Online Tuning of Dynamic Power Management for Efficient Execution of Interactive Workloads

James R. B. Bantock, Vasileios Tenentes, Bashir M. Al-Hashimi, Geoff V. Merrett
Department of Electronics and Computer Science, University of Southampton
{jrbb1g11, v.tenentes, bmah, gvm}@ecs.soton.ac.uk

***Abstract*—Modern mobile devices contain powerful Multi-Processor System-on-Chips (MPSoCs) that are performance throttled by Dynamic Power Management (DPM) runtime systems to extend battery lifetime. Applications on mobile devices commonly generate highly interactive workloads, dependent on interaction between the processor cores, peripherals, external resources and the user, such as touch input during web-browsing. Inevitably, a subset of interactive workloads are affected by delays caused by data unavailability, e.g. loss or delay of data packets during voice-over-IP. At the same time, the system is required to respond quickly upon data retrieval to ensure that the user Quality of Experience (QoE) metrics (frame-rate, latency, etc.) are not degraded. Traditionally, operating systems have mitigated this problem with periodic sampling or event-driven approaches. Through experimentation using a mobile MPSoC platform, however, we demonstrate that improving the tuning of DPM parameters for certain interactive user inputs can provide energy savings of up to 21% or QoE improvements of up to 36%, when compared with the traditional approach. To capture these improvements, we propose a dynamic modeling of user input and data resource access times (e.g. mobile network bandwidth and latency) for interactive workloads, which is based on workload profiling and which we refer to herein as inelasticity analysis. The proposed approach is implemented through online tuning of a DPM runtime in the Android operating system and is validated through a Monte Carlo simulation of interactive workloads. In comparison to the default DPM tuning, the proposed approach achieves energy savings of 13% or QoE improvement of 27% or a selectable trade-off, e.g. 9% energy savings and 15% QoE improvement.**

I. Introduction

Multi-Processor System-on-Chips (MPSoCs) used in modern mobile devices allow execution of increasingly demanding workloads. These workloads are commonly interactive, such that they are dependent on user input and external resources during execution; this is a trend which is expected to escalate with networked Internet-of-Things applications. Unlike throughput-based computation, interactive workloads are primarily benchmarked by their response latency to touch, speech or other sensor input [1]. Variation in access time of data resources may limit the achievable latency. At the same time, preservation of battery lifetime of a mobile device is a significant challenge and must be considered during system optimization [2]. Operating systems can artificially limit the performance of an MPSoC using power management levers in order to increase battery lifetime; however, increased battery life must be traded off against potential regression of latency performance.

978-1-5090-6023-8/17/$31.00

Ubiquitous power management levers available within MPSoCs that may be set by Dynamic Power Management (DPM) include Dynamic Voltage and Frequency Scaling (DVFS) and processor core idle states. Traditionally, there have been two approaches to DPM, namely individual control loops such as the *cpufreq* and *cpuidle* governors and approaches at the system level [3], [4]. System level approaches integrate awareness of application context, in addition to the metrics considered by the low-level control loop approach. DPM runtimes have a number of parameters, which are typically tuned to a static value during system configuration.

Mobile device workloads share the characteristic of having a single user interface task running in the foreground. The user Quality of Experience (QoE) - which is the quantitative measurement of perceived performance from the perspective of the user [5] - may be derived from the Quality of Service (QoS) metrics of this task. QoS metrics include: (a) the latency in response to a user input; and (b) the redrawing rate of the graphics buffer. The former is an absolute time measurement from the moment of input to completion of the associated event [1]. The latter is caused by the necessity to update the graphics buffer of the user interface at a certain frame-rate [6].

The user QoE for a subset of interactive workloads can suffer significantly from aggressive DPM; this is primarily due to the lag in control loops for individual power management levers. In response to this, a new *cpufreq* governor was implemented for the Android operating system; the *interactive* governor is scheduled to execute immediately after the CPU wakes up from idle to allow faster ramping up of frequency compared to previous governors [7]. A comparable event-based approach has been proposed at the system level, reliant on detection of the type of task to determine the length of the following period of elevated CPU frequency [3]. Setting a fixed high performance period requires accurate deadline prediction of interactive tasks; the study proposing this did not consider tasks affected by high variability in resource access time such as in a mobile network scenario; it is likely that this variability would erode the accuracy of deadline prediction.

An alternative approach that does not require deadline prediction is to maintain a constant DPM lever setting throughout an interactive workload. Offline profiling may be used to evaluate the correct DPM lever setting without significant degradation in QoE, exploiting the fact that the QoE of a

number of interactive workloads is inelastic with respect to the DPM lever setting over a certain range of values. Accurate determination of this operating point has been demonstrated on a mobile device for a number of fixed workloads [8], however, the study considers neither workloads that are mixed by nature and require a set of correct DPM operating points, nor the effect of variability on resource access time that may occur at runtime.

In the context of interactive workloads, evaluation of the variability in resource access time on loading of web-pages within a mobile web-browser has shown that mobile network conditions may affect not only energy consumption [9], but also the optimum DVFS lever setting [10]. In this paper, three major contributions are included:

1) We present experiments from executing interactive workloads on the Samsung Exynos-5422 mobile MP-SoC, which runs the Android operating system. These demonstrate that changing the tuning configuration of DPM parameters from the default tuning for certain interactive user inputs can provide energy savings of up to 21% or QoE improvements of up to 36%.
2) To capture these improvements, we propose a dynamic modeling of user input and data resource access times (e.g. mobile network bandwidth and latency) for interactive workloads, based on offline workload profiling. During the profiling, which we refer to as inelasticity analysis, QoE metrics are evaluated with respect to varying DPM runtime tuning for individual user inputs to interactive workloads.
3) Additionally, we show how normalized QoE metrics may be traded off with energy in a single weighted optimization function to achieve a full range of Pareto-optimal points in the performance vs. energy trade-off.

The dynamic aspect proposed is implemented through online tuning of DPM runtime parameters in the Android operating system and is validated though a Monte Carlo simulation. This simulation explores the achievable trade-offs between QoE and energy for a set of user inputs to interactive workloads under varying resource access times. In comparison to the default DPM tuning, the proposed approach achieves energy savings of 13% or QoE improvements of 27% or a selectable trade-off, e.g. 9% energy savings and 15% QoE improvement.

The remainder of the paper is organized as follows: in Section II, we present the experimental demonstration of opportunities to exploit energy savings and QoE improvements for interactive workloads under varying user input and resource access times. In Section III, we present the proposed offline inelasticity analysis of interactive workloads and online tuning of DPM runtime parameters for exploiting these opportunities. The trade-off between QoE and energy consumption achieved by the proposed approach is presented in Section IV and validation results are shown in Section V. Finally, conclusions are drawn in Section VI.

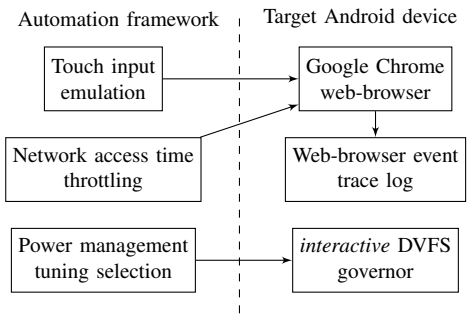


Fig. 1. Experimental setup for evaluating the effect of user input and resource access time variability on QoE and energy consumption for interactive workloads.

II. Motivation

In this section, we present experimental results demonstrating the effect on the execution of interactive workloads of different tuning configurations of DPM runtime parameters taking into account variability in user input and resource access times during such execution.

A. Platform

The hardware platform used for experimentation is the Samsung Exynos-5422 MPSoC with an ARM big.LITTLE cluster architecture consisting of A7 and A15 cores, running the Android 6.0 Marshmallow operating system.

B. Experimental setup for emulation of interactive workloads

The experimental setup, which is shown in Figure 1, consists of two components. The first is the target device executing the Google Chrome web-browser application in Debug mode in which a debugging protocol over WebSocket is provided. The second component of the setup is the automation framework, which uses the aforementioned protocol to configure in-built tracing capabilities and emulate interactive workload features.
Interactions: The web-browser debugger protocol enables repeatable user input to be emulated, such as touch gestures to the screen display. Additionally, resource access time variability for mobile network conditions is induced using the network emulation which artificially throttles content loading time by adding to the round trip time latency, and limiting upload and download bandwidths.
QoE and energy data collected: Events collected in the web-browser trace log are post processed to extract rendering frame-rate and content loading times for various web-pages requests in order to measure the QoE for each individual emulated user input. At the same time, through a device driver Android provides direct access to the on-board Exynos-5422 power sensors for energy consumption measurements.
DPM parameters: The DPM parameters selected for tuning were for the *interactive* DVFS governor in the underlying Linux kernel, such as the sampling timer and target CPU utilization level.

TABLE I
DEFAULT AND IMPROVED DPM TUNING CONFIGURATIONS TAKING INTO ACCOUNT VARYING USER INPUT AND RESOURCE ACCESS TIMES FOR AN INTERACTIVE WORKLOAD.

DPM tuning configuration (ID)	Quality of Experience (ms) (vs. default)	Energy consumption (J) (vs. default)
Web-page scroll: QoE = Time below 60 frames-per-second rendered		
Default (1)	774 (0.0%)	4.46 (0.0%)
Max QoE (46)	493 (-36.3%)	7.26 (62.8%)
Min energy (3)	805 (4.0%)	4.27 (-4.3%)
Web-page navigate (WiFi): QoE = Web-page load-time		
Default (1)	2035 (0.0%)	4.92 (0.0%)
Max QoE (21)	1992 (-2.1%)	4.94 (0.4%)
Min energy (5)	2724 (33.9%)	3.92 (-20.3%)
Web-page navigate (3G): QoE = Web-page load-time		
Default (1)	4718 (0.0%)	5.03 (0.0%)
Max QoE (46)	4665 (-1.1%)	7.02 (39.6%)
Min energy (31)	4791 (1.5%)	3.99 (-20.7%)

C. Motivational experiment

In order to quantify the range of QoE and energy consumption values achievable by using various DPM tunings, 50 unique cases are generated including the default DPM tuning. These tunings are evaluated for two different types of user input, namely, a scroll touch input and web-page navigation; the latter's dependency on mobile network resource access time is considered over two emulated mobile network conditions, WiFi and 3G. Table I presents a comparison between the default tuning (labeled as "Default") and the best tunings for the objectives of QoE (labeled as "Max QoE") and energy consumption (labeled as "Min energy"). We observe that the default tuning is not the best choice for any of the user inputs or resource access times, with none of the 49 alternative tunings being optimal for both QoE and energy for all user inputs. More importantly, we observe that the tuning for minimum energy for web-page navigate workloads under 3G network conditions is only 1.5% slower than the default tuning, even though it consumes 20.7% less energy.

These observations support our assumption that there are opportunities during the selection of DPM tuning configurations for improving the QoE and reducing the energy consumption for interactive workloads. In order to capture these opportunities in the context of interactive workloads, we propose an online approach to adapt the DPM tuning configuration dependent on the relevant workload.

III. PROPOSED ONLINE POWER MANAGEMENT TUNING

We propose a hybrid offline and online approach to improve the configuration of DPM parameters, see Figure 2. Expected user QoE is quantitatively derived from individual QoS metrics profiled offline for different DPM tuning configurations in order to weight an optimization function evaluated online.

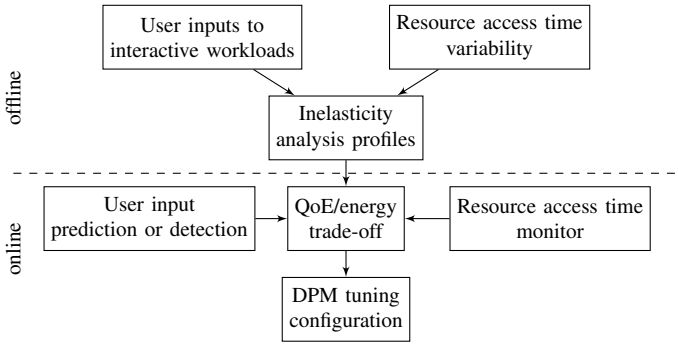


Fig. 2. Proposed interactive workload inelasticity analysis and online DPM tuning configuration selection

Variability in user input and resource access times is monitored online and considered as part of the optimization function.

A. Inelasticity Analysis

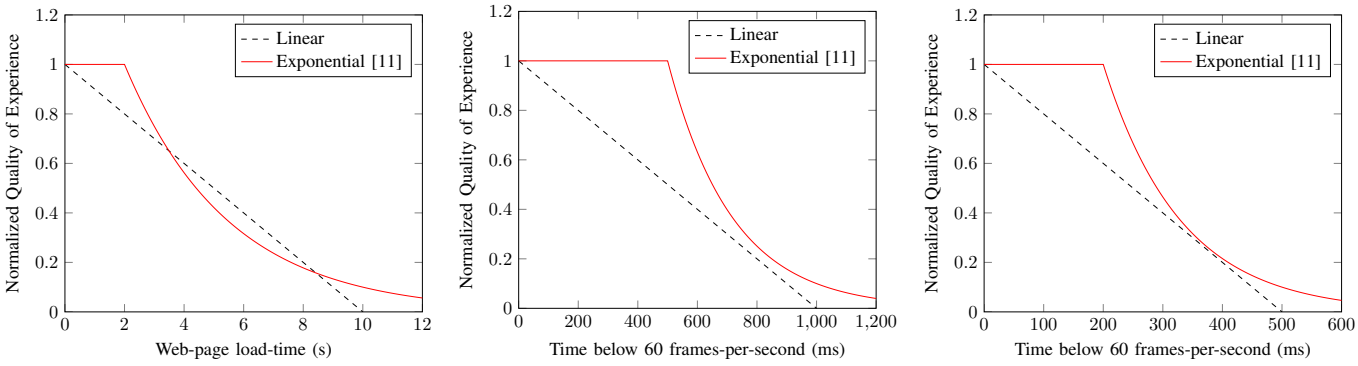
A subset of user inputs to interactive workloads may have dependencies on external resources, for example retrieving data from a mobile network source. The critical computation path for each of the affected user inputs may depend on the access time of that resource and therefore the requisite CPU performance will also vary, this is the inelasticity of the workload with respect to CPU performance. In many cases, a reasonable estimation of the resource access time may be calculated based on previous accesses. For a mobile network - assuming deterministic routing and server response time - metrics of round trip time, upload and download bandwidth may be estimated. During inelasticity analysis of user inputs under different DPM configurations, simulated resource access time variability (such as network throttling) is used to build a model that may be exploited at runtime based on online monitoring of resource access times.

B. User Input Prediction or Detection

A characteristic of an interactive mobile workload is that there is a user interface (UI) task operating in the foreground which accepts the majority of interactive user input. The proposed approach considers the future workload to be the next user input to this UI task at all times.

In an ideal environment, the next user input is always known. This would be achieved through raising an interrupt upon user input detailing its characteristics; this method has been proposed and demonstrated previously [3]. To accurately determine an individual input, an interrupt must be raised by the application to the runtime system when it occurs.

Alternatively, input prediction models may be used online. Probabilities of user inputs occurring may be gathered using a fusion of offline and online characterization. There exist statistical techniques to model ordering of user input which may use this characterization as a heuristic, including analysis of application choices [12] and user input timing [13], [14]. It is assumed that input prediction will always be inferior to input detection in terms of accuracy; if the prediction of the



(a) Web-page navigate (b) Web-page scroll (c) Web-page zoom

Fig. 3. Linear and exponential [11] models to derive normalized user Quality of Experience for (a) web-page navigate, (b) scroll and (c) zoom inputs.

model differs beyond a configured delta from the actual inputs, then the runtime may resort to a default DPM configuration.

C. QoE metrics for interactive workloads

In order to apply the scoring of the DPM tuning configurations discussed in the proposed approach, the QoS metrics between the different input actions must be normalized to derive a comparable QoE. To demonstrate that the proposed approach is applicable to different interpretations of quantitative derivation of QoE, two methods of normalization are used simultaneously, namely (a) a straightforward linear function and (b) an exponential decay as proposed by [11]. A graphical representation of these functions is shown in Figure 3 where QoS metrics of web-page load-time and time spent below 60 frames-per-second are translated to normalized QoE values. However, it is important to stress that the proposed approach can work with any given QoE models and those presented are simply for illustration. We demonstrate this independence with respect to the QoE model in Section V.

D. Online Optimization of QoE and Energy

Online, the DPM parameter selection will establish a current probability $P(A)$ for the next input being each of the possible choices $A_{0..k}$ and estimate the access times for each resource $R_{0..m}$. Given a pool of n DPM tuning configurations $T_{0..n}$, each configuration will be given a score for QoE and energy using equations 1 and 2 respectively.

$$\forall T : T_{QoE} = \sum_{A=0}^{k} P(A_k)(1 - A_{QoE}(R_{0..m}, T)) \quad (1)$$

$$\forall T : T_{energy} = \sum_{A=0}^{k} P(A_k) A_{energy}(R_{0..m}, T) \quad (2)$$

QoE and energy must be weighted at the system level to determine a trade-off denoted by w_{QoE} and w_{energy}. A composite score of QoE and energy is then calculated for each DPM tuning configuration using equation 3. Finally, the minimum score is determined to determine the tuning configuration that will be chosen for the next user input using equation 4.

$$\forall T : T_{score} = w_{QoE} T_{QoE} + w_{energy} T_{energy} \quad (3)$$

$$T_{choice} = min(T_{score}) \quad (4)$$

The DPM tuning configuration is then implemented using the relevant system calls, following which the system runtime sleeps until the input has finished before repeating the process.

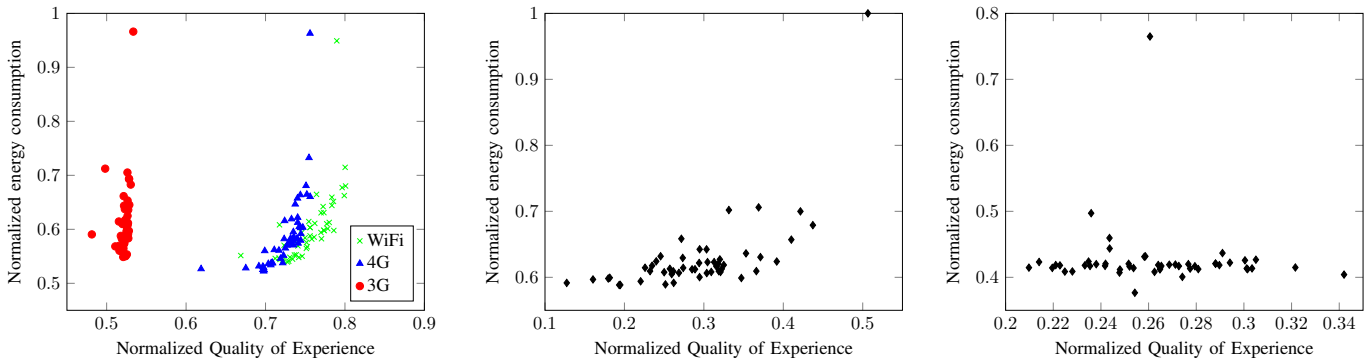
IV. Inelasticity Analysis Results

An inelasticity analysis was conducted using the experimental platform described in Section II. In addition to the web-page navigation and scroll inputs, a zoom touch input was evaluated along with an additional 4G network condition. Four DPM parameters were considered from the *interactive* DVFS governor. In total 50 different DPM tuning configurations for these parameters were included for inelasticity analysis, including the default tuning configuration and 49 randomly generated alternatives. The random generation was implemented by uniformly selecting across the available range of each parameter. Where possible, the minimum and maximum value for each parameter were chosen to be significantly lower or higher (as applicable) than the default DPM configuration. In table II, the four DPM parameters, their default value and ranges are presented in columns from left to right.

TABLE II
DPM tuning parameters used for inelasticity analysis.

Parameter reference	Default value	Min value	Max value
timer_rate	20000	1000	100000
target_loads	99	60	99
scaling_max_freq	2000000	800000	2000000
min_sample_time	80000	5000	500000

Each DPM tuning configuration was profiled offline for all of the user inputs and network conditions over 50 repeats



(a) Web-page navigate (b) Web-page scroll (c) Web-page zoom

Fig. 4. Quality of Experience and energy trade-off for (a) web-page navigate, (b) scroll and (c) zoom inputs over 50 power management tuning configurations.

and averaged to obtain mean values for QoS and energy. The QoE normalization functions were then applied, together with an energy normalization with respect to the highest energy recorded. The resultant trade-offs between QoE and energy are shown in Figure 4.

Variation in QoE and energy metrics are observable for all DPM tuning configurations for all user inputs. It should be noted, however, that one tuning configuration is a distant outlier, this is because a highly inefficient tuning for energy conservation was randomly generated. The user inputs which trigger interactive workloads lasting longer timer periods, such as web-page navigation and scrolling, show larger variation than the shorter zoom user input. This result is not surprising as there is a fixed overhead for waking up and then sleeping a processor core.

The effect of resource access time variability on the QoE of the navigate input is evident, when the network is relatively unthrottled in the WiFi scenario, CPU performance is clearly driving a variation in QoE. However, when the network is throttled in the 3G case, extra energy is consumed by some tuning configurations for no change in QoE. The 4G case is a combination of the two, there still exists a trade-off between QoE and energy but there is a steeper gradient than the WiFi case.

V. Validation of Proposed Approach

The inelasticity analysis results highlight ample opportunities to capture energy savings or QoE improvements for all of the user inputs profiled. To validate how the proposed approach would exploit these opportunities online, we use a Monte Carlo simulation to generate random interactive workloads and network resource access time conditions. The simulation evaluates the optimization function presented in Section III for six variants of the proposed approach comparing the resultant mean QoE and energy consumption to the default DPM tuning configuration.

The simulation was executed for 100,000 iterations to observe convergence and repeated for both linear and exponential methods of calculating the normalized QoE result. The weightings for QoE and energy, w_{QoE} and w_{energy}, were swept from $\{0, 1\}$ to $\{1, 0\}$ to expose the full trade-off curve between QoE and energy dependent on system goals.

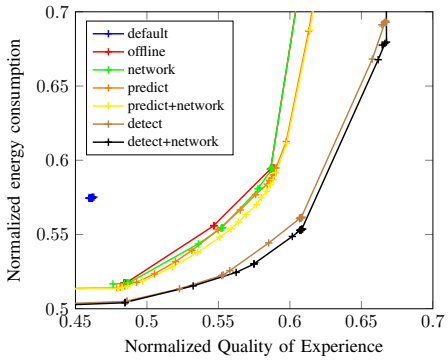
The mean QoE and energy consumption values for the variants of the proposed approach under the generated interactive workloads are shown in Table III. The first two columns are results for when $w_{QoE} = 1$, such that there is a full weighting towards QoE improvement. The second two columns are results for when $w_{energy} = 1$, such that there is a full weighting towards energy improvement. The third two columns are results for when $w_{QoE} = 0.5$ and $w_{energy} = 0.5$ or an equal weighting between QoE and energy.

The *default* approach is the default tuning for the DPM parameters. The *offline* approach considers usage of the inelasticity profiles with equal probability of each user input and network condition occurring. The *network* approach adds accurate detection of current network resource access time. The *predict* and *detect* approaches emulate accurate user input prediction and detection respectively. The *predict+network* and *detect+network* combine accurate detection of current network resource access time with user input prediction and detection respectively.

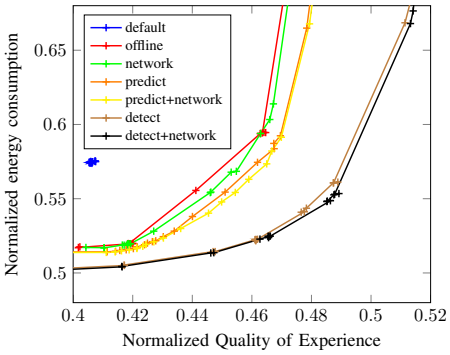
TABLE III
Proposed online tuning approach compared to the default tuning configuration, for three QoE and energy weightings.

	$w_{QoE}(1)$ $w_{energy}(0)$		$w_{QoE}(0)$ $w_{energy}(1)$		$w_{QoE}(0.5)$ $w_{energy}(0.5)$	
Approach	QoE	Energy	QoE	Energy	QoE	Energy
default	0.407	0.575	0.406	0.575	0.406	0.574
offline	0.488	0.909	0.402	0.518	0.418	0.519
network	0.488	0.909	0.404	0.517	0.418	0.519
predict	0.491	0.862	0.391	0.514	0.428	0.522
predict+network	0.491	0.850	0.391	0.514	0.430	0.524
detect	0.516	0.693	0.366	0.502	0.461	0.522
detect+network	0.516	0.725	0.365	0.501	0.466	0.524

In the majority of operating conditions, a suitable trade-off between energy and QoE is more desirable than fully weighting one or the other as shown in Table III. Instead



(a) Exponentially derived Quality of Experience



(b) Linearly derived Quality of Experience

Fig. 5. Monte Carlo simulation results for proposed online DPM tuning approach versus the default DPM tuning.

there is a full range of Pareto-optimal points which are shown in Figure 5 for both methods of normalization of the QoE metric presented in Section III. The claim of independence with respect to the QoE normalization model used is evidenced by the similar response for both linear and exponential QoE models.

All variants of the proposed approach demonstrate energy savings and QoE improvements, when compared to the default DPM configuration. The resource access time detection used by the *network* variant achieves marginal improvement over the *offline* variant that models current network conditions with equal probability. Prediction of user input is shown to be more effective by the *predict* and *predict+network* variants. However, detection of user input is the clear optimal choice as shown by the *detect* and *detect+network* variants which achieved QoE improvements of 27% or energy savings of 13%, in comparison to the default DPM tuning configuration.

VI. Conclusions

We have demonstrated through experimentation that, where inelasticity analysis is conducted in respect of variability in user input and resource access time for interactive workloads, this will result in increased opportunities to capture energy savings and QoE improvements, in comparison with current operating systems. A single weighted optimization function for QoE and energy is proposed as a proxy for a performance vs. energy trade-off for interactive workloads. The function has been validated through simulations to confirm its efficacy, independent of the model used to normalize QoE. A novel approach to online tuning of DPM parameters has been demonstrated as an alternative to setting individual DPM levers. A hybrid offline and online approach has been proposed that generates inelasticity profiles for user inputs to interactive workloads. These profiles are opportunistically exploited online to optimize for weighted system goals of QoE and energy. The approach is validated on a modern mobile MPSoC platform demonstrating energy savings of up to 13% or QoE improvements of up to 27% or a selectable trade-off, e.g. energy savings of 9% and QoE improvements of 15%, when compared to the default DPM tuning configuration. These results could feasibly be improved by consideration of additional DPM parameters available for tuning.

Acknowledgment

This work was supported in part by the ESPRC grant EP/K034448/1, (www.prime-project.org). All data openly available at http://doi.org/10.5258/SOTON/D0100.

References

[1] Y. Endo, Z. Wang, J. Chen, and M. Seltzer, "Using latency to evaluate interactive system performance," in *Proceedings of the Second USENIX Symposium on Operating Systems Design and Implementation*, 1996, pp. 185–199.

[2] G. Forman and J. Zahorjan, "The challenges of mobile computing," *Computer*, vol. 27, no. 4, pp. 38–47, 1994.

[3] J. Lorch and A. Smith, "Using user interface event information in dynamic voltage scaling algorithms," in *11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer Telecommunications Systems*, 2003, pp. 46–55.

[4] L. Yan, L. Zhong, and N. Jha, "User-perceived latency driven voltage scaling for interactive applications," in *Proceedings of the 42nd Annual Design Automation Conference*, 2005, pp. 624–627.

[5] S. Bischoff, A. Hansson, and B. Al-Hashimi, "Applying of quality of experience to system optimisation," in *23rd International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, 2013, pp. 91–98.

[6] N. Duca and T. Wiltzius, "Jank free: Chrome rendering performance," Google I/O 2013.

[7] M. Chan. (2015) cpufreq: interactive: New 'interactive' governor. [Online]. Available: https://lwn.net/Articles/662209//

[8] V. Seeker, P. Petoumenos, H. Leather, and B. Franke, "Measuring qoe of interactive workloads and characterising frequency governors on mobile devices," in *Proceedings of the IEEE International Symposium on Workload Characterization (IISWC)*, 2014, pp. 61–70.

[9] D. Bui, Y. Liu, H. Kim, I. Shin, and F. Zhao, "Rethinking energy-performance trade-off in mobile web page loading," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, 2015, pp. 14–26.

[10] Y. Zhu and V. Reddi, "High-performance and energy-efficient mobile web browsing on big/little systems," in *IEEE 19th International Symposium on High Performance Computer Architecture (HPCA)*, 2013, pp. 13–24.

[11] M. Fiedler, T. Hossfeld, and P. Tran-Gia, "A generic quantitative relationship between quality of experience and quality of service," *IEEE Network*, vol. 24, no. 2, pp. 36–41, 2010.

[12] C. Shin, J. Hong, and A. Dey, "Understanding and prediction of mobile application usage for smart phones," in *Proceedings of the ACM Conference on Ubiquitous Computing*, 2012, pp. 173–182.

[13] S. Card, T. Moran, and A. Newell, "The keystroke-level model for user performance time with interactive systems," *Commun. ACM*, vol. 23, no. 7, pp. 396–410, 1980.

[14] P. Holleis, F. Otto, H. Hussmann, and A. Schmidt, "Keystroke-level model for advanced mobile phone interaction," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2007, pp. 1505–1514.