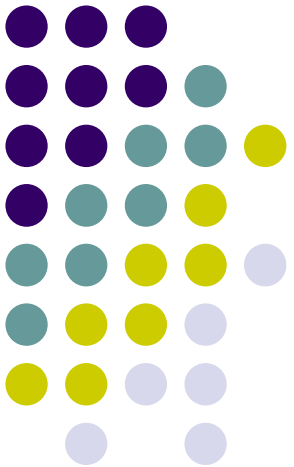
Ubiquitous and Mobile Computing CS 528:EnergyEfficiency Comparison of Mobile Platforms and Applications: A Quantitative Approach

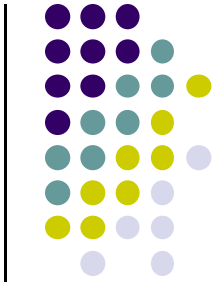
Norberto Luna-Cano

Computer Science Dept.

Worcester Polytechnic Institute (WPI)

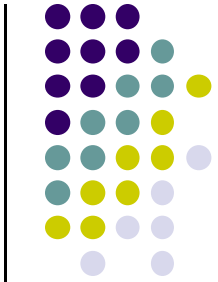


Introduction



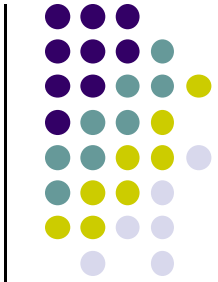
- A comparison of the relative energy-efficiency of apps in the same category on multiple platforms.
- **The Platforms**:
  - Android
  - Windows
  - Apple
- **The Categories**:
  - Browsers
  - Video/Music streaming
  - Social Networking

Challenges Measuring efficiency is a difficult task. Why?



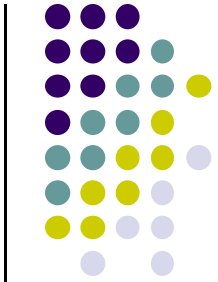
- Each platform requires different tools
  - *Different accuracy, introduced overhead.*
- Test environment has to be kept constant
  - *Light, WIFI signal strength*.
- Background activities affect differently
- Dynamic content in apps affect differently
  - *e.g. Facebook friend updates*
- Apps may have logic in the cloud

Experimental Approach –Tools



- Windows
  - Intel Soc Watch 2-5% overhead
    - CPU idle states
    - CPU frequency
    - Wakeups
    - Timer resolution
    - Threads per app
  - EnergyMeter (developed)
    - Battery consumption
    - Package, core and GPU energy consumption

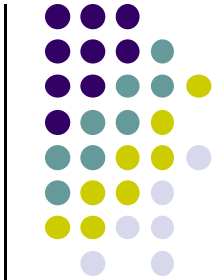
Experimental Approach –Tools



- In iOS
  - *Energy Profiler Instrument*
    - Energy consumption in scale from 0 to 20
    - CPU utilization
    - GPU utilization
    - Packets sent and received
- In Android
  - Trepn profiler (~40% overhead)
    - CPU and virtual memory utilization per app, avg. power consumption, wakelocks, wifilocks, and threads per app.
  - SoftPowerMon (1-2% overhead)
    - CPU idle states and frequency.

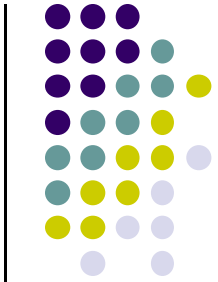
Experimental Approach –Tools

Case Studies –Browsers



- Case 1 (Browsers)
  - **Surface 2 Pro –*Firefox 3-tabs vs. Chrome***
    - *More efficient: Chrome*
      - *Better use of concurrency enabled longer idle sleep states*
- Case 2 (Browsers)
  - **iPad Air –*Bing vs. Chrome***
    - *Bing*
      - *had highest CPU utilization, 2 second interval for 60 byte packets sent/receive*
    - *Chrome*
      - *Sent large packets initially, then 80 byte packets spaced out at about 30 second intervals*

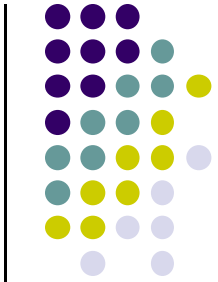
Case Studies –Browsers



- Case 3 (Browsers)
  - **Nexus 7 –*Firefox 3-tabs vs. Firefox 1-tab***
    - *More efficient: Firefox 3-tabs*
      - *Increased CPU frequency led to less core active duration.*

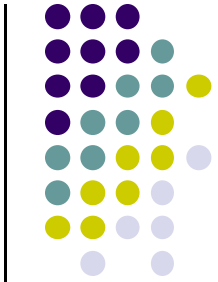
- Case 4 (Browsers)
  - **Nexus 7 –*Chrome vs. Bing.***
    - *More efficient: Chrome*
      - *Bing consumed 3 times as much power. Possible reason: Chrome has higher multithreading index.*

Case Studies –video and music streaming



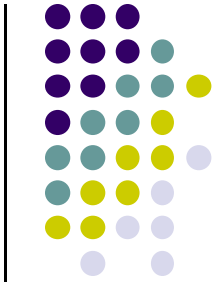
- Case 5 (video streaming)
  - **iPad Air *–YouTube app vs. browser.***
    - *More efficient: App.*
      - *It received packets more constantly –counterintuitive*
    - *Browser*
      - *used less CPU and graphics, but larger buffer used more memory.*
- Case 6 (music streaming)
  - **iPad Air *–Pandora vs. iTunes***
    - *More energy efficient: iTunes*
      - *but utilized more CPU*
    - *Pandora: utilized less CPU. More use of high power WIFI radio state.*

Case Studies –music streaming



- Case 7 (music streaming)
  - **Nexus 7 –*XBOX vs. Spotify***
    - *Most energy efficient: XBOX but had more jitter*
      - *Poor buffering.*
      - *Sacrificed user experience to save energy or due to poor app design.*

Case Studies –Social Networking



- Case 8 (social networking)
  - **Surface 2 Pro –*Facebook metro vs. browser***
    - *More energy efficient: Metro*
    - *Browser.*
      - Higher timer resolution kept WIFI radio more active. Larger number of wakeups.
- Case 9 (social networking)
  - **Nexus 7 –*Facebook App vs. browser***
    - *More energy efficient: browser.*
      - *App consumed 25.5% more energy possibly due to app sophisticated interface leading to 17% more GPU utilization.*

Table 3

Energy in Joules on Surface Pro 2

Scenario	Application	Platform	Package	Core	GPU
Browsers	Chrome	734	148	22	0.69
	Chrome 3-tabs	763	165	34	1.2
	IE	824	177	36	4
	IE 3-tabs	853	247	111	0.89
	IE Metro	882	548	279	5.28
	Firefox	828	181	32	7.77
	Firefox 3-tabs	878	532	280	7.31
Video Streaming	Amazon Desk.	2023	635	135	39.83
	Amazon Browser	3063	1873	866	225.82
	Netflix Metro	1836	589	147	29.72
	Netflix Browser	3009	1493	496	259.14
	YouTube Browser	2476	1290	487	130.93
Music Streaming	Pandora Browser	1757	650	149	31.98
	Spotify Desk.	1598	404	65	0.3
	XBOX Metro	1465	332	57	3.5
Social Networking	Facebook Browser	853	165	23	1.37
	Facebook Metro	770	201	59	2.35
	LinkedIn Browser	799	160	32	1.17
	LinkedIn Metro	745	149	26	2.26

Figure 1

Timer resolution on Surface 2 Pro

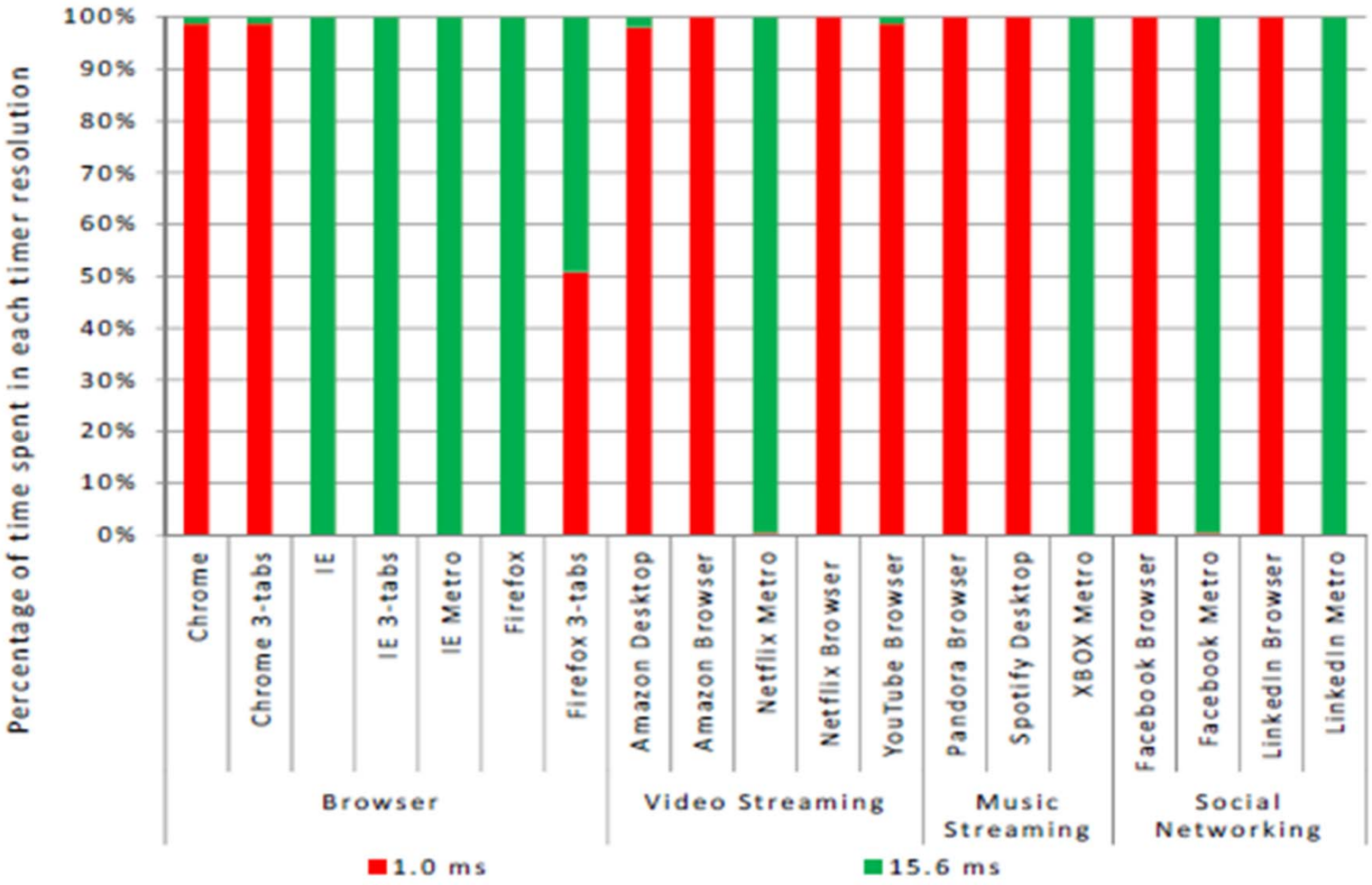


Table 4

Metrics collected on Surface Pro 2

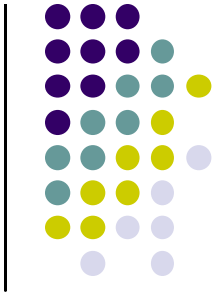
Scenario	Application	Active Duration in ms.	Average Package Wakeups	Average Core Wakeups
Browsers	Chome	6,371	1015	1507
	Chrome 3-tabs	10,736	1061	1595
	IE	15,786	208	754
	IE 3-tabs	5,386	145	363
	IE Metro	10,647	316	1319
	Firefox	7,996	118	457
	Firefox 3-tabs	6,260	600	1141
Video Streaming	Amazon Desktop	75,638	1,094	1,900
	Amazon Browser	379,754	575	2,648
	Netflix Metro	28,750	393	1,019
	Netflix Browser	346,010	611	2,019
	YouTube Browser	361,663	622	1,628
Music Streaming	Pandora Browser	111,995	1,079	1,956
	Spotify Desktop	24,633	1,287	1,892
	XBOX Metro	18,774	286	567
Social Networking	Facebook Browser	9098	1,011	1,451
	Facebook Metro	19,618	260	551
	LinkedIn Browser	10,911	1,067	1,584
	LinkedIn Metro	15,420	232	569

Figure 2

CPU idle states on Surface 2 Pro

Table 5

Metrics collected on Nexus 7



Scenario	App Name	Ave. Power in mW	Ave. CPU Percent	Ave. Virtual Memory	Thr-ead Count	Total wake-locks
Browsers	Chrome	237	0.68	2966.63	88	1171
	Chrome 3-tabs	374	1.99	1989	66	1401
	Bing	745	7.04	912	20	0
	Firefox	235	0.09	1943	53	0
	Firefox 3-tabs	221	0.31	1955	52	0
Video Stream-ing	YouTube App	1,280	1.91	1003.22	65	0
	YouTube Browser	1,468	1.17	2146	77	1515
	Netflix App	1,387	3.04	1023.78	65	0
Music Stream-ing	Pandora	705	0.59	973.38	42	1714
	Spotify	683	4.01	1860.03	107	1646
	XBOX	483	2.08	981.23	45	1550
Social Network-ing	Facebook Browser	965	1.35	2089.94	72	1023
	Facebook App	1,211	1.74	1817.81	49	4
	LinkedIn Browser	640	0.55	2131.36	70	1024
	LinkedIn App	426	0.04	946.38	31	0

Figure 4

CPU idle states on Nexus 7.

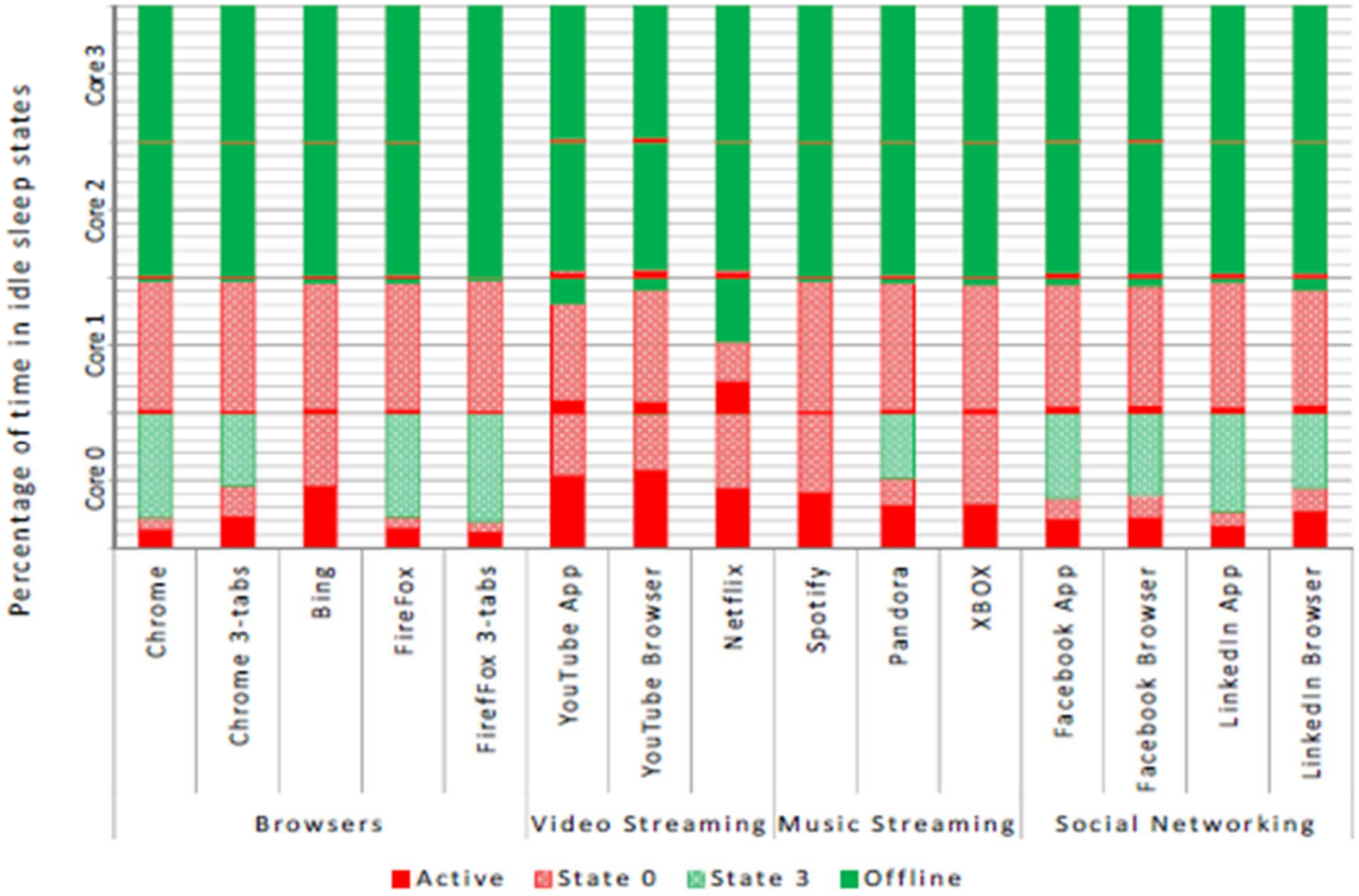


Figure 5

CPU frequency in MHz on Nexus 7.

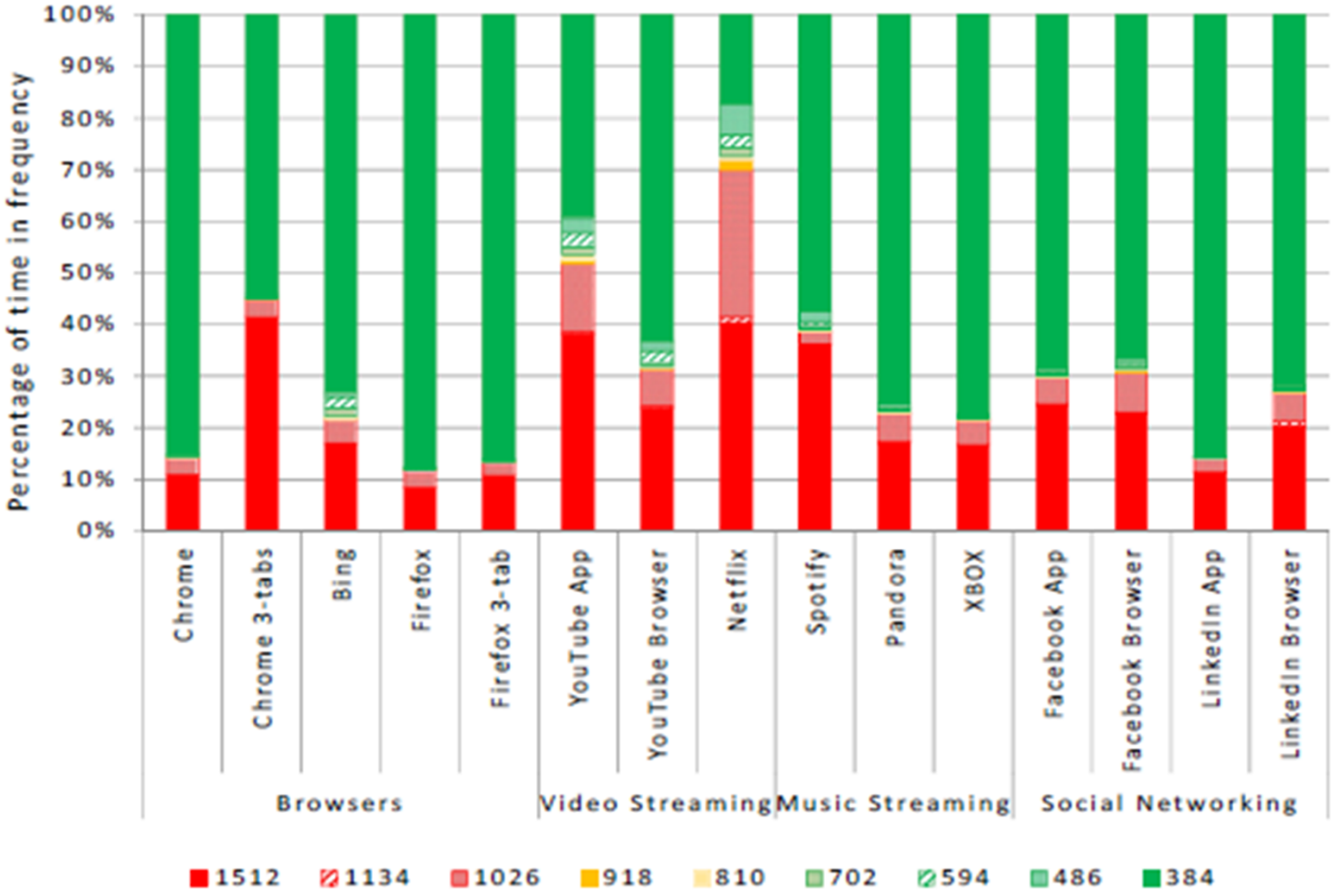
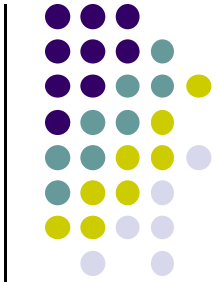


Figure 3

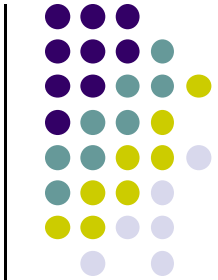
CPU frequency in MHz on Surface 2 Pro

Insights



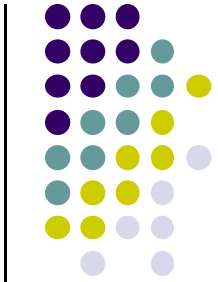
- Need for new tools and rules that allow:
  - *Increased accuracy of comparisons.*
  - *Define procedures to avoid measurement errors.*
- Approaches to energy efficiency differ:
  - *Apple could improve the precision of its data collection tool.*
  - *Google favors performance over energy efficiency. Case 1 and Table 5.*
  - *Windows was more concerned with energy efficiency than performance. Case 8.*
- App developer practices
  - *More use of power profiling tools e.g. WattsOn.*
  - *Need for power benchmarks for each category of apps.*
  - *Developer decisions: changing timer resolution on Windows, holding wakelocks on Android seem common practice.*

Insights



- Energy efficient app design:
  - *In general, native apps consume less energy than web apps.*
    - *Is Google at a disadvantage? Further research is recommended.*
  - *Buffer sizes should be balanced.*
    - *Increased buffer size allows more WIFI radio idle states.*
    - *To much buffer also increases memory usage and thus consumes more energy.*
  - *Multithreading.*
    - *Increases the energy efficiency of an app if execution is balanced across cores.*
  - *Time interrupts and network communication that are negative to efficiency:*
    - *Low resource (e.g. CPU) utilization but high average wakeup.*
    - *High average wakeup of platform's idle components (e.g. CPU or WIFI)*

Related work



- Taxonomy of sleep bugs on Android:
  - Jindal *et al.* [13] categorized root causes on android phones.
- Measurement of energy usage of top 100 apps in Google Play:
  - Chen *et al.* [14] tried to determine the energy savings from prefetching ads
- A collaborative approach –120000 Android users:
  - Wang *et al.* [15] built a power estimation model on the collected data.

References

[1] Google, “Data compression proxy,” https://developer.chrome.com/multidevice/data-compression.

[2] Intel, “Intel SoC Watch for Windows,”https://software.intel.com/sites/default/files/managed/aa/4a/socwatch_windows.pdf.

[3] Intel, “Intel 64 and IA-32 architectures software developers manual combined volumes: 1, 2A, 2B, 2C, 3A, 3B and 3C,” http://www.intel.com/content/dam/ www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-manual-325462.pdf.

[4] Qualcomm, “Trepn profiler,” https://developer.qualcomm.com/mobile-development/ increase-app-performance/trepn-profiler.

[5] G. Metri, A. Agrawal, R. Peri, M. Brockmeyer, and W. Shi, “A simplistic way for power profiling of mobile devices,” in Proc. IEEE ICEAC, 2012.

[6] Apple, “About instruments,” https://developer.apple.com/library/mac/ documentation/developertools/conceptual/ instrumentsuserguide/Introduction/Introduction.html.

[7] A. Charland and B. Leroux, “Mobile application development: web vs. native,” Communications of the ACM, vol. 54, no. 5, 2011.

[8] M. Sabharwal, A. Agrawal, and G. Metri, “Enabling green IT through energy-aware software,” IT Professional, 2013.

[9] A. Carroll and G. Heiser, “Mobile multicores: use them or waste them,” in Proc. HotPower, 2013.

[10] B. Steigerwald and A. Agrawal, “Developing green software,” Intel White Paper, 2011.

[11] A. Kansal and F. Zhao, “Fine-grained energy profiling for power-aware application design,” ACM SIGMETRICS Performance Evaluation Review, vol. 36, no. 2, 2008.

[12] R. Mittal, A. Kansal, and R. Chandra, “Empoweringdevelopers to estimate app energy consumption,” in Proc. ACM MobiCom, 2012.

[13] A. Jindal, A. Pathak, Y. C. Hu, and S. Midkiff, “On death, taxes, and sleep disorder bugs in smartphones,” in Proc. HotPower, 2013.

[14] X. Chen, A. Jindal, and Y. C. Hu, “How much energy can we save from prefetching ads?: energy drain analysis of top 100 apps,” in Proc. HotPower, 2013.

[15] C. Wang, F. Yan, Y. Guo, and X. Chen, “Power estimation for mobile applications with profile-drivenbattery traces,” in Proc. IEEE/ACM ISLPED, 2013.

Questions