Model-Based Conformance Testing of Software Product Lines
23rd CREST Open Workshop

Malte Lochau, Real-Time Systems Lab, TU Darmstadt
Nov. 20th, 2012
Model-Based Testing [UL07]

- Black-box assumption for implementation under test (iut)
- Automated derivation and application of test cases from a behavioral specification (test model)
Model-Based Conformance Testing [Tre99]

Test model (tm) conforms?  i_k, ..., i_2, i_1

Observational equivalent?  o_1, o_2, ..., o_k

Implementation model (im)

Test Hypothesis

- Test Hypothesis for test result confidence and reproducibility [Ber91]
- Partial verification of the observable behavioral conformance [NH84]
Testing Preorder Relations

Implementation relation – equivalent behaviors:

\[
\text{impl} \equiv \text{spec}
\]
Testing Preorder Relations

Implementation relation – equivalent behaviors:

\[ \text{impl} \equiv \text{spec} \]

Preorder relation – implementation conforms specification:

\[ \text{impl} \sqsubseteq \text{spec} \]
Testing Preorder Relations

Implementation relation – equivalent behaviors:

\[ \text{impl} \equiv \text{spec} \]

Preorder relation – implementation conforms specification:

\[ \text{impl} \subseteq \text{spec} \]

Model-based testing – test model as behavioral specification:

\[ \text{impl} \subseteq \text{tm} \]
Testing Preorder Relations

Implementation relation – equivalent behaviors:

\[ \text{impl} \equiv \text{spec} \]

Preorder relation – implementation conforms specification:

\[ \text{impl} \subseteq \text{spec} \]

Model-based testing – test model as behavioral specification:

\[ \text{impl} \subseteq \text{tm} \]

Black-box assumption – imaginary implementation model:

\[ \text{im} \subseteq \text{tm} \]
Testing Preorder Relations

Implementation relation – equivalent behaviors:

\[ \text{impl} \equiv \text{spec} \]

Preorder relation – implementation conforms specification:

\[ \text{impl} \sqsubseteq \text{spec} \]

Model-based testing – test model as behavioral specification:

\[ \text{impl} \sqsubseteq \text{tm} \]

Black-box assumption – imaginary implementation model:

\[ \text{im} \sqsubseteq \text{tm} \]

Weakened implementation relation – testing equivalence:

\[ \text{im} \sqsubseteq_{te} \text{tm} \]
Testing Preorder Relations

Implementation relation – equivalent behaviors:

\[ \text{impl} \equiv \text{spec} \]

Preorder relation – implementation conforms specification:

\[ \text{impl} \subseteq \text{spec} \]

Model-based testing – test model as behavioral specification:

\[ \text{impl} \subseteq \text{tm} \]

Black-box assumption – imaginary implementation model:

\[ \text{im} \subseteq \text{tm} \]

Weakened implementation relation – testing equivalence:

\[ \text{im} \subseteq_{te} \text{tm} \]

Parameterized implementation relation – finite set of behaviors:

\[ \text{im} \subseteq^{TC}_{te} \text{tm} \]
Labeled Transition Systems (LTS)

- Labeled State-Transition Graph \((\text{Proc}, \text{Act}, \rightarrow)\)
- LTS trace semantics \(tr = (a_1, a_2, \ldots, a_n) \in Tr(s_0, lts) \subseteq \text{Act}^*\), iff

\[
\begin{align*}
s_0 & \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \cdots \xrightarrow{a_n} s_n = s_0 \xrightarrow{tr} s_n
\end{align*}
\]
Labeled Transition Systems (LTS)

- Labeled State-Transition Graph \((\text{Proc}, \text{Act}, \rightarrow)\)
- LTS trace semantics \(tr = (a_1, a_2, \ldots, a_n) \in Tr(s_0, lts) \subseteq \text{Act}^*, \) iff

\[
s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \cdots \xrightarrow{a_n} s_n = s_0 \xrightarrow{tr} s_n
\]

- \(Tr(s_0, lts_1) = \{a, ab, ac\}\)
Labeled Transition Systems (LTS)

- Labeled State-Transition Graph \((\text{Proc}, \text{Act}, \rightarrow)\)
- LTS trace semantics \(tr = (a_1, a_2, \ldots, a_n) \in Tr(s_0, lts) \subseteq \text{Act}^*, \text{iff} \)
  \[
  s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \cdots \xrightarrow{a_n} s_n = s_0 \xrightarrow{tr} s_n
  \]
- \(\text{Tr}(s_0, lts_1) = \{a, ab, ac\}\)
- Trace Preorder as Testing Preorder Relation:
  \[
  \text{im} \subseteq_T \text{tm} :\iff \text{Tr}(s_0, \text{im}) \subseteq \text{Tr}(s_0, \text{tm})
  \]
Labeled Transition Systems (LTS)

- Labeled State-Transition Graph \((\text{Proc}, \text{Act}, \rightarrow)\)
- LTS trace semantics \(tr = (a_1, a_2, \ldots, a_n) \in Tr(s_0, lts) \subseteq \text{Act}^*\), iff

\[
\begin{align*}
s_0 & \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \cdots \xrightarrow{a_n} s_n = s_0 \xrightarrow{tr} s_n
\end{align*}
\]

- \(Tr(s_0, lts_1) = \{a, ab, ac\}\)

- Trace Preorder as Testing Preorder Relation:

\[
im \sqsubseteq_T tm \iff Tr(s_0, im) \subseteq Tr(s_0, tm)
\]

- Parameterized Testing Preorder Relation:

\[
im \sqsubseteq_{T^C} tm \iff (Tr(s_0, im) \cap TC) \subseteq (Tr(s_0, tm) \cap TC)
\]

where \(TC \subseteq Tr(s_0, im)\)
Example

\[(e) \ lts_1 \ \ \ (f) \ lts_2 \ \ \ (g) \ lts_3\]

\[\lts_1 \equiv_T \ lts_2 \equiv_T \ lts_3\]
Example

- $lts_1 \equiv_T lts_2 \equiv_T lts_3$

- But: different behaviors after composition with environment emitting input action $a$. 
Decorated Trace Semantics

- Trace equivalence is a *weak* equivalence
- Stricter notions of behavioral equivalence discriminate different decision structures within the state-transition graphs [Abr87]

Example: Failures and Readies

A pair $p, X q$ with $p \in \text{Act}$ and $X \subseteq \text{Act}$ is a failure of state $s_0$ if $s_0 \overset{p}{\rightarrow} s_n$ for some state $s_n$ and initials $p s_n q X$. 

A pair $p, X q$ with $p \in \text{Act}$ and $X \subseteq \text{Act}$ is a ready of state $s_0$ if $s_0 \overset{p}{\rightarrow} s_n$ for some state $s_n$ and initials $p s_n q X$. 
Decorated Trace Semantics

- Trace equivalence is a *weak* equivalence
- Stricter notions of behavioral equivalence discriminate different decision structures within the state-transition graphs [Abr87]
- **But:** testing is limited to observable behaviors

\[
\text{initials}(s) = \{ a \in Act \mid s \xrightarrow{a} \} \subseteq Act
\]
Decorated Trace Semantics

- Trace equivalence is a *weak* equivalence
- Stricter notions of behavioral equivalence discriminate different decision structures within the state-transition graphs [Abr87]
- **But:** testing is limited to observable behaviors

\[
\text{initials}(s) = \{ a \in \text{Act} \mid s \xrightarrow{a} \} \subseteq \text{Act}
\]

Example: Failures and Readies
Decorated Trace Semantics

- Trace equivalence is a weak equivalence.
- Stricter notions of behavioral equivalence discriminate different decision structures within the state-transition graphs [Abr87].

**But:** testing is limited to observable behaviors

\[
\text{initials}(s) = \{ a \in \text{Act} \mid s \xrightarrow{a} \} \subseteq \text{Act}
\]

Example: Failures and Readies

- A pair \((tr, X)\) with \(tr \in \text{Act}^*\) and \(X \subseteq \text{Act}\) is a failure of state \(s_0\) if \(s_0 \xrightarrow{tr} s_n\) for some state \(s_n\) and \(\text{initials}(s_n) \cap X = \emptyset\). 

Decorated Trace Semantics

- Trace equivalence is a *weak* equivalence
- Stricter notions of behavioral equivalence discriminate different decision structures within the state-transition graphs [Abr87]
- **But:** testing is limited to observable behaviors

\[
initials(s) = \{ a \in Act \mid s \xrightarrow{a} \} \subseteq Act
\]

Example: Failures and Readies

- A pair \((tr, X)\) with \(tr \in Act^*\) and \(X \subseteq Act\) is a *failure* of state \(s_0\) if \(s_0 \xrightarrow{tr} s_n\) for some state \(s_n\) and \(initials(s_n) \cap X = \emptyset\).
- A pair \((tr, X)\) with \(tr \in Act^*\) and \(X \subseteq Act\) is a *ready* of state \(s_0\) if \(s_0 \xrightarrow{tr} s_n\) for some state \(s_n\) and \(initials(s_n) = X\).
Preorder Relation Inclusion Hierarchy [BFvG04]
Example – Revisited

- $lts_3$ has completed trace $a$
- $lts_2 \sqsubseteq_F lts_1$
- $lts_2$ and $lts_1$ are incomparable under $\sqsubseteq_R$
Model-Based SPL Testing

- Reusable generic test model specification parameterized over features $F$
Model-Based SPL Testing

- Reusable generic test model specification parameterized over features $F$
- Reuse of test cases $T C' \subseteq T C$ of product $iut$ for product $iut'$ if
  $$tm \sqsubseteq_{te}^{T C'} tm'$$
Model-Based SPL Testing

- Extending the Test Hypothesis to SPLs under test
Model-Based SPL Testing

- Extending the Test Hypothesis to SPLs under test
- Reuse of test results $TC'' \subseteq TC'$ of product $iut$ for product $iut'$ if
  
  $$im \sqsubseteq_{te}^{TC''} im'$$
Feature-Annotated LTS [CHSL11]
LTS with transition annotations
\( \sigma(s, a, s') \in \mathcal{B}(F) \)
Feature-Annotated LTS [CHSL11]

- LTS with transition annotations
  \( \sigma(s, a, s') \in \mathbb{B}(F) \)
- Constraints by feature model
  \( fm \in \mathbb{B}(F) \)
Feature-Annotated LTS [CHSL11]

- LTS with transition annotations \( \sigma(s, a, s') \in \mathbb{B}(F) \)
- Constraints by feature model \( fm \in \mathbb{B}(F) \)
- Product configuration \( \Gamma : F \rightarrow \mathbb{B} \) (full, partial)
Feature-Annotated LTS [CHSL11]

- LTS with transition annotations 
  \( \sigma(s, a, s') \in \mathbb{B}(F) \)

- Constraints by feature model 
  \( fm \in \mathbb{B}(F) \)

- Product configuration \( \Gamma : F \rightarrow \mathbb{B} \)
  (full, partial)

- Product space 
  \( PC_{fm} = \{ \Gamma : F \rightarrow \mathbb{B} \mid \Gamma \models fm \} \)
Feature-Annotated LTS [CHSL11]

- LTS with transition annotations $\sigma(s, a, s') \in \mathcal{B}(F)$
- Constraints by feature model $fm \in \mathcal{B}(F)$
- Product configuration $\Gamma : F \rightarrow \mathbb{B}$ (full, partial)
- Product space $PC_{fm} = \{ \Gamma : F \rightarrow \mathbb{B} \mid \Gamma \models fm \}$
- Feature model refinement $fm' \sqsubseteq_{fm} fm$ is product space refinement
Transition Modalities [LT88]

\[ fm = f_1 \land (f_2 \lor f_3) \]
Transition Modalities [LT88]

\[ fm = f_1 \land (f_2 \lor f_3) \]

- **may-transitions** $\rightarrow_{may} \subseteq \rightarrow$, where
  \[ s \xrightarrow{a}_{may} s' :\Leftrightarrow \exists \Gamma \in PC_{fm} : \Gamma \models \sigma(s, a, s') \]
Transition Modalities \[LT88\]

\[
f_{m} = f_{1} \land (f_{2} \lor f_{3})
\]

- \textit{may-transitions} \(\rightarrow_{\text{may}} \subseteq \rightarrow\), where
  \[
  s \xrightarrow{a}_{\text{may}} s' \iff \exists \Gamma \in PC_{fm} : \Gamma \models \sigma(s, a, s')
  \]
- \textit{must-transitions} \(\rightarrow_{\text{must}} \subseteq \rightarrow\), where
  \[
  s \xrightarrow{a}_{\text{must}} s' \iff \forall \Gamma \in PC_{fm} : \Gamma \models \sigma(s, a, s')
  \]
Transition Modalities [LT88]

- **may-transitions** $\rightarrow_{\text{may}} \subseteq \rightarrow$, where
  
  $s \xrightarrow{a}_{\text{may}} s' \iff \exists \Gamma \in PC_{fm} : \Gamma \models \sigma(s, a, s')$

- **must-transitions** $\rightarrow_{\text{must}} \subseteq \rightarrow$, where
  
  $s \xrightarrow{a}_{\text{must}} s' \iff \forall \Gamma \in PC_{fm} : \Gamma \models \sigma(s, a, s')$

- **prohibited-transitions** $\rightarrow \subseteq \text{Proc} \times \text{Act} \times \text{Proc}$, where
  
  $s \xrightarrow{a} s' \iff \nexists \Gamma \in PC_{fm} : \Gamma \models \sigma(s, a, s')$
Transition Modalities [LT88]

\[ fm = f_1 \land (f_2 \lor f_3) \]

- **may-transitions** \( \rightarrow_{\text{may}} \subseteq \rightarrow \), where
  \[ s \xrightarrow{a}_{\text{may}} s' :\iff \exists \Gamma \in PC_{fm} : \Gamma \models \sigma(s, a, s') \]

- **must-transitions** \( \rightarrow_{\text{must}} \subseteq \rightarrow \), where
  \[ s \xrightarrow{a}_{\text{must}} s' :\iff \forall \Gamma \in PC_{fm} : \Gamma \models \sigma(s, a, s') \]

- **prohibited-transitions** \( \rightarrow \subseteq Proc \times Act \times Proc \), where
  \[ s \xrightarrow{a} s' :\iff \not\exists \Gamma \in PC_{fm} : \Gamma \models \sigma(s, a, s') \]

- \( \rightarrow_{\text{must}} \subseteq \rightarrow_{\text{may}} \)
- \( \rightarrow \cap \rightarrow_{\text{may}} = \emptyset \)
F-LTS Refinement

From $fm' \subseteq_{FM} fm$ it follows that $lts_{\Gamma'} \sqsubseteq_T lts_{\Gamma}$

- $\rightarrow\text{may}' \subseteq \rightarrow\text{may}$
- $\rightarrow\text{must} \subseteq \rightarrow\text{must}'$
- $\rightarrow \subseteq \rightarrow'$

But: this does not hold for decorated trace semantics
- Set of failures increases under refinement
- Set of readies is not subset closed
- May-transitions may become failures as well as readies after refinement
F-LTS Refinement

From $fm' \subseteq_{FM} fm$ it follows that $lts_{\Gamma'} \subseteq_T lts_{\Gamma}$

- $\rightarrow'_{\text{may}} \subseteq \rightarrow_{\text{may}}$
- $\rightarrow'_{\text{must}} \subseteq \rightarrow_{\text{must}}$
- $\rightarrow \subseteq \rightarrow'$

**But:** this does not hold for decorated trace semantics

- Set of failures increases under refinement
- Set of readies is not subset closed

$\Rightarrow$ May-transitions may become failures as well as readies after refinement
A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\) and \(X \subseteq Act\) is a \textit{may-failure} of state \(s_0\) if for each \(a \in Act\) with \(s \xrightarrow{a_{\text{must}}} s'\) it holds that \(a \notin X\).
Decorated May-Trace Semantics

- A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\) and \(X \subseteq Act\) is a \textit{may-failure} of state \(s_0\) if for each \(a \in Act\) with \(s \xrightarrow{a}_{\text{must}} s'\) it holds that \(a \notin X\).

- A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\) and \(X \subseteq Act\) is a \textit{may-ready} of state \(s_0\) if (1) for each \(a \in Act\) with \(s \xrightarrow{a}_{\text{must}} s'\) it holds that \(a \in X\), and (2) for each \(a \in Act\) with \(s \xrightarrow{a} s'\) it holds that \(a \notin X\).
A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\) and \(X \subseteq Act\) is a \textit{may-failure} of state \(s_0\) if for each \(a \in Act\) with \(s \xrightarrow{a\text{ must}} s'\) it holds that \(a \notin X\).

A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\) and \(X \subseteq Act\) is a \textit{may-ready} of state \(s_0\) if (1) for each \(a \in Act\) with \(s \xrightarrow{a\text{ must}} s'\) it holds that \(a \in X\), and (2) for each \(a \in Act\) with \(s \xrightarrow{a} s'\) it holds that \(a \notin X\).

From \(fm' \sqsubseteq_{FM} fm\) it follows that \(lts_{\Gamma}, \sqsubseteq_{te-may} lts_{\Gamma}\) holds.
Decorated May-Trace Semantics

- A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\) and \(X \subseteq Act\) is a may-failure of state \(s_0\) if for each \(a \in Act\) with \(s \xrightarrow{a} \text{must} s'\) it holds that \(a \notin X\).
- A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\) and \(X \subseteq Act\) is a may-ready of state \(s_0\) if (1) for each \(a \in Act\) with \(s \xrightarrow{a} \text{must} s'\) it holds that \(a \in X\), and (2) for each \(a \in Act\) with \(s \xrightarrow{a} s'\) it holds that \(a \notin X\).

From \(fm' \subseteq_{FM} fm\) it follows that \(lts_\Gamma', \sqsubseteq_{te\text{-may}} lts_\Gamma\) holds.

**But:** full product configurations are incomparable under \(\sqsubseteq_{te\text{-may}}\).
Must-Trace Semantics

- A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\), where \(s_i \xrightarrow{a_{must}} s_{i+1}\), for \(0 \leq i < n\), and \(X \subseteq Act\) is a \textit{must}-failure of state \(s_0\) if for each \(a \in Act\) with \(s \xrightarrow{a_{may}} s'\) it holds that \(a \notin X\)

- A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\), where \(s_i \xrightarrow{a_{must}} s_{i+1}\), for \(0 \leq i < n\), and \(X \subseteq Act\) is a \textit{must}-ready of state \(s_0\) if (1) for each \(a \in Act\) with \(s \xrightarrow{a_{must}} s'\) it holds that \(a \in X\), and (2) there is no \(a' \in Act\) with \(s \xrightarrow{a'_{may}}\) and not \(s \xrightarrow{a'_{must}}\).
Must-Trace Semantics

- A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\), where \(s_i \xrightarrow{a}_{\text{must}} s_{i+1}\), for \(0 \leq i < n\), and \(X \subseteq \text{Act}\) is a \textit{must}-failure of state \(s_0\) if for each \(a \in \text{Act}\) with \(s \xrightarrow{a}_{\text{may}} s'\) it holds that \(a \notin X\).

- A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\), where \(s_i \xrightarrow{a}_{\text{must}} s_{i+1}\), for \(0 \leq i < n\), and \(X \subseteq \text{Act}\) is a \textit{must}-ready of state \(s_0\) if (1) for each \(a \in \text{Act}\) with \(s \xrightarrow{a}_{\text{must}} s'\) it holds that \(a \in X\), and (2) there is no \(a' \in \text{Act}\) with \(s \xrightarrow{a'}_{\text{may}}\) and not \(s \xrightarrow{a'}_{\text{must}}\).

\(\Rightarrow\) From \(fm' \sqsubseteq_{FM} fm\) it follows that \(lts_{\Gamma} \sqsubseteq_{\text{te-must}} lts_{\Gamma'}\) holds.
Must-Trace Semantics

- A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\), where \(s_i \xrightarrow{a_{must}} s_{i+1}\), for \(0 \leq i < n\), and \(X \subseteq Act\) is a must-failure of state \(s_0\) if for each \(a \in Act\) with \(s \xrightarrow{a_{may}} s'\) it holds that \(a \notin X\).

- A pair \((tr, X)\) with \(s_0 \xrightarrow{tr} s\), where \(s_i \xrightarrow{a_{must}} s_{i+1}\), for \(0 \leq i < n\), and \(X \subseteq Act\) is a must-ready of state \(s_0\) if (1) for each \(a \in Act\) with \(s \xrightarrow{a_{must}} s'\) it holds that \(a \in X\), and (2) there is no \(a' \in Act\) with \(s \xrightarrow{a'_{may}} \) and not \(s \xrightarrow{a'_{must}}\).

⇒ From \(fm' \sqsubseteq FM fm\) it follows that \(lts_\Gamma \sqsubseteq_{te-must} lts_{\Gamma'}\) holds.

⇒ From \(\Gamma'' = lub(\Gamma, \Gamma')\) and \(TC = Tr_{te-must}(s_0, f-lts'')\) it follows that \(lts_\Gamma \sqsubseteq_{TC} lts_{\Gamma'}\) holds.
F-LTS Refinement Hierarchy

\[
\text{F-LTS} \quad \text{150\% \textit{tm}}
\]

\[
\text{PC-LTS} \quad \text{\&te-may}
\]

\[
\text{PC-LTS} \quad \text{\&te-must}
\]
A trace $s_0 \xrightarrow{tr} s_n$ is an *fm-constraint may-trace* if
\[ \wedge_{1 \leq i \leq n} \sigma(s_{i-1}, a_i, s_i) \models fm \] holds.

A may-failure $(tr, X)$ is an *fm-constraint may-failure* if (1) $s_0 \xrightarrow{tr} s_n$ is an *FM-constraint may-trace*, and (2) \[ \wedge_{a \in X} \neg \sigma(s_n, a, s') \models fm \] holds.

A may-ready $(tr, X)$ is an *fm-constraint may-ready* if (1) $s_0 \xrightarrow{tr} s_n$ is an *FM-constraint may-trace*, and (2) \[ \wedge_{a \in X} \sigma(s_n, a, s') \models fm \] holds.
Conclusions & Future Work

- Sample implementation for trace preorder semantics [LSKL12, LLSG12]
- Test result reuse via test model slicing [KLB12]

Future Work

- Variability-aware test result reuse criteria
- Feature-Unit testing
- Testing Equivalences with $\tau$-sensitivity $\rightarrow$ pl-ioco
- Automated SPL test suite generation
Any Questions?
Samson Abramsky.
Observation Equivalence as a Testing Equivalence.

Gilles Bernot.
10.1007/354053981663.

Precongruence Formats for Decorated Trace Semantics.

Andreas Classen, Patrick Heymans, Pierre-Yves Schobbens, and Axel Legay.
Symbolic Model Checking of Software Product Lines.
Jochen Kamischke, Malte Lochau, and Hauke Baller.
Conditioned Model Slicing of Feature-Annotated State Machines.
In 4th International Workshop on Feature-Oriented Software Development (FOSD), 2012.

Sascha Lity, Malte Lochau, Ina Schaefer, and Ursula Goltz.
Delta-oriented Model-based SPL Regression Testing.

Malte Lochau, Ina Schaefer, Jochen Kamischke, and Sascha Lity.

Kim Guldstrand Larsen and Bent Thomsen.
A Modal Process Logic.
