

# Eficiência Energética em Navegação Web usando DVFS

Renato Melo, Vinicius Petrucci

PGCOMP - Programa de Pós-graduação em Ciência da Computação  
Universidade Federal da Bahia (UFBA)  
Salvador – BA – Brasil

**Abstract.** *Energy consumption has become a major concern in data centers and mobile devices. This work proposes an intelligent energy manager, Web Governor, that explores DVFS (Dynamic Voltage and Frequency Scaling) technique available in modern processors to reduce energy consumption for Web browsing applications, leveraging characteristics of the communication network. Experimental results show that Web Governor is able to reduce the system energy consumption by 12% (average) and 18% (max) when compared to Linux governor developed by Google/Android, while loading the web pages faster (on average, 4%). When compared to a performance-focused Linux DVFS manager, Web Governor improves energy efficiency by 29% on average and up to 49%, maintaining similar load time for a variety of web pages.*

**Resumo.** *O consumo de energia tornou-se uma grande preocupação em data centers e dispositivos móveis. Este trabalho propõe um gerenciador inteligente de energia, Web Governor, que explora a técnica DVFS (Dynamic Voltage and Frequency Scaling) disponível em processadores modernos para reduzir o consumo de energia em aplicações de navegação na Web, com o diferencial em explorar características da rede de comunicação. Resultados experimentais mostram que o Web Governor é capaz de reduzir o consumo de energia do sistema em 12% (média) e 18% (max) quando comparado ao governador DVFS do Linux desenvolvido pelo Google/Android, enquanto carrega as páginas mais rapidamente (em média, 4%). Quando comparado a um gerenciador DVFS do Linux focado no desempenho, o Web Governor melhora a eficiência energética em 29% em média e até 49%, mantendo semelhante o tempo de carregamento para uma variedade de páginas Web.*

## 1. Introdução

Atualmente, os usuários estão cada vez mais conectados à Web e, para isso, têm usado os mais variados tipos de dispositivos como *Laptops*, *Tablets*, *Smartwatches* e *Smartphones*. Grande parte das informações requisitadas pelos usuários são obtidas por páginas da Web com conteúdos variados. Ao longo dos últimos anos, as páginas Web têm se mostrado cada vez mais sofisticadas e têm exigido maior poder de processamento pelos dispositivos móveis. Em contraste à crescente melhoria em desempenho nos dispositivos móveis, não observa-se uma melhora significativa na eficiência energética desses dispositivos [20], o que nos leva a investigar novas formas de prover melhor balanceamento entre o consumo energético e o desempenho em aplicações importantes como a navegação Web [16, 23].

Fabricantes de *hardware* (processador, memória, disco, etc) têm se preocupado com o custo energético dos seus componentes; por exemplo, permitindo ajustar a velocidade e consumo de cada componente individualmente, tais como CPU, adaptadores

de rede, discos ou ventoinhas. Além do mais, tais dispositivos podem entrar em estado de suspensão quando não estiverem sendo completamente utilizados [2]. Essas técnicas permitem tornar o *hardware* capaz de se adaptar à carga atual. Um dos componentes que representa maior gasto de energia nos dispositivos é o processador; assim, a maioria das técnicas de redução do consumo de potência atacam primeiramente processadores. A técnica de DVFS (*Dynamic Voltage and Frequency Scaling*), que envolve o ajuste dinâmico da tensão e da velocidade/frequência da CPU, tornou-se uma abordagem bem estudada e consolidada em economia de energia em processadores modernos [9, 17, 18].

Neste trabalho, exploramos o uso de processadores com suporte a DVFS a fim de prover eficiência energética para execução de aplicações envolvendo navegação *Web*. Nossa solução apoia-se na intuição e observação de que, além de observar as métricas de utilização dos processadores, devemos observar também o comportamento da rede de comunicação usada no acesso à *Web* [20]. Naturalmente, a rede de comunicação influencia no tempo requerido para carregar e exibir os dados no navegador para os usuários. De fato, observar o atraso da rede de comunicação na navegação *Web* mostra-se um aspecto crítico para permitir selecionar a frequência de maior eficiência energética, aproveitando assim melhor o potencial da técnica de DVFS fornecida pelos processadores. Contudo, não é uma tarefa trivial identificar a melhor frequência para os diferentes cenários de atrasos na rede de comunicação e complexidades das páginas *Web* que afetam o comportamento da aplicação de navegação *Web*.

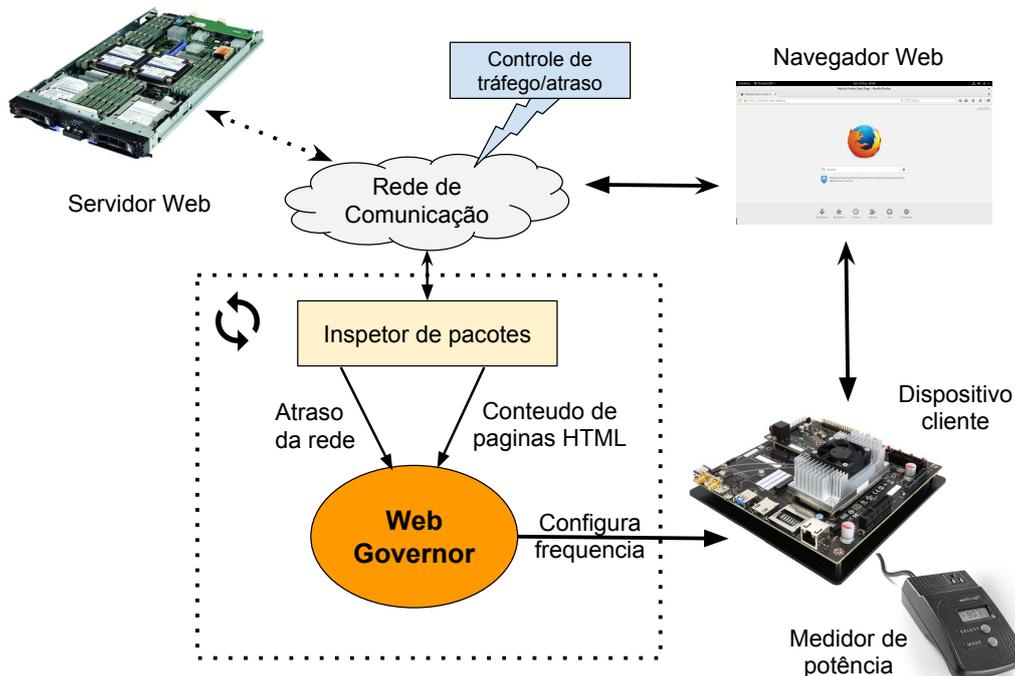
No contexto descrito acima, este trabalho apresenta as seguintes contribuições:

- Experimentos em um ambiente cliente/servidor demonstrando o impacto em eficiência energética do chaveamento da frequência do processador (DVFS) e do atraso da rede de comunicação durante a navegação *Web*;
- Projeto e implementação de um novo gerenciador de energia inteligente, chamado *Web Governor*, que explora conhecimento da aplicação e da rede de comunicação para decidir qual a melhor frequência de operação do processador, através de aprendizado de máquina,
- Resultados comparativos com diferentes políticas de gerenciamento do Linux: *Performance Governor (Baseline)*, *Interactive Governor (Google/Android)* e *Web Governor*, considerando distintas páginas *Web* e cenários de atraso da rede de comunicação em um ambiente cliente/servidor.

Nossos resultados demonstram que o gerenciador do sistema operacional Linux (*Interactive Governor*), desenvolvido pelo Google como parte do *Android* para aplicações interativas como a navegação *Web*, porém agnóstico em relação à rede, consome em média 21% (e até 33%) mais energia do que a configuração de frequência ideal<sup>1</sup> que deveria ser usada pela CPU se a velocidade da rede fosse levada em consideração. Observa-se também que o *Web Governor* consegue obter desempenho melhor do que os alcançados pelo *Interactive Governor*, com média de 12% de economia de energia (chegando a 18%). Outro ponto a ser destacado sobre gerenciador de energia inteligente desenvolvido por este trabalho é que ele degrada somente 1% (em média) do tempo requerido para carregar as páginas *Web* comparado ao *Performance Governor*, com enfoque em desempenho máximo.

---

<sup>1</sup>Para este experimento, a frequência ideal foi identificada dado uma tolerância do usuário (10%) ao atraso no carregamento das páginas em relação à frequência máxima.



**Figura 1. Proposta Web Governor situa-se entre a rede de comunicação e o dispositivo cliente para efetuar a mudança de frequência da CPU, de acordo com características atualizadas do processamento da página e da rede de comunicação**

Neste trabalho, definimos como **frequência ótima ou ideal** aquela que permita reduzir ao mínimo o consumo de energia e que o atraso no carregamento das páginas seja menor ou igual a  $K\%$  em relação à frequência máxima, onde  $K$  é um parâmetro escolhido pelo usuário; salvo indicação em contrário, utilizamos o valor padrão de 20%.

O restante deste artigo está organizado da seguinte forma. Na Seção 2 descrevemos nossa proposta, *Web Governor*, no contexto de um ambiente experimental cliente/servidor usado na avaliação da eficiência energética de uma aplicação de navegação *Web*. Resultados e análises dos nossos experimentos são mostrados na Seção 3. Trabalhos relacionados são apresentados na Seção 4. Por fim, a conclusão e trabalhos futuros são apresentados na Seção 5.

## 2. Web Governor

Nesta seção apresentamos nossa proposta, *Web Governor*, no contexto do mecanismo DVFS do Linux em um ambiente cliente/servidor. Neste ambiente, o dispositivo cliente possui um processador com suporte DVFS que permite chavear dinamicamente sua frequência de operação. O cliente realiza acesso a um servidor *Web* que pode apresentar diferentes situações de atraso no acesso à rede durante a navegação *Web*.

### 2.1. Mecanismo de DVFS

O sistema operacional Linux dispõe de um mecanismo de DVFS que permite a utilização de *Governors* implementando diferentes políticas de ajuste de frequência da CPU. O DVFS permite ajustar o *Governor* de formas distintas. As diferentes configurações que

podem ser usadas pelo DVFS no sistema operacional Linux são descritas a seguir. **Performance:** Permite definir, estaticamente, que o processador trabalhe com a frequência mais alta obtendo consequentemente o melhor tempo de resposta com relação às tarefas executadas; **Powersave:** Configura o sistema para usar a frequência mais lenta do processador para máxima economia de energia; **Userspace:** Habilita que o usuário defina livremente qual frequência será usada pela CPU para atender condições específicas; **Ondemand:** Estabelece o uso da CPU de acordo com a demanda. Assim, o processador irá alterar a frequência rapidamente ajustando-se à demanda dinâmica de uso de CPU; **Interactive:** A velocidade da CPU varia com sua utilização, assim como acontece com o *Ondemand*. No entanto, o *Interactive* é mais agressivo e responsivo no chaveamento de frequência da CPU, obtendo melhor resposta à variação na carga de trabalho.

O *Interactive Governor* foi desenvolvido pelo *Google* para o sistema *Android*, de forma a adequar-se melhor a cargas de trabalho interativas, como a navegação *Web*. É importante enfatizar que todos os mecanismos de DVFS listados somente levam em consideração métricas de carga da CPU. Dessa forma, a velocidade da rede de comunicação tem um impacto indireto na utilização da CPU. Nossa proposta é um mecanismo de DVFS que leve em consideração os diferentes cenários de velocidade da rede de comunicação durante a navegação *Web*. O *Web Governor* utiliza-se do processador em modo *userspace* de forma que possa ser decidido por um mecanismo externo qual frequência será usada para executar a aplicação de navegação *Web*.

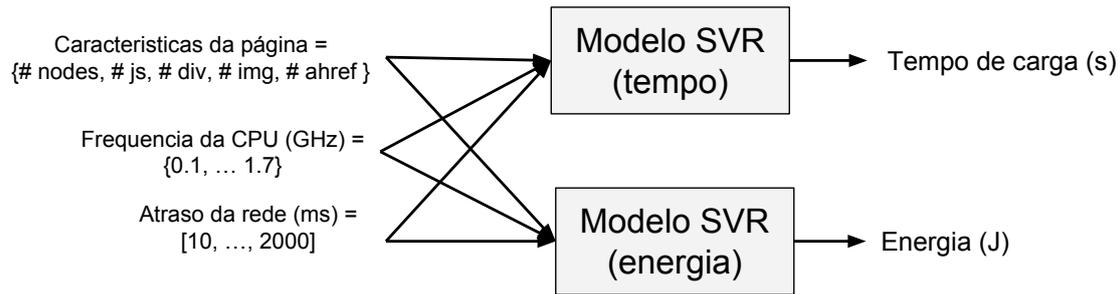
## 2.2. Fase de monitoração

O *Web Governor*, conforme descrito na Figura 1, utiliza-se de um inspetor de pacotes da rede de comunicação [7] que permite extrair informações sobre o atraso atual da rede e interceptar o conteúdo da página inicial requisitada pelo usuário; tipicamente, *index.html*. O tempo requerido para interceptação dos pacotes é rápido o suficiente para adaptar a frequência da CPU para processamento logo após a interceptação. A mudança de frequência também é rápida, na ordem de dezenas de microssegundos [5]. O objetivo é que o mecanismo que envolve desde a monitoração até a decisão sobre a frequência usada pelo processador não gere por si só um gasto adicional de tempo e energia.

Após inspecionar o conteúdo da página, o *Web Governor* extrai um conjunto de indicadores que permitem caracterizar o tempo requerido para processamento/renderização daquela página pelo navegador *Web*. As características usadas para representar a complexidade de uma página HTML foram selecionadas baseando-se em trabalho anterior [8], sendo elas: *número de objetos elementos no documento*, *número de referências Javascript*, *número de links externos*, e *números de tags div e img*. Apesar da diversidade das páginas, elas têm características comuns usadas amplamente no contexto da *Web*. Portanto, notou-se que essas características podem ser usadas pelo *Web Governor* junto com a informação da rede para auxiliar na seleção da frequência que produza maior eficiência energética.

## 2.3. Fase de decisão

O *Web Governor* monitora os dados das páginas que passam pela porta padrão HTTP (80). Assim que o usuário faz a requisição de uma página, as características das páginas são extraídas pelo módulo monitor para então acionar o módulo gerenciador que irá selecionar a frequência mais eficiente para processar aquela página *Web* específica. Para



**Figura 2. Modelos de previsão (SVR) para estimar o tempo de carga e o consumo de energia em função das características da página, da frequência da CPU, e do atraso da rede de comunicação.**

isso, o mecanismo de decisão utiliza um modelo de previsão baseado em aprendizado de máquina.

Utilizamos a biblioteca Scikit-learn [15] para implementar o mecanismo de decisão com base na técnica de SVR (Support Vector Regression), permitindo instanciar dois modelos de previsão: tempo de carregamento de página e consumo de energia, como ilustrado na Figura 2. O mecanismo de decisão permite prever qual o tempo de carga da página no navegador *Web* a partir dos dados monitorados (características da página e da rede de comunicação) e de uma determinada configuração de frequência de CPU. Assim, os modelos treinados a partir de páginas já navegadas pelos usuários anteriormente são usados para encontrar a melhor configuração de frequência que minimize o consumo de energia.

Para implementar os modelos de previsão, outras técnicas mais simples como *regressão linear* e *árvores de regressão* foram testadas, porém apresentaram resultados piores do que os obtidos com o mecanismo SVR. O uso de redes neurais (*Perceptron multicamadas*) apresentou resultados semelhantes aos do SVR. Escolhemos o SVR por este possuir melhor compromisso entre a qualidade e custo de treinamento/processamento das previsões em nossos testes.

#### 2.4. Qualidade do modelo de previsão

A Figura 3 apresenta o resultado da precisão do modelo durante a fase de treinamento e teste. Na criação dos modelos de previsão, separamos  $2/3$  dos dados coletados para treinamento e  $1/3$  para validação. Utilizamos o *radial basis kernel* e variamos os parâmetros  $C$  e  $\gamma$  que influenciam na precisão do modelo durante a fase de treinamento; os melhores parâmetros encontrados foram:  $C=1000$  e  $\gamma = 0.1$ , com medida de *R-squared* em 95% (energia) e 99% (tempo). *R-squared*, ou coeficiente de determinação, é uma medida estatística de quão perto os dados estão do modelo de regressão treinado; um valor de 0% indica que o modelo não explica nenhuma variabilidade dos dados.

As páginas utilizadas nos experimentos representam uma variedade de 11 páginas com menor ou maior requisito computacional e de comunicação, e seus conteúdos incluem mecanismo de buscas, sites de notícias e conteúdos variados, como Amazon, Google, ESPN etc, selecionadas a partir de trabalho anterior [19].

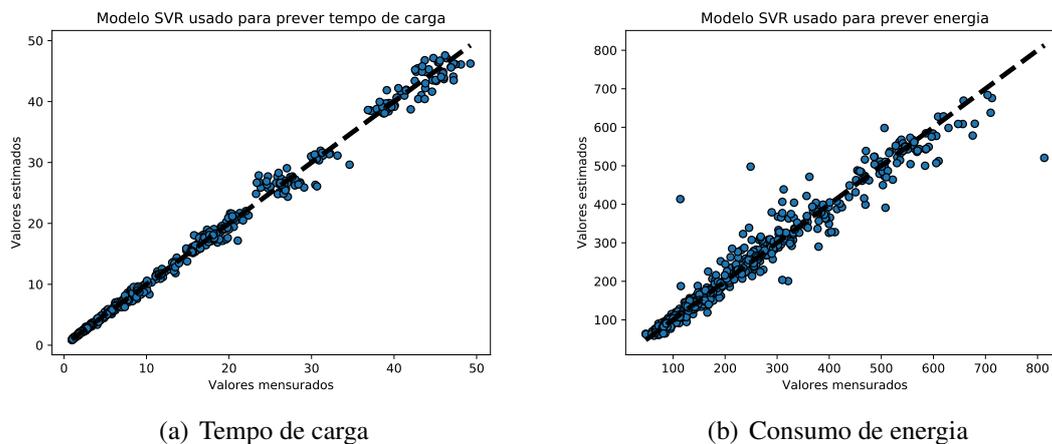


Figura 3. Precisão dos modelos de previsão: valor mensurado x valor estimado.

### 3. Resultados experimentais e análises

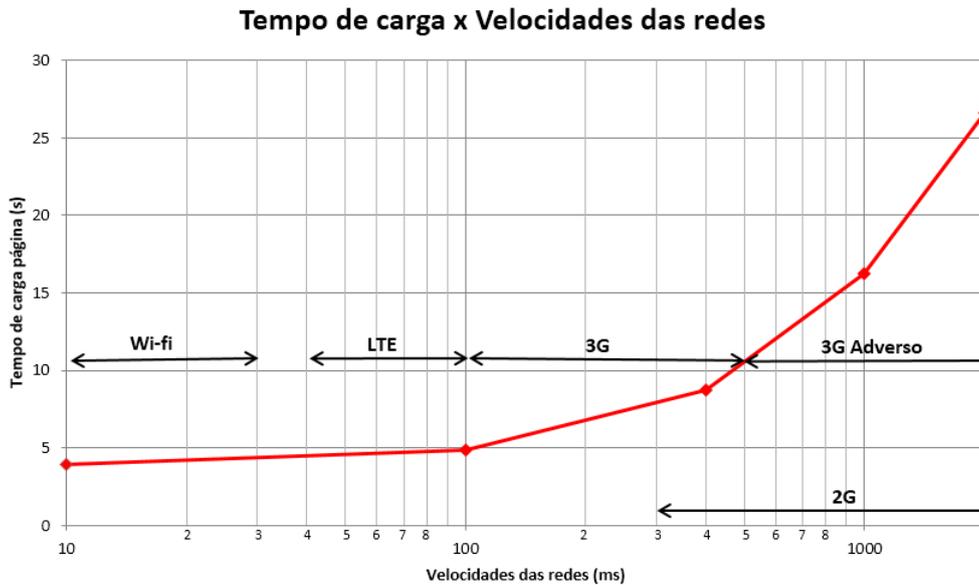
Nesta seção apresentamos resultados e análise dos experimentos realizados neste trabalho executando uma aplicação de navegação *Web* em um ambiente cliente/servidor controlado, contemplando diferentes configurações de acesso à rede.

#### 3.1. Ambiente experimental

**Servidor de páginas Web.** Utilizamos um servidor com processador *Intel Xeon* 8 núcleos E5-2407 2.20 GHz, com 64 GB de memória RAM. Esse servidor opera com sistema operacional *Linux*, distribuição *CentOS release 7.3.1611 (Core)*, *kernel 3.10.0-514.10.2.el7.x86\_64*. Esse equipamento hospeda um conjunto de 11 páginas *Web* mais utilizadas mundialmente (como Amazon, Google, CNN, ESPN, etc) localmente para serem acessadas pela máquina cliente. O servidor executa o Apache HTTP server, versão 2.4.6. A metodologia adotada assemelha-se a usada em [10] e [20], em que as páginas armazenadas no servidor local representam importantes cargas de trabalho da *Web* com conteúdos variados.

**Aplicação e Dispositivo cliente** No lado do cliente, utilizamos a ferramenta *PhantomJS* [11] que usa o software *WebKit* para processamento e renderização de páginas, de forma a caracterizar o comportamento real de uma aplicação acessando páginas *Web*. A aplicação é executada na placa *Jetson TX1* da *NVIDIA* [1], equipada com processador de 4 núcleos, com capacidade para variar sua velocidade entre as opções: 100 MHz até 1.7 GHz, com espaçamento de 100 MHz, totalizando 17 opções de frequências. As frequências são aplicadas igualmente a todos os núcleos do processador. A aplicação de navegação, todavia, utiliza apenas um núcleo durante sua execução. A placa possui 4 GB de memória RAM e executa o sistema operacional *Linux Ubuntu, kernel 3.10.96-tegra aarch64*. Importante frisar que a placa *Jetson TX1* usa processadores da arquitetura *ARM* que é a mesma utilizada atualmente na maior parte dos dispositivos móveis, como celulares *Samsung/Android* e *iPhone*.

**Medidor de potência.** Para aferir o consumo de energia do dispositivo cliente durante o carregamento das páginas *Web* foi utilizado o medidor real de consumo *Wattsup Pro* [12]. Esse instrumento é um dispositivo externo que foi colocado entre a fonte de energia da



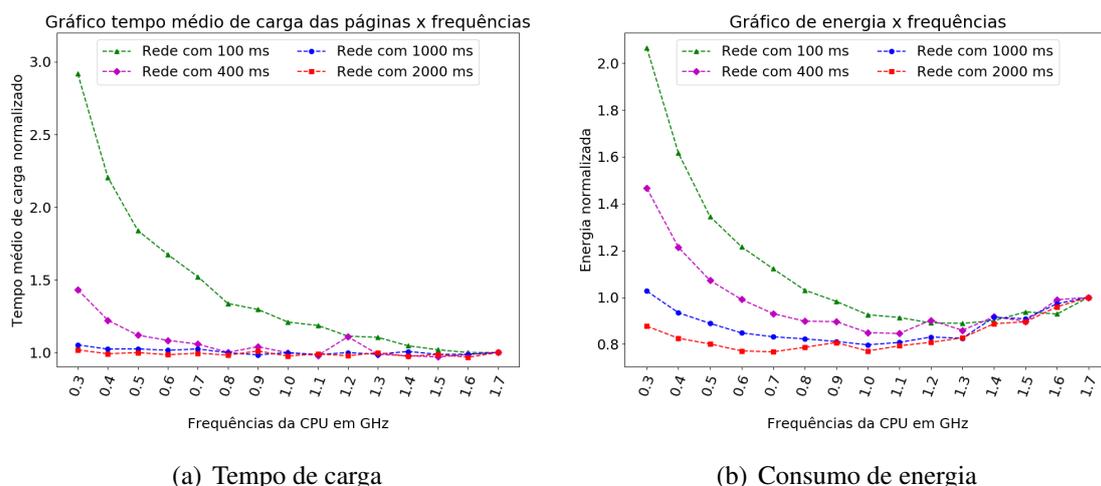
**Figura 4. Tempo de carga de páginas Web em velocidades de rede diferentes**

tomada e a fonte de energia do dispositivo cliente. Uma interface USB é usada para coletar os dados do medidor, e para isso usamos uma máquina externa de forma a permitir a leitura dos dados de potência sem interferir com a coleta dos dados de desempenho no dispositivo cliente.

**Controle do atraso da rede de comunicação.** Para realizar experimentos com diferentes atrasos da rede de comunicação, utilizamos a ferramenta `tc` [13] para alterar as configurações do controle de tráfego no *kernel* do Linux. Isso permite injetar uma latência na rede de comunicação entre o cliente e o servidor, possibilitando caracterizar cenários envolvendo redes mais lentas ou rápidas. Assim, os pacotes que trafegaram entre o servidor e a máquina cliente sofreram um retardo no tráfego até o destino. Através deste mecanismo, o ambiente tornou-se o mais próximo possível de situações reais, encontradas pelo usuário no dia a dia, quando utiliza-se tanto redes mais velozes como *Wi-Fi* ou *LTE/4G*, ou redes mais lentas como *2G* ou *3G*. Os atrasos de rede utilizadas entre a máquina cliente e o servidor foram de 100 ms, 400 ms, 1000 ms e 2000 ms.

### 3.2. Efeito da velocidade da rede na navegação Web

A Figura 4 ilustra a relação entre a velocidade da rede e tempo de carga das páginas quando o processador está na frequência máxima. Observa-se que a velocidade da rede de comunicação experimentada pelo usuário durante a navegação Web podem variar significativamente. Muitas vezes o usuário navega na Web utilizando *Wi-fi*, cuja velocidade é rápida o suficiente para carregar os dados dos servidores remotos, e outras vezes o usuário está navegando em uma rede de comunicação mais lenta (e.g., *3G* ou *3G adversa*) cuja velocidade tende a carregar as páginas mais lentamente; o gargalo aqui estaria na rede de comunicação e não no processador. A fim de melhorar a eficiência energética durante a navegação Web, os processadores com DVFS surgem como uma oportunidade para explorar características da rede e permitir utilizar o processador em frequências diferenciadas para melhor balancear o consumo de energia e o tempo de carregar as páginas Web [20].



**Figura 5. Impacto da variação da frequência no tempo de carga das páginas e no consumo de energia, normalizado pela frequência máxima (1.7 GHz)**

### 3.3. Efeito da variação de frequência da CPU

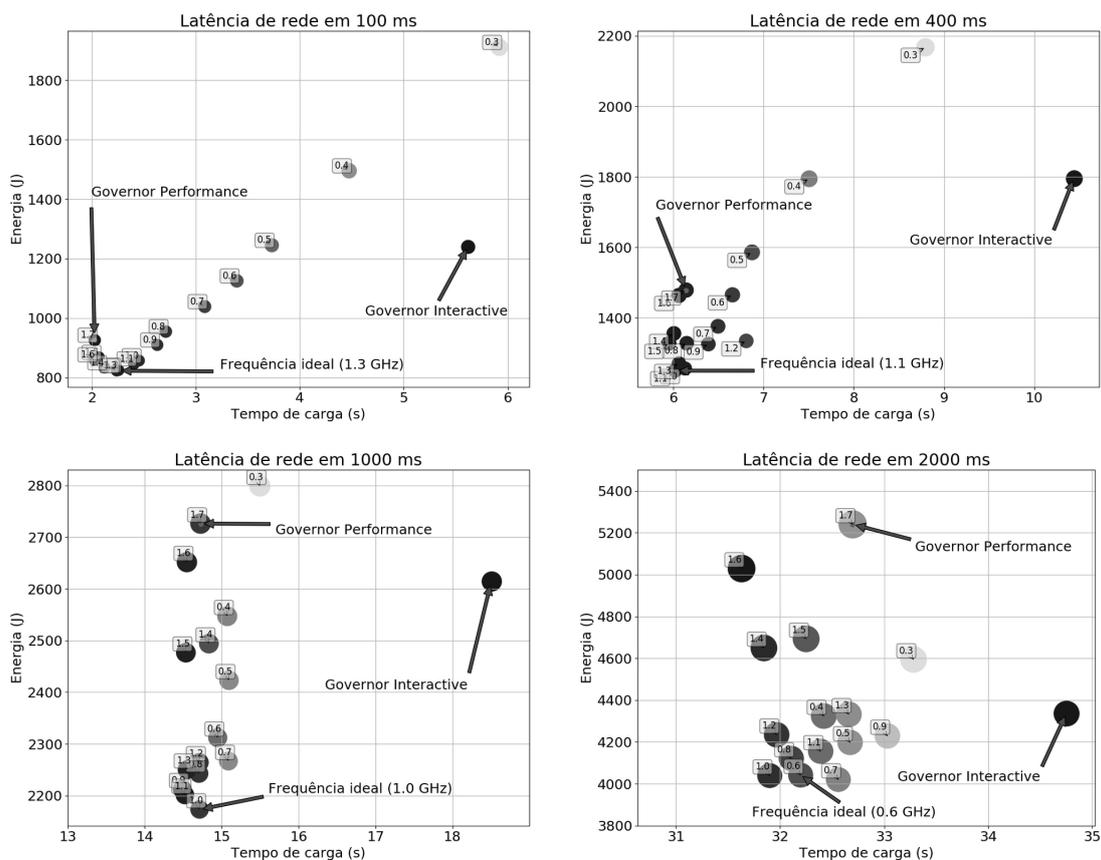
A Figura 5 apresenta um experimento variando as configurações de frequência da CPU de forma a avaliar o comportamento da aplicação em relação ao tempo de carga e consumo de energia nas diferentes opções de frequência e em diferentes atrasos na comunicação entre o cliente e o servidor. Cada página foi executada por 10 vezes e a média das execuções foi reportada em cada uma das frequências da máquina cliente. Esse número de vezes foi suficiente para garantir que os experimentos sejam reproduzíveis com baixa variância entre uma execução e outra.

Os resultados da Figura 5 corroboram com a intuição de que quando a aplicação utiliza uma rede rápida (100 ms ou 400 ms), não é vantajoso usar as frequências mais lentas do processador porque fazer isso acaba demandando excessivo tempo para a carga total da página *Web*. Portanto a eficiência energética fica prejudicada já que quanto mais demora-se para carregar os dados, maior é a energia gasta devido a atividade prolongada da CPU, mesmo usando uma frequência que consome menos potência. Essa situação se deve ao fato de, nessas condições, haver um impacto maior do tempo de carga no processamento pela CPU do que da rede de comunicação.

Observa-se ainda que em uma rede lenta (1000 ms ou 2000 ms) há pouca variação do tempo médio de carga quando comparamos frequências menores e frequências maiores; por esse motivo não há vantagem em usar uma frequência altas como 1.3-1.7 GHz. Ou seja, mesmo que se use frequências rápidas do processador, os dados não chegarão mais rápido ao usuário porque o gargalo está na rede de comunicação. Apesar de ser uma observação simples e intuitiva, identificamos que os gerenciadores de DVFS atuais no Linux não conseguem explorar tal aspecto, deixando de lado oportunidades de melhorias em eficiência energética para esse tipo de aplicação.

### 3.4. Comportamento dos gerenciadores DVFS implementados no Linux

Neste experimento, avaliamos os seguintes gerenciadores do Linux: *Interactive*, cujo foco está em balancear desempenho com economia de energia, e o *Performance* cujo objetivo é maximar o desempenho; isto é, reduzir ao máximo o tempo total de carga das páginas

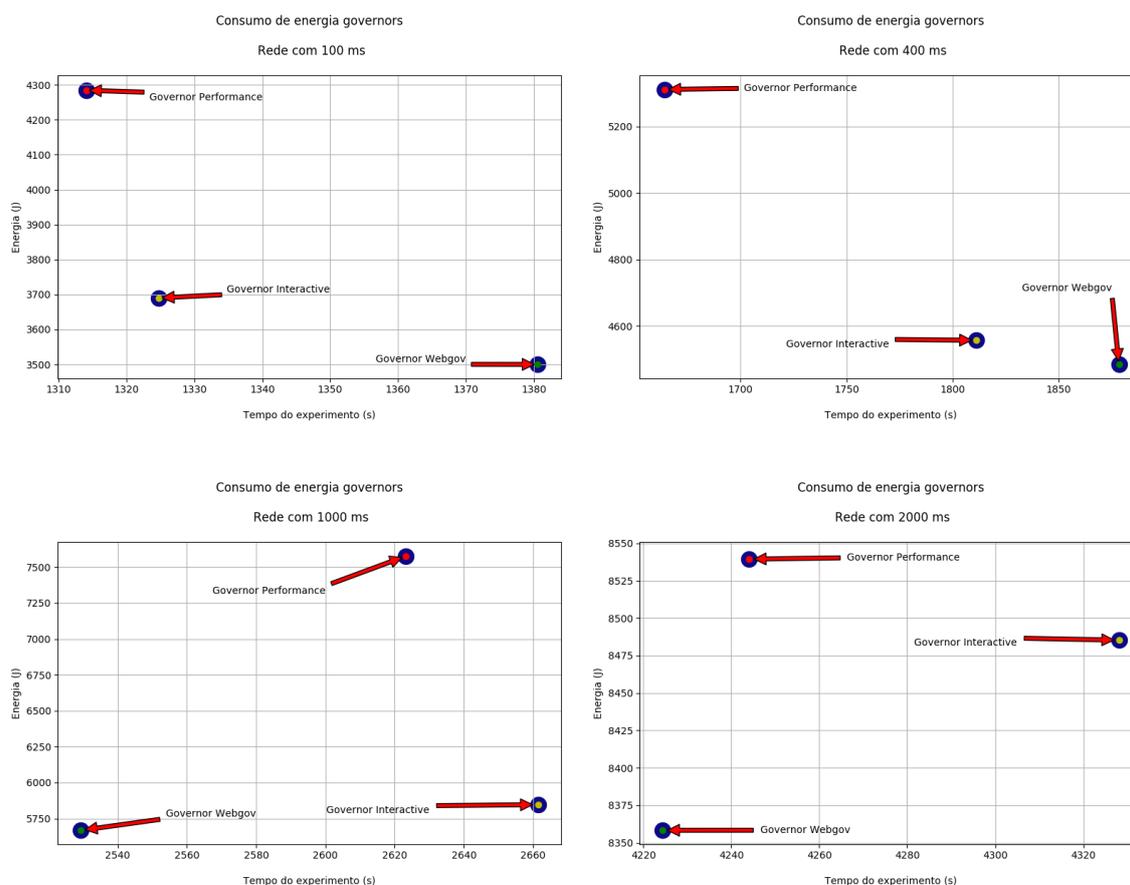


**Figura 6. Consumo de energia e tempo médio de carga das páginas com atrasos na rede de 100 ms, 400 ms, 1000 ms, e 2000 ms. O gráfico contrasta o comportamento das políticas *Interactive* e *Performance*, com a frequência ideal escolhida em cada cenário**

usando a frequência máxima do processador, que em alguns casos pode ser uma decisão energeticamente eficiente.

O resultado do comportamento das políticas *Interactive* e *Performance* é apresentado na Figura 6, em contraste com as outras opções de frequência. Os gráficos relacionam o consumo de energia com o tempo médio requerido para a carga total dos dados das páginas *Web* considerando os diferentes atrasos de rede (100 ms, 400 ms, 1000 ms e 2000 ms). A partir dos resultados desses experimentos, é possível observar que existe uma configuração de frequência ideal que deveria ser usada pelo processador em função do atraso da rede de comunicação. A frequência ideal é aquela que possui menor consumo de energia até um limite de degradação (máximo 20%) em tempo de execução em relação ao *Performance Governor*. Note que a configuração mais energeticamente eficiente é diferente tanto na política *Performance* quanto na política *Interactive*.

Observe que as frequências apontadas como ideais nos gráficos da Figura 6 demonstram que o atraso da rede de comunicação traz uma informação adicional que pode ser usada como indicador para melhor escolha de uma determinada frequência do processador. A frequência ideal identificada através dos experimentos considerando os diferentes atrasos na rede, teve um desempenho médio melhor comparado ao gasto de energia



**Figura 7. Comportamento das políticas *Interactive Governor*, *Performance Governor*, e *Web Governor*, em diferentes cenários de velocidades da rede de comunicação**

da política *Performance* (17%), aumentando o tempo médio de carga em somente 5%. Os valores percentuais reportados consideram a média de todos os atrasos de rede, considerando 11 páginas distintas. Além disso, é possível observar que a frequência ideal foi uma escolha mais eficiente do que a política *Interactive*, já que conseguiu gastar menos energia (21%) e permitiu carregar as páginas mais rapidamente (32%).

### 3.5. Comportamento do Web Governor

Nesta seção comparamos o comportamento do *Web Governor* em relação ao *Interactive Governor* e o *Performance Governor*. A Figura 7 apresenta os resultados do tempo de carga e consumo de energia quando atuando em diferentes velocidades da rede de comunicação. O valor reportado é a média navegação de 11 páginas distintas; 10 execuções cada uma. Naturalmente, o *Performance Governor*, por manter a CPU na frequência máxima, obtém melhor tempo de carga da página, porém o consumo de potência da CPU é muito maior nesta configuração, o que impacta negativamente no consumo de energia.

Sumarizamos na Figura 8 que o *Web Governor* é capaz de reduzir o consumo de energia do sistema em 12% (média) e 18% (máximo) quando comparado ao governador DVFS do Linux desenvolvido pelo Google/Android, enquanto carrega as páginas mais

### Média economia de energia e tempo de carga das webpages do Web Governor comparado com duas políticas do Linux (Intervalo de confiança 95%)

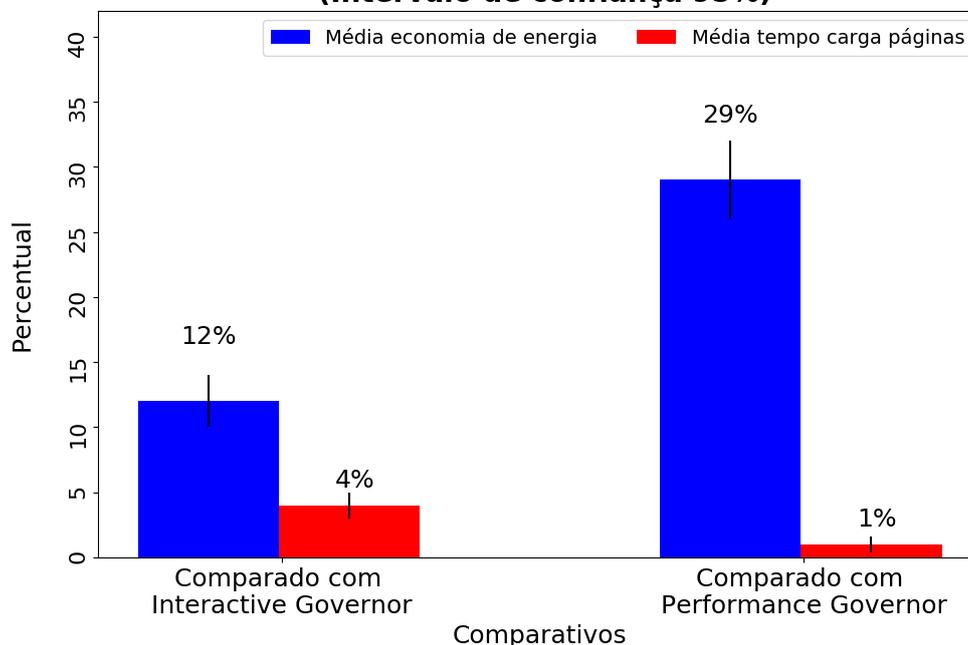


Figura 8. Média economia de energia e tempo de carga das páginas Web com intervalo de confiança de 95%

rapidamente (em média, 4%). Quando comparado a um gerenciador DVFS do Linux focado no desempenho, o *Web Governor* melhora a eficiência energética em 29% em média e até 49%, degradando apenas 1% de carregamento para uma variedade de páginas *Web*. Mesmo em redes com atrasos de 100ms ou 400ms, o *Web Governor* permite reduzir energia, mantendo o aumento do tempo de carregamento das páginas abaixo de 5%.

#### 4. Trabalhos relacionados

No âmbito de eficiência energética, diversas pesquisas têm sido desenvolvidas buscando encontrar uma forma eficaz de economizar energia sem degradar o desempenho. Citamos nesta seção alguns trabalhos relacionados ao tema.

O trabalho mais relacionado ao nosso está descrito em [20]. Nele, foram apresentados experimentos usando *smartphones* de gerações entre 2009 e 2014 para acessar páginas *Web* representando uma evolução em termos de maior requisito computacional. Também foi introduzido atrasos na rede para avaliar o impacto do acesso a página e consumo energético. Diferente do nosso trabalho, eles não identificaram frequências ideais para cada atraso de rede, simplesmente fizeram uma comparação com modo de máximo desempenho (*Performance Governor*) e chaveamento dinâmico de frequência do Linux (*Ondemand Governor*).

Um conjunto de heurísticas de propósito geral para economizar energia em dispositivos móveis foi introduzido por [14]. O objetivo foi detectar desperdício de energia para prolongar a vida útil da bateria. Esse trabalho utiliza técnicas de reconhecimento de

padrões e *crowdsourcing* para identificar instâncias de desperdício de energia no mundo real e também para recomendar soluções de remediações que sejam de interesse dos usuários. Os autores também visam apresentar aos desenvolvedores de aplicativos móveis orientações para identificar/corrigir erros relacionados ao desempenho energético.

No trabalho apresentado em [4] os autores propõem uma técnica de combinação do DVFS com o DPM (*Dynamic Power Management* - Gerenciamento Dinâmico de Potência). Eles propuseram um escalonador não preemptivo disparado por tempo otimizado para as tarefas individualmente e uma atribuição de frequência ideal para cada tarefa minimizar o consumo de energia. Eles usam programação linear e utilizam *benchmarks* para demonstrar a eficácia dos experimentos comparado às abordagens existentes.

Os benefícios de usar um sistema assimétrico com núcleos *big.LITTLE* foi demonstrado em [21], buscando encontrar uma relação ideal entre a alta performance e eficiência energética na abertura de algumas páginas *Web*. No artigo, os autores citam o fato de falhas na carga de páginas *Web* poderem ter como consequência o abandono do site pelo usuário ou grande perda de lucro nos casos de sites de *e-commerce*. O objetivo desse trabalho é demonstrar os benefícios dos processadores heterogêneos para processamento de páginas *Web*.

Os autores em [22] pesquisaram áreas estratégicas relacionadas à economia de energia usando três tipos de mecanismos que permitem o gerenciamento da CPU com relação ao consumo de potência, sendo eles: o DVFS em conjunto com o DPM, o gerenciamento térmico e por fim, os processadores assimétricos. Nesse trabalho, é citado que os resultados em questões relacionadas a energia, estão se tornando uma tendência comum em ambientes que vão desde grandes parques de servidores, que buscam minimizar os custos de resfriamento, até dispositivos móveis que precisam prolongar a vida útil da bateria.

O potencial dos processadores assimétricos em economizar energia foi apresentado em [6] através de implementação de estratégias para escalonamento ciente da assimetria do processador usando algoritmos que podem ser aplicados em sistemas reais e usados nos processadores heterogêneos ou assimétricos, sem sobrecarga significativa. O gerenciamento dinâmico de energia foi desenvolvido para reconfigurar dinamicamente o sistema e assim fornecer os serviços solicitados e os níveis de desempenho com um número mínimo de componentes ativos ou uma carga mínima em tais componentes [3]. No entanto, tais abordagens não levam em consideração a rede de comunicação.

Embora os trabalhos citados sejam boas referências para a área de economia de energia eles não levam em consideração a rede de comunicação. Isso motiva a averiguar possibilidades ainda não investigadas por outros pesquisadores. A latência da rede de comunicação é usada neste trabalho como informação adicional para identificação da frequência ideal da CPU durante a navegação *Web* e assim economizar energia.

## 5. Conclusão

Neste trabalho apresentamos o *Web Governor*, que permite decidir a melhor configuração de velocidade dos processadores na execução de uma aplicação de navegação *Web*. Observamos que há uma relação a ser melhor explorada em relação à rede de comunicação, o tempo gasto para carregar uma página *Web*, e a frequência que deve ser usada pelo

processador. Percebe-se ainda que é possível identificar uma frequência mais otimizada usando a informação da rede. Apresentamos um ambiente experimental com objetivo de avaliar situações encontradas no dia a dia do usuário final, com experiência de atrasos de rede durante a navegação *Web*. Resultados experimentais, considerando diferentes páginas *Web* e cenários de atraso da rede de comunicação, mostram que o *Web Governor* permite reduzir o consumo de energia com garantias de desempenho quando comparado com diferentes políticas de gerenciamento do Linux: *Performance (Linux)* e *Interactive (Google/Android)*.

#### AGRADECIMENTOS

Os autores agradecem aos revisores anônimos pelas sugestões de melhoria do trabalho e à *NVIDIA Corporation* pelo apoio e doação da placa *Jetson TX1* [1] usada nos experimentos desta pesquisa.

#### Referências

- [1] Nvidia® jetson™ - tx1 system-on-module, author = NVIDIA, howpublished = <http://images.nvidia.com/content/tegra/embedded-systems/pdf/JTX1-Module-Product-sheet.pdf> note = Acessado: 2017-07-03, year = 2016,.
- [2] Luiz André Barroso and Urs Hölzle. The case for energy-proportional. 40:33–37, 2007.
- [3] L. Benini, A. Bogliolo, and G. De Micheli. A survey of design techniques for system-level dynamic power management. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 8(3):299–316, June 2000.
- [4] Gang Chen, Kai Huang, and Alois Knoll. Energy optimization for real-time multiprocessor system-on-chip with optimal dvfs and dpm combination. *ACM Trans. Embed. Comput. Syst.*, 13(3s):111:1–111:21, March 2014.
- [5] Stijn Eyerman and Lieven Eeckhout. Fine-grained dvfs using on-chip regulators. *ACM Trans. Archit. Code Optim.*, 8(1):1:1–1:24, February 2011.
- [6] Alexandra Fedorova, Juan Carlos Saez, Daniel Shelepov, and Manuel Prieto. Maximizing power efficiency with asymmetric multicore systems. *Queue*, 7(10):30:30–30:45, November 2009.
- [7] Simson L. Garfinkel. TCPFLOW - TCP/IP packet demultiplexer. <https://github.com/simsong/tcpflow>, 2017. Acessado: 2017-12-10.
- [8] Ilya Grigorik. *High Performance Browser Networking*. O'Reilly Media, 2013.
- [9] Yan Gu, Samarjit Chakraborty, and Wei Tsang Ooi. Games are up for dvfs. In *Proceedings of the 43rd Annual Design Automation Conference, DAC '06*, pages 598–603, New York, NY, USA, 2006. ACM.
- [10] Anthony Gutierrez, Ronald G. Dreslinski, Thomas F. Wenisch, Ali Saidi Trevor Mudge, Chris Emmons, and Nigel Paver. Bbench version 3.0. <http://bbench.eecs.umich.edu/>, 2013. Acessado: 2016-04-11.
- [11] Ariya Hidayat. Phantomjs - WebKit scriptable with a JavaScript API. <http://phantomjs.org/>, 2010. Acessado: 2016-04-15.

- [12] Jason M. Hirst, Jonathan R. Miller, Brent A. Kaplan, and Derek D. Reed. Watts up? pro ac power meter for automated energy recording. *Behavior Analysis in Practice*, 6(1):82–95, Jun 2013.
- [13] Alexey N. Kuznetsov. tc - traffic control. <https://linux.die.net/man/8/tc>, 2017. Acessado: 2017-12-10.
- [14] Guangtai Liang and Shaochun Li. An energy-saving framework for mobile devices based on crowdsourcing intelligences. In *Proceedings of the 3rd International Workshop on Mobile Development Lifecycle*, MobileDeLi 2015, pages 5–6, New York, NY, USA, 2015. ACM.
- [15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [16] Gustavo Pinto and Fernando Castor. Energy efficiency: A new concern for application software developers. *Commun. ACM*, 60(12):68–75, November 2017.
- [17] Barry Rountree, David K. Lownenthal, Bronis R. de Supinski, Martin Schulz, Vincent W. Freeh, and Tyler Bletsch. Adagio: Making dvs practical for complex hpc applications. In *Proceedings of the 23rd International Conference on Supercomputing*, ICS '09, pages 460–469, New York, NY, USA, 2009. ACM.
- [18] Wanghong Yuan and Klara Nahrstedt. Energy-efficient soft real-time cpu scheduling for mobile multimedia systems. In *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, SOSP '03, pages 149–163, New York, NY, USA, 2003. ACM.
- [19] Y. Zhu and V. J. Reddi. Webcore: Architectural support for mobile web browsing. In *2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, pages 541–552, June 2014.
- [20] Yuhao Zhu, Matthew Halpern, and Vijay Janapa Reddi. The role of the CPU in energy-efficient mobile web browsing. *IEEE Micro*, 35(1):26–33, 2015.
- [21] Yuhao Zhu and Vijay Janapa Reddi. High-performance and energy-efficient mobile web browsing on big/little systems. In *Proceedings of the 2013 IEEE 19th International Symposium on High Performance Computer Architecture (HPCA)*, HPCA '13, pages 13–24, Washington, DC, USA, 2013. IEEE Computer Society.
- [22] Zhuravlev, Sergey, Saez, Juan Carlos, Blagodurov, Sergey, Fedorova, Alexandra, and Manuel Prieto. Survey of scheduling techniques for addressing shared resources in multicore processors. *ACM Comput. Surv.*, 45(1):4:1–4:28, December 2012.
- [23] S. Zhuravlev, J. C. Saez, S. Blagodurov, A. Fedorova, and M. Prieto. Survey of energy-cognizant scheduling techniques. *IEEE Transactions on Parallel and Distributed Systems*, 24(7):1447–1464, July 2013.