Empowering Developers to Estimate App Energy Consumption

Raghu Rangan

Computer Science Dept.
Worcester Polytechnic Institute (WPI)
Introduction

● In the world of smartphones there are a number of mobile applications available
  ● Games, calendars, social media

● Poorly written apps can drain the battery of a phone
  ● Very frustrating for users
Battery Problems

- Battery life for smartphones has improved significantly over the past several years
- Lot of work has been done to improve battery life
  - Focus on the platform itself
    - Battery density, low power processors, the cloud
- But this work only focuses on the platform itself
  - Poorly written programs can still destroy battery life
Goal

- Create a system which allows developers “to estimate the energy consumed by his/her app in the development environment itself”
Current Offerings for Users

PowerTutor Screenshot
Related Work

- Large body of work on energy modeling for phones
  - Specifically for Palm device
  - Models for specific components (OLED displays, 3G)
- Looked at app energy accounting at run time
  - PowerScope: tracks app with active context on CPU
  - eProf: traces system calls and power state models
Related Work

● Energy emulation at development time
  ● Power TOSSIM
  ● Problem: event based simulation does not directly apply to mobile app emulation
    ● Interaction with external resources (i.e. web services)
WattsOn System Design

- Two major techniques in design
- Power Modeling
  - Alternative to using physical meter equipment
  - Compute energy of resource utilization using power models
- Resource Scaling
  - Resource counter measured on workstation cannot be fed directly into power models
  - Timing events may be different
WattsOn System Design

- Target Mobile Device
  - Resource Profiling
  - Power Profiling
  - Power Model Generation
- Mobile Phone Emulator
  - Resource Scaling
  - Hardware Resources
  - Developer Workstation
- Test Application
- Resource Profiling
- Energy Calculation

Flowchart shows the process starting with the Target Mobile Device, moving through Resource Profiling and Power Profiling, leading to Power Model Generation. The process continues through Mobile Phone Emulator, Resource Scaling, and Hardware Resources to the Test Application. Resource Profiling is involved at multiple stages, culminating in Energy Calculation and ultimately determining App Energy use.
3G Network Modeling

- Resource Scaling
  - Link Shaping
    - Shape network link bandwidth and latency
    - Emulated network in terms of packets similar to 3G link
  - Method better than Virtual Clock and Trace Stretching

- Power Model
  - Active energy consumption when communicating data
  - “Tail” time: active state after comm activity
  - ARO model used to calculate power state
3G Network
3G Network

Network Tail energy Measurement for Sprint.

Tail State Time for Various Mobile Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>DCH</th>
<th>FACH</th>
<th>PCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT&amp;T (3G)</td>
<td>5s</td>
<td>12s</td>
<td>0</td>
</tr>
<tr>
<td>T-Mobile (3G)</td>
<td>5s</td>
<td>1s</td>
<td>1s</td>
</tr>
<tr>
<td>T-Mobile (4G HSPA)</td>
<td>4s</td>
<td>2s</td>
<td>1s</td>
</tr>
<tr>
<td>Verizon (3G)</td>
<td>6s</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sprint (3G)</td>
<td>10s</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
WiFi Network Modeling

- Resource Scaling
  - Same approach 3G modeling if dev machine not on WiFi

- Power Model
  - PSM state model
  - Deep Sleep(10mW), Light Sleep(120mW), Idle(400mW), and High(600mW)
Display Modeling

- Resource Scaling
  - Existing mobile device emulators perform this
  - Emulator window can be resized accordingly

- Power Model
  - Models exist for LCD and OLED displays
  - Modern devices use Active Matrix OLED (AMOLED)
    - Does not fit existing models
Display Modeling

Figure 4: Power measurements for different colors on an AMOLED display.
Display Modeling

Figure 5: AMOLED power changes as the fraction of white colored pixels changes.

Figure 6: Change in power with color.

Resulting Model Equation

\[ P_{\text{display}} = \beta(s) \times L(s) + (1 - \beta(s)) \times O(s) \]  

(1)
CPU Modeling

- **Resource Scaling**
  - Scale down the performance of emulated app running on dev machine
  - Restrict # of processor cycles available to emulator

- **Power Model**
  - Power models exist for CPUs
  - Simple utilization based power model

\[
P_{\text{cpu}} = \alpha \cdot u_{\text{cpu}} \quad (2)
\]
Implementation

- WattsOn integrated with Windows Phone Emulator
- GUI allows users choose network carrier, strength, phone brand
Performance Evaluation

- **Application 1: Display Only**
  - Evaluates display power model
  - Two tests (100 random colors and 30 different images)

![Graph 1: Power vs Color Number](image1.png)
**Figure 8:** Testing the AMOLED display power model with 100 random colors.

![Graph 2: Power vs Image Number](image2.png)
**Figure 9:** Measured and emulated energy for Application 1, with 30 different images. Images are sorted by the measured energy used.
Application 2: Local Computation

- Test designed to model applications that use the processor and display
  - No heavy network use or heavy graphics

Figure 11: Comparing measured and emulated energy for Application 2.

Figure 12: Experimental evaluation of emulation accuracy for Application 2.
Application 3: Networked Apps

- Consider applications which use the network in addition to CPU and display
  - Test is to download files of varying sizes
  - Average error: 4.73%

![Graph showing power consumption over time](image1)

**Figure 13:** Example data capture for Application 3.

![Bar chart showing energy consumption](image2)

**Figure 14:** Estimated and measured energy for Application 3 with varying download sizes.
Application 4: Internet Browsing

- Download a webpage and render it on display
- Variations across multiple runs
  - Due to network and web server availability
- Average error: 4.64%

Figure 15: Accuracy evaluation for Application 4.
Case Study

- Consider an application which uses multiple components
  - i.e. a simple weather app
- Multiple design decisions for developer of app
  - Portability
  - Rich Graphics
  - Animation
- Quantitative energy cost would help designer make decisions
Case Study

Figure 17: Energy breakdown into multiple components.
Conclusion

- Presented a system to estimate energy consumption of apps during development
  - Fairly close to real world measurements
  - Leverages known power modeling and resource scaling concepts
Future Work

- Currently only prototyped for Windows Phone Platform
  - Which has a very small market share currently
  - Need to expand to other mobile platforms
- Improve models with real world data
References


- **Power Tutor**: powertutor.org

QUESTIONS?