Stoat: Guided, Stochastic Model-based GUI Testing of Android Apps

Ting Su
Research Fellow@NTU
Mobile Apps (Android)

Ensuring app quality is *challenging*

**Time-to-market**
- Short delivery cycle
- 50,000+ new apps per month

**Event-centric**
- User and system events
- 30+ GUI widgets & 10+ user gestures

**Environment Interplay**
- Users, SDKs, sensors, and other apps
- 24,000+ different device models
## Existing Mobile App Testing Techniques

<table>
<thead>
<tr>
<th>Approach</th>
<th>Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Testing/Fuzzing</td>
<td>Google Monkey, WCTester[FSE’16-ind], Dynodroid[FSE’13]</td>
</tr>
<tr>
<td>Symbolic Execution</td>
<td>ACTeve[FSE’12], JPF-Android[SSEN’12]</td>
</tr>
<tr>
<td>Evolutionary (Genetic) Algorithm</td>
<td>Evodroid[FSE’14], Sapienz[ISSTA’16]</td>
</tr>
<tr>
<td>Other Approaches</td>
<td>MonkeyLab[MSR’15], CrashScope[ICST’16], TrimDroid[ICSE’16], EHB_Droid [ASE’17]</td>
</tr>
</tbody>
</table>
Our Approach --- Stoat

- Stoat (*Stochastic model App Tester*)
  - A *guided, stochastic model-based* GUI testing approach
  - A *fully-automatic* tool for testing/fuzzing Android apps

- Given an app as input,
  1. Model Construction
     * Use *dynamic/static analysis* to learn a stochastic model
  2. Test Generation and Optimization
     * Adopt *Gibbs sampling* to iteratively mutate/refine the model
     * Validate apps with various *user/system-level events*
Evaluation & Effectiveness

• Subjects
  • 93 benchmark apps

• Outperform existing techniques

17~31% higher code coverage
3X more unique crashes
Evaluation & Effectiveness

• Subjects
  • 93 benchmark apps
  • 1661 Google Play apps

• Contribute to real-world apps

<table>
<thead>
<tr>
<th>ID</th>
<th>Exception Type</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NullPointerException</td>
<td>1226</td>
</tr>
<tr>
<td>2</td>
<td>Windows Leaked Exception</td>
<td>255</td>
</tr>
<tr>
<td>3</td>
<td>ActivityNotFoundException</td>
<td>191</td>
</tr>
<tr>
<td>4</td>
<td>SQLite Related Exception</td>
<td>71</td>
</tr>
<tr>
<td>5</td>
<td>IllegalStateException</td>
<td>47</td>
</tr>
<tr>
<td>6</td>
<td>IllegalArgumentException</td>
<td>37</td>
</tr>
<tr>
<td>7</td>
<td>RuntimeException</td>
<td>21</td>
</tr>
<tr>
<td>8</td>
<td>ClassCastException</td>
<td>9</td>
</tr>
<tr>
<td>9</td>
<td>UnsatisfiedLinkError</td>
<td>8</td>
</tr>
<tr>
<td>10</td>
<td>WindowManager$BadTokenException</td>
<td>4</td>
</tr>
<tr>
<td>11</td>
<td>Other Exceptions</td>
<td>233</td>
</tr>
</tbody>
</table>

2110 unique previously-unknown crashes from 691 apps
Contribute to the apps with billions of users
Evaluation & Effectiveness

• Subjects
  • 93 benchmark apps
  • 1661 Google Play apps
  • 2104 F-droid apps (total 4560 versions)

• Effective bug detection
  • Detected 3535 unique app crashes
  • Categorized into 75 types of errors
Technical Innovation

Key Technique

- Learn a behaviour model for an app
- Sample tests to optimize test generation
- Enforce various user/system interactions

https://youtu.be/v4UkJgdcWDQ
(or https://v.youku.com/v_show/id_XMzA0Nzc4MTcyNA )

https://youtu.be/Xk7A7wczLj0
(or https://v.youku.com/v_show/id_XMzA0Nzc4NjYxMg )
Workflow of Stoat

Phase 1. Model Construction
1. static event identification
2. dynamic UI exploration (weighted UI exploration)
3. static analysis

Events → States → System-level Events → initial stochastic FSM

Phase 2. Model Mutation, Test Generation, and Execution
7. test execution
6. inject events
5. test generation (probability-based test generation)
4. mutate probabilities

Test Suite

Bug Coverage & Diversity

stochastic FSM

Test Optimization Goal
✓ Statement coverage
✓ Model coverage
✓ Event sequence diversity
✓ Inject 113+ user/system events

Features & Usability

- **End-to-end, server-client** distributed testing;
- Support *binary* and *open-source* apps on *real devices* and *emulators*;
- Generate model/class/method/line coverage reports; Bug-triggering tests/screenshots
Summary

• Tool: Stoat (Stochastic model App Tester)
  • A Guided, Stochastic model-based GUI testing approach
  • Tested 6000+ APKs, detected 5800+ fatal crashes

• Goal
  • Thoroughly test various usage scenarios of an app;
  • Enforce environmental interplay

• Publication
  • Guided, Stochastic Model-Based GUI Testing of Android Apps (ESEC/FSE’17)
  • FSMdroid: Guided GUI Testing of Android Apps (First Prize of ACM SRC@ICSE 2016)
  • Large-Scale Analysis of Framework-Specific Exceptions in Android Apps (ACM SIGSOFT Distinguished Paper Award@ICSE 2018)
  • Efficiently Manifesting Asynchronous Programming Errors in Android Apps (ASE 2018)

• https://tingsu.github.io/files/stoat.html