Um Ambiente para Avaliação do Consumo de Energia em Múltiplos Dispositivos Baseado na Web

Sidartha Carvalho[1], Rafael Lima[1], Carolina Barbosa[1], Abel Silva-Filho[1]

[1]Centro de Informática (CIn) – Universidade Federal de Pernambuco (UFPE) – Recife – PE – Brazil

{salc,rnl,cmab,agsf}@cin.ufpe.br

Abstract. *Recently can be observed a growth in mobile devices sales, including smartphones and tablets, with its high quantity and diversity of hardware and software models, it has highlighted the problem of fragmentation. The increase of features highlighted the need for greater capacity in mobile batteries to provide an extended time of use. This work proposes an environment for simultaneously tests on mobile devices with support for energy analysis. The tested devices in the environment were: Samsung Galaxy S3 and Galaxy S4. Experiments were conducted to provide energy efficiency optimizations in frequency scaling. From the analysis made in the environment were created techniques using DVFS that allowed energy savings of up to 25%.*

Resumo. *Com o crescimento das vendas dos dispositivos móveis, entre eles os smartphones e tablets, com sua alta quantidade e diversidade de modelos de hardware e software destacaram o problema da fragmentação. O incremento de funcionalidades destacou a necessidade de maior capacidade nas baterias para prover um tempo de uso mais prolongado dos aparelhos. Neste trabalho foi proposto um ambiente para testes simultâneos em dispositivos móveis com suporte a análises energéticas. Os dispositivos testados no ambiente foram: Samsung Galaxy S3 e Galaxy S4. Foram realizados experimentos para prover eficiência energética com otimizações no escalonamento de frequência. A partir das análises feitas no ambiente foram criadas técnicas utilizando DVFS que permitiram uma economia de energia de até 25%.*

1. Introdução

Dispositivos como *smartphones* e *tablets* que utilizam o sistema operacional *Android* cresceram surpreendentemente, aumentando de 200 000 ativados por dia em 2010 para 1,5 milhões de dispositivos ativados por dia em 2013 e atualmente é a plataforma com mais dispositivos ativos [Statista 2014]. Esse crescimento pode ser justificado pela quantidade de funcionalidades que são agregadas aos *smartphones* e *tablets* a cada ano, redução de custos dos aparelhos e a diversidade de opções que englobam os vários perfis de usuários, das soluções mais simples e baratas às mais complexas.

O sistema operacional mais utilizado por esses dispositivos no mundo é o *Android*. Ele é construído com base nas contribuições da comunidade *open-source Linux*, possuindo mais de 300 versões de *hardware* e *software*. O código aberto do *Android* e a facilidade em publicar aplicativos na loja virtual favoreceram para torná-lo

o favorito por consumidores e desenvolvedores. Atualmente possui mais de 1,5 bilhões de aplicativos no *Google Play*, loja virtual do sistema operacional [Android Developer 2015].

Os *smartphones* e *tablets* possuem diversos modelos (*hardware* diferentes) e várias versões do sistema operacional, cada um com sua especificidade. Essa alta taxa de diferentes dispositivos e sistemas operacionais destaca um problema chamado de fragmentação. Esse problema é caracterizado pela necessidade de construção de *software* que funcione perfeitamente nos diferentes modelos de *hardware* (diferentes implementações) e *software* (diferentes sistemas operacionais).

Para diminuir o impacto deste problema é necessário garantir que um único aplicativo possa ser executado corretamente em todas as diferentes plataformas e sistemas operacionais. O problema acontece com maior intensidade no sistema operacional *Android* por possuir uma quantidade maior de versões do sistema, mas o problema também é recorrente em outros sistemas operacionais, como o *iOS* ou *Windows Phone* [Fuerstenberg 2014 apud Huang 2014].

Uma alternativa para amenizar o problema da fragmentação é testar o aplicativo na maior quantidade de dispositivos, com a maior variedade de modelos e versões do sistema operacional. É impraticável para a maioria dos desenvolvedores comprarem todos os modelos para realizarem os testes.

Outra alternativa é o uso de emuladores, mas não são substitutos ideais pois não podem emular todos os aspectos dos dispositivos, tais como: diferentes tamanhos de tela, coordenadas reais de *GPS* ou outros sensores, realizar chamadas e enviar mensagens de texto. Além disso, não há simuladores suficientes para emular todos os tipos de dispositivos disponíveis.

Pela necessidade de prolongar o tempo de uso dos dispositivos, o forte crescimento dos dispositivos móveis e do sistema operacional *Android*, o avanço no desenvolvimento de processadores com vários núcleos, torna-se necessário o desenvolvimento de técnicas que diminuam o consumo energético desses dispositivos e, por conseguinte, aumente seu tempo de uso. Esta é uma preocupação no *design* da maioria dos dispositivos móveis, dentre esses estão os *smartphones* e *tablets*.

Dispositivos com mais funcionalidades exigem maior poder de processamento e consequentemente maior gasto energético. Esse poder de processamento é mensurado pela quantidade de processadores e núcleos e pela frequência do *clock* de cada núcleo do processador. Para o desenvolvimento de técnicas que promovam eficiência energética é necessário entender o problema abordado e a caracterização do seu gasto energético no dispositivo móvel. Para realizar a caracterização é necessário possuir uma infraestrutura de medição energética.

A motivação desse trabalho é a falta de um ambiente de medição (integração de *hardware* e *software*) que auxilie na caracterização do consumo energético de dispositivos móveis e permita automatizar testes. A caracterização é um passo necessário no desenvolvimento e teste de novas técnicas que visam reduzir o consumo de energia em dispositivos móveis. Será utilizada uma interface *Web* para manipulação do ambiente proposto a fim de permitir o uso por pesquisadores espalhados geograficamente.

O ambiente proposto poderá ser usado para caracterizar energeticamente *smartphones* e *tablets Android* de forma automática e escalável. A automatização na reprodução de testes em diversos dispositivos simultaneamente possibilita redução do tempo na proposta de novas técnicas que melhorem o consumo energético de aplicativos. Também poderá ser usado para diminuir o problema da fragmentação, permitindo testes em múltiplos dispositivos simultaneamente. Empresas fabricantes de dispositivos *Android* poderão utilizar os dados gerados pelo ambiente proposto para criarem aplicativos adaptados a cada tipo específico de *hardware* e versão do sistema operacional, provendo eficiência energética no produto.

O ambiente de medição energética proposto neste trabalho automatiza a avaliação do consumo energético de diversos trabalhos, disponibilizando a infraestrutura necessária para a medição e o teste das funcionalidades ou aplicativos. É possível replicar trabalhos da literatura no ambiente de medição proposto para os dispositivos disponíveis. Exemplos de trabalhos que podem ser parcialmente automatizados pela proposta são: Carroll e Heiser (2013), Chang et al (2013), Dong, Lai e Li (2013), Huang et al (2012), Liang, Lai e Chiou (2010), Shin et al (2013), Silva-Filho et al (2012), Xu et al (2013), dentre outros.

O restante deste trabalho é organizado como segue: na seção 2 têm-se os conceitos necessários para o entendimento do restante deste trabalho, na seção 3 o ambiente proposto é detalhado, o ambiente de medição e *software,* juntamente com a arquitetura e os módulos. Na seção 4 são detalhados os experimentos e é feita a análise dos resultados, na seção 5 os trabalhos relacionados ao proposto são comparados e por fim, na seção 6 estão as conclusões e trabalhos futuros.

2. *Background*

Medição energética é a avaliação da quantidade de energia necessária para realizar uma quantidade de trabalho. Energia é a quantidade de *joules* gastos em um determinado tempo e potência é a taxa com que a energia é gasta, o último é medido em *watts* (*joules*/segundo). Para mensurar quanto de energia é gasta em um dispositivo é necessário saber a potência gasta em um determinado tempo. Para calcular a potência é necessário possuir amostras da corrente e tensão que passam pelo dispositivo.

Para estimar a quantidade de energia gasta é preciso calcular a área abaixo do gráfico da potência instantânea pelo tempo do experimento. O método do trapézio [Atkinson 1989] pode ser usado para calcular a área do gráfico que representa a energia consumida em um dispositivo. A partir das amostras de corrente e tensão, é calculado a potência instantânea que é plotada em um gráfico pelo tempo. O método funciona basicamente em dividir a área do gráfico em ŋ pedaços, onde cada pedaço forma um trapézio e então a área pode ser mensurada. A integral de cada intervalo é calculada e ao final, todos os intervalos são somados para obter a aproximação do valor da energia gasta.

2.1. *Linux* e *Android*

O sistema operacional *Linux* usa *governors* para controle operacional do sistema. *Governors* são módulos reguladores do *kernel Linux* que se comunicam com o *hardware*. A principal função é regular as frequências de operação dos núcleos do

processador. O sistema operacional *Android* é baseado no *kernel Linux* e também usa *governors*. Há *governors* que focam em desempenho, enquanto outros, em economia de energia. Essa escolha deve ser feita de acordo com o perfil de uso do usuário. [Pallipadi e Starikovskiy 2006].

Desempenho e redução energética são variáveis opostas. Para melhorar uma, consequentemente piora-se a outra. É necessário balancear essas variáveis para prover ao usuário uma experiência de uso melhor. Os *governors* mais utilizados no sistema operacional *Linux* são: *Ondemand, Performance* e *PowerSave*. Para o sistema *Android*, tem-se os anteriores mais outros adaptados para dispositivos móveis, como o *Interactive, PegasusQ, Userspace,* dentre outros.

2.2. Técnicas para Redução do Consumo Energético

Várias técnicas que reduzem o consumo energético em dispositivos móveis estão disponíveis na literatura. É possível realizar otimizações energéticas em diversas áreas, como na transmissão de dados: escolher de forma inteligente e em tempo de execução qual rede de dados utilizar, *Wifi* ou redes móveis (2G/3G/4G) [Perrucci et al 2009], diminuição do tempo ocioso no final das transmissões, essa técnica é chamada de *fast dormancy* [Xu et al 2013], agrupar pacotes de dados e enviar por rajadas (*batching*) [Palit, Naik e Singh 2011] [Huang et al 2012], desacoplar o processamento da transmissão dos dados [Xu et al 2013], *off-load* de processamento para a nuvem, DVFS (*Dynamic Voltage and Frequency Scaling*), entre outras.

Na técnica de DVFS (*Dynamic Voltage and Frequency Scaling*), é possível diminuir a frequência de operação do processador ou a tensão de alimentação para diminuir a potência gasta. Em alguns casos, o desempenho é afetado pela diminuição da frequência de operação, enquanto que em alguns casos é indiferente por não haver muito processamento. O consumo energético é calculado através da fórmula da potência instantânea. A potência é proporcional à frequência e a tensão ao quadrado: $P \propto f * v^2$ [Domeika 2008].

3. Ambiente Proposto

O ambiente de medição proposto fornece uma plataforma para medição energética de dispositivos móveis. A partir de uma sequência de ações, cria-se um teste que é replicado em vários dispositivos simultaneamente e o consumo energético individual de cada aparelho é medido. O processo de criar uma sequência de teste pode ser feito pelo usuário na interface *Web* do ambiente ou realizando o *upload* de um arquivo com os testes.

As sequências de ações são testes em funcionalidades do dispositivo, por exemplo, fazer chamada telefônica, enviar *e-mail*, abrir o navegador *Web*, enviar *SMS*, reproduzir vídeos em *stream* ou ações dentro de um aplicativo não nativo do sistema operacional *Android*. As sequências podem ser replicadas em diferentes condições do dispositivo, como diferentes redes de dados, frequência mínima e máxima de operação de cada núcleo do processador, escolha do *governor Android*, o número de iterações que o teste vai ser repetido e em quais dispositivos o teste será replicado. Todas essas configurações são ajustadas pelo usuário através da interface *Web* do ambiente.

O ambiente proposto (Figura 1) é dividido em: Ambiente de Medição e Ambiente de *Software*. O primeiro responsável pela infraestrutura física necessária para realizar medições energéticas nos dispositivos enquanto o segundo é responsável pelo processamento dos dados, comunicação com os componentes da plataforma, interface e comunicação com o usuário e geração de resultados.

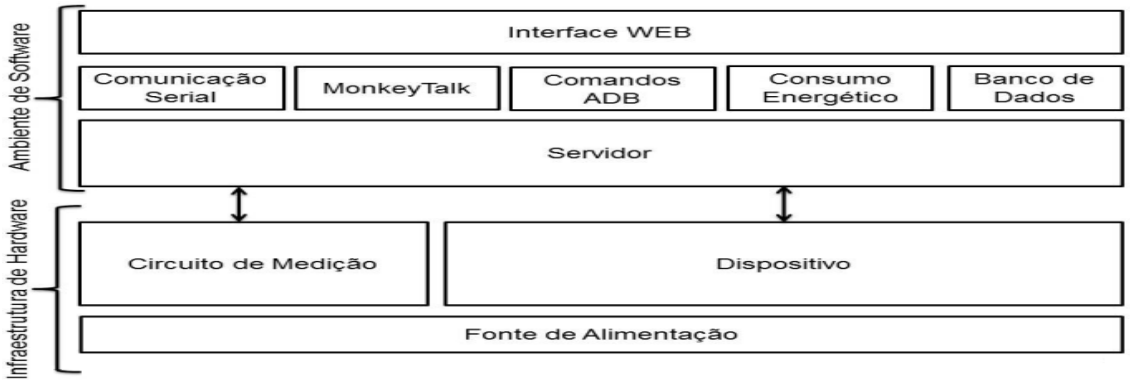

Figura 1. Ambientes de *Hardware* e *Software*

O ambiente de medição é composto por uma fonte de alimentação que fornece a energia necessária para o funcionamento do dispositivo, o circuito de medição, o dispositivo em análise e o servidor. Qualquer *hardware* que necessite de alimentação energética externa pode ser incluído nessa infraestrutura, tais como: *smartphones*, *tablets*, computadores embarcados, sensores, entre outros.

No ambiente de medição, cada dispositivo requer um circuito de medição independente. A fonte de alimentação envia energia para o circuito de medição que por sua vez fornece a energia necessária para o funcionamento do dispositivo. Do dispositivo há uma conexão *USB* para envio dos comandos pelo servidor e outra conexão do dispositivo para o circuito de medição para a alimentação energética. A conexão *USB* entre o circuito de medição e o servidor serve para o envio das amostras coletadas de corrente e tensão.

O ambiente de *software* é composto por módulos responsáveis por: geração da interface *Web* para o usuário, realizar a comunicação serial com o circuito de medição, o de automatização dos testes que utiliza a biblioteca *MonkeyTalk*, gerência de comandos *ADB* para manipulação do processador, redes móveis e outras configurações de baixo nível no dispositivo medido, o de cálculo do consumo energético com a implementação do método do trapézio e o módulo de armazenagem das medições realizadas em um banco de dados relacional.

3.1. Circuito de Medição

O circuito de medição é composto por um micro controlador, um circuito integrado responsável por coletar as amostras de corrente e tensão, um conversor mine *USB* e outros componentes eletrônicos. O circuito responsável pela coleta das amostras de corrente e tensão é conectado entre a fonte de alimentação e o dispositivo que está recebendo a carga. O circuito envia os dados para o micro controlador que envia para o servidor através da conexão *USB*. O circuito de medição pode ser visto na Figura 2.

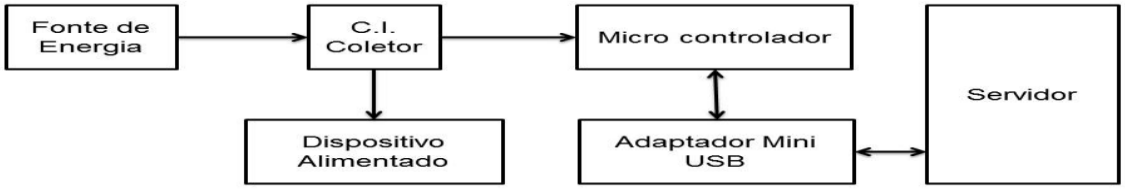

Figura 2. Circuito de Medição

3.2. Dispositivos no Ambiente

Na infraestrutura proposta foram incluídos dois *smartphones* com o sistema operacional *Android*, um *Samsung Galaxy S3 GT-9305T* e um *Samsung Galaxy S4 GT-I9505*. As especificações técnicas dos aparelhos estão disponíveis na Tabela 1.

Tabela 1. *Smartphones* disponíveis na infraestrutura proposta

	GT-9305T	**GT-I9505**
SoC	Exynos 4412	Snapdragon 600
CPU	ARM Cortex A9	Krait 300
Núcleos	4	4
Frequências Disponíveis (MHz)	200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100, 1200, 1300 e 1400 (13 Frequências)	384, 486, 594, 702, 810, 918, 1026, 1134, 1242, 1350, 1458, 1566, 1674, 1782 e 1890 (15 frequências)
Versão *Android*	4.1.2	4.4.2

Com pequenas modificações é possível adicionar novos dispositivos na infraestrutura de medição, isso contribui para tornar o ambiente escalável. É possível adicionar *smartphones* ou *tablets* com outros sistemas operacionais, como *iOS*, *Windows Phone*, entre outros. No sistema operacional *iOS* é possível adaptar os comandos *ADB* utilizados no *Android* através de uma conexão *SSH* estabelecida entre o servidor e o dispositivo que está sendo medido. Essa conexão permite executar comandos na *shell* do dispositivo, assim como é feito com comandos *ADB* no *Android*.

As informações de cada aparelho que são mostradas ao usuário através da interface, algumas como: quantidade de núcleos, frequências do processador, *governors* disponíveis ou redes de dados disponíveis, podem ser adquiridas em tempo de execução através de comandos *ADB*.

3.3. Funcionalidades do Ambiente Proposto

O ambiente de medição permite a seleção de qual rede de dados o dispositivo em teste deve utilizar (*Wifi*, *2G*, *3G* ou *4G*), a frequência de operação de cada núcleo do processador do dispositivo, quais dispositivos serão testados, quantidade de iterações e tudo isso feito através de uma interface *Web*. Todas as configurações são definidas no dispositivo utilizando comandos *ADB* antes de ser iniciada a sequência de reprodução dos testes.

Seleção de Rede: O usuário pode escolher qual rede o dispositivo deve utilizar durante a realização dos testes. As opções disponíveis são: Wifi, 2G, 3G ou 4G.

Seleção de frequência do processador: Além da definição das frequências de operação de cada núcleo do processador individualmente, o usuário também pode definir quantos núcleos estarão ativos.

Dispositivos para teste: O usuário seleciona quais dispositivos serão testados de forma interativa, todas as configurações definidas anteriormente são aplicadas a cada dispositivo selecionado.

Quantidade de iterações: O ambiente permite que o usuário selecione quantas iterações o teste deve ser repetido. Calcula a média e o desvio padrão dos experimentos e exibe ao usuário.

4. Análise dos Resultados

Foram realizados três experimentos para demonstrar o uso do ambiente proposto. Nos experimentos 1 e 2, foram reproduzidos vídeos do *YouTube* em *stream,* a cada iteração o aplicativo *YouTube* era inicializado com o *link* e o vídeo reproduzido por 120 segundos. No experimento 1 foram utilizadas as combinações de frequências do processador disponíveis nos aparelhos do ambiente com 4 (quatro) núcleos ativos e o *governor* definido para o *Userspace*, ou seja, não será realizado chaveamentos entre as frequências durante o experimento. No experimento 2, a configuração que obteve menor consumo energético no experimento 1 foi comparada com os *governors* padrões do *Android*.

No experimento 3 um aplicativo *Android* foi criado para estressar o processador até uma porcentagem desejada. Esse aplicativo foi enviado através da interface *Web* para a plataforma de medição que se encarrega de instalar, executar e repetir a sequência de testes em todos os aparelhos. Foi criado um teste que ajustava o aplicativo para estressar o processador em 25%, 50% e 75% de carga.

Cada experimento foi repetido 30 vezes para obter-se uma consistência maior nos resultados. Abaixo são demonstrados os valores médios de todos os experimentos, nenhum teve desvio-padrão maior que 7%.

4.1. Experimento 1 - *Stream* de Vídeo

No gráfico da Figura 3 estão os dados coletados durante o experimento *stream* de vídeo para o *Samsung Galaxy S3*. É perceptível que a frequência de 700MHz e adjacentes são as mais energeticamente eficientes. Essa informação pode ser utilizada para gerar algoritmos de escalonamento inteligentes e serem incorporados na produção de novos *governors* para o sistema operacional *Android*.

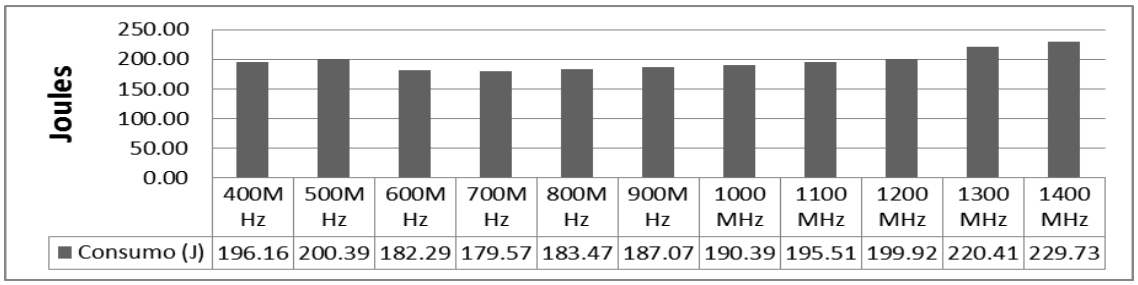

Figura 3. Gráfico do Consumo Energético com quatro núcleos ativos no *smartphone Samsung Galaxy S3* (Experimento *Stream* de Vídeo)

No gráfico da Figura 4 são apresentados os dados para o experimento de *stream* de vídeo para o *smartphone Samsung Galaxy S4*. Novamente os experimentos mostram o consumo mínimo sendo com a frequência de 702MHz e adjacentes.

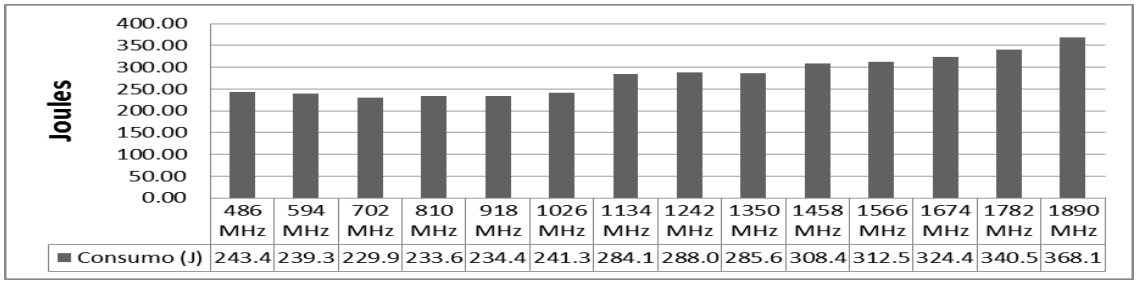

Figura 4. Gráfico do Consumo Energético com quatro núcleos ativos no smartphone Samsung Galaxy S4 (Experimento Stream de Vídeo)

Com esses resultados, pode-se indicar que as frequências próximas de 700MHz são as mais energeticamente eficientes para uso com *stream* de vídeo. Esse resultado é verdade tanto para o *smartphone Samsung Galaxy S3* como para o *Samsung Galaxy S4*.

4.2. Experimento 2 - *Stream* de Vídeo com *Governors*

Neste experimento os valores energéticos dos *governors* padrões do *Android* são comparados com a frequência que obteve menor consumo energético no macro experimento anterior.

No gráfico da Figura 5 estão os valores energéticos para o *smartphone Samsung Galaxy S3*. O eixo vertical indica o consumo energético enquanto o horizontal são os *governors* ou a frequência fixa (sem chaveamentos). É perceptível que o menor consumo energético pertence à frequência fixa de 700MHz (4C-700MHz) com quatro núcleos ativos. Os *governors PegasusQ* e *Performance* foram os que apresentaram consumos energéticos mais próximo do mínimo e valores quase iguais entre si. Essa aproximação se dá pela constante necessidade de processamento em aplicações do tipo *stream* de vídeo, enquanto o *Ondemand* ficou com um consumo mais elevado. Isso pode se justificar pela alta taxa de chaveamentos de frequências que acontecem no dispositivo com o *governor* operando.

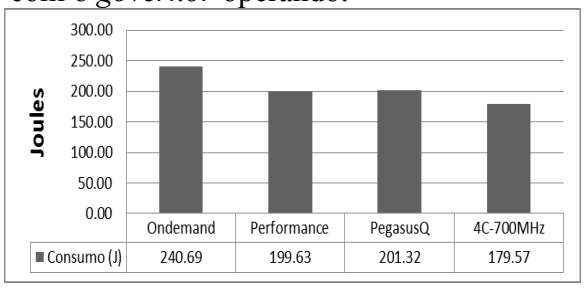

Figura 5. Consumo Energético dos *Governors* no *Samsung Galaxy S3*

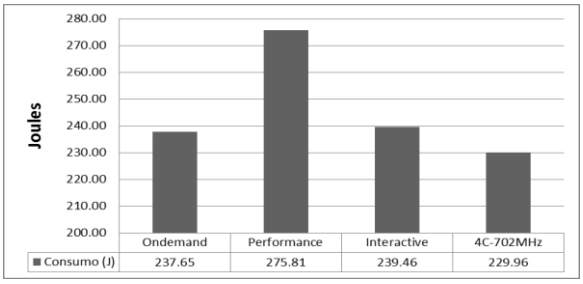

Figura 6. Consumo Energético dos *Governors* no *Samsung Galaxy S4*

No gráfico da Figura 6 estão os valores coletados para os experimentos energéticos de *stream* de vídeo com os principais *governors* do aparelho *Samsung Galaxy S4*. Novamente a frequência fixa de 702MHz com quatro núcleos ativos (4C-702MHz) obteve o menor consumo energético comparado aos demais governors.

Os resultados energéticos dos demais *governors* saíram como o esperado, o *Interactive* que é o padrão no *Samsung Galaxy S4* obteve o segundo menor valor de consumo enquanto o *Ondemand* estava em terceiro lugar. O *Performance* foi o que apresentou maior consumo energético.

Pode-se perceber nos experimentos 1 e 2 que a frequência ideal para prover eficiência energética para a funcionalidade *stream* de vídeo é igual ou próxima de 700MHz com o número máximo de núcleos ativos, tanto para o aparelho *Samsung Galaxy S3* quanto para o *Samsung Galaxy S4*. Provavelmente essa característica pode ser generalizada para os demais *smartphones*, mas serão necessários mais testes nos dispositivos alvos para comprovar a hipótese.

A técnica de não realizar chaveamentos entre as frequências (manter frequência fixa) permitiu uma economia de energia de 25.39% para o *Samsung Galaxy S3* quando comparado com o *governor Ondemand* e de 10.80% comparado ao *PegasusQ*. Para o *Samsung Galaxy S4* teve-se economia de 16.62% na comparação com o *governor Performance* e de 3.96% comparado ao *governor Interactive*.

4.3. Experimento 3 - Carga de Processamento

Neste último experimento, foi feita a avaliação energética de um aplicativo *Android* não nativo do sistema operacional, ou seja, que não pertence ao conjunto de aplicativos padrões do *Android*.

O aplicativo consiste em criar *threads* que realizam operações matemáticas para sobrecarregar o processador até o nível desejado. Os níveis desejados nos experimentos são: 25%, 50% e 75% de carga no processador. Inicialmente a porcentagem de 100% foi testada, porém foi retirada dos experimentos por ocasionar travamentos nos aparelhos e impossibilitar a continuação do teste. A função para calcular o nível de carga foi implementada no aplicativo juntamente com uma interface gráfica. A interface é utilizada pelo ambiente proposto para automatizar os testes simulando ações humanas de toque na tela. O nível de carga é mantido destruindo ou criando *threads*.

O experimento foi realizado para a combinação de 1 (um) à 4 (quatro) núcleos ativos do processador com a frequência máxima de operação, sendo a mesma para todos os núcleos ativos. Os consumos energéticos para as combinações de núcleos ativos e a porcentagem de carga de processador podem ser vistos na Figura 7. O eixo vertical apresenta os valores de corrente consumida em miliampéres enquanto o eixo horizontal representa os núcleos ativos e a porcentagem de carga do processador. Como foi visto anteriormente, a energia consumida é proporcional à tensão e a corrente consumida pelo dispositivo, como foi utilizada uma tensão constante de 4.8V para o *Samsung Galaxy S3* e de 4.9V para o *Samsung Galaxy S4*, a potência consumida é diretamente proporcional a corrente.

Como o esperado, os consumos energéticos são maiores com o incremento da porcentagem de carga do processador e também aumentam com o incremento da quantidade de núcleos ativos.

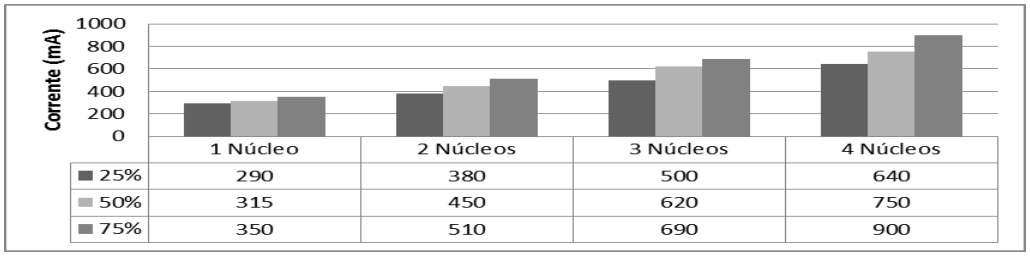

Figura 7. Gráfico da Corrente pela Carga de Processamento no *smartphone Samsung Galaxy* S4 (Experimento Carga de Processamento)

A partir dos dados no gráfico da Figura 7 é possível afirmar que para processos *CPU-bound* é mais energeticamente eficiente utilizar a quantidade máxima de núcleos ativos na frequência máxima. Para chegar a essa conclusão, foi realizada uma proporção para saber o valor médio de corrente necessário para a execução de 1% de carga de processamento. Fazendo a proporção para 4 núcleos ativos temos para 25%, 50% e 75% de carga: 640/25 = 25.6mA, 750/50 = 15mA e 900/75 = 12mA por 1% de processamento. É perceptível um menor consumo energético para realizar a mesma quantidade de trabalho utilizando uma quantidade maior de núcleos. O resultado seguiu verdade para as proporções de 1, 2 e 3 núcleos ativos.

Deve haver uma análise mais aprofundada para verificar qual frequência de operação é mais eficiente com a combinação dos núcleos ativos. Isso não é o foco deste trabalho, mas permite demonstrar que o ambiente possibilita que seja feita essa análise de forma parcialmente automatizada, assim como foi feito no trabalho de Carroll e Heiser (2013) de forma manual.

5. Trabalhos Relacionados

O teste de *software* em dispositivos móveis é necessário para combater o problema da fragmentação. Esse teste pode ser escalável, realizando o mesmo teste em diversos dispositivos simultaneamente e economizando tempo nos testes. Várias empresas oferecem o serviço na nuvem para testes em dispositivos móveis, o que diminui o gasto com compra de aparelhos e tempo de implantação do *software* no dispositivo.

O trabalho proposto visa complementar os trabalhos relacionados descritos na Tabela 2. Nenhum dos trabalhos consultados permitia **testes de *software* e análises energéticas**, mesmo sendo esse último essencial no desenvolvimento de técnicas que aumentem o tempo de uso dos dispositivos dependentes de bateria.

Tabela 2. Características dos trabalhos relacionados

Característica	Este Trabalho	Power Monitor[1]	Energino[2]	Xamarin[3]	*APPThwack*[4]	*Monkey Mobile* Cloud[5]
Consumo Energético	Sim	Sim	Sim	Não	Não	Não
Controle do Processador	Sim	Não	Não	Não	Não	Não

Interface *Web*	Sim	Não	Não	Sim	Sim	Sim
Multidispositivo	Sim	Não	Não	Sim	Sim	Sim
Multiplataforma	Sim	Não	Não	Sim	Sim	Sim
Multilinguagem	Sim	Não	Não	Sim	Sim	Sim

[1]https://www.msoon.com/LabEquipment/PowerMonitor/ [2]http://www.energino-project.org/ [3]http://www.xamarin.com/
[4]https://www.appthwack.com/ [5]https://www.cloudmonkeymobile.com

6. Conclusões e Trabalhos Futuros

Dada à necessidade de um ambiente que auxiliasse a diminuir o problema da fragmentação e permitisse análises energéticas, o trabalho proposto ofereceu uma combinação de infraestrutura e um ambiente de *software* que auxilia na caracterização energética de testes em multidispositivos através de uma interface *Web*.

Para testar o ambiente proposto foram realizados três macro experimentos abrangendo reprodução de vídeos em *stream* e carga de processamento. O primeiro teste apresentou como resultado uma frequência ótima sendo próxima de 700MHz para a reprodução de vídeos em *stream*. Essa informação possibilitou até 25% de economia de energia em comparação com os *governors* padrões do *Android* nos *smartphones* testados. No experimento sobre carga de processamento foi possível indicar a utilização do máximo número de núcleos na frequência máxima quando há processos *CPU-bound*, essa informação pode proporcionar economia de energia na execução de processos que fazem uso intenso de processamento.

Como trabalhos futuros, temos a necessidade da análise de mais cenários para a identificação da frequência ótima que promova eficiência energética. Esse conhecimento será aplicado no desenvolvimento de um novo *governor* para o sistema operacional *Android*.

Outra linha de evolução é a disponibilização do ambiente como um serviço por meio da *Web* de forma gratuita e a disponibilização do código fonte para a comunidade acadêmica. Pesquisadores poderão caracterizar do ponto de vista energético suas técnicas em dispositivos reais.

Agradecimentos

A Fundação de Amparo à Ciência e Tecnologia de Pernambuco (FACEPE) pelo financiamento parcial deste trabalho, processo IBPG-0731-1.03/12 e IBPG-1269-1.03/14 e ao Centro de Informática (CIn) da Universidade Federal de Pernambuco.

Referências

Android Developer. (2015) “Android, the world's most popular mobile platform”. 2015. Disponível em < http://developer.android.com/about/index.html >. Acesso em: 02 mar. 2015.

Carroll, A. e Heiser, G. (2013) “Mobile Multicores: Use Them or Waste Them”. Proceedings of the 5th Workshop on Power-Aware Computing and Systems (HotPower’13). 2013.

Chang, Y.M., Hsiu, P.C., Chang, Y.H. e Chang, C.W. (2013) “A Resource-Driven DVFS Scheme for Smart Handheld Devices”. ACM Trans. on Embedded Computer Systems, 13(3):53:1–53:22, 2013

Domeika, M. (2008) “Software Development for Embedded Multi-core Systems: A Practical Guide Using Embedded Intel Architecture”. Newnes, Newton, MA, 2008.

Dong, M., Lai, P. e Li, Z. (2013) “Can We Identify Smartphone App by Power Trace?”. Proceedings of the 8th Asia and South Pacific Design Automation Conference (ASP-DAC). 2013.

Huang, J. (2014) “AppACTS: Mobile App Automated Compatibility Testing Service”. International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud), IEEE. 2014.

Huang, J., Qian, F., Mao, Z., Sen, S. and Spatscheck, O. (2012) “Screen-off traffic characterization and optimization in 3g/4g networks”. In IMC ’12.

Liang, Y., Lai, P. e Chiou, C. (2010) “An energy conservation dvfs algorithm for the Android operating system”. Journal of Convergence, Vol. 1, pp.93-100. 2010.

Palit, R., Naik, K. e Singh, A. (2011) “Impact of Packet Aggregation on Energy Consumption in Smartphones”, In Proceedings of the 7th International Wireless Communications and Mobile Computing Conference, Istanbul, Turkey, 2011, pp. 589-594

Pallipadi, V. e Starikovskiy, A. (2006) “The ondemand governor: past, present and future.” In Proceedings of Linux Symposium, vol. 2, pp. 223-238, 2006.

Perrucci, G.P., Fitzek, F.H.P., Sasso, G. , Kellerer, W. and Widmer, J. (2009) “On the Impact of 2G and 3G Networks Usage for Mobile Phones' Battery Life”, in European Wireless Conference.

Shin, D., Lee, W., Kim, K., Wang, Y., Xie, Q., Pedram, M. e Chang, N. (2013) “Online estimation of the remaining energy capacity in mobile systems considering system-wide power consumption and battery characteristics”. In Proc. of Asia South Pacific Design Automation Conference (ASP-DAC), 2013.

Silva-Filho, A.G., Bezerra, P.T.L., Silva, F.Q.B., Junior, A.L.O.C., Santos, A.L.M., Costa, P.H.R. e Miranda, R.C.G. (2012) “Energy-Aware technology-based DVFS mechanism for the Android operating system”. Brazilian Symposium on Computing System Engineering (SBESC), 2012.

Statista. (2014) “The Statistics Portal”. Disponível em: < http://www.statista.com/statistics/278305/daily-activations-of-Android-devices >. Acesos em: 24 nov. 2014.

Xu, F., Liu, Y., Moscibroda, T., Chandra, R., Jin, L., Zhang, Y. and Li, Q. (2013) “Optimizing background email sync on smartphones”. In Proceeding of the 11th International Conference on Mobile Systems, Applications, and Services, MobiSys ’13, pages 55–68, New York, NY, USA, 2013. ACM.