Challenge Problem: Subject-Observer Specification with Component-Interaction Automata

Pavlína Vařeková and Barbora Zimmerova

Faculty of Informatics, Masaryk University
Brno, Czech Republic

SAVCBS’07, Dubrovnik, Croatia

September 3, 2007
1 Challenge Problem
   Subject-Observer Specification

2 Specification of the system
   Component-interaction automata
   Model with one Subject
   Model with several Subjects

3 Verification of the system
   Verification technique and optimizations
   Examples of verified properties

4 Conclusion
1 Challenge Problem
   Subject-Observer Specification

2 Specification of the system
   Component-interaction automata
   Model with one Subject
   Model with several Subjects

3 Verification of the system
   Verification technique and optimizations
   Examples of verified properties

4 Conclusion
The problem statement

- **Subject-Observer system** - many Subjects, many Observers
- Update of a Subject $\rightarrow$ notification of registered Observers
- When a Subject is notifying Observers, **no state changes allowed**
- Each Observer is called **at most once per state change**

In addition

- We add a possibility of Observers to **deregister** from Subjects
- The number of **Subjects** is **fixed**, number of **Observers** is not
- Asynchronous updating
1 Challenge Problem
   Subject-Observer Specification

2 Specification of the system
   Component-interaction automata
   Model with one Subject
   Model with several Subjects

3 Verification of the system
   Verification technique and optimizations
   Examples of verified properties

4 Conclusion
Component-interaction automata

A component-Interaction automaton (CI automaton)

- States (initial)
- Labeled transitions
- Labels (structured - component names, actions)
  - input, output and internal
- Hierarchy

\[
\begin{align*}
C_1 : & \quad p \rightarrow r \rightarrow q \\
& \quad (1, c, -) \rightarrow (1, b, 1) \\
& \quad (2, a, -) \rightarrow (3, d, 4) \\
& \quad Hierarchy: (1)
\end{align*}
\]

\[
\begin{align*}
C_2 : & \quad p \rightarrow q \\
& \quad (2, a, -) \rightarrow (3, c, 2) \\
& \quad Hierarchy: (2)
\end{align*}
\]

\[
\begin{align*}
C_3 : & \quad p \rightarrow q \\
& \quad (4, e, 3) \rightarrow (4, d, 4) \\
& \quad Hierarchy: ((3),(4))
\end{align*}
\]
Composition of CI automata

Handshake-like composition via operator $\otimes \mathcal{F}$

→ composite automaton $C = \otimes \mathcal{F}\{C_1, C_2\}$ where $\mathcal{F} = \{(2, a, 1), (1, b, 1), (1, c, 2)\}$
Model of an Observer $O^j$

- Models of all Observers $O^1, O^2, \ldots$ are the same
- Each method, e.g. `register()`, is assigned a tuple of actions: `register` represents its start, `register'` its return

```
O^j : 1 --> 2
     |   \ (j, register, -) \\
     V   (j, deregister, -) \\
        4   3 \\
            \ (j, notify', -) \\
3   \ (j, getValue', -) \\
    ^        5 \\
    (j, getValue, -) \\
    |   \ (j, notify, -) \\
    V   (j, deregister', -) \\
6   7   \ (j, register', -) \\
```

Hierarchy: (j)
Model of a Subject \( S \)

- Subject implements four methods
  - model consists of four parts connected via \( \otimes \)
- \( S = \otimes \{ S_1, S_2, S_3, S_4 \} \)
Model of an Observer $O^j$

- Suppose $n$ Subjects $S_1, S_2, \ldots, S_n$
- The model of an Observer consists of $n$ identical parts, each for communication with one Subject
- $O^j = \otimes \{O^j,i\}_{i \in \{1,\ldots,n\}}$

\[ O^j,i : (j, register_i, -) \]
\[ (-, deregister_i', j) \]
\[ (j, deregister_i, -) \]
\[ (j, notify_i', -) \]
\[ (-, notify_i, j) \]
\[ (-, getValue_i', j) \]
\[ (-, getValue_i, -) \]
\[ (j, getValue_i', j) \]
\[ (j, notify_i', j) \]
\[ (-, getValue_i, j) \]
\[ (-, notify_i, j) \]
\[ (j, register_i, -) \]

Hierarchy: $(j)$
Model with several Subjects

**Model of a Subject** $S^i$

- Each model $S^i$ analogical to $S$
- $S^i = \bigotimes\{S^i_1, S^i_2, S^i_3, S^i_4\}$
Model with several Subjects

The composite model $\mathcal{D}$

- Subjects $S^1, S^2, \ldots, S^n$
- composite Subject $S = \bigotimes\{S^i\}_{i \in \{1, \ldots, n\}}$
- Observers $O^1, O^2, \ldots$
- $\mathcal{F}$ realizing the handshake-like composition of these
1 Challenge Problem
   Subject-Observer Specification

2 Specification of the system
   Component-interaction automata
   Model with one Subject
   Model with several Subjects

3 Verification of the system
   Verification technique and optimizations
   Examples of verified properties

4 Conclusion
Verification technique

Complexity of the model $\mathcal{D}$

- Estimate the maximal number of clients that the provider is able to regard w.r.t. observable labels $X$
- Denote the number $|\mathcal{D}|_X$

Complexity of the temporal property $\{\varphi_i\}_{i \in \mathbb{N}}$

- Find the minimal number $m$ of clients that suffice to violate the property
- Then $\{\varphi_i\}_{i \in \mathbb{N}} \in \text{Property}(\mathcal{D}, m)$

A number of clients needed for the verification

- It suffices to verify the model with $0, 1, 2, \ldots, k = |\mathcal{D}|_X + m$ to conclude on the general validity of the property
Problem: In our model, the maximal number of regarded clients $|\mathcal{D}|_X$ is often $\infty$.

Solution: We introduce the following optimizations

- Move $m$ clients inside the provider $\rightarrow \overline{\mathcal{D}}$
- Narrow the property down to these clients $\rightarrow \{\overline{\varphi}_i\}_{i \in \mathbb{N}}$
- Minimize $X$ used in the computation of $|\mathcal{D}|_X$ to observe only the clients inside the provider $\rightarrow \overline{X}$

Then $\overline{\mathcal{D}}_n \models \overline{\varphi}_n$ iff $\mathcal{D}_{n+m} \models \varphi_{n+m}$ and $\{\overline{\varphi}_i\}_{i \in \mathbb{N}_0} \in Property(\overline{\mathcal{D}}, 0)$. Hence we only need to verify $\{\overline{\varphi}_i\}_{i \in \mathbb{N}}$ on $\overline{\mathcal{D}}_0, \overline{\mathcal{D}}_1, ..., \overline{\mathcal{D}}_{|\overline{\mathcal{D}}|_X}$ and $\{\varphi_i\}_{i \in \mathbb{N}}$ on $\mathcal{D}_0, \mathcal{D}_1, ..., \mathcal{D}_{m-1}$.
Property 1

If a run contains infinitely many steps concerning some Observer, then the Observer is infinitely many times enabled to receive notifications.

• Temporal formulas \( \{ \varphi_i \}_{i \in \mathbb{N}_0}, \ \varphi_i = \wedge_{j \leq i} \varphi(\alpha, j) \) where
  \[
  \varphi(\alpha, j) = [G F \bigvee_{l \in Lab_j} P(l)] \Rightarrow [G F E(\alpha, notify_1, j)]
  \]
  \[
  Lab_j = \{(j, register_1, \alpha), (\alpha, register'_1, j), (j, deregister_1, \alpha), (\alpha, deregister'_1, j), \ldots \}
  \]

• If \( \pi \not\models \varphi_i \) then \( \exists j \in \mathbb{N} : \pi \not\models \varphi(\alpha, j) \)
  Hence it suffices to observe one client to confirm the violation \( \{ \varphi_i \}_{i \in \mathbb{N}_0} \in Property(D, 1) \)

• Optimization: We modify the model to \( \overline{D} \) by moving Observer \( \beta \) inside the provider, then \( \overline{X} = Lab_\beta \) and \( |\overline{D}_{\overline{X}}| = 0 \).

• The verification (2 models) confirms that the property is valid!
Property 2

After any update, each Observer receives at most one notification about value change. This reflects that each Observer is called at most once per state change.

- Temporal formulas \( \{ \varphi_i \}_{i \in \mathbb{N}_0} \), \( \varphi_i = \bigwedge_{j \leq i} \neg \mathcal{F} \varphi(\alpha, j) \) where
  \[
  \varphi(\alpha, j) = \mathcal{P}(\alpha, \text{notify}_1, j) \land \mathcal{X} [\neg \mathcal{P}(\_, \text{update}_1, \alpha) \cup \mathcal{P}(\alpha, \text{notify}_1, j)]
  \]
  Modified to \( \varphi(\alpha, j) = \mathcal{P}(\alpha, \text{notify}_1, j) \land [\neg \mathcal{P}(\_, \text{update}_1, \alpha) \cup \{\mathcal{P}(j, \text{notify}_1', \alpha) \land [\neg \mathcal{P}(\_, \text{update}_1, \alpha) \cup \mathcal{P}(\alpha, \text{notify}_1, j)]\}] \)

- Again \( \{ \varphi_i \}_{i \in \mathbb{N}_0} \in \text{Property}(D, 1) \)

- **Optimization:** We modify the model to \( \bar{D} \) by moving Observer \( \beta \) inside the provider, then
  \[
  \bar{X} = \{(-, \text{update}_1, \alpha), (\alpha, \text{notify}_1, \beta), (\beta, \text{notify}_1', \alpha)\} \text{ and } |\bar{D} \bar{X}| = 1.
  \]

- The verification (3 models) confirms that the property is valid!
Property 3

If one of the registered Observers receives a notification and some other Observer is also ready to receive one (is registered and has not receive it yet), it will receive the notification too.

- Formulas \( \{ \varphi_i \}_{i \in \mathbb{N}_0}, \varphi_i = \bigwedge_{j_1,j_2 \leq i, j_1 \neq j_2} \varphi(\alpha, j_1, j_2) \) where
  \( \varphi(\alpha, j_1, j_2) = G \left[ (P(\alpha, \text{notify}_1, j_1) \land E(\alpha, \text{notify}_1, j_2)) \Rightarrow (\text{true} U P(\alpha, \text{notify}_1, j_2)) \right] \)

- It suffice to observe two distinct Observers, hence
  \( \{ \varphi_i \}_{i \in \mathbb{N}_0} \in \text{Property}(D, 2) \)

- Optimization: Two Observers \( \beta, \beta \beta \) need to be moved inside the provider. Then \( \overline{X} \) regards only these two, and \( |\overline{D}_{\overline{X}}| = 0 \).

- The verification (3 models) shows the property is not valid!
1 Challenge Problem
   Subject-Observer Specification

2 Specification of the system
   Component-interaction automata
   Model with one Subject
   Model with several Subjects

3 Verification of the system
   Verification technique and optimizations
   Examples of verified properties

4 Conclusion
Summary of the talk

- Solution to the *Subject-Observer challenge problem*
- Specification via *Component-Interaction automata*
- Verification via a technique presented at the workshop for verification of systems with a dynamic number of components
Thank you for your attention