Semantic Differencing of Activity Diagrams by a Translation into Finite Automata

Kautz, Rumpe
Software Engineering
RWTH Aachen

\[ \text{sem}(\text{ad}_1) \subseteq \text{sem}(\text{ad}_2) \iff \mathcal{L}(\text{aut}_1) \subseteq \mathcal{L}(\text{aut}_2) \]

http://www.se-rwth.de/
1. Context
2. Example
3. Activity Diagram Variant
4. Translation to Finite Automata
Activity Diagram Evolution Analysis

- **Syntactic AD evolution analysis**
  - \( ad_1 \oplus \Delta(ad_1, ad_2) = ad_2 \)
  - But what about semantics?

- \( ad_1 \neq ad_2 \) but \( \text{sem}(ad_1) = \text{sem}(ad_2) \) is possible

- **Semantic AD evolution analysis**
  - \( \text{sem}(ad_1) \subseteq \text{sem}(ad_2) ? \)
  - If not, output witness \( w \in \text{sem}(ad_1) \setminus \text{sem}(ad_2) \)

- **Contribution:**
  - Simple translation from AD subclass to automata
  - Reduction of \( \text{sem}(ad_1) \subseteq \text{sem}(ad_2) \) to automata language inclusion
Agenda

1. Context
2. Example
3. Activity Diagram Variant
4. Translation to Finite Automata
Hiring Workflow

Δ(hire1, hire2): Action labeled “Assign Keys” added
Δ(hire2, hire1): Action labeled “Assign Keys” removed

No information about the semantic impact of the syntactic changes
Hiring Workflow

```
semantic differencing

\[ \text{sem}(\text{hire1}) \subseteq \text{sem}(\text{hire2}) \]

Every execution trace of hire1 is also an execution trace of hire2
```
There are execution traces of hire2 that are no execution traces of hire1

**semantic differencing**

\[ w \in \text{sem}(\text{hire2}) \setminus \text{sem}(\text{hire1}) \]
Agenda

1. Context
2. Example
3. Activity Diagram Variant
4. Translation to Finite Automata
Activity Diagram Variant

- Action, Initial, Final, Fork, Join, Decison, Merge nodes
- Multiple actions with same label
- Loops permitted
- Nesting of control flow nodes and loops
- Semantics: Set of all execution traces – no true parallelism

\[ \langle A, B, C, B, E, H, G, C \rangle \in \text{sem}(ad) \]
Agenda

1. Context

2. Example

3. Activity Diagram Variant

4. Translation to Finite Automata
Semantic Differencing

- Define translation $t: AD \rightarrow Aut$ from ADs to automata
- Define semantics $\text{sem}(ad) = \mathcal{L}(t(ad))$
- Reduce semantic AD differencing to language inclusion checking

\[ \text{sem}(ad1) \subseteq \text{sem}(ad2) \iff \mathcal{L}(aut1) \subseteq \mathcal{L}(aut2) \]
Translation to Automata

- Set of AD edges $\leftrightarrow$ Automaton state
- AD action label $\leftrightarrow$ Automaton transition label
Thanks

Hiring Workflow

There are execution traces of hire2 that are no execution traces of hire1

Semantic Differencing

- Define translation $t : AD \rightarrow Aut$ from ADs to automata
- Define semantics $sem(ad) = L(t(ad))$
- Reduce semantic AD differing to language inclusion checking

$sem(ad1) \subseteq sem(ad2) \iff L(aut1) \subseteq L(aut2)$

Activity Diagram Variant

- Action, Initial, Final, Fork, Join, Decision, Merge nodes
- Multiple actions with same label
- Loops permitted
- Nesting of control flow nodes and loops
- Semantics: Set of all execution traces – no true parallelism

Translation to Automata

- Set of AD transitions $\leftrightarrow$ Automaton state
- AD action label $\leftrightarrow$ Automaton transition label