Self-Adaptive Automata

Aimee Borda (Lero & Trinity College Dublin)
Vasileios Koutavas (Lero & Trinity College Dublin)
Self-Adaptive Systems

• Systems with the “ability to adapt at run-time to changing user needs, system intrusions or faults, changing operational environment, and resource variability”

• “Has been proposed as a means to cope with the complexity of today's software-intensive systems”

(Dagstuhl Seminar 10431)
Search Operation by Unmanned Vehicles

Requirements:
1. No Collisions
2. Vehicles stay in search area
3. Eventually search the whole area
Search Operation by SA Unmanned Vehicles
Time-Triggered Adaptation

Every 3 time units

Coordinator
(collision avoidance,...)
Time-Triggered Adaptation

Every 10 time units

Coordinator
(collision avoidance,...)

T0  →  T1  →  T2  →  T3  →  T4
Event-Triggered Adaptation

Vehicles are dangerously close to each other
Adaptation Pattern is crucial!

Execution points where adaptation is triggered
• Time-based, event-based, history-based...
• Important (for correctness)
• Necessary (for implementation)
**Goal**: Enable Experimentation and Verification of Adaptation Patterns

- Model at a **high-level** of abstraction
- **Modularity**: localise adaptation pattern
- Leverage existing verification technology
Our Model: Self-Adaptive Automata (SAA)

Adaptation pattern

Adaptation Pattern

Must-Adapt

SAA

Managed System

Adaptation: distinguished symbol

Adaptation Manager

May-Adapt
Our Model: Self-Adaptive Automata (SAA)

\[ \text{SAA} \overset{\text{def}}{=} \langle Q, \Sigma, \Delta, q_0, \delta_0, \Pi \rangle \]

- \( Q \): Set of States
- \( \Sigma \): Set of Events
- \( \Delta \): Set of Transition Functions
- \( q_0 \): Initial State
- \( \delta_0 \): Initial Transition Function
\[ q_0 : Q, \delta_0 : Q \times \Sigma \rightarrow Q \]
- \( \Pi \): Adaptation Function
\[ \Pi : Q \rightarrow Q \times \Delta \]
## Model: Self-Adaptive Automata

\[ \text{SAA} \overset{\text{def}}{=} \langle Q, \Sigma, \Delta, q_0, \delta_0, \Pi \rangle \]

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Q )</td>
<td>Set of States</td>
</tr>
<tr>
<td>( \Sigma )</td>
<td>Set of Events</td>
</tr>
<tr>
<td>( \Delta )</td>
<td>Set of Transition Functions</td>
</tr>
<tr>
<td>( q_0 )</td>
<td>Initial State</td>
</tr>
<tr>
<td>( \delta_0 )</td>
<td>Initial Transition Function</td>
</tr>
<tr>
<td>( \Pi )</td>
<td>Adaptation Function</td>
</tr>
</tbody>
</table>

**Operational semantics**

\[ \langle q, \delta \rangle \xrightarrow{a} \langle q', \delta \rangle \]

such that \( \delta(q, a) = q' \)

\[ \langle q, \delta \rangle \xrightarrow{\ast} \langle q', \delta' \rangle \]

such that \( \Pi(q) = \langle q', \delta' \rangle \)

Model: Self-Adaptive Automata (SAA)

\[ SAA \overset{\text{def}}{=} \langle Q, \Sigma, \Delta, q_0, \delta_0, \Pi \rangle \]
Modelling the Adaptation Pattern with SAA

Adaptation Pattern

Base Model

SAA

Adaptation Manager

Managed System

May-Adapt
Implementing the **Adaptation Pattern** with SAA

Time-Triggered (every 3 time units)
Modelling the **Adaptation Pattern** with SAA

**Time-Triggered (every 3 time units)**

![Adaptation Pattern Diagram]

- Enabled */-transition is the only outgoing transition

**Must-Adapt**
Modelling the **Adaptation Pattern** with SAA

Event Triggered: Vehicles are dangerously close to each other
Modelling the **Adaptation Pattern** with SAA

**Event Triggered:** Vehicles are dangerously close to each other

**Adaptation Pattern**

- Safe moves
- Unsafe moves

- **Enabled** ∗-transition is the only outgoing transition

**Must-Adapt**
Self-Adaptive System

Adaptation pattern

Base Model

Must-Adapt

May-Adapt

Coordinator

Vehicles

SAA

safe moves

0 safe

tick

1 tick unsafe moves

2 tick unsafe

3

★
Verifying Adaptation Patterns

Translation to FDR (a refinement-based verification tool)
Refinement-based Verification

\[ \mathcal{M} \] (refines) \[ \mathcal{A} \]

TRANSLATION!

\[ \text{Requirement}_{FDR4} \]

\[ \text{Requirement}_{FDR4} \] (refines)
Refinement-based Verification

Translation is a bisimulation

\[ [\mathcal{M}]_{\text{FDR4}} \equiv [\mathcal{A}]_{\text{FDR4}} \text{ (refines)} \]

SAA

Adaptation Pattern

Adaptation Manager

System
Translation: SAA $\rightarrow$ FDR

$$M = \langle Q, \Sigma, \Delta, q_0, \delta_0, \Pi \rangle$$

$Q$ $\rightarrow$ 0, 1, 2... (FDR States)

$\Sigma$ $\rightarrow$ a, b, c... (FDR Events)

$\delta(0) = a \rightarrow \delta(1)$

$\delta(0) = b \rightarrow \delta(2)$

$(0) \rightarrow$ STOP

adapt + state

*** SEMANTICS

$Q, \delta > a \rightarrow <q', \delta>$ such that $\delta'(q')$

$\delta_0(q_0) \Delta \{ \cdot \} \Pi$

FDR parallel & interrupt

such that $\Pi(q) = <q, \delta'>$
Verifying different Adaptation Patterns

No Collision

Vehicles Stay In Area

Search All Area

Coordinator

Vehicles

Event-Triggered

Time-Triggered (3 steps)

Time-Triggered (10 steps)

23
Expressivity of SAA
Model: Self-Adaptive Automata

\[ \text{SAA} \triangleq \langle Q, \Sigma, \Delta, q, \delta, \Pi \rangle \]

- **\( Q \):** Set of States
- **\( \Sigma \):** Set of Events
- **\( \Delta \):** Set of Transition Functions
- **\( q \):** Initial State
- **\( \delta \):** Initial Transition Function
  \[ Q \times \Sigma \to Q \]
- **\( \Pi \):** Adaptation Function
  \[ Q \to Q \times \Delta \]

Operational Semantics:

Sets of States \( Q \) and Transition Functions \( \Delta \) are fixed

Such that \( \delta(q, a) = q' \)

\[ \langle q, \delta \rangle \to \langle q', \delta' \rangle \]

Such that \( \Pi(q) = \langle q', \delta' \rangle \)
SAA vs other self-modifying Models

SMFA [Schutt et al 1994]
+ Compact representation of dynamic behaviour
+ Add significant expressivity to base Model
- No verification tools

SAA
+ Compact representation of dynamic behaviour
- Does not add expressivity to base Model
+ Leverage existing verification tools

Proven through a bidirectional translation to Execution Monitor
(see paper)

Translation to FDR
Conclusion

• High-level Model for Self-Adaptive Systems
  • Modularizes Adaptation patterns
  • Enables **experimentation** with Adaptation Patterns

• Leverage existing verification technologies
  • Enables **verification** of Adaptation Patterns

• Future Work: improve usability, use other verification technologies
Thank you!

Questions?