Interpreting a successful testing process: risk and actual coverage

Mariëlle Stoelinga, Mark Timmer
University of Twente

3th Dutch Workshop on Formal Testing Techniques
April 23, 2009
Why testing?

- Software becomes more and more complex
- Research showed that billions can be saved by testing better
- No need for the source code (black-box perspective)
# Why testing?

- Software becomes more and more complex
- Research showed that billions can be saved by testing better
- No need for the source code (black-box perspective)

# Model-based testing

- Precise and formal
- Automatic generation and evaluations of tests
- Repeatable and scientific basis for product testing
Why do we need risk and coverage?

- Testing is inherently incomplete
- Testing does increase our confidence in the system
- A notion of *quality* of a test suite is necessary
- Two fundamental concepts: risk and coverage

**Informal calculation**

**Coverage:** $\frac{6}{13} = 46\%$

**Risk:** $7 \cdot 0.1 \cdot $10 = $7$
Why do we need risk and coverage?

- Testing is inherently incomplete
- Testing does increase our confidence in the system
- A notion of *quality* of a test suite is necessary
- Two fundamental concepts: risk and coverage
Why do we need risk and coverage?

- Testing is inherently incomplete
- Testing does increase our confidence in the system
- A notion of *quality* of a test suite is necessary
- Two fundamental concepts: risk and coverage
Why do we need risk and coverage?

- Testing is inherently incomplete
- Testing does increase our confidence in the system
- A notion of *quality* of a test suite is necessary
- Two fundamental concepts: risk and coverage
Why do we need risk and coverage?

- Testing is inherently incomplete
- Testing does increase our confidence in the system
- A notion of *quality* of a test suite is necessary
- Two fundamental concepts: risk and coverage

Informal calculation

Coverage: \( \frac{6}{13} = 46\% \)
Why do we need risk and coverage?

- Testing is inherently incomplete
- Testing does increase our confidence in the system
- A notion of *quality* of a test suite is necessary
- Two fundamental concepts: risk and coverage

Informal calculation

Coverage: \( \frac{6}{13} = 46\% \)

Risk: \( 7 \cdot 0.1 \cdot \$10 = \$7 \)
## Existing coverage measures

- **Statement coverage**
- **State/transition coverage**

**Limitations:**
- All faults are considered of equal severity
- Likely locations for fault occurrence are not taken into account
- Syntactic point of view

**Existing risk measures**

- **Bach**
- **Amland**

**Limitations:**
- Informal
- Based on heuristics
- Only identify testing order for components
Existing coverage measures

- Statement coverage
- State/transition coverage

Limitations:

- all faults are considered of equal severity
- likely locations for fault occurrence are not taken into account
- syntactic point of view
## Introduction – Existing approaches

**Existing coverage measures**

- Statement coverage
- State/transition coverage

**Limitations:**

- All faults are considered of equal severity
- Likely locations for fault occurrence are not taken into account
- Syntactic point of view

**Existing risk measures**

- Bach
- Amland
### Existing coverage measures

- Statement coverage
- State/transition coverage

**Limitations:**
- All faults are considered of equal severity
- Likely locations for fault occurrence are not taken into account
- Syntactic point of view

### Existing risk measures

- Bach
- Amland

**Limitations:**
- Informal
- Based on heuristics
- Only identify testing order for components
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga

- System considered as black box
- Semantic point of view
- Fault weights
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga
- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

\[ s_0 \]
- 10ct?
- 20ct?
- tea!
- coffee!
- coffee!
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga
- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

\[
\delta
\]

\[
\begin{array}{c}
\text{10ct?} \\
\text{20ct?} \\
\text{coffee!} \\
\text{tea!} \\
\text{coffee!}
\end{array}
\]
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga
- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

\[ \delta \]

10ct? $\xrightarrow{\delta} 20ct? \\text{coffee!} \quad \text{tea!} \quad 20ct? \quad 10ct? \quad \text{coffee!} \]
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga
- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

Test cases

Interpreting a successful testing process: risk and actual coverage
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga
- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

Test cases
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga
- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems:

\[
\delta
\]
\[
10\text{ct}? \quad 20\text{ct}?
\]
\[
s_0 \quad \text{tea!}
\]
\[
\text{coffee!}
\]
\[
s_1 \quad s_2
\]

10ct? coffee! 20ct? tea! \(\delta\)

Test cases:

\[
\delta
\]
\[
10\text{ct}?
\]
\[
\text{coffee!}
\]
\[
\text{tea!}
\]
\[
\text{fail}
\]

Interpreting a successful testing process: risk and actual coverage
Starting point: semantic coverage

Previous work by Brandán Briones, Brinksma and Stoelinga

- System considered as black box
- Semantic point of view
- Fault weights

Labelled transition systems

\[ s_1 \xrightarrow{10\text{ct}?} s_0 \xrightarrow{\text{coffee}!} s_2 \xleftarrow{20\text{ct}?} s_0 \xrightarrow{\text{tea}!} s_0 \]

Test cases

\[ 10\text{ct}? \xrightarrow{\delta} \text{coffee}! \xrightarrow{\delta} \text{tea}! \]

Interpreting a successful testing process: risk and actual coverage
Weighted fault specification

A WFS consists of

- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)

\[ \delta(p_{\text{err}}(10\text{ct? coffee!}) = 0.02 \]
\[ p_{\text{err}}(20\text{ct? tea!}) = 0.03 \]
\[ w(\epsilon) = 10 \]
\[ w(10\text{ct?}) = 15 \]
\[ w(10\text{ct? coffee!}) = 9.5 \]

(For more details see TechRep)
A WFS consists of

- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)

```
δ

s0

s1

10ct? 20ct?

coffee! tea!

s2

coffee!
```

For more details see TechRep.
Weighted fault specification

A WFS consists of:
- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)

\[ p_{\text{err}}(10\text{ct? coffee!}) = 0.02 \]
\[ p_{\text{err}}(20\text{ct? tea!}) = 0.03 \]
**The WFS Model – Weighted Fault Specifications**

### Weighted fault specification

A WFS consists of:

- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)

\[ p_{\text{err}}(10\text{ct? coffee!}) = 0.02 \]
\[ p_{\text{err}}(20\text{ct? tea!}) = 0.03 \]

\[ w(\epsilon) = 10 \]
\[ w(10\text{ct?}) = 15 \]
\[ w(10\text{ct? coffee!}) = 9.5 \]
A WFS consists of

- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)

\[ p_{err}(10\text{ct? coffee!}) = 0.02 \]
\[ p_{err}(20\text{ct? tea!}) = 0.03 \]

\[ w(\epsilon) = 10 \]
\[ w(10\text{ct?}) = 15 \]
\[ w(10\text{ct? coffee!}) = 9.5 \]

(For more details see TechRep)
The WFS Model – Fault Weight

Interpreting a successful testing process: risk and actual coverage

The WFS Model

Nov. 27, 2008 8 / 18
The WFS Model – Fault Weight

Interpreting a successful testing process: risk and actual coverage

Fault weight: 10 + 15 = 25

We are only interested in whether a fault can occur, not in which one.

\[ w(\epsilon) = 10 \]
\[ w(10\text{ct?}) = 15 \]
\[ w(10\text{ct? coffee!}) = 9.5 \]
The WFS Model – Fault Weight

Fault weight: $10 + 15 = 25$

We are only interested in whether a fault can occur, not in which one.

Interpreting a successful testing process: risk and actual coverage

$w(\epsilon) = 10$

$w(10\text{ct}) = 15$

$w(10\text{ct? coffee!}) = 9.5$
Fault weight: $10 + 15 = 25$

$w(\epsilon) = 10$

$w(10\text{ct}?\ coffee!) = 9.5$
Fault weight: 10 + 15 = 25

(We are only interested in whether a fault can occur, not in which one)
Given a test suite $T$ and a passing execution $E$, we define

$$ \text{risk}(T, E) = \mathbb{E}[w(\text{Impl}) \mid \text{observe } E] $$

i.e., the fault weight still expected to be present after observing $E$. 

**Definition**

Interpreting a successful testing process: risk and actual coverage
Definition

Given a test suite $T$ and a passing execution $E$, we define

$$\text{risk}(T, E) = \mathbb{E}[w(\text{Impl}) \mid \text{observe } E]$$

i.e., the fault weight still expected to be present after observing $E$.

Observe:

$$\text{risk}(\langle \rangle, \langle \rangle) = \sum_{\sigma} w(\sigma) \cdot p_{\text{err}}(\sigma)$$
Risk

Definition

Given a test suite $T$ and a passing execution $E$, we define

$$\text{risk}(T, E) = \mathbb{E}[w(\text{Impl}) | \text{observe } E]$$

i.e., the fault weight still expected to be present after observing $E$.

Observe:

$$\text{risk}(\langle \rangle, \langle \rangle) = \sum_{\sigma} w(\sigma) \cdot p_{\text{err}}(\sigma)$$
Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?
Risk

Nondeterministic output behaviour yields difficulties. How to calculate risk (expected fault presence)?

Interpreting a successful testing process: risk and actual coverage
Interpreting a successful testing process: risk and actual coverage

Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?

Risk fail pass fail x δ x coffee! tea! s′ 0 s′ 10ct? 20ct? coffee! tea! s 1 δ s 0 s 2 s 1 10ct? 20ct? coffee! tea! s 0 coffee! tea! s 2 δ
Nondeterministic output behaviour yields difficulties.
Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?
Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?

$$\text{risk}(\langle \rangle, \langle \rangle) = \sum_{\sigma} w(\sigma) \cdot p_{err}(\sigma)$$
Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?

\[
\text{risk}(T, E) = \sum_{\sigma \neq 10\text{ct}} w(\sigma) \cdot p_{\text{err}}(\sigma) + f(10\text{ct})
\]
Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?

\[
\text{risk}(T, E) = \sum_{\sigma \neq 10\text{ct?}} w(\sigma) \cdot p_{\text{err}}(\sigma) + f(10\text{ct?})
\]
Nondeterministic output behaviour yields difficulties.

How to calculate risk (expected fault presence)?

$$\text{risk}(T, E) = \sum_{\sigma \neq 10\text{ct?}} w(\sigma) \cdot p_{\text{err}}(\sigma) + w(10\text{ct?}) \cdot \mathbb{P}[\text{error after 10ct?} | E]$$
Weighted fault specification

A WFS consists of

- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)
- A failure function (probability of failure in case of fault)
Weighted Fault Specifications (revisited)

A WFS consists of
- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)
- A failure function (probability of failure in case of fault)

\[ s_0 \xrightarrow{\delta} s_1 \]
\[ s_0 \xrightarrow{\text{tea!}} s_2 \]
\[ s_0 \xrightarrow{10\text{ct}?} s_0 \]
\[ s_0 \xrightarrow{20\text{ct}?} s_0 \]

\[ p_{\text{fail}}(\epsilon) = 1.0 \]
\[ p_{\text{fail}}(10\text{ct?}) = 0.5 \]
Weighted fault specification

A WFS consists of

- An LTS (expected system behaviour)
- An error function (probability of faults)
- A weight function (severity of faults)
- A failure function (probability of failure in case of fault)

\[
p_{\text{fail}}(\varepsilon) = 1.0 \\
p_{\text{fail}}(10\text{ct?}) = 0.5
\]
Interpreting a successful testing process: risk and actual coverage

$P_{\text{error after } 10\text{ct?} \mid \text{observation of } E}$

$p_{\text{fail}}(\epsilon) = 1.0$
$p_{\text{fail}}(10\text{ct?}) = 0.5$
\[ p_{\text{fail}}(\epsilon) = 1.0 \]
\[ p_{\text{fail}}(10\text{ct?}) = 0.5 \]

\[ \Pr[\text{error after 10ct?} \mid \text{observation of } E] \]
\[ = \Pr[\text{error after 10ct?} \mid \text{correct after 10ct? once}] \]
Risk

\[ P[A \mid B] = \frac{P[B \mid A] \cdot P[A]}{P[B]} \]

\[ p_{\text{fail}}(\epsilon) = 1.0 \]

\[ p_{\text{fail}}(\text{10ct?}) = 0.5 \]

Interpreting a successful testing process: risk and actual coverage

\[
P[\text{error after 10ct?} \mid \text{observation of } E] = P[\text{error after 10ct?} \mid \text{correct after 10ct? once}] \]

\[
\text{Bayes} \quad \frac{P[\text{correct after 10ct? once} \mid \text{error after 10ct?}] \cdot P[\text{error after 10ct?}]}{P[\text{correct after 10ct? once}]} \]
\[ p_{\text{fail}}(\epsilon) = 1.0 \]
\[ p_{\text{fail}}(10\text{ct}?) = 0.5 \]

\[
\mathbb{P}\left[\text{error after } 10\text{ct?} \mid \text{observation of } E\right] \\
= \mathbb{P}\left[\text{error after } 10\text{ct?} \mid \text{correct after } 10\text{ct? \ once}\right] \\
\overset{\text{Bayes}}{=} \frac{\mathbb{P}\left[\text{correct after } 10\text{ct? \ once} \mid \text{error after } 10\text{ct?}\right] \cdot \mathbb{P}\left[\text{error after } 10\text{ct?}\right]}{\mathbb{P}\left[\text{correct after } 10\text{ct? \ once}\right]} \\
= (1 - p_{\text{fail}}(10\text{ct?})) \cdot p_{\text{err}}(10\text{ct?})
\[ \Pr[A] = \Pr[A \mid B] \cdot \Pr[B] + \Pr[A \mid \neg B] \cdot \Pr[\neg B] \]

\[
\begin{align*}
\Pr[\text{error after 10ct? | observation of } E] &= \Pr[\text{error after 10ct? | correct after 10ct? once}] \\
&\stackrel{\text{Bayes}}{=} \frac{\Pr[\text{correct after 10ct? once | error after 10ct?}] \cdot \Pr[\text{error after 10ct?}]}{\Pr[\text{correct after 10ct? once}]}
\end{align*}
\]

\[
= (1 - p_{\text{fail}}(10\text{ct?})) \cdot p_{\text{err}}(10\text{ct?})
\]

\[
p_{\text{fail}}(\epsilon) = 1.0 \\
p_{\text{fail}}(10\text{ct?}) = 0.5
\]
\[ P[A] = P[A \mid B] \cdot P[B] + P[A \mid \neg B] \cdot P[\neg B] \]

\[ p_{\text{fail}}(\epsilon) = 1.0 \]
\[ p_{\text{fail}}(10\text{ct?}) = 0.5 \]

\[ P[\text{error after } 10\text{ct?} \mid \text{observation of } E] \]
\[ = P[\text{error after } 10\text{ct?} \mid \text{correct after } 10\text{ct? once}] \]
\[ \text{Bayes} \quad \frac{P[\text{correct after } 10\text{ct? once} \mid \text{error after } 10\text{ct?}] \cdot P[\text{error after } 10\text{ct?}]}{P[\text{correct after } 10\text{ct? once}]} \]
\[ = \frac{(1 - p_{\text{fail}}(10\text{ct?}))^1 \cdot p_{\text{err}}(10\text{ct?})}{(1 - p_{\text{fail}}(10\text{ct?}))^1 \cdot p_{\text{err}}(10\text{ct?})} \]
\[ P[A] = P[A \mid B] \cdot P[B] + P[A \mid \neg B] \cdot P[\neg B] \]

\[ p_{\text{fail}}(\epsilon) = 1.0 \]
\[ p_{\text{fail}}(10\text{ct?}) = 0.5 \]

\[ P[\text{error after 10ct?} \mid \text{observation of } E] \]
\[ = P[\text{error after 10ct?} \mid \text{correct after 10ct? once}] \]
\[ \text{Bayes} \]
\[ = \frac{P[\text{correct after 10ct? once} \mid \text{error after 10ct?}] \cdot P[\text{error after 10ct?}]}{P[\text{correct after 10ct? once}]} \]
\[ = \frac{(1 - p_{\text{fail}}(10\text{ct?}))^1 \cdot p_{\text{err}}(10\text{ct?})}{(1 - p_{\text{fail}}(10\text{ct?}))^1 \cdot p_{\text{err}}(10\text{ct?}) + (1 - p_{\text{err}}(10\text{ct?}))} \]
Risk

\[
\text{risk}(T, E) = \sum_{\sigma \neq 10\text{ct}} w(\sigma) \cdot p_{\text{err}}(\sigma) + w(10\text{ct}) \cdot \mathbb{P}[\text{error after 10ct? | E}]
\]
\[
\text{risk}(T, E) = \sum_{\sigma \neq 10\text{ct?}} w(\sigma) \cdot p_{\text{err}}(\sigma) + w(10\text{ct?}) \cdot \mathbb{P}[\text{error after } 10\text{ct?} \mid E]
\]

\[
= \sum_{\sigma \neq 10\text{ct?}} w(\sigma) \cdot p_{\text{err}}(\sigma) + w(10\text{ct?}) \cdot \frac{(1 - p_{\text{fail}}(10\text{ct?}))^1 \cdot p_{\text{err}}(10\text{ct?})}{(1 - p_{\text{fail}}(10\text{ct?}))^1 \cdot p_{\text{err}}(10\text{ct?}) + (1 - p_{\text{err}}(10\text{ct?}))}
\]
\( \text{risk}(T, E) \) \\
\[ = \sum_{\sigma \neq 10\text{ct}?} w(\sigma) \cdot p_{\text{err}}(\sigma) + w(10\text{ct}?) \cdot \mathbb{P}[\text{error after } 10\text{ct}? \mid E] \]
\[ = \sum_{\sigma \neq 10\text{ct}?} w(\sigma) \cdot p_{\text{err}}(\sigma) + \]
\[ w(10\text{ct}?) \cdot \frac{(1 - p_{\text{fail}}(10\text{ct}?))^n \cdot p_{\text{err}}(10\text{ct}?)}{(1 - p_{\text{fail}}(10\text{ct}?))^n \cdot p_{\text{err}}(10\text{ct}?) + (1 - p_{\text{err}}(10\text{ct}?))} \]
Calculation of risk

\[
\text{risk}(T, E) = \text{risk}(\langle \rangle, \langle \rangle) - \sum_{\sigma \in E} w(\sigma) \cdot \left( p_{\text{err}}(\sigma) - \frac{(1 - p_{\text{fail}}(\sigma))^{\text{obs}(\sigma,E)} \cdot p_{\text{err}}(\sigma)}{(1 - p_{\text{fail}}(\sigma))^{\text{obs}(\sigma,E)} \cdot p_{\text{err}}(\sigma) + 1 - p_{\text{err}}(\sigma)} \right)
\]

with \(\text{obs}(\sigma, E)\) the number of observations in \(E\) after \(\sigma\).
Calculation of risk

\[
\text{risk}(T, E) = \text{risk}(\langle \rangle, \langle \rangle) - \sum_{\sigma \in E} w(\sigma) \cdot \left( p_{\text{err}}(\sigma) - \frac{(1 - p_{\text{fail}}(\sigma))^{\text{obs}(\sigma, E)} \cdot p_{\text{err}}(\sigma)}{(1 - p_{\text{fail}}(\sigma))^{\text{obs}(\sigma, E)} \cdot p_{\text{err}}(\sigma) + 1 - p_{\text{err}}(\sigma)} \right)
\]

with \(\text{obs}(\sigma, E)\) the number of observations in \(E\) after \(\sigma\).

Although \(\text{risk}(\langle \rangle, \langle \rangle) = \sum_{\sigma} w(\sigma) \cdot p_{\text{err}}(\sigma)\) seems infinite, it can be calculated smartly:

- \(w\) defined by truncation: the sum is already finite
- \(w\) defined by discounting: system of linear equations
Other Applications

Optimisations

- Find the optimal test suite of a given size
- Apply history-dependent backwards induction (Markov Decision Theory)
Other Applications

Optimisations

- Find the optimal test suite of a given size
- Apply history-dependent backwards induction (Markov Decision Theory)

Actual Coverage

- Only consider the traces that were actually tested
- Use error probability reduction as coverage measure
- Methods very similar to risk
Probabilities might be hard to find, but

- We show what can be calculated, and the required ingredients
- We facilitate sensitivity analysis
- To compute numbers, we have to start with numbers...
Probabilities might be hard to find, but
- We show what can be calculated, and the required ingredients
- We facilitate sensitivity analysis
- To compute numbers, we have to start with numbers...

It looks like we need many probabilities and weights, but
- The framework can be applied at higher levels of abstraction
- Compute risk based on error / failure probabilities of modules
Main results

- Formal notion of risk
- Both evaluation of risk \textit{and} computation of optimal test suite
- Easily adaptable to be used as a coverage measure
Main results

- Formal notion of risk
- Both evaluation of risk and computation of optimal test suite
- Easily adaptable to be used as a coverage measure

Directions for Future Work

- Validation of the framework: tool support, case studies
- Dependencies between errors
- On-the-fly test derivation
Conclusions and Future Work

Main results

- Formal notion of risk
- Both evaluation of risk and computation of optimal test suite
- Easily adaptable to be used as a coverage measure

Directions for Future Work

- Validation of the framework: tool support, case studies
- Dependencies between errors
- On-the-fly test derivation

For more details, see the technical report
(http://fmt.cs.utwente.nl/~timmer)