.t. strong and branching bisimulation.
More recent challenging issues:

- Timed, hybrid, **probabilistic**, mobile systems, e.g. Uppaal, Kronos, PRISM, ?;
- Software verification, e.g. Spin;
- **Source code verification**, e.g. Bandera;
- Infinite-state systems, e.g. regular model checkers Fast, Trex;
- Challenging case studies, e.g. NASA Mars exploration rover.
Simplifying Itai-Rodeh Leader Election for Anonymous Rings
Figure 1: Mutual exclusion: token recovery.
Leader Election

Many algorithms:

- Communication mechanism: *asynchronous* vs. *synchronous*;
- Process names: *unique identities* vs. *anonymous*;
- Network topology: *ring, tree, complete graph*;
- ...
Plan of the Talk

1. The Chang-Roberts algorithm and the Itai-Rodeh algorithm;

2. Algorithm $\mathcal{A}$: leader election without round numbers;

3. Algorithm $\mathcal{B}$: leader election without bits;

4. Performance analysis in PRISM;

5. Conclusions and future works.
The Chang-Roberts Algorithm

Processes have unique identity and send messages with identity; process with maximal identity is elected as the leader.

States of processes: \{active, passive, leader\}
Anonymous Rings

Some cases where processes cannot be distinguished by means of unique identities:

1. as the number of processes increases, it is difficult to keep all the identities of processes distinct;

2. identities cannot always be sent around the network, e.g. FireWire, the IEEE 1394 high performance serial bus.
The Itai-Rodeh Algorithm

Probabilistic method to break symmetry!

Assumption: processes have the knowledge of the ring size $n$.

Difficulties: each process selects a random identity from a finite set, so different processes may carry the same identity. Each process needs to

- recognize the message sent by its own – $hop$ counter;
- realize name clashes – $bit$; and
- recognize old messages and start a new round – $round$ number.
The Itai-Rodeh Algorithm

- Initially, all processes are active, and each process $p_i$ randomly selects its identity $id_i \in \{1, \ldots, k\}$ and sends the message $(id_i, 1, 1, true)$.

- Upon receipt of a message $(id, round, hop, bit)$, a passive process $p_i$ ($state_i = passive$) passes on the message, increasing the counter $hop$ by one; an active process $p_i$ ($state_i = active$) behaves according to one of the following steps:
The Itai-Rodeh Algorithm

1. if $hop = n$ and $bit = true$, then $p_i$ becomes the leader ($state'_i = leader$);

2. if $hop = n$ and $bit = false$, then $p_i$ selects a new random identity $id'_i \in \{1, \ldots, k\}$, moves to the next round ($round'_i = round_i + 1$), and sends the message $(id'_i, round'_i, 1, true)$;

3. if $hop < n$ and $(round, id) = (round_i, id_i)$, then $p_i$ passes on the message $(id, round, hop + 1, false)$;

4. if $(round, id) > (round_i, id_i)$, then $p_i$ becomes passive ($state'_i = passive$) and passes on the message $(id, round, hop + 1, bit)$;

5. if $(round, id) < (round_i, id_i)$, then $p_i$ purges the message.
The Itai-Rodeh Algorithm

**Theorem 1** [Itai and Rodeh 1981] The Itai-Rodeh algorithm terminates with probability one, and upon termination a unique leader has been elected.
The Itai-Rodeh Algorithm

Figure 2: The Itai-Rodeh Algorithm: an example $n = 3$
The Itai-Rodeh Algorithm

(u, 1, 2, true)  (u, 1, 2, false)

u > v

Figure 3: The Itai-Rodeh Algorithm: an example $n = 3$
The Itai-Rodeh Algorithm

\[(u, 1, 3, false)\]

\[u \gg v\]

Figure 4: The Itai-Rodeh Algorithm: an example \(n = 3\)
The Itai-Rodeh Algorithm

\[ (w, 2, 1, \text{true}) \]

\[ (x, 2, 1, \text{true}) \]

\[ w > x \]

Figure 5: The Itai-Rodeh Algorithm: an example \( n = 3 \)
The Itai-Rodeh Algorithm

Figure 6: The Itai-Rodeh Algorithm: an example $n = 3$
The Itai-Rodeh Algorithm

\[ w \xrightarrow{w > x} x \]

\[ (w, 2, 3, true) \]

Figure 7: The Itai-Rodeh Algorithm: an example \( n = 3 \)
Round Number are Essential!

Observations:

- Itai-Rodeh leader election has infinite state space, (due to round numbers).

Round numbers are essential if channels are not FIFO.
Leader Election without Round Numbers

Observations:

- if channels are FIFO, round numbers are redundant.

**Proposition 2** Consider the Itai-Rodeh algorithm where all channels are FIFO. When an active process receives a message, then the round numbers of the process and the message are always the same.
Leader Election without Round Numbers

Algorithm $\mathcal{A}$: messages have the form of $(id, hop, bit)$. Passive processes behave the same as before.

1. if $hop = n$ and $bit = true$, then $p_i$ becomes the leader ($state'_i = leader$);

2. if $hop = n$ and $bit = false$, then $p_i$ selects a new random identity $id'_i \in \{1, \ldots, k\}$ and sends the message $(id'_i, 1, true)$;

3. if $hop < n$ and $id = id_i$, then $p_i$ passes on the message $(id, hop + 1, false)$;

4. if $id > id_i$, then $p_i$ becomes passive ($state'_i = passive$) and passes on the message $(id, hop + 1, bit)$;

5. if $id < id$, then $p_i$ purges the message.
Leader Election without Round Numbers

\[
\begin{align*}
(u, 1, \text{true}) \\
(u, 1, \text{true}) \\
(u, 1, \text{true})
\end{align*}
\]

\[ u > v \]

Figure 8: Algorithm $\mathcal{A}$: an example $n = 3$ (step 1)
Leader Election without Round Numbers

\[ u \rightarrow u \quad (u, 2, false) \]

\[ u \quad (u, 2, true) \rightarrow v \]

\[ u > v \]

Figure 9: Algorithm \( \mathcal{A} \): an example \( n = 3 \) (step 2)
Leader Election without Round Numbers

\begin{center}
\begin{tikzpicture}[node distance=2cm,thick]
  \node (u) {$u$};
  \node (v) [below of=u] {$v$};
  \node (u2) [right of=u] {$u$};
  \node (u3) at ([yshift=-1cm]u2) {$u$};

  \draw[->] (u) -- (u2);
  \draw[->] (v) -- (u3);
  \draw[->] (u2) -- (u3);
  \draw[->] (u) -- (v);

  \node (label1) [above of=u2] {$(u, 3, false)$};
  \node (label2) [above of=u3] {$(u, 3, false)$};

  \draw[->] (u2) -- (label1);
  \draw[->] (u3) -- (label2);

  \node (u4) [above of=v] {$(u, 3, false)$};
\end{tikzpicture}
\end{center}

\begin{align*}
u > v
\end{align*}

Figure 10: Algorithm $\mathcal{A}$: an example $n = 3$ (step 3)
Leader Election without Round Numbers

Figure 11: Algorithm $\mathcal{A}$: an example $n = 3$ (step 4)
Leader Election without Round Numbers

Figure 12: Algorithm $\mathcal{A}$: an example $n = 3$ (step 5)
Leader Election without Round Numbers

Figure 13: Algorithm $\mathcal{A}$: an example $n = 3$ (step 6)
Leader Election without Round Numbers

<table>
<thead>
<tr>
<th></th>
<th>Processes</th>
<th>Identities</th>
<th>Channel size</th>
<th>FIFO</th>
<th>States</th>
<th>Transitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ex.1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>yes</td>
<td>127</td>
<td>216</td>
</tr>
<tr>
<td>Ex.2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>yes</td>
<td>5,467</td>
<td>12,360</td>
</tr>
<tr>
<td>Ex.3</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>yes</td>
<td>99,329</td>
<td>283,872</td>
</tr>
</tbody>
</table>

Table 1: Model checking result for Algorithm \( \mathcal{A} \) with FIFO channels in PRISM

**Theorem 3** Let channels be FIFO. Then Algorithm \( \mathcal{A} \) terminates with probability one, and upon termination exactly one leader is elected.

**Proof.** Reuse the proof of Itai and Rodeh and by Proposition 2.
Leader Election without Bits

Observation:

- an active process $p_i$ detects a name clash, it is not necessary for $p_i$ to wait for its own message to return.

Algorithm $B$: messages have the form of $(id, hop)$.

1. if $hop = n$ and $id = id_i$, then $p_i$ becomes the leader ($state'_i = leader$);

2. if $hop < n$ and $id = id_i$, then $p_i$ selects a new random identity $id'_i \in \{1, \ldots, k\}$ and sends the message $(id'_i, 1)$;

3. if $id > id_i$, then $p_i$ becomes passive ($state'_i = passive$) and passes on the message $(id, hop + 1)$;

4. if $id < id_i$, then $p_i$ purges the message.
Leader Election without Bits

\[ u > v \]

Figure 14: Algorithm \( B \): an example \( n = 3 \) (step 1)
Leader Election without Bits

Figure 15: Algorithm $\mathcal{B}$: an example $n = 3$ (step 2)
Leader Election without Bits

\[ w > x \]

Figure 16: Algorithm \( B \): an example \( n = 3 \) (step 3)
Leader Election without Bits

Figure 17: Algorithm $B$: an example $n = 3$ (step 4)
Leader Election without Bits

$w \rightarrow x$

$(w, 3) \rightarrow v$

$w > x$

Figure 18: Algorithm $B$: an example $n = 3$ (step 5)
### Leader Election without Bits

<table>
<thead>
<tr>
<th>Processes</th>
<th>Identities</th>
<th>Channel size</th>
<th>FIFO</th>
<th>States</th>
<th>Transitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ex.1</td>
<td>2</td>
<td>2</td>
<td>yes</td>
<td>97</td>
<td>168</td>
</tr>
<tr>
<td>Ex.2</td>
<td>3</td>
<td>3</td>
<td>yes</td>
<td>6,019</td>
<td>14,115</td>
</tr>
<tr>
<td>Ex.3</td>
<td>4</td>
<td>3</td>
<td>yes</td>
<td>176,068</td>
<td>521,452</td>
</tr>
<tr>
<td>Ex.4</td>
<td>4</td>
<td>4</td>
<td>yes</td>
<td>537,467</td>
<td>1,615,408</td>
</tr>
<tr>
<td>Ex.5</td>
<td>5</td>
<td>2</td>
<td>yes</td>
<td>752,047</td>
<td>2,626,405</td>
</tr>
</tbody>
</table>

Table 2: Model checking result for Algorithm $\mathcal{B}$ with FIFO channels
Leader Election without Bits

**Theorem 4** Let channels be FIFO. Then Algorithm $B$ terminates with probability one, and upon termination exactly one leader is elected.

Proof is not easy!
Performance Analysis in PRISM

The probability that Algorithms $\mathcal{A}$ and $\mathcal{B}$ terminate within a given number of transitions.

Figure 19: The probability of electing a leader with deadlines.
Performance Analysis in PRISM

The probability that Algorithms $A$ and $B$ terminate within a given number of transitions.

![Graph showing the probability of electing a leader with deadlines.](image)

Figure 20: The probability of electing a leader with deadlines.
### Performance Analysis in PRISM

The expected number of steps before a unique leader is elected for each algorithm.

<table>
<thead>
<tr>
<th></th>
<th>Processes</th>
<th>Identities</th>
<th>Channel size</th>
<th>Steps (A)</th>
<th>Steps (B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ex.1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>25.0</td>
<td>19.0</td>
</tr>
<tr>
<td>Ex.2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>33.6</td>
<td>29.3</td>
</tr>
<tr>
<td>Ex.3</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>52.5</td>
<td>46.0</td>
</tr>
</tbody>
</table>
Conclusions and Future Works

- We developed two new leader election algorithms for anonymous rings;
- Model checking and performance analysis of both algorithms in PRISM;
- We gave a manual correctness proof for each algorithm;
- When $k = 2$, both algorithms $A$ and $B$ are correct even if channels are not FIFO;
- We developed two more probabilistic leader election algorithms, based on the Dolev-Klawe-Rodeh algorithm;
- We are going to check the proofs in PVS.
The Proof of Theorem 4

Definition 5 The processes and messages between a process $p$ and a message $m$ are the ones that are encountered when traveling in the ring from $p$ to $m$.

Lemma 6 Let active process $p$ have identity $id_p$ and message $m$ have identity $id_m$. If $id_p \neq id_m$, then there is an active process or message between $p$ and $m$ with an identity $\geq \min\{id_p, id_m\}$.

Proof. We apply induction on execution sequences.

\[ \Box \]
The Proof of Theorem 4

Definition 7 An active process $p$ is related to a message $m$ if they have the same identity $id$, and all active processes and messages between $p$ and $m$ have an identity smaller than $id$.

Lemma 8 Let active process $p$ be related to message $m$. Let $\xi$ be the maximum of all identities of active processes and messages between $p$ and $m$ ($\xi = 0$ if there are none).

1. Between $p$ and $m$, there is an equal number of active processes and of messages with identity $\xi$; and

2. if $p$ is not the originator of $m$, then there is an active process or message between $p$ and $m$.

Proof. We apply induction on execution sequences.
The Proof of Theorem 4

**Definition 9** We say that an active process or message is *maximal* if its identity is maximal among the active processes or messages in the ring, respectively. In the following proposition we write $\xi_\pi$ and $\xi_\mu$ for the identity of maximal active processes and messages, respectively. We write $\#_\pi$ and $\#_\mu$ for the number of maximal active processes and messages, respectively.

**Proposition 10** Until a leader is elected, there exist active processes and messages in the ring, and $\xi_\pi = \xi_\mu$ and $\#_\pi = \#_\mu$.

**Proof.** We apply induction on execution sequences.

Finally, by induction on execution sequences, and use Proposition 10, we can prove Theorem 4.