Towards Verifiable Model Transformations:
A Finite State Example

Gabor Karsai, Anantha Narayanan,
Sandeep Neema
Vanderbilt University/ISIS
Outline

- Problem description
- Goal-Directed Certification
- Bisimilarity
- Statechart to EHA transformation
- Checking bisimilarity between Statechart and EHA models
- Conclusion and future work
Correctness of Model Transformations is central to the success of a model driven development process.

Systems are designed using a design language, and transformed into an analysis language for analysis.

The results of the analysis hold on the analysis model.

They will hold on the design model only if the transformation preserved the semantics.
Goal-Directed Certification

- NASA ARC project (Denney, Fischer, Schumann, Lowry, et al, GPCE 2005) for generating assurances for code produced by automatic program synthesis tools
  1. Add annotations to capture pre- and post-conditions
  2. Capture safety properties in some logic notation
  3. Translate 1 and 2 into verification conditions
  4. Use an automatic theorem prover to generate formal proofs
  5. Use a proof checker to check proofs

- This generates a *certificate* that the generated code satisfies the desired safety properties
- This does not verify the generator itself, but provides a certificate for the *specific* generated code
Application: Reachability in Finite State Systems

- Applying Goal-Directed Certification to Model Transformations
  - Design Language – Statecharts
  - Analysis Language - EHA
  - Property to be Verified – Reachability

- If the transition systems defined by the two models simulate each other (i.e. they are bisimilar), then reachability in one model will be equivalent to reachability in the other model
Providing a Certificate

- A certificate for a certain model for a certain property will consist of:
  - A result of the analysis on the analysis model
  - A result of the bisimilarity check

- This certificate guarantees that the analysis results hold on the design model
Let $S_A$ and $S_A'$ be two states in a transition system $A$, and $S_B$ and $S_B'$ be two states in a transition system $B$. We define a binary relation $R$ between elements of $A$ and $B$. Consider a transition $t : S_A \rightarrow S_A'$ in $A$, and $(S_A, S_B) \in R$.

The two systems are bisimilar if

- There is a transition $t' : S_B \rightarrow S_B'$ in $B$,
- and $(S_A', S_B') \in R$

$R$ is a bisimilarity relation.

If we have a way to specify $R$, we can algorithmically check if the two systems are bisimilar.
## Statechart to EHA Transformation

### Source - Statechart

- A
- B
- C
- D
- E
- F
- G
- H

### Target - EHA

- A'
- B'
- C'
- F'
- G'
- H'
- I'

### Transition Table

<table>
<thead>
<tr>
<th>Transition Label</th>
<th>SR</th>
<th>TD</th>
</tr>
</thead>
<tbody>
<tr>
<td>2'</td>
<td>G'</td>
<td></td>
</tr>
<tr>
<td>3'</td>
<td>I', F'</td>
<td></td>
</tr>
</tbody>
</table>

### Steps

1. Create top-level Sequential Automaton
2. Create a Basic State for each top level state
3. Create cross-links as elements
4. Split each Basic State into individual Sequential Automata
5. Create simple transitions
6. Create and annotate inter-level transitions

---

Cross-link

---

Sequential Automata
Verifying the Transformation

- When the target elements are created, we know what source elements they correspond to.
- But we do not know whether
  - all the source elements were considered
  - all compound states were refined correctly
  - all transitions were connected between the correct corresponding elements
  - all inter-level transitions were annotated correctly
- To verify these conditions, we check if the two models are bisimilar.
  - Using the cross-links to trace the equivalence relation $R$. 
Checking Bisimilarity

At the end of the transformation, the *cross-links* are preserved and sent to the bisimilarity checker, which performs the following steps:

- For every transition $t: S_{SC} \rightarrow S_{SC}'$ in the Statechart, find the equivalent transition $t': S_{EHA} \rightarrow S_{EHA}'$ in the EHA.
- Check if $S_{SC}$ and $S_{EHA}$ are equivalent.
- Check if $S_{SC}'$ and $S_{EHA}'$ are equivalent.

The result of the bisimilarity checker will guarantee whether the results of the analysis on the analysis model can be applied to the design model.
Conclusions

- We have shown how Goal-Directed certification can be used to guarantee properties about a design model by transforming it into an analysis model and performing the analysis on the analysis model.
- The bisimilarity checker and the model checker must be *trusted components* for the certificate to be valid.
Future Work

- Application for verifying other kinds of properties
  - Properties other than reachability
  - Constraints on values of variables
- Application to other types of transformations
  - Abstractions
- Use of theorem provers