Context-aware, Perception-guided Workload Characterization and Resource Scheduling on Mobile Phones for Interactive Applications

CHUNG-TA KING, NATIONAL TSING HUA UNIV., TAIWAN
CHUN-HAN LIN, NATIONAL TAIWAN NORMAL UNIV., TAIWAN
Acknowledgement

• Research contributions
  ◦ Dr. Pi-Cheng Hsiu, Academia Sinica
  ◦ Prof. Yung-Ju (Stanley) Chang, National Chiao Tung Univ.
  ◦ Dr. Bhaskar Das, National Tsing Hua Univ.
  ◦ NTHU team: Chi-Kai Ho, Kuan-Ting Ho, Johnny Liu, Eva Chen, Shuan-Yi Lin
  ◦ Academic Sinica team: Chih-Kai Kang

• Financial supports
  ◦ MediaTek
  ◦ Ministry of Science and Technology
Introduction

• Mobile phones are indispensable in our daily life
Introduction

• We use our phone primarily by interacting with it through rich multimedia
  ◦ We instruct the phone by touch, voice, button, ...
  ◦ The phone responds through visual, audio, vibration, ...

• Workload to the phone depends heavily on how we interact
  ◦ Dynamic, bursty, user/context dependent
Introduction

• Workloads need to be executed by processors
  ◦ Classical CPU performance equation:
    \[
    \text{CPU time} = \frac{\text{Instruction count} \times \text{CPI}}{\text{Frequency}}
    \]
  ◦ For mobile phones, performance is constrained by power consumption

Where is the sweet spot?
  ◦ Constrain one, optimize the other
  ◦ Optimize weighted sum
  ◦ ...

http://wccftech.com/iphone-7-coming-with-32gb-base-storage-model/
Introduction

• Since mobile phones are designed primarily to interact with their users, we thus should throw in human factor and consider user satisfaction
• Optimization strategy: *Just enough performance before user becomes unsatisfied*
  ◦ Constrain performance based on user satisfaction while optimizing power
• For interactive applications, user experience is affected mainly by perceived display quality *(Quality of Experience, QoE)*

Picture Source: T Zinner et al. (2010)
Introduction

• Three intertwined factors:
  ◦ Workload
  ◦ Quality of Experience (QoE)
  ◦ Resource scheduling

• Scheduling problem of mobile phones:
  ◦ Schedule the resources just enough to execute the given workload so that the required QoE is satisfied
  ➔ the power consumption can be minimized
Introduction

- However, workload and QoE are dynamic and often affected by the user contexts and activities

- Workload:
  - Types of apps: games (shooting, round-based, ...), movie/music player, Facebook, Twitter, browser, shopping app, ...
  - Contexts/activities: time of day, at home, on train, in meeting, walking, jogging, at low battery, with WiFi, ...

- Types of apps used, user interaction patterns

- QoE requirements:
  - Types of apps: games, Facebook, ...
  - Contexts/activities: walking, sitting, chatting, length of use, eating, ...
Issues to Discuss

• How to characterize and predict workload?
  ◦ Response time of user, app functions/phases, ...

• How to measure QoE, particularly expected display quality?

• How much influence can contexts have on workload and QoE?

• How to know the relationship between resources allocated and workload executed, and the effects on display quality?

• How to schedule resources to execute the predicted workload to meet QoE and minimize power?
Outline

• Context-aware design for OLED displays on mobile phones

• Android display system and CPU frequency governor

• QoE-aware resource scheduling strategies for energy optimization

• How to collect data from real users to characterize workload and QoE requirements?
OLED Displays

• Different types of displays are available for mobile phones
• OLED is deemed a promising technology to replace LCD for mobile displays
  ◦ Brighter colors, wider viewing angles, faster response times, etc.
• Power consumption increases dramatically with the pixel values of the displayed image
Structure of OLED
Pixel Power Models of OLED Displays

Picture sources:
Partial Display Disabling/Dimming Techniques

- A user usually focuses on just half of the screen for most (but not all) interactive applications ← context-depend.
- Darken contents that are not of interest
- Impact user perception

Color Remapping Techniques

• Change colors into colors that consume less power
  ◦ Acceptable colors are also user and context dependent
• Suitable for GUI but not natural images

OLED Dynamic Voltage Scaling Techniques

- Decrease the supply voltage of each pixel’s circuit
- Restore luminance with image compensation
  - Acceptable distortion is user dependent
- Require hardware support and partition the display into rectangular regions

![OLED device structure and equivalent circuit model](image)

Image Compensation Results

• Resultant distortion is context dependent

<table>
<thead>
<tr>
<th>Image</th>
<th>Original image</th>
<th>10</th>
<th>Original image</th>
<th>300</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viewed image</td>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Color histogram</th>
<th>Count</th>
<th>Intensity</th>
<th>Count</th>
<th>Intensity</th>
<th>Count</th>
<th>Intensity</th>
<th>Count</th>
<th>Intensity</th>
</tr>
</thead>
<tbody>
<tr>
<td>V_{DD} (V)</td>
<td>15.0</td>
<td>8.7</td>
<td>15.0</td>
<td>12.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power (mW)</td>
<td>399.9</td>
<td>189.8</td>
<td>731.7</td>
<td>572.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Saving (%)</td>
<td>-</td>
<td>52.5</td>
<td>-</td>
<td>21.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Image Pixel Scaling Techniques

- Scale down pixel values of different shaped regions
- Regions of an image receive varying degrees of visual attention
- Different regions can tolerate different degrees of image distortion
  - Acceptable distortion is user dependent
- Not every change is noticeable

Segmentation Regions and Processed Images

(a) Weather  (b) Twitter  (c) Facebook  (d) CNN

Outline

• Context-aware design for OLED displays on mobile phones

• Android display system and CPU frequency governor

• QoE-aware resource scheduling strategies for energy optimization

• How to collect data from real users to characterize workload and QoE requirements?
Overview of Android Display

- Activity
- BufferQueue
- SurfaceFlinger
- Window Manager
- Activity Manager
- OpenGL ES
- HWComposer
- Gralloc FB
- Gralloc alloc
- OpenGL vendor libraries
- GPU driver
- FB driver
- ION
- Android framework
- Vendor HAL library
- Kernel driver

Picture Source: http://elinux.org/images/2/2b/Android_graphics_path--chis_simmonds.pdf
Composition of the Display

[Diagram showing the composition of the display, including wallpaper, SurfaceFlinger, navigation bar, status bar, and launcher.]

Picture Source: [http://elinux.org/images/2/2b/Android_graphics_path--chis_simmonds.pdf](http://elinux.org/images/2/2b/Android_graphics_path--chis_simmonds.pdf)
Android Graphics Components

- Everything is rendered onto a "surface"
  - Every window is backed by a surface
- All of the visible surfaces rendered are composited onto the display by SurfaceFlinger

Picture Source: https://source.android.com/devices/graphics/
Android Graphics Display

- **Frame rate** (frames per second or FPS): the rate at which an imaging device generates consecutive images.
- **Refresh rate**: the rate at which the display hardware refreshes the frames.

Frame rate = 40 fps  
Refresh rate = 60 Hz

VSync and Display Buffering

• Synchronizes events to the refresh cycle of the display
  ◦ Applications always start drawing on a VSync boundary, and SurfaceFlinger always composites on a VSync boundary
• If the display is refreshing at 60 Hz, Android has 16 ms to process user input and draw on the display
• Android uses double buffering for drawing by default
  ◦ Front buffer: currently visible screen contents
  ◦ Back buffer: screen contents of the next frame
  ◦ On the next VSync event, these two buffers are swapped
  ◦ Frame dropped events may occur due to complexity of the view → mismatch between workload and resources allocated
VSync and Display Buffering

- When system missed VSync when producing a buffer, can only start to process next buffer at following VSync
  - Frames dropped and user perceived QoE is affected
- Triple buffering
  - On frame drop, switch to use a third buffer

CPU Governor

• To maintain a certain display quality, e.g. 60 frames per second (FPS), the frequency of the processors must be set high enough to process the workload within the deadline (e.g. 16 ms)

• Android provides several built-in CPU frequency governors (but also allows manual setting)
Common CPU Governors

- **Performance:**
  - Sets CPU frequency statically to the highest possible frequency

- **PowerSave:**
  - Sets CPU frequency statically to the lowest possible frequency

- **OnDemand:**
  - Quickly scales up CPU frequency on sufficient load, e.g. user input, and quickly scales down frequency once the load is no longer present

- **Interactive:**
  - Quickly ramps up to the maximum allowed frequency, and then slowly drops the frequency once no longer under load

http://wiki.rootzwiki.com/CPU_Governors
Outline

• Context-aware design for OLED displays on mobile phones

• Android display system and CPU frequency governor

• QoE-aware resource scheduling strategies for energy optimization

• How to collect data from real users to characterize workload and QoE requirements?
Quality of Experience, QoE

- For interactive apps on mobile phones, user experience is affected mainly by perceived display quality.
- A common metric of QoE is thus the frame rate, *frames per second* (FPS), of the display.
  - Duration between two frames defines deadline to execute the given workload, e.g. 60 FPS limits the time to draw a frame, including all the associated computations, to 16 ms.
- What is the right value for FPS that the user may feel satisfied regarding the display quality?
Quality of Experience, QoE

• Ideally, we need to experiment on a large number of real users to find out the right FPS to use for different types of applications under different contexts
• However, reports showed that normal users cannot perceive any significant difference between 45 and 60 FPS

• Is 45 or 60 FPS a good value to use for QoE?

Quality of Experience, QoE

- Application: CandyCrush
- Measurement tool: GameBench

- The game application draws continuously on the display
- Foreground app normally dominates computations
Quality of Experience, QoE

• However, other types of applications may not be applied, and experiments on real users may be needed.
• Not surprisingly, most surveyed works focus on games and set a fixed target QoE at 60 FPS.
Workload

• What metric to use for workload?
• For CPU/GPU scheduling, a commonly used metric is *CPU/GPU utilization*

• Two issues related to workload for CPU/GPU scheduling:
  ◦ What is the workload for the next scheduling interval, e.g. every 16 ms?
  ◦ How to set the CPU/GPU frequency to execute the predicted workload so that the deadline, e.g. 16 ms, can be met?
  → relationship between utilization and processor frequency
• More complex for heterogeneous multicore
How to Predict Workload?

- Use workload (CPU/GPU utilization) of the last scheduling interval
  - Or weighted average of last \( n \) intervals
  - Work fine with short scheduling intervals and homogeneous workload, e.g. games

- Use characteristics of applications (app-aware)
  - Study activities in using apps, e.g. scrolling, tapping, typing, watching, to identify different “states/phases” of the apps
    → build Markov model to predict next state
  - For example: round-based games, FB
CPU Utilization and Frequency

• In the last scheduling interval, CPU utilization was measured to be x% with a clock rate of y GHz, assuming workload remains the same, what should the clock rate be for the next interval? → CPU/GPU governor

• What is the target utilization for the next interval?
  ◦ Low vs. high
  ◦ Is 80% a good value?

• Given the target utilization, z%, what is the clock rate?
  ◦ Linear model, table lookup, ...
  ◦ How to take account of user input events?
CPU Utilization and Frequency

- **CPU-GPU co-scheduling**
  - Adjust CPU and GPU frequency independently
  - Consider relative computing power of CPU and GPU
  - Consider performance bound at CPU or GPU

- **Heterogeneous multicore and core scheduling**
  - Relative computing power of the cores

CPU Utilization and Frequency

- Power consumption is directly proportional to the CPU Utilization
  - Ex: MOBA (multiplayer online battle arena) game on Nexus 5x
  - Power consumption increases with the number of active cores
  - Gameplay is very smooth with two active cores (~80% utilization)
  - Playing the game with more than two cores is not profitable
    - Consumes more power without much improvements in the gameplay
  - Users may not enjoy the game with only one active core
Context Awareness

- Contexts may affect workload and acceptable QoE
- How many contexts and how to identify them?
- User annotated:
  - Types of annotated contexts are limited
  - Same context, behave differently, e.g. context too broad
- Machine learned:
  - Unsupervised machine learning to classify context data, such as time, place, activity, ...

Context Awareness

- FB usage of one user for 8 weeks
- Machine learned contexts:

<table>
<thead>
<tr>
<th>Context 1</th>
<th>Context 2</th>
<th>Context 3</th>
<th>Context 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weekday Moring</td>
<td>N/A</td>
<td>Before Sleep After wake up</td>
<td>N/A</td>
</tr>
<tr>
<td>Location</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Home</td>
<td>Outside</td>
<td>Home</td>
<td>Outside</td>
</tr>
<tr>
<td>Network</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wi-Fi</td>
<td>GSM</td>
<td>Wi-Fi</td>
<td>GSM</td>
</tr>
<tr>
<td>Status</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Charging</td>
<td>Hand-on</td>
<td>N/A</td>
<td>Walking</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Context 5</th>
<th>Context 6</th>
<th>Context 7</th>
<th>Context 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weekday Night</td>
<td>Weekend</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Location</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Home</td>
<td>Home</td>
<td>Not Home</td>
<td>Outside</td>
</tr>
<tr>
<td>Network</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wi-Fi</td>
<td>Wi-Fi</td>
<td>Wi-Fi</td>
<td>GSM</td>
</tr>
<tr>
<td>Status</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Charging</td>
<td>Charging</td>
<td>N/A</td>
<td>Still</td>
</tr>
</tbody>
</table>
Context Awareness

- User behaviors (workloads) under different contexts
  - Convert number of times of user’s operations into frequency to eliminate the temporal influence
  - Five scales: Least, Less, Medium, More, Much

<table>
<thead>
<tr>
<th></th>
<th>Context 1</th>
<th>Context 2</th>
<th>Context 3</th>
<th>Context 4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Duration</strong></td>
<td>Much</td>
<td>Least</td>
<td>Much</td>
<td>Least</td>
</tr>
<tr>
<td><strong>Scroll</strong></td>
<td>Much</td>
<td>Much</td>
<td>Less</td>
<td>Much</td>
</tr>
<tr>
<td><strong>Click</strong></td>
<td>Less</td>
<td>Much</td>
<td>Less</td>
<td>More</td>
</tr>
<tr>
<td><strong>FocusChanged</strong></td>
<td>Least</td>
<td>Much</td>
<td>Medium</td>
<td>Much</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Context 5</th>
<th>Context 6</th>
<th>Context 7</th>
<th>Context 8</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Duration</strong></td>
<td>More</td>
<td>Medium</td>
<td>Less</td>
<td>Medium</td>
</tr>
<tr>
<td><strong>Scroll</strong></td>
<td>More</td>
<td>Less</td>
<td>Medium</td>
<td>Least</td>
</tr>
<tr>
<td><strong>Click</strong></td>
<td>Less</td>
<td>More</td>
<td>Less</td>
<td>Least</td>
</tr>
<tr>
<td><strong>FocusChanged</strong></td>
<td>Less</td>
<td>Least</td>
<td>Medium</td>
<td>More</td>
</tr>
</tbody>
</table>
Context Awareness

• Context and acceptable QoE
  ◦ Shakiness is a first-order measure of user contexts, e.g. high for jogging; medium for walking/riding; low for sitting/standing
  ◦ Users tend to tolerate lower display qualify if the phone is shaky
• Preliminary data from a single user

<table>
<thead>
<tr>
<th>Shakiness</th>
<th>User Experience</th>
<th>No. of Cores</th>
<th>Max. Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>Good</td>
<td>2</td>
<td>600MHz</td>
</tr>
<tr>
<td>Low</td>
<td>Fair</td>
<td>2</td>
<td>460MHz</td>
</tr>
<tr>
<td>Low</td>
<td>Poor</td>
<td>2</td>
<td>384MHz</td>
</tr>
<tr>
<td>Medium</td>
<td>Good</td>
<td>2</td>
<td>384MHz</td>
</tr>
</tbody>
</table>
Outline

• Context-aware design for OLED displays on mobile phones

• Android display system and CPU frequency governor

• QoE-aware resource scheduling strategies for energy optimization

• How to collect data from real users to characterize workload and QoE requirements?
Real User Experiments

• QoE is very subjective, personalized, hard to quantify
  ◦ Particularly, it varies with user context, e.g. walking, sitting,…
• Most existing works on resource scheduling for mobile phones bypass the problem of quantifying QoE
  ◦ e.g. use a fixed 60 FPS as the target QoE
• Ignoring QoE variations under different contexts may set an unnecessary bound on the scheduling target, leading to a waste of power
• Need to better quantify users’ QoE under different contexts for tighter resource scheduling
Real User Experiments

- Recruit users to collect data on their usages of mobile phones and their satisfaction of the display quality
  - Be transparent to the testers regarding rights and obligations on recruiting
- Develop an app for data collection
  - Creating display conditions: randomly fix CPU/GPU frequency at a certain level when the tester uses specific interactive apps
  - Phone logging: sample and collect usage data, e.g. CPU/GPU utilization, mobility (activity + location), sensor values, FPS, etc.
  - Questionnaire: (discussed next)
  - Data uploading: upload collected data and ensure quality
  - Heartbeat service: check liveness of data collection app
Questionnaire Design

• Challenge:
  ◦ Ensure the quality of data gathered from the testers
  ◦ Do not annoy and disturb the testers

• Methodology: Experience Sampling Method (ESM)
  ◦ Ask only when the tester uses the app long enough
  ◦ Ask only when the tester leaves the app, with a random prob.
  ◦ Ask at most a certain times per hour and per day

• Also record data related to questionnaire
  ◦ Time that a target app started, the questionnaire popped up, the questionnaire answered, ...
Data Collection App
Summary

- Mobile phones are interactive
  - Workloads are dynamic, bursty, and user/context dependent
  - User experience affected by display quality
  - Processor scheduling for low power must consider workloads and user desired QoE, which in turn are affected by contexts
- Many works on processor scheduling for phones
  - Mainly focus on game-type applications
  - Address user experience but with static metric (e.g. 60 FPS)
- The more we know about device, application workloads, user preferences, contexts, (aware of this, aware of that) the better we can optimize the phone
References

• Application-aware and User-interaction-aware Energy Optimization Framework
• Using HSFMs to Model Mobile Gaming Behavior for Energy Efficient DVFS Governor
• Frame-based and Thread-based Power Management for Mobile Games on HMP Platforms
• Time Series Characterization of Gaming Workload for Runtime Power Management
• A User-Centric CPU-GPU Governing Framework for 3D Games on Mobile Devices
• Personalized Power Saving Profiles Generation Analyzing Smart Device Usage Patterns
• User-Centric Scheduling and Governing on Mobile Devices with big.LITTLE Processors
Thank You

Any questions?