Reasoning on Reasoning Robots
in honour of Kokichi Futatsugi

Martin Wirsing
Ludwig-Maximilians-Universität München

In cooperation with
Lenz Belzner, Rocco De Nicola, Andrea Vandin
My relationship to Kokichi

■ Kokichi and MW are members of IFIP WG 1.3 on Foundations of System Specification

■ 1996-1998 MW partner of CafeOBJ Project
  ▪ Alexander Knapp, MW: CafeOBJ specification and analysis of
    ▪ Airport "Munich II"
    ▪ Operational semantics of multi-threaded Java
Autonomic systems are distributed computing systems whose
- components act autonomously and
- can adapt to environment changes.
ASCENS Project

- **Goal of ASCENS:**
  Develop methods, tools, and theories for **modelling and analysing autonomic self-aware systems**
  that combine
  - software engineering based on formal methods with
  - methods from autonomic, adaptive, and self-aware systems

- **Partners:**
  - LMU (Coordinator), U Pisa, U Firenze with ISTI Pisa, Fraunhofer, Verimag, U Modena e Reggio Emilia, U Libre de Bruxelles, EPFL, Volkswagen, Zimory GmbH, U Limerick, Charles U Prague, IMT Lucca, Mobsya

- **Case studies:**
  - Robotics, cloud computing, and e-mobility
Simple Robot Case Study

- Swarm of
  - blind (randomly moving) robots and
  - informed and reasoning robots
  - seeking for collision avoidance
Simple Robot Case Study

What should I do?
1) Move up
2) Move right
3) Move down
4) Move left
5) Stand still
Simple Robot Case Study

What should I do?
1) Move up
2) Move right
3) Move down
4) Move left
5) Stand still

Takes random decisions

Observes the environment and reasons to minimize collisions
Simple Robot Case Study

Perception range

Robots perceived on the right

Current number of collisions

Current number of collisions: 5

Numbers on the right side of the perception range:
- 8
- 4
- 4
- 9
- 13
- 7
- 10
- 10
Video
Goal

Use rewriting logic as a tool for
- modelling,
- reasoning about,
- simulating and
- analyzing autonomic systems
Goal and Outline of this Talk

Goal
Use rewriting logic as a tool for
- modelling,
- reasoning about,
- simulating and
- analyzing autonomic systems
Modelling Behaviour

- Modelling behaviour
  - SCEL behaviour specification language
  - MISSCEL rewriting implementation

- Reasoning
  - Action-oriented reasoning
  - PiRLo rewriting implementation

- Integrating behaviour and reasoning

- Analysis
SCEL
Service Component Ensemble Language

- Kernel language for programming autonomic computing systems
  [De Nicola et al. 2013]
- Distributed Linda-like components
- Structured operational semantics

Processes explicitly represent the behaviour of a component, and interact with the local or remote knowledge repositories.

Policies regulate the interaction within the component and with external components.

Martin Wirsing
Execution of operational semantics of SCEL in Maude

Designed and implemented by Andrea Vandin [Vandin 2013]

Supports

- Qualitative analysis
  - Invariant checking (Maude search command)
  - LTL model checking (Maude LTL model checker)

- Debugging
  - Animated probabilistic simulations

- Quantitative analysis
  - Distributed Statistical Model Checking (MultiVeStA)
MISSCEL Example

- **Uninformed robot**

  Interface with robot id

  $\text{SC}(I, \ \text{tId('SCId'))};$

  $\text{K}(< \text{tId('SCId')}; \ \text{av(id('robot-normal-1'))} >; \ < \text{tId('type')}; \ \text{av('normal')} >; \ < \text{tId('pos')}; \ \text{av(1) av(2)} >; \ < \text{tId('collisions')}; \ \text{av(13)} > ),$

  $\text{Pi}(\ \text{INTERLEAVING-INTERACTION-PREDICATE}),$

  $\text{P}(\ \text{qry(< tId('pos'); ?x('x) ?x('y) @ self} . \ \text{put(< tId('dir'); randomDirection(x('x), x('y)) @ self} . \ \text{put(< av('terminated') @ self}} \ [\ \text{get(< av('terminated') @ self}. \ \text{pDef('PnormalRobot')}] )

- **Actuate robot movement**

  $\text{ceq}\ \text{SC}(I,\ \text{K}(< \text{tId('pos')}; \ \text{av(x) av(y)} >; \ < \text{tId('dir')}; \ \text{av(dir)} >, \ k), \ \text{Pi}, \ \text{P}) = \text{SC}(I, \ \text{K}(< \text{tId('pos')}; \ \text{av(x2) av(y2)} >, \ k), \ \text{Pi}, \ \text{P})$

  if $\text{av(x2) av(y2)} :=$

  computeNeighbouringPosition($\text{av(x)}, \ \text{av(y)}, \ \text{av(dir)})$. 

Martin Wirsing
(MIS)SCEL

Integration

Visual Analysis

Statistical Model Checking

PiRLo

Modelling behaviour
- SCEL behaviour specification language
- MISSCEL rewriting implementation

Reasoning
- Action-oriented reasoning
- PiRLo rewriting implementation

Integrating behaviour and reasoning

Analysis

Martin Wirsing
Reasoning with PiRLo

- Action Programming [Thielscher 2008] in Maude
  - Reasoning about actions and change (using fluents)
  - Construction & evaluation of behavioural alternatives
- Designed and implemented by Lenz Belzner [Belzner 2013]

- Domain specification
  - Domain objects and properties
    $\text{pos}(\text{Robot}, X, Y)$
  - Action preconditions and their effects
    $\text{pos}(\text{Robot}, X, Y) \text{ and } \text{move}(\text{Robot}, \text{right}) \Rightarrow \text{pos}(\text{Robot}, X+1, Y)$
- Action programs
  - Nondeterministic plans
    $(\text{move}(\text{ca, d1}) \neq \text{move}(\text{ca, d2})) \text{ ; update(a)}; \ldots$
Determine probability of collision avoidance

- Move probability is uniformly distributed
  - noop, up, down, left, right occur with probability 0.2
- Precompute lookup tables
  - E.g. for a single agent

Current position of uncontrolled agent

Probability of avoiding collision when moving here
PiRLo Example: Compute move with lowest collision probability

eq moveProgram(A) =
move(A,0,0) # move(A,1,0) # move(A,-1,0)
# move(A,0,1) # move(A,0,-1).

Current State

Controlled Agent

Other Agent

Update Action

Collision Avoidance Probability

Collision avoidance probability accounting for agent A according to lookup table

Future State

Behavioral alternatives for controlled agent

pos(CA,P) and pos(A,P') and update(A) and p(Prob)

if Prob' := lookup(P', P).

moveProgram(ca); update(a); update(a');...
Integration

- Modelling behaviour
  - SCEL behaviour specification language
  - MISSCEL rewriting implementation

- Reasoning
  - Action-oriented reasoning
  - PiRLo rewriting implementation

- Integrating behaviour and reasoning

- Analysis

(MIS)SCEL

PiRLo

Reasoner Integrator

Visual Analysis

Statistical Model Checking

Martin Wirsing
Integration
Integration

![Integration Diagram]

- **$R$** Reasoner
- **$K$** Knowledge
- **$RI$** Reasoner Integrator
- **$\Pi$** Policies
- **$P$** Processes
- **$I$** Interface

**Normal flow**

**Reasoning request**
Integration

Reasoner Integrator

Abstract Interface

Definitions of

\texttt{scel2reasoner()}
\texttt{invokeReasoner()}
\texttt{reasoner2scel()}

Concrete Adapter

Implementation of abstract methods and

\textit{reasoning services}

specific to reasoner and domain

SCEL

Reasoner

Martin Wirsing
Analysis

- Modelling behaviour
  - SCEL behaviour specification language
  - MISSCEL rewriting implementation

- Reasoning
  - Action-oriented reasoning
  - PiRLo rewriting implementation

- Integrating behaviour and reasoning

- Analysis
Statistical Modelchecking

- Statistical analysis technique
  - No absolute confidence (in contrast to classical model checking)
  - Errors bounded by a given confidence interval

- We use the tool **MultiVeStA** [Vandin 2013]
  - Extends PVesTA [AlTurki, Meseguer 2011]
  - Monte Carlo based analysis via parallel/distributed simulations
  - Allows to query expected values of real-typed expressions
    - \( E[\text{collisions of informed robot after } n \text{ steps }], 0 < n < 6000 \)
### Statistical Modelchecking

- **Avg collisions of random walkers**
- **Collisions of informed robot - perceive 4 dirs**
- **Collisions of informed robot - perceive 8 dirs**

<table>
<thead>
<tr>
<th>Number of steps (consumed SCEL actions)</th>
<th>Collisions</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1000</td>
<td>5</td>
</tr>
<tr>
<td>2000</td>
<td>10</td>
</tr>
<tr>
<td>3000</td>
<td>15</td>
</tr>
<tr>
<td>4000</td>
<td>20</td>
</tr>
<tr>
<td>5000</td>
<td>25</td>
</tr>
<tr>
<td>6000</td>
<td>30</td>
</tr>
</tbody>
</table>
Conclusion and Outlook

- ASCENS is developing a systematic approach for constructing Autonomic Service-Component Ensembles
- Simple case study illustrating
  - SCEL behaviour specification language
  - MISSCEL rewriting logic implementation
  - PiRLo reasoning (using rewriting logic)
  - Integration of behaviour and reasoning
  - Analysis through animated simulation and statistical model checking
- In the future
  - Applying ASCENS methods to complete case studies
- But most importantly
Thank you, Kokichi!
And all our best wishes for many further happy and successful years!