Program Testing and Analysis: GUI Testing

Dr. Michael Pradel

Software Lab, TU Darmstadt
Warm-up Quiz

What does the following code print?

```javascript
var a = (0.1 + 0.2) + 0.3;
var b = 0.1 + (0.2 + 0.3);
console.log(a === b);
```

true  false  Something else
Warm-up Quiz

What does the following code print?

```javascript
var a = (0.1 + 0.2) + 0.3;
var b = 0.1 + (0.2 + 0.3);
console.log(a === b);
```

true  false  Something else
What does the following code print?

```javascript
var a = (0.1 + 0.2) + 0.3;
var b = 0.1 + (0.2 + 0.3);
console.log(a === b);
```

Floating point numbers are represented with finite precision (not only in JavaScript)

true  false  Something else
Warm-up Quiz

What does the following code print?

```javascript
var a = (0.1 + 0.2) + 0.3;
var b = 0.1 + (0.2 + 0.3);
console.log(a === b);
```

0.30000000000000004
(due to rounding)

true false Something else
GUI Testing

- Test application via its **graphical user interface (GUI)**

- Possible approaches
  - Manual testing
  - Semi-automated, e.g., Selenium

- Here: **Automated GUI testing**
  - Purely random testing, e.g., Monkey for Android
  - Today: Sophisticated approaches beyond purely random testing
Challenges

- **Two input dimensions**
  - Sequences of events, e.g., clicks, mouse movements (focus of today’s lecture)
  - Input values, e.g., strings entered into form

- **Not all events are known ahead of time**
  - Web apps load content dynamically

- **Need a test oracle**
  - When does an execution expose a bug?

- **Huge search space**
Huge Search Space

Challenge: **Huge search space**

- Too large to explore exhaustively
Huge Search Space

Approach: Steer search toward potential bugs or not yet explored behavior
Outline

■ Feedback-directed GUI testing
  Based on A Framework for Automated Testing of JavaScript Web Applications, Artzi et al., ICSE 2011

■ Model inference-based testing
  Based on Guided GUI Testing of Android Apps with Minimal Restart and Approximate Learning, Choi et al., OOPSLA 2013

■ Responsiveness testing
  Based on EventBreak: Analyzing the Responsiveness of User Interfaces through Performance-Guided Test Generation, Pradel et al., OOPSLA 2014
Artemis

Feedback-directed generation of GUI tests

- Start with randomly selected events
- Gather feedback from execution
- Steer toward particularly interesting behavior
- Implemented for web applications
- Test oracle: JavaScript exceptions and invalid HTML
Example

Application with 3 buttons:

```javascript
if (prepared1)
  if (prepared2)
    window.alert("hi");
```

Initially, `prepared1=prepared2=false`
Gathering Feedback

Feedback gathered while executing generated sequences of events:

- Available events
- Source code of handlers attached to events
- Memory locations read & written
- Branch coverage
Artemis: Algorithm

- Input: URL $u$
- Add sequence $[load \ u]$ to worklist
- While worklist not empty
  - Execute next sequence and gather feedback
  - Add new sequences to worklist
    - Modify inputs of existing sequence
    - Extend sequence with additional event
    - Create new sequence with new start URL
  - Prioritize worklist
Prioritization by Coverage

Coverage-guided prioritization

- Keep track of branch points in each handler
  - Branch here means entry of handler or control flow branch
- Prioritize sequences that trigger handlers with low coverage

\[
P(e_1, \ldots, e_k) = 1 - cov(e_1) \cdot \ldots \cdot cov(e_k)
\]

where \( cov(e) = \frac{\text{covered branches of } e\text{'s handler}}{\text{all discovered branches of } e\text{'s handler}} \)
Example

Application with 3 buttons:

Initially, \texttt{prepared1=prepared2=false}

```javascript
prepared1 = true;
p
```
Suppose to have executed:

Prepare 2, Process

\[
\begin{align*}
\text{cov (Prepare1)} &= \frac{0}{1} = 0 \\
\text{cov (Prepare2)} &= \frac{1}{1} = 1 \\
\text{cov (Process)} &= \frac{2}{5}
\end{align*}
\]

Possible next sequences:

- Prepare 2, Process, Prepare 1 \rightarrow P = 1 - 1 \cdot \frac{2}{5} \cdot 0 = \frac{3}{5}
- Prepare 2, Prepare 1 \rightarrow P = 1 - 1 \cdot \frac{2}{5} \cdot 1 = \frac{3}{5}
- Process, Prepare 1 \rightarrow P = 1 - 1 \cdot \frac{2}{5} \cdot \frac{2}{5} = \frac{24}{25}

\[\text{highest priority} \rightarrow \text{execute next}\]
Prioritization by Memory Accesses

Prioritize based on read/write sets

- Keep track of memory locations read/written by each handler
- Prioritize sequences where some handlers write values read by a subsequence handler
- Intuition: Can cover interesting behavior only after some handlers have set the right pre-conditions
Outline

- Feedback-directed GUI testing
  Based on A Framework for Automated Testing of JavaScript Web Applications, Artzi et al., ICSE 2011

- Model inference-based testing
  Based on Guided GUI Testing of Android Apps with Minimal Restart and Approximate Learning, Choi et al., OOPSLA 2013

- Responsiveness testing
  Based on EventBreak: Analyzing the Responsiveness of User Interfaces through Performance-Guided Test Generation, Pradel et al., OOPSLA 2014
SwiftHand

- Challenge: Restarting the application is expensive
- Learn finite-state model of application while exploring it
  - Explore states with unknown outgoing transitions
  - Continuously refine model by splitting states
- Explores application with small number of restarts
Outline

- Feedback-directed GUI testing
  Based on *A Framework for Automated Testing of JavaScript Web Applications*, Artzi et al., ICSE 2011

- Model inference-based testing
  Based on *Guided GUI Testing of Android Apps with Minimal Restart and Approximate Learning*, Choi et al., OOPSLA 2013

- Responsiveness testing
  Based on *EventBreak: Analyzing the Responsiveness of User Interfaces through Performance-Guided Test Generation*, Pradel et al., OOPSLA 2014
Motivation

Event-based UI applications should be responsive

single thread of execution
Motivation

Event-based UI applications should be responsive

Click

single thread of execution
Motivation

Event-based UI applications should be responsive

click

single thread of execution
Motivation

Event-based UI applications should be responsive

click  click

single thread of execution
Motivation

Event-based UI applications should be responsive
Real-World Example

My great site

Getting Started

It's easy to get started creating your website. Knowing some of the basics will help.

What is a Content Management System?

A content management system is software that allows you to create and manage webpages easily by separating the creation of your content from the mechanics required to present it on the web.

In this site, the content is stored in a database. The look and feel are created by a template. Joomla! brings together the template and your content to create web pages.

Joomla bug 30274
Real-World Example

My great site

Getting Started

What is a Content Management System?

A content management system is software that allows you to create and manage webpages easily by separating the creation of your content from the mechanics required to present it on the web.

In this site, the content is stored in a database. The look and feel are created by a template. Joomla! brings together the template and your content to create web pages.
Real-World Example

Admin

Menu Items

Menu

Joomla bug 30274
Real-World Example

Joomla bug 30274
Real-World Example

Joomla bug 30274
Real-World Example

Joomla bug 30274
Real-World Example

Joomla bug 30274
Real-World Example

Unresponsive

Joomla bug 30274
Real-World Example

Cost plot for responsiveness problem

Cost of saving menu

Number of menu items

Joomla bug 30274
Real-World Example

Cost plot for responsiveness problem

Unbounded growth: Unresponsive application
EventBreak: Idea

Analyze responsiveness of web applications through automated testing

Focus: Slowdown pairs

Event $E_{cause}$ increases cost of event $E_{effect}$
Overview

Dynamic analysis of application

Event-cost history

Infer potential slowdown pairs
Infer finite state model of application

Targeted test generation:
Verify slowdown pairs

Slowdown pairs with cost plots
Overview

Dynamic analysis of application

Event-cost history

Infer potential slowdown pairs

Infer finite state model of application

Targeted test generation: Verify slowdown pairs

Slowdown pairs with cost plots
Event-Cost History

Sequence of event-cost pairs

- DOM element
- Type of event
- Pre-state
- Post-state
- Number of conditionals evaluated in event handler
## Potential Slowdown Pairs

### Does A increase cost of B?

<table>
<thead>
<tr>
<th>Event</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>5</td>
</tr>
<tr>
<td>A</td>
<td>3</td>
</tr>
<tr>
<td>B</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td>12</td>
</tr>
<tr>
<td>A</td>
<td>3</td>
</tr>
<tr>
<td>B</td>
<td>12</td>
</tr>
</tbody>
</table>

**Supporting evidence $S$**

**Refuting evidence $R$**

$$Supp = |S| = 1$$

$$Conf = \frac{|S|}{|S| + |R|} = 33\%$$
Algorithm

- **Input:** History of event-cost pairs \((e, c)\) with unique events \(E\)

- **For each** \((e_1, e_2) \in E \times E\)
  - Initialize \(S(e_1, e_2) = 0\) and \(R(e_1, e_2) = 0\)
  - **For each** minimal subsequence \((e_2, c_2^1), \ldots, (e_2, c_2^2)\) in history
    - * If \(c_2^1 < c_2^2\) and \(e_1\) in subsequence, \(S(e_1, e_2)++\)
    - * If \(c_2^1 < c_2^2\) xor \(e_1\) in subsequence, \(R(e_1, e_2)++\)

- **Output** \((e_1, e_2)\) if \(S(e_1, e_2) > \text{min}_{\text{supp}}\) and
  \[
  \frac{S(e_1, e_2)}{S(e_1, e_2) + R(e_1, e_2)} > \text{min}_{\text{conf}}
  \]
Quiz

What are support and confidence for the potential slowdown pair (A,B)?

<table>
<thead>
<tr>
<th>Event</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>14</td>
</tr>
<tr>
<td>A</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>7</td>
</tr>
<tr>
<td>B</td>
<td>18</td>
</tr>
<tr>
<td>A</td>
<td>4</td>
</tr>
<tr>
<td>B</td>
<td>18</td>
</tr>
<tr>
<td>A</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>9</td>
</tr>
<tr>
<td>B</td>
<td>22</td>
</tr>
<tr>
<td>Event</td>
<td>Cost</td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>B</td>
<td>14</td>
</tr>
<tr>
<td>A</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>7</td>
</tr>
<tr>
<td>B</td>
<td>18</td>
</tr>
<tr>
<td>A</td>
<td>14</td>
</tr>
<tr>
<td>B</td>
<td>18</td>
</tr>
<tr>
<td>A</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>9</td>
</tr>
<tr>
<td>B</td>
<td>22</td>
</tr>
</tbody>
</table>

\[ L_1 = 2 \]

\[ 1R_1 = 1 \]

\[ \text{support} = 2 \]

\[ \text{confidence} = \frac{2}{1+2} = \frac{2}{3} \]
Targeted Test Generation

Confirm or reject slowdown pair:
Alternate between $E_{effect}$ and $E_{cause}$
Targeted Test Generation

Confirm or reject slowdown pair:
Alternate between $E_{\text{effect}}$ and $E_{\text{cause}}$

$E_{\text{cause}} = \text{Save new item}$

$E_{\text{effect}} = \text{Save menu}$

Costs of $E_{\text{effect}}$:

- current state
- target event
Targeted Test Generation

Confirm or reject slowdown pair:
Alternate between $E_{effect}$ and $E_{cause}$

$E_{cause} = \text{Save new item}$

$E_{effect} = \text{Save menu}$

Costs of $E_{effect}$:

Current state: saves the menu.

Target event: saves a new item.
Targeted Test Generation

Confirm or reject slowdown pair:
Alternate between $E_{effect}$ and $E_{cause}$

$E_{cause} = \text{Save new item}$

$E_{effect} = \text{Save menu}$

Costs of $E_{effect}$:
Confirm or reject slowdown pair:
Alternate between $E_{effect}$ and $E_{cause}$

$E_{cause} = \text{Save new item}$

$E_{effect} = \text{Save menu}$

Costs of $E_{effect}$: 5

Current state

Target event
Targeted Test Generation

Confirm or reject slowdown pair:
Alternate between $E_{effect}$ and $E_{cause}$

$E_{cause} = \text{Save new item}$

$E_{effect} = \text{Save menu}$

Costs of $E_{effect}$:
5

current state  \rightarrow  target event
Targeted Test Generation

Confirm or reject slowdown pair:
Alternate between $E_{effect}$ and $E_{cause}$

$E_{cause} = \text{Save new item}$

$E_{effect} = \text{Save menu}$

Costs of $E_{effect}$: 5
Targeted Test Generation

Confirm or reject slowdown pair:
Alternate between $E_{effect}$ and $E_{cause}$

$E_{cause} = \text{Save new item}$

$E_{effect} = \text{Save menu}$

Costs of $E_{effect}$: 5

current state  \rightarrow  target event
Targeted Test Generation

Confirm or reject slowdown pair:
Alternate between $E_{effect}$ and $E_{cause}$

$E_{cause} = \text{Save new item}$

$E_{effect} = \text{Save menu}$

Costs of $E_{effect}$:
5

current state \hspace{1cm} target event
Targeted Test Generation

Confirm or reject slowdown pair:
Alternate between $E_{effect}$ and $E_{cause}$

$E_{cause} = \text{Save new item}$

$E_{effect} = \text{Save menu}$

Costs of $E_{effect}$:
5, 10, 15, etc.
Navigation: Challenge

Shortest event sequence in model may be infeasible in application
Navigation: Challenge

Shortest event sequence in model may be infeasible in application

\[ E_{\text{cause}} = \text{Save new item} \]

\[ E_{\text{effect}} = \text{Save menu} \]
Navigation: Challenge

Shortest event sequence in model may be infeasible in application

$E_{cause} = \text{Save new item}$

$E_{effect} = \text{Save menu}$

Insert text,

Save new item

current state  target event
Navigation: Challenge

Shortest event sequence in model may be infeasible in application

\[ E_{cause} = \text{Save new item} \]

\[ E_{effect} = \text{Save menu} \]

Insert text, Save new item
Navigation: Approach

- Compute sequences to target event
- Randomly pick from set of first steps

\[ E_{\text{cause}} = \text{Save new item} \]

\[ E_{\text{effect}} = \text{Save menu} \]

Insert text, Save new item
Evaluation

Find responsiveness problems?
■ 2 known + 4 previously unknown

Effectiveness of targeted test generation?
■ Reaches 89% of all target events
■ Invalidates $> 99\%$ of all potential slowdown pairs
Results: Joomla

![Graph showing cost of saving menu versus number of menu items]

Cost of saving menu

Number of menu items

Joomla bug 30274
Several similar examples:

- $E_{cause}$ accumulates data items
- $E_{effect}$ processes all of them and has unbounded cost
Summary: GUI Testing

- Automated system-level testing
- Black-box and white-box approaches to explore huge search space
  - Artemis: Whitebox
  - SwiftHand and EventBreak: Mostly blackbox
- Different test oracles possible
  - Application crashes (robustness testing)
  - Consistency criterion, e.g., HTML validation
  - Responsiveness (performance testing)