Improving the Energy Consumption of Mobile Web Applications

William G.J. Halfond
Assistant Professor
University of Southern California
Mobile Web Applications

Mobile Device \leftrightarrow \text{Network} \leftrightarrow \text{Servers}
Mobile Web Apps – A Brief History

• First mobile “app” marketplace 2008
  – Now generates over $8.5 billion
• Originally, native apps
  – Choose a side: iOS vs. Android X.X vs. …
  – Bad for developers
  – Bad for users
• HTML 5 and AJAX
  – Better for everybody (except the testers!)
Verification and Validation of Mobile Web Applications

1. Energy consumption
2. User interfaces
3. Software abstractions
4. Security vulnerabilities
Energy Consumption

• Important quality metric
• Strongly influences usability
• Affects perception of app quality ★★★★★
• Goal: Help developers to understand and change energy consumption in apps
• Current techniques
  – Heavyweight
  – Too course grained
  – Very slow
Areas of Investigation

1. Understand energy consumption in apps
2. Identify best practices and optimizations
3. Automated refactorings and transformations of the implementation
Energy Consumption Projects

**vLens**: Map online measurements to source code lines

**eLens**: Predict energy consumption

**Nyx**: Automated transformation of web app pages for energy optimization

**EDTSO**: Energy directed test suite optimization for in-field testing
**vLens**: Online Power Attribution

**Goal**: Where is the energy being consumed in the application?

**Challenges**: Insufficient sampling speed, relationships between instructions, minimal runtime overhead

**Key insight**: Combine static program analysis and statistical analysis techniques
vLens: Process Overview

Runtime Measurement Phase

1. App Instrumenter
2. Power Measurement Platform

Offline Analysis Phase

3. Path Adjuster
4. Analyzer
5. Annotator

Test Cases

App

Energy Report

Annotated Code
vLens: Statistical Methods

Iterative

\[
\sum_i \varphi(y_i - \sum_k x_{ik}\theta_k)x_{ij} = 0
\]

\[
\varphi_k = \begin{cases} 
  x(k\sigma - x^2)^2, & -k\sigma < x < k\sigma \\
  0, & \text{otherwise}
\end{cases}
\]

- Handle non-linear API invocations
- Attribute tail energy
- Allocate energy among threads
- Eliminate garbage collection costs
- Remove IPC overhead
Handle Non-Linear Cost Invocations

Calculate and remove non-linear API cost

Sum up energy between entry and exit time stamps

\[
\frac{t_2 - t_1}{t_4 - t_3} E_{t_3,t_4}
\]

Minimal sampling interval
Assign tail energy to corresponding API

\[ TE_{API_1} = \frac{T_s(API_2) - T_E(API_1)}{T_{tail}} E_{tail} \]
vLens: Achievements

- High accuracy
  - 9% error
- High granularity
  - On source line level
- Low overhead
  - 4% runtime cost
eLens: Predict Energy Consumption

Combine program analysis and per instruction cost modeling

1. Lightweight → no OS changes or specialized hardware required
2. Fine-grained → feedback at the source line level
3. Accurate → within 10% of ground truth
4. Fast → estimates within minutes
eLens: Process Overview

1. Generate workload
2. Identify corresponding executed paths
3. Compute power values for paths
4. Annotate source lines
eLens: Energy Calculations

\[ \text{Energy} = \sum_{h \in \text{Hardware}} \sum_{i \in \text{path}} C_h(i) \]

- Cost functions \((C_h)\) for each component \((h)\)
- Instruction’s energy cost is either:
  - Path-independent: “fixed-cost” energy
  - Path-dependent: varies based on path

- Cost functions provided by a Software Environment Energy Profile (SEEP)
**eLens: Accuracy**

- **Application level**: eLens differs from Ground Truth, on average, 8.8%.
- **Method level**: eLens differs from Ground Truth, on average, 7.1%.

### Application level

<table>
<thead>
<tr>
<th>Application</th>
<th>Error Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBC Reader</td>
<td>-6.2</td>
</tr>
<tr>
<td>Bubble Blaster II</td>
<td>-11.6</td>
</tr>
<tr>
<td>Classic Alchemy</td>
<td>-4.4</td>
</tr>
<tr>
<td>Location</td>
<td>-7.8</td>
</tr>
<tr>
<td>Skyfire</td>
<td>-7.9</td>
</tr>
<tr>
<td>Textgram</td>
<td>5.2</td>
</tr>
</tbody>
</table>

### Method level

<table>
<thead>
<tr>
<th>Application</th>
<th>Error Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>execute</td>
<td>-6.8</td>
</tr>
<tr>
<td>createAds</td>
<td>-6.8</td>
</tr>
<tr>
<td>doInBg</td>
<td>-6.8</td>
</tr>
<tr>
<td>getInput</td>
<td>-6.8</td>
</tr>
<tr>
<td>w.a</td>
<td>-6.8</td>
</tr>
<tr>
<td>e.a</td>
<td>-6.8</td>
</tr>
</tbody>
</table>

**No more than 12% difference from Ground Truth**
Research Progression

- vLens and eLens focus on understanding energy consumption
- But we want to optimize!
- What is an energy bug?
  - Informally: Lots of energy is consumed and a more efficient way is possible
- New questions:
  - What are these best practices?
  - How do we automate them?
Nyx: Optimizing Mobile Web Apps Display

**Problem:** Display consumes a significant amount of energy on smartphones

**Goal:** Automatically transform web applications so that their display consumes less energy

**Key insight:** Darker colors consume less energy on OLED screens
Nyx: Process

1. HTML Output Analysis
   - Leverage string analysis techniques by Bultan (UCSB) and Moeller (Aarhaus)

2. Color Transformation
   - Model color relationships on page
   - Generate new color mapping scheme

3. Output Modification
# Nyx: Results

<table>
<thead>
<tr>
<th>App</th>
<th>Loading</th>
<th>Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bookstore</td>
<td>26.7</td>
<td>47.2</td>
</tr>
<tr>
<td>Portal</td>
<td>24.7</td>
<td>44.2</td>
</tr>
<tr>
<td>JavaLib</td>
<td></td>
<td>55.8</td>
</tr>
<tr>
<td>ClassRoom</td>
<td></td>
<td>51.6</td>
</tr>
<tr>
<td>Roller</td>
<td>10.4</td>
<td>18.0</td>
</tr>
<tr>
<td>Scarab</td>
<td>27.1</td>
<td>47.8</td>
</tr>
<tr>
<td>jForum</td>
<td>26.7</td>
<td>47.8</td>
</tr>
</tbody>
</table>

Average 40% savings in display power

Energy savings of transformed applications
Nyx: Transformations
Research Summary

- Verification and validation of mobile web applications
- Energy is an important quality metric
- Step 1: Understand energy consumption
- Step 2: Change energy consumption

Thank you!
halfond@usc.edu