TouchGuru: Static Analysis for Mobile Development Environments

Peter Müller
ETH Zurich

Joint work with Lucas Brutschy and Pietro Ferrara
Challenges of Mobile App Development

- Event-based execution
- Mobile environment
- Persistent storage
- Diverse platforms

- Education
- End-user programming
- Independent developers
Cloud-Based Static Analysis

```
action main ()
| var pic := senses → take camera picture
| pic → resize(100, 100)
end action
```

TouchGuru

Analyze app for all:
- event schedules
- platforms
- environment interactions

```
action main ()
| var pic := senses → take camera picture
| pic → resize(100, 100)
| ✽ When calling resize: Object whose field/method is accessed might be invalid
end action
```
TouchDevelop Language

- Statically-typed, imperative
- Sequential execution
- Code is structured into
  - actions (procedures)
  - event handlers
- Data is stored in
  - local variables
  - persistent global variables
  - heap records
- Rich API
Overview of Static Analysis: Actions

- Semantics of statements

\[ S : Stmt \rightarrow (\Sigma \rightarrow (Label \rightarrow \Sigma)) \]

- Semantics of any action

\[ \mathcal{A} (\sigma) := \lambda l. \bigcup_{a \in Actions} (S[a] (\sigma) (l)) \]
Overview of Static Analysis: Events

- Semantics of any event

\[ E(\tau) := \lambda l. \bigcup_{e \in \text{Events}} S[e] \left( \bigcup_{l' \in \text{Exit} \tau(l')} l \right) \]

- Semantics of an app

\[ L(\sigma) := \text{lfp}_{A(\sigma)} \lambda \tau. (\tau \triangledown E(\tau)) \]
Challenge 1: Late Failing

- To increase robustness, many erroneous operations result in invalid value

```latex
\begin{align*}
\textbf{if} & \ 0 \leq \text{index} \leq \text{boards} \rightarrow \text{size} \\
\textbf{then} & \quad b := \text{boards} \rightarrow \text{at( index )}; \\
\textbf{else} & \quad b := \text{media} \rightarrow \text{create board}; \\
& \quad b \rightarrow \text{post to wall};
\end{align*}
```

- Abstract domain tracks origin of invalid values for better error reporting

\[
\text{InvalidDom} := \text{Id} \rightarrow \mathcal{P} ( \{ \text{Valid} \} \cup ( \{ \text{Invalid} \} \times \text{Label} ) )
\]
Challenge 2: Mobile Environment

- Some aspects of the mobile environment
  - are stable
  - change occasionally
  - change frequently

```plaintext
var x := senses \rightarrow acceleration \text{ stable} \rightarrow x \text{ X}

if senses \rightarrow has accelerator \text{ then}
  var x := senses \rightarrow acceleration \text{ stable} \rightarrow x \checkmark

\begin{align*}
\text{media} & \rightarrow \text{songs} \rightarrow \text{random} \rightarrow \text{play} \text{ X} \\
\text{if not media} & \rightarrow \text{songs} \rightarrow \text{is empty} \text{ then} \\
\text{media} & \rightarrow \text{songs} \rightarrow \text{random} \rightarrow \text{play} \checkmark
\end{align*}

\begin{align*}
\text{if senses} & \rightarrow \text{has accelerometer} \text{ then} \\
  \text{var x1 := senses} & \rightarrow \text{acceleration} \text{ stable} \rightarrow x \\
  \text{var x2 := senses} & \rightarrow \text{acceleration} \text{ stable} \rightarrow x \\
  \text{if} x1 & \neq x2 \text{ then abort} \text{ X}
\end{align*}
```
Mobile Environment: Solution

- We model the environment via three sets of abstract identifiers

\[ EId = EId_{stable} \cup EId_{occasional} \cup EId_{volatile} \]

- No information is tracked for “volatile” identifiers
- Information for “occasional” identifiers is reset before each event handler

Semantics of any event

\[ E(\tau) := \lambda l. \bigcup_{e \in Events} S[e] \left( \bigcup_{l' \in Exit} \tau(l') \bigg| EId_{occasional} \right)(l) \]
Challenge 3: Persistent Storage

Global variables are persistent

Assume that boards has three elements

Intended invariant: $0 \leq \text{level} \leq 2$

Execution may be terminated here

```data
level: Number
```
```
action main {
  ... 
  boards → at( level ) → post to wall;
  ... 
  level := level + 1;
  if level = 3 then
    "Game over!" → post to wall;
    level := 0;
  end
}```
Persistent Storage: Solution

- Fresh execution from $\sigma$
- Next execution

$\mathbb{L}(\sigma) := \text{lfp}^{\subseteq}_{\mathbb{A}(\sigma)} \lambda \tau. (\tau \triangledown E(\tau))$

$\mathbb{R}(\tau) := \mathbb{L}\left(\bigcup_{l \in \text{Label}^\tau(\tau)} | \text{Id}_{\text{Persistent}} \cup \text{EId}_{\text{stable}} \right)$
Persistent Storage: Solution (cont’d)

- Next execution
- Semantics of program

\[ \mathcal{R}(\tau) := L\left(\bigcup_{l \in \text{Label}\tau(l)}|\text{Id}_{\text{Persistent}} \cup \text{EId}_{\text{stable}}\right) \]

\[ \mathcal{P} := \text{lfp}_{\tau_0}^\in \lambda \tau. (\tau \triangledown \mathcal{R}(\tau)) \]
Experimental Results: Performance

- Implemented in Sample
  - Inter-procedural analysis
  - Value domain: octagons, strings, invalid
  - Heap abstraction: points-to and may/must container analysis

<table>
<thead>
<tr>
<th>#Scripts</th>
<th>#LOC</th>
<th>#Alarms</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sum</td>
<td>Avg.</td>
<td>Sum</td>
</tr>
<tr>
<td>root set</td>
<td>50,431</td>
<td>17,149,776</td>
<td>340.06</td>
</tr>
</tbody>
</table>

- Complex analysis (three nested fixed points)
Experimental Results: Precision

- Manual inspection of 51 random scripts
  - 34 alarms
  - 85% precision: 29 true and 5 false alarms

- Ten most frequently executed scripts

<table>
<thead>
<tr>
<th>Script</th>
<th>PID</th>
<th>#LOC</th>
<th>#Runs</th>
<th>Time [s]</th>
<th>#True</th>
<th>#False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flip a Virtual Coin!</td>
<td>htmh</td>
<td>30</td>
<td>45,300</td>
<td>0.01</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>My Online Meetings</td>
<td>mptuj</td>
<td>333</td>
<td>41,100</td>
<td>1.47</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>WiFi/3G Swap</td>
<td>kmjn</td>
<td>44</td>
<td>40,800</td>
<td>0.07</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Line Runner</td>
<td>dvvx</td>
<td>261</td>
<td>16,600</td>
<td>11.48</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>internet speedtest</td>
<td>qwzu</td>
<td>39</td>
<td>9,150</td>
<td>0.05</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>doodle jump</td>
<td>ajkc</td>
<td>221</td>
<td>9,000</td>
<td>9.14</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>where am I ??</td>
<td>kblp</td>
<td>98</td>
<td>8,100</td>
<td>0.40</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>CloudHopper</td>
<td>wbxs</td>
<td>255</td>
<td>8,050</td>
<td>19.43</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>BreakIt! Touch</td>
<td>zids</td>
<td>180</td>
<td>7,700</td>
<td>3.98</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>doodle jump</td>
<td>ybce</td>
<td>221</td>
<td>6,700</td>
<td>6.10</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td>1,682</td>
<td>52.13</td>
<td></td>
<td>6</td>
<td>14</td>
</tr>
</tbody>
</table>
Conclusions

- TouchGuru is an efficient and precise analysis for TouchDevelop

- Some of the addressed challenges occur also in other mobile platforms

- TouchDevelop is an attractive research platform
  - Simple language, small scripts
  - Cloud support