Odysseys in Testing and Analyzing Mobile Apps

Mayur Naik
Georgia Institute of Technology
Odyssey

: A long journey full of adventures

: A series of experiences that give knowledge or understanding to someone

Source: Merriam-Webster's Dictionary
The Mobile App Life Cycle

Reliability ➔ Security ➔ Performance

Development and Testing ➔ Pre-deployment Certification ➔ Post-deployment Adaptation

New software engineering problems in all stages need new tools based on program analysis.
Three Odysseys

Reliability
Development and Testing
Dynodroid

Security
Pre-deployment Certification
Stamp

Performance
Post-deployment Adaptation
CirrusCloud
Dynodroid
Automated Testing of Mobile Apps

Key Idea: View app as an event-driven program

\[ s_0 \xrightarrow{e_1} s_1 \xrightarrow{e_2} s_2 \xrightarrow{e_3} \ldots \]

Broadly two kinds of events:

**UI event:** LongTap(245, 310), Drag(0, 0, 245, 310), ...

**System event:** BatteryLow, SMSReceived("hello"), ...

Assumption: Fixed concrete data in environment (sdcard, network, . . .)

May cause loss of coverage
Automated Testing of Mobile Apps

Key Idea: View app as an event-driven program

$s_0 \xrightarrow{- e_1} s_1 \xrightarrow{- e_2} s_2 \xrightarrow{- e_3} \ldots$

[Diagram of events and screens with 'tap(305, 544)']
Automated Testing of Mobile Apps

Key challenge: Large number of possible events
E.g., 108 system events in Android Gingerbread

Insight #1: Few events are relevant in any state

Insight #2: Many event sequences are equivalent

Our solution: Identify both conditions by specializing to app framework
Example of Equivalent Event Sequences
Example of Equivalent Event Sequences
Dynodroid achieves higher coverage than Monkey for 30 of the 50 apps.
Code Coverage: Dynodroid vs. Humans

Automation Degree = $\frac{C(\text{Dynodroid} \cap \text{Human})}{C(\text{Human})}$

Range = $8-100\%$, Average $= 83\%$, S.D. $= 21\%$
Sample Feedback from Participants

“Tried to cancel download to raise exception.”

“Human cannot trigger change to AudioFocus.”

“Many, many options and lots of clicking but no actions really involved human intelligence.”

“There are too many combinations of state changes (play -> pause, etc.) for a human to track.”
A Problem: Path Divergence

- Results from missed propagation of symbolic values in uninstrumented code
  Primarily native (C/C++) code, occasionally Java code (e.g., object serialization)

- Divergence can be your friend: served as a beacon for bugs in our implementation
  ~ 0% today in our implementation for Android (compared to ~ 40% for SAGE)
Automated Testing – Looking Ahead

Some old problems ...
Scalability (path explosion)
Demand-driven?
Which inputs to treat symbolically and when?

... and some new ones
Framework model synthesis
Debugging support to localize false alarms?
Engage user-in-the-loop?

CirrusCloud
## Smartphone Trends

<table>
<thead>
<tr>
<th>Model</th>
<th>CPU (GHz)</th>
<th>Screen Res. (thousand pixels)</th>
<th>Rear Camera (MP)</th>
<th>Front Camera (MP)</th>
<th>Sensors</th>
<th>Battery (mAh)</th>
</tr>
</thead>
<tbody>
<tr>
<td>iPhone</td>
<td>0.4</td>
<td>153</td>
<td>2</td>
<td>-</td>
<td>3 (light, accelerometer, proximity)</td>
<td>1,400</td>
</tr>
<tr>
<td>iPhone 3</td>
<td>0.6</td>
<td>153</td>
<td>3</td>
<td>-</td>
<td>4 (light, accelerometer, proximity, compass)</td>
<td>1,150</td>
</tr>
<tr>
<td>iPhone 4</td>
<td>0.8</td>
<td>614</td>
<td>5</td>
<td>0.3</td>
<td>6 (light, accelerometer, proximity, compass, gyroscope, infrared)</td>
<td>1,420</td>
</tr>
<tr>
<td>iPhone 5</td>
<td>1.3</td>
<td>727</td>
<td>8</td>
<td>1.2</td>
<td>7 (light, accelerometer, proximity, compass, gyroscope, infrared, fingerprint)</td>
<td>1,560</td>
</tr>
<tr>
<td>iPhone 6</td>
<td>2.0</td>
<td>1000</td>
<td>12</td>
<td>5.0</td>
<td>8 (light, accelerometer, proximity, compass, gyroscope, infrared, fingerprint, barometer)</td>
<td>1,715</td>
</tr>
</tbody>
</table>
A Challenge: Data growth vs. Performance

- Data growth trends: IDC’s Digital Universe Study, December 2012
An Opportunity: Mobile-Cloud Computing

![Diagram showing the relationship between mobile devices and cloud computing systems]

- Performance vs. years graph
- Mobile devices resident on vehicles
- Central Cloud
- Cloudlets
- Mobile Devices

Microsoft Research
Faculty Summit 2016
It’s Already Here

“Dialing 123-456-7890”

“Call Mary!”

Call (Mary)
Challenges to Broader Use

1. Interleaved I/O and computation

2. Network latency

3. Diverse and dynamic environments
Our Contributions

Interleaved I/O and computation

Diverse and dynamic environments

Optimization Problem

ILP → Min-Cut

Remote State

Network latency

Transient → Persistent

Bi-directional

Uni-directional

Communication Pattern

No installation. No storage. Use & forget.
Overall Approach

offload
optimal and valid
resume

offloading

traces
models

Microsoft Research
Faculty Summit 2016
Previous Approaches: ILP

30 million instructions > 8 hours

traces → optimal and valid offloading

models → ILOG CPLEX

resumes → offload

3G → WiFi

Microsoft Research Faculty Summit 2016
Our Approach: Min Cut

- traces -> Min-Cut
- 450 million instructions
- optimal and valid offloading
- < 5 seconds
- models
- offload
- resume
Speedup on Galaxy S3 – WiFi and LTE

![Graphs showing speedup comparison between Transient Unidirectional and Persistent Bidirectional for various applications.]
Speedup on Galaxy S2 – WiFi and LTE

- Transient Unidirectional
- Persistent Bidirectional

![Graph 1](https://via.placeholder.com/150)

- aMetro
- collision
- fileSearch
- mosaic
- objTrack
- stabilize
- sudoku
- zxing
- geomean

![Graph 2](https://via.placeholder.com/150)

- aMetro
- collision
- fileSearch
- mosaic
- objTrack
- stabilize
- sudoku
- zxing
- geomean

Microsoft Research
Faculty Summit 2016
Speedup of S3 over S2
Quality for Video Apps

![Graph showing frames per second for different applications and methods: local, transient unidirectional, and persistent bidirectional.]
Demo: Video Stabilization
Partitioning and Offloading – Looking Ahead

Leveraging Accelerators in the Cloud

Specialized Communication Protocols

Programming Models

- Approximate computing [safety, quality requirements, ...]
- Distributed computing [consistency, fault tolerance, ...]

Sequential  Concurrent  Weakly consistent  Partially consistent
Challenge or Opportunity?

- Hard to analyze (e.g., native code, reflection)
- Very large
- Mostly irrelevant to the analysis

Application
Framework (e.g. Android)
Challenge or Opportunity?

Application → Program Analysis Tool → False Positives/Negatives
Challenge or Opportunity?

- Summarizes behaviors relevant to analysis
- Built once and for all
- Improves scalability of analysis