Análise da Eficiência Energética de uma Aplicação HPC de Geofísica em um Cluster de Baixo Consumo

Jean Luca Bez[1], Eliezer E. Bernart[1], Fernando F. dos Santos[1], Lucas Mello Schnorr[1], Philippe O. A. Navaux[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

{jlbez,eebernart,ffsantos,schnorr,navaux}@inf.ufrgs.br

Abstract. *In this paper, we aimed at analyzing the feasibility and energy efficiency when using an unconventional cluster of ARM processors to execute a scientific application. For this purpose, we used Ondes3D to simulate geophysical events. We present a comparison using different compilation flags and distinct values for the processors frequency. It was possible to observe gains of up to* 54.24% *in Time-to-Solution and* 53.24% *in Energy-to-Solution when compared to executing the application without any prior optimization. This suggests that it is possible to exploit this platform through the correct cluster configuration to get a good balance between performance and energy efficiency.*

Resumo. *Neste artigo buscamos analisar a viabilidade e a eficiência energética ao utilizar um cluster não convencional de processadores ARM para executar uma aplicação científica real. Utilizamos, para este fim, o Ondes3D que permite simular eventos geofísicos. Apresentamos um comparativo entre a execução da aplicação empregando diferentes flags de compilação e diferentes valores para a frequência do processador. Os resultados demonstraram ganhos de até* 54.24% *no tempo para a solução e* 53.24% *na energia para a solução, quando comparado com a execução sem nenhuma otimização. Estes resultados apontam que é possível explorar tal plataforma através da correta configuração do ambiente para se obter um bom equilíbrio entre desempenho e custo energético.*

1. Introdução

A relação entre desempenho e consumo de energia em HPC (*High Performance Computing*) é um tema que vem sendo amplamente discutido. Isso acontece devido a busca de um modelo ou de alternativas que permitam a construção de *clusters* e supercomputadores que apresentem, ao mesmo tempo, grande poder de processamento e o uso moderado de recursos energéticos.

Pesquisas descrevem a eficiência energética na utilização de processadores de baixo consumo quando comparados com arquiteturas tradicionais no processamento de *benchmarks* conhecidos [Padoin et al. 2012b, Padoin et al. 2012a, Padoin et al. 2014]. Nesse contexto, processadores baseados em diferentes versões da arquitetura ARM vêm

Este projeto foi parcialmente financiado pela CAPES, CNPq e Microsoft e desenvolvido no âmbito do Laboratório LICIA e do projeto HPC4E.

sendo cada vez mais utilizados como base para a construção de *clusters* de alto desempenho. Esse crescimento é motivado principalmente pelo custo acessível e pelo baixo consumo de energia apresentado por esta família de processadores.

Neste trabalho buscamos relacionar o custo energético com o tempo de execução de uma aplicação de simulação geofísica chamada *Ondes3D* [Dupros et al. 2008]. Para isso foi utilizado um *cluster* construído com processadores ARM Allwinner A20 [Allwinner 2013] de baixo consumo, presente na plataforma de desenvolvimento Cubietruck [Cubieboard 2015].

Abordamos o impacto da utilização de diferentes quantidades de nós para a simulação, combinando diferentes diretivas de compilação e definindo diferentes frequências de operação para o processador. A execução do *Ondes3D* demanda grande quantidade de processamento e troca de dados, o que permite uma análise mais ampla e próxima a uma situação real, se comparada aos conjuntos de testes tradicionais com *benchmarks*.

O artigo está organizado da seguinte forma: a Seção 2 apresenta trabalhos relacionados a utilização de processadores de baixo consumo para processamento de alto desempenho. Nas seções 3 e 4 a arquitetura ARM utilizada e a aplicação para simulação de modelos geofísicos são brevemente descritas. As métricas de avaliação e os experimentos são abordados na Seção 5. O ambiente e os resultados são apresentados e analisados na Seção 6. Por fim, a Seção 7 conclui este trabalho e aponta perspectivas de trabalhos futuros.

2. Trabalhos Relacionados

A busca por alternativas, em HPC, que obtenham uma melhor relação entre consumo de energia e desempenho vem sendo tema central de diversas pesquisas. Estas abordam a construção de *clusters* não-convencionais baseados em arquiteturas ARM, comparativos entre a execução de *benchmarks* em diferentes plataformas ARMs e a otimização de aplicações para estas arquiteturas de baixo consumo.

[Ou et al. 2012] analisam *clusters* de processadores ARM (Cortex A9) e *clusters* de processadores Intel x86 (Intel Core2-Q9400) quanto a eficiência energética e custo, através de três aplicações com diferentes características. Os resultados demonstram que é possível empregar ARMs na construção de *data centers*, e que estes apresentam mais vantagens para aplicações com baixo custo computacional. No entanto, aplicações computacionalmente intensivas apresentaram menos vantagens pois o número de processadores ARM necessários para prover semelhante desempenho é maior.

Restrições no consumo de energia são impostas para se atingir o *exascale* [Reed and Dongarra 2015]. Alternativas não convencionais, como o uso de processadores ARM, estão sendo estudadas para determinar sua aplicabilidade em aplicações HPC. [Padoin et al. 2012b] avaliam três métricas: *Time-to-Solution*, *Peak-Power* e *Energy-to-Solution*, na perspectiva do usuário, a partir das quais conclui que o emprego de processadores de baixo consumo ainda é questionável para aplicações computacionalmente intensivas.

No mesmo sentido, [Padoin et al. 2014] estudam o uso de processadores de baixo consumo buscando alternativas que não reduzam o desempenho computacional. Através

dos experimentos, onde são comparados 6 diferentes processadores ARM, fica demonstrado que ajustes na compilação das aplicações para melhor explorar essa arquitetura permitem obter o melhor desempenho oferecido pelos MPSoCs baseados em ARM.

Em [Laurenzano et al. 2014] é apresentada uma análise sobre o impacto dos processadores ARM em *benchmarks* computacionais de diferentes tipos. Os resultados indicam que a eficiência energética de sistemas com essa arquitetura pode variar de acordo com as características computacionais e com as características de memória da aplicação. Esta variabilidade pode ser atribuída a dois subsistemas do processador: a unidade de ponto flutuante/SIMD e a interação com o subsistema de memória.

[Göddeke et al. 2014] avaliam a eficiência de processadores de baixo consumo, utilizando a arquitetura ARM v7-A na resolução de equações diferenciais parciais, presentes em diferentes problemas da física, matemática e engenharia. Nas aplicações utilizadas algumas considerações foram tomadas, como a validação da escalabilidade. Os resultados obtidos, ao medir a escalabilidade fraca e forte das aplicações, observou-se uma superioridade do *cluster* de ARM em eficiência energética por um fator de até $2.5\times$ sobre o *cluster* de processadores x86. Ao final concluiu-se que são possíveis reduções substanciais no consumo de energia, com moderadas quedas de desempenho.

Discussão e motivação deste trabalho

Avaliar a viabilidade de execução de uma aplicação científica real com alto custo de processamento em *clusters* não convencionais, permite a obtenção de informações sobre o desempenho do processador quando estão presentes combinações de diferentes características, comumente avaliadas individualmente em *benchmarks* científicos conhecidos.

Como forma de avaliar os resultados obtidos foram utilizadas as métricas apresentadas em trabalhos anteriores: *Time-to-Solution* e *Energy-to-Solution*. Além disso, foram utilizadas diretivas de compilação visando a melhoria no desempenho da aplicação para a plataforma de destino, sendo que as configurações de frequência de operação do processador também foram modificadas buscando uma combinação ideal para o *cluster* e a aplicação em uso.

3. Processadores ARM

Os processadores baseados nas arquiteturas ARM vêm gradativamente ganhando popularidade, principalmente devido ao baixo consumo de energia aliado ao poder computacional oferecido. Essa relação resulta em uma ótima eficiência energética, tornando processadores ARM uma alternativa para diferentes segmentos de mercado, como o setor de dispositivos móveis e de automação.

A família de arquiteturas ARM tem como principais características a similaridade com arquiteturas RISC (*Reduced Instruction Set Computer*), suporte à OOO (*Out-of-Order Exectution*), e em suas versões mais recentes (ARMv7 e ARMv8) apresenta uma unidade vetorial SIMD (*Single Instruction, Multiple Data*) chamada NEON [ARM 2015]. Na arquitetura ARMv8, a unidade NEON oferece suporte ao cálculo de valores com ponto flutuante de precisão dupla, funcionalidade não presente em especificações anteriores que permitem apenas o cálculo de números inteiros [Cebrián et al. 2014].

Neste artigo utilizamos um processador da família ARM Cortex™-A7, desenvolvido pela Allwinner [Allwinner 2015], denominado ARM A20. Algumas das carac-

terísticas presentes nesta família de processadores de baixo consumo são: suporte a até quatro *cores*, com *cache* L1 de dados e instruções variando de 8 KB a 64 KB; *cache* L2 de tamanhos variáveis de 128 a 1024KB; suporte a SIMD (*Single Instruction Multiple Data*) com o coprocessador NEON e suporte a ponto flutuante com a arquitetura VFPv4 da ARM [Cortex-A7 2013]. A Figura 1 apresenta a organização arquitetural de processadores da família ARM Cortex™-A7.

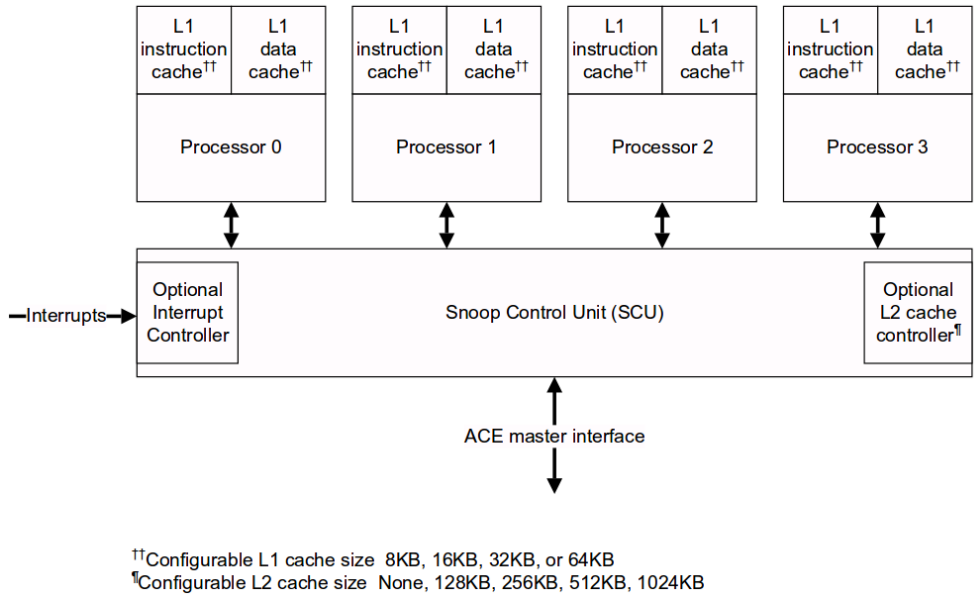


Figura 1. Diagrama de blocos da família ARMv7 [Cortex-A7 2013].

A especificação do processador ARM utilizado neste trabalho – SoC (*System on Chip*) A20 – possui dois *cores*, com 32 KB de cache de instruções e 32 KB para dados. Sendo que este possui suporte a memória DDR3, DDR3L e DDR2. O processamento gráfico é feito por uma GPU Mali400 MP2 integrada com suporte a OpenGL 2.0 [Allwinner 2013].

Esse processador possui frequência de operação de 1.01 GHz, com suporte a DVFS (*Dynamic Voltage and Frequency Scaling*), o que permite ao processador aumentar ou diminuir a frequência de operação de acordo com a necessidade ou com as configurações definidas pelo usuário. Esta funcionalidade pode acarretar em economia de energia, quando não há processamento, e maior desempenho, quando há processamento útil a ser feito [Allwinner 2013].

4. Aplicação de Simulação Geofísica *Ondes3D*

Ondes3D é um simulador de ondas sísmicas utilizado para estimar danos futuros ocasionados por terremotos [Dupros et al. 2010]. Detalhes sobre o modelo e sua implementação em C podem ser conferidos no artigo [Dupros et al. 2010]. Esta aplicação simula modelos de propagação de ondas sísmicas em um espaço tridimensional, através de equações de elastodinâmica, utilizando o método de diferenças finitas explícitas [Appel and Petersson 2009].

Na paralelização, a simulação de ondas sísmicas é particionada utilizando decomposição cartesiana em MPI. Cada processador soluciona parte do subproblema,

atualizando o campo da onda dentro da sua porção da grade e trocando informações com os vizinhos relativo as bordas [Dupros et al. 2008]. Porém, ao simular uma região de espaço limitada, é necessário tratar condições de borda, para absorver a energia que sai desta região simulada [Tesser et al. 2014].

O tamanho do problema simulado é determinado pela quantidade de *timesteps* e pelo tamanho da malha $x \times y \times z$. Os testes conduzidos neste artigo consideraram 100 *timesteps*, cada um representando 0.008 segundos na simulação. A malha utilizada possui dimensões $200 \times 200 \times 200$.

5. Metodologia de Avaliação Utilizando *Time-to-Solution* e *Energy-to-Solution*

As métricas utilizadas na avaliação foram o tempo de execução (*Time-to-Solution*) e o consumo de energia para a solução (*Energy-to-Solution*). Nos testes cujo objetivo era medir o tempo médio de execução e o consumo de energia, a simulação geofísica foi executada com 4, 8 e 16 *cores*. A distribuição de tarefas entre os *cores* foi a ideal em relação a alocação de recursos, isto é, tarefas são distribuídas visando ocupar os dois *cores* de cada nó. A equação 1 permite obter a quantidade de energia gasta no processamento (*Energy-to-solution*). Nela, *P* é a potência consumida no instante t, e Δt é de $1s$ [Keller 2012].

$$E = \int_{t_{inicio}}^{t_{final}} P(t)dt \approx \sum_{t_{inicio}}^{t_{final}} P(t)\Delta t \quad (1)$$

Cada experimento foi repetido 10 vezes, diminuindo a possibilidade que erros de medição mudem a interpretação dos resultados. Assumindo uma distribuição padrão (*t-student*), a probabilidade que a média verdadeira esteja entre os limites inferiores e superiores do intervalo confiança é de 95%. Estes limites são equivalentes a duas vezes o desvio padrão dividido pela raiz quadrada do número de medições.

Utilização de *flags* de compilação

Considerando os resultados obtidos em [Padoin et al. 2012a], também foram realizados testes utilizando a melhor combinação de diretivas de compilação apontada pelos autores por obterem os melhores resultados: *flag* de otimização (`-O3`) e as *flags* específicas para a plataforma ARM (`-march=armv7-a -mtune=cortex-a7 -mfpu=neon -ftree-vectorize`).

Variando a frequência do processador

O processador ARM utilizado possui suporte à *Dynamic Voltage and Frequency Scaling* (DVFS) o que possibilita fazer uso da ferramenta CPUFreq [CPUFreq 2015], responsável por aumentar ou diminuir a frequência de operação do SoC. A frequência do processador pode ser configurada manualmente ou utilizar um perfil pré-estabelecido pelo sistema, denominado *governor*. Estes perfis definem as características de energia da CPU, o que afeta diretamente o desempenho. Cada *governor* possui um comportamento e propósito único, e é mais recomendado para determinada carga de trabalho. Neste contexto foram analisados os perfis: *performance* e *powersave*.

O perfil ***performance*** define que o processador irá utilizar uma frequência de operação voltada para o alto desempenho, que no caso do A20 é de 912 MHz. O CPUFreq não utiliza o máximo por medida de segurança definidas na BIOS da plataforma. Por sua vez, o perfil ***powersave*** apresenta comportamento oposto ao *performance*, ou seja, ele irá utilizar a frequência definida no arquivo de configuração de frequências do sistema para o perfil *powersave*, que neste caso é de 720 MHz.

6. Resultados dos Experimentos

Nesta Seção são apresentados a plataforma experimental e os resultados obtidos a partir dos experimentos, considerando as métricas descritas na Seção 5. Também são analisados o impacto das *flags* de compilação nestas métricas, e o impacto dos perfis de frequência de CPU suportados pela plataforma.

6.1. Descrição da Plataforma

Todos os experimentos foram conduzidos no *cluster* Yggdrasil composto por processadores ARM de baixo consumo. A configuração completa está descrita na Tabela 1, incluindo o somatório do total de processadores, *cores*, memória e armazenamento para o *cluster* como um todo.

	Um Nó	**Cluster** ($\times 8$)
Fabricante	Allwinner	
Processador	ARM Cortex-A7 1 GHz	
Processadores	1	8
Núcleos *cores*	2	16
L1 Cache de instruções	32KB (por *core*)	
L1 Cache de dados	32KB (por *core*)	
L2 Cache	256KB (compartilhado)	
Memória	2 GBytes DDR3 480 MHz	16 GBytes
Armazenamento	8 GBytes NAND	64 GBytes
Sistema Operacional	Ubuntu 14.04 LTS ARM	
Kernel	Linux 3.4.61	
GCC	gcc 4.8.3	
MPI	OpenMPI 1.6.5	
Rede	Gigabit Ethernet	

Tabela 1. Configuração de um nó do cluster Yggdrasil, e somatório para o total de oito nós interconectados por um *switch gigabit* dedicado.

6.2. Visão Geral do Desempenho Obtido

A Figura 6.2 mostra o *speedup* obtido com a melhor combinação de diretivas de compilação, conforme será demonstrado na Seção 6.3 a seguir. No gráfico, é possível observar que, aumentando o número de *cores* empregados na simulação do problema, o *speedup* (linha com símbolo $\times$) vai afastando-se do ideal (linha sem marcação). Empregando 2 nós (4 *cores*) o *speedup* foi de $3,90$. Com 4 nós (8 *cores*) foi de $6,80$ e utilizando todos os 8 nós (16 *cores*) foi de $12,37$. A distância em relação ao *speedup* ideal pode ter várias origens: peso da comunicação entre os processos, que é bloqueante; utilização

de dois *cores* por nó, que pode sobrecarregar a interconexão entre os nós e prejudicar os acessos à memória, entre outras. Mesmo assim, observa-se que embora a distância do ideal aumente conforme se aumenta a quantidade de *cores*, o comportamento da melhor versão paralela se mantém linear. Como o objetivo deste trabalho é analisar as métricas *Time-to-Solution* e *Energy-to-Solution*, as próximas seções detalham a relação entre o aumento do número de *cores* e tais métricas.

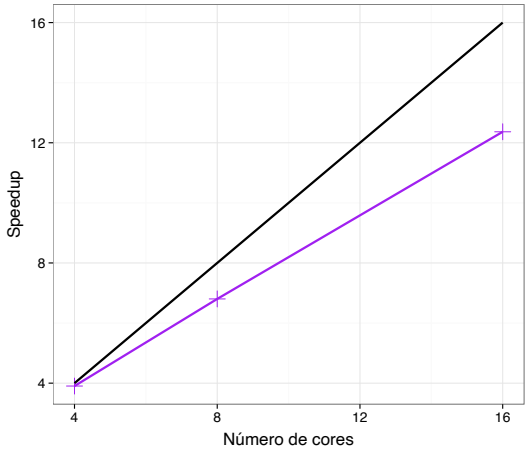


Figura 2. *Speedup* observado nos testes com o uso das diretivas de compilação.

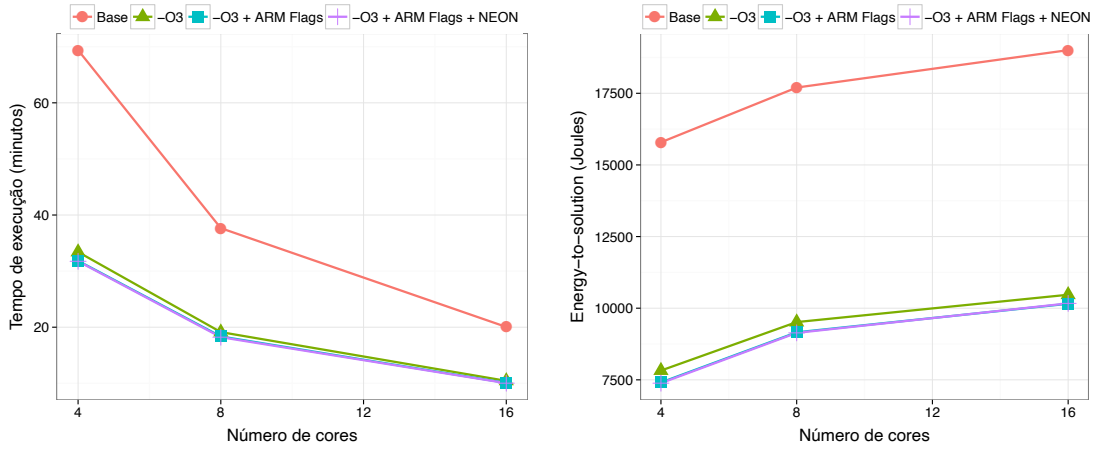
6.3. Análise do Impacto das Diretivas de Compilação

Time-to-Solution

A Figura 3(a) apresenta os tempos de execução médios para as diferentes abordagens. Em abscissa, temos a quantidade de *cores* utilizada nos experimentos, podendo ser 4, 8 ou 16. Em ordenada, temos os tempos médios de execução em minutos. As marcações nas linhas no gráfico representam as diferentes abordagens. O tempo considerado inicial (**Base** – marcado com círculos) é baseado na execução do *Ondes3D* sem nenhuma otimização, enquanto que as outras abordagens representam as diferentes otimizações. A análise desta figura permite observar que o ganho de desempenho com o uso das *flags* de compilação foi de $54,24\%$ ao utilizar 2 nós (4 *cores*), $51,62\%$ com 4 nós (8 *cores*) e $49,99\%$ com 8 nós (16 *cores*) quando considera-se que o tempo de referência é a execução do *Ondes3D* sem nenhuma otimização.

Os resultados apresentados na Figura 3(a), confirmam o observado por [Padoin et al. 2012a] nos testes com diferentes *benchmarks*, que o simples uso da *flag* `-O3` proporciona grandes ganhos de desempenho mesmo ao executar uma aplicação real como o *Ondes3D*. Por outro lado o uso de diretivas específicas para ARM apresentam apenas uma pequena redução do tempo de execução.

O uso da unidade NEON também não apresentou grande diferença no tempo de execução, o que era esperado, visto que a mesma somente possui suporte para operações envolvendo números inteiros nesta versão da arquitetura e, considerando a natureza da aplicação, a maioria das operações envolvem ponto flutuante. Estas pequenas diferenças podem ser observadas nos tempos médios apresentados na Tabela 2.



(a) *Time-to-Solution* (b) *Energy-to-Solution*

Figura 3. Impacto das diferentes *flags* no tempo e energia para solução.

	4 *cores* (minutos)	**8 *cores*** (minutos)	**16 *cores*** (minutos)
Base	69,393	37,658	20,045
`-O3`	33,409	19,127	10,435
`-O3 + ARM Flags`	31,841	18,354	10,074
`-O3 + ARM Flags + NEON`	31,751	18,221	10,024

Tabela 2. *Time-to-Solution* utilizando as diferentes *flags* de compilação.

Energy-to-Solution

A Figura 3(b) apresenta a média da estimativa da métrica *Energy-to-Solution* para as diferentes abordagens, variando-se a quantidade de *cores* nos experimentos. As estimativas são feitas baseando-se em medidas realizadas em apenas um dos nós, e em seguida se extrapolando através da multiplicação pela quantidade de nós de fato utilizados no experimento. Como no gráfico da Figura 3(a), o tempo de base (marcado com círculos) é baseado na execução do *Ondes3D* sem nenhuma otimização. A análise da métrica *Energy-to-Solution* permite verificar qual o consumo total de energia do *cluster* durante a execução da aplicação. A redução de consumo observada foi de $53,24\%$ ao utilizar apenas 2 nós (4 *cores*), $48,37\%$ utilizando 4 nós (8 *cores*) e $46,50\%$ ao utilizar todos os 8 nós (16 *cores*) do ambiente, conforme demonstra a Figura 3(b), utilizando como base a energia gasta pela aplicação sem nenhuma otimização. Quando a utilização da unidade NEON, que permite a realização de operações SIMD em inteiros, foi habilitada durante a compilação do código, os resultados não apresentaram grandes diferenças no quesito consumo de energia. Isto indica que o cálculo de números inteiros, do *Ondes3D*, não apresenta carga significativa para o sistema.

A Tabela 3 apresenta a estimativa de energia gasta para a solução considerando cada um dos experimentos. A melhor relação *Energy-to-Solution* observada em todos os casos foi ao habilitar todas as *flags* (`-O3 -march=armv7-a -mtune=cortex-a7 -mfpu=neon -ftree-vectorize`).

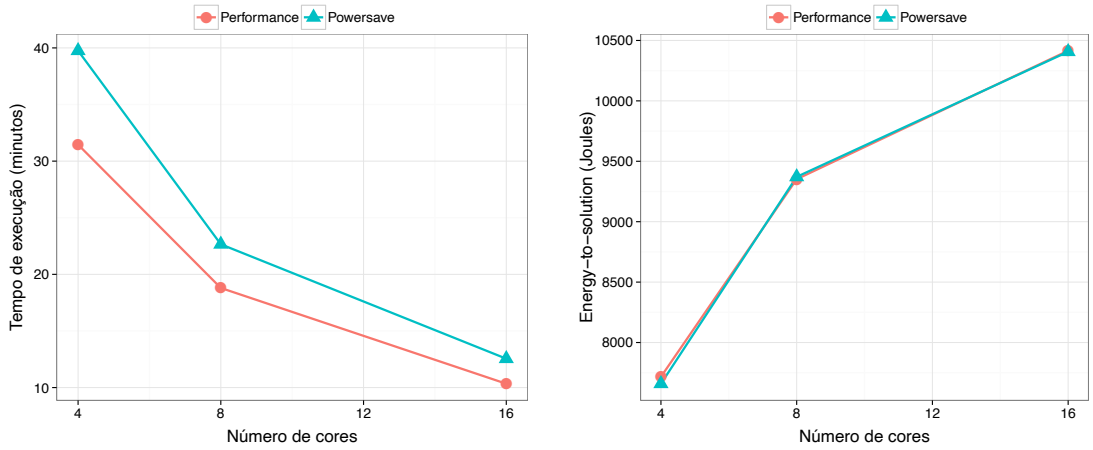
	4 *cores* (Joules)	**8 *cores* (Joules)**	**16 *cores* (Joules)**
Base	15789, 352	17700, 002	19006, 241
`-O3`	7823, 696	9515, 459	10469, 231
`-O3 + ARM Flags`	7408, 087	9159, 194	10154, 453
`-O3 + ARM Flags + NEON`	7383, 493	9138, 861	10168, 053

Tabela 3. *Energy-to-Solution* utilizando as diferentes *flags* de compilação.

6.4. Análise do Impacto dos Perfis de Frequência

Time-to-Solution

A Figura 4(a) apresenta os tempos de execução médios para as diferentes abordagens. Em abscissa, temos a quantidade de *cores* utilizada nos experimentos, podendo ser 4, 8 ou 16. Em ordenada, temos os tempos médios de execução em minutos. As diferentes marcações no gráfico representam os perfis de frequência *powersave* (marcado com triângulos) e *performance* (marcado com círculos). Nesta Figura é possível observar que o tempo de execução (*Time-to-Solution*) foi influenciado negativamente pelo uso do perfil *powersave*. Tendo como base para comparação o perfil *performance*, o tempo de execução aumentou 26, 38% para 4 *cores*, 21, 61% para 8 *cores* e 22, 65% para 16 *cores*.



(a) *Time-to-Solution* (b) *Energy-to-Solution*

Figura 4. Impacto dos diferentes perfis de frequência no tempo e energia para solução.

A Tabela 4 apresenta a diferença, em minutos, para a execução de cada experimento envolvendo 4, 8 e 16 *cores* empregando os perfis de frequência previamente descritos. É possível notar que a diferença na métrica *Time-to-Solution* entre os dois perfis diminui conforme a quantidade de cores aumenta.

Energy-to-Solution

Do ponto de vista energia para a solução (*Energy-to-Solution*) não houve diferença significativa entre as abordagens, conforme ilustra a Figura 4(b). A Tabela 5 suporta esta

	4 *cores* (minutos)	**8 *cores*** (minutos)	**16 *cores*** (minutos)
Performance	$31,458$	$18,798$	$10,339$
Powersave	$39,754$	$22,657$	$12,550$

Tabela 4. *Time-to-Solution* utilizando os diferentes perfis de frequência.

afirmação ao apresentar os resultados dos experimentos envolvendo 4, 8 e 16 *cores* com os perfis de frequência previamente descritos. Os resultados são semelhantes pois há um equilíbrio entre a frequência e o tempo de processamento, isto é, quando a frequência foi reduzida o processador passou mais tempo computando a simulação e quando esta foi aumentada o tempo de processamento diminuiu.

	4 *cores* (Joules)	**8 *cores*** (Joules)	**16 *cores*** (Joules)
Performance	$7716,362$	$9351,849$	$10415,242$
Powersave	$7658,830$	$9371,098$	$10406,959$

Tabela 5. *Energy-to-solution* utilizando os diferentes perfis de frequência.

7. Conclusões e Trabalhos Futuros

7.1. Conclusões

Neste artigo, demonstramos que é viável empregar um *cluster* de baixo consumo para executar aplicações científicas, como o simulador de modelos geofísicos *Ondes3D*. A correta configuração da aplicação durante a compilação, através do uso de *flags* de otimização e *flags* específicas para a plataforma demonstraram uma redução no tempo de execução (*Time-to-Solution*) de até 54.24% com 2 nós (4 *cores*).

Considerando o consumo total de energia para a solução (*Energy-to-Solution*), utilizando as *flags* de compilação específicas para plataformas ARM e *flags* de otimização, observou-se redução de consumo de até 53.24% ao utilizar 2 nós (4 *cores*) do ambiente.

Em relação ao uso dos perfis de frequência (configuração do CPUFreq) os resultados demonstram um aumento de até 26.38%, no tempo de processamento utilizando 4 *cores* ao utilizar o *governor powersave*. Por outro lado, o gasto energético de ambos não teve diferença significativa, ou seja, o processador apresentou um gasto de energia instantâneo menor ao trabalhar com uma frequência menor, porém passou mais tempo executando a tarefa. Sendo assim, é possível concluir que a diminuição da frequência é uma ferramenta útil para quando o processador está em *idle*, sem carga de trabalho significativa, não sendo eficiente para aplicações reais na plataforma testada.

7.2. Trabalhos Futuros

Dando continuidade a este trabalho, uma análise das comunicações pode ser feita, buscando determinar o impacto destas sobre as métricas *Energy-to-Solution* e *Time-to-Solution*, em um *cluster* de ARMs. Pretende-se também analisar outras aplicações, com focos distintos do *Ondes3D*, tendo pelo menos um representante com comportamento *compute-bound* e outro *communication-bound*.

Como o objetivo deste trabalho estava na avaliação de desempenho em um *cluster* de processadores ARM e na escolha de configurações e parâmetros que venham a otimizar seu funcionamento para a aplicação em questão, não abordamos a comparação deste com um *cluster* tipicamente utilizado para tais simulações. Este compativo caracteriza-se como trabalho futuro.

Referências

Allwinner (2013). A20 User Manual. Technical Report 1, Allwinner Technology.

Allwinner (2015). Allwinner Technology Company. `http://www.allwinnertech.com/en/about/company.htm`. Accessed: 2015-05-30.

Appel, D. and Petersson, N. A. (2009). A Stable Finite Difference Method for the Elastic Wave Equation on Complex Geometries with Free Surfaces. *Communications in Computational Physics*, 5(1):84–107.

ARM (2015). ARM Processor Architecture. `http://www.arm.com/products/processors/instruction-set-architectures/index.php`. Accessed: 2015-05-11.

Cebrián, J. M., Natvig, L., and Meyer, J. C. (2014). Performance and energy impact of parallelization and vectorization techniques in modern microprocessors. *Computing*, 96(12):1179–1193.

Cortex-A7 (2013). Cortex-A7 MPCore. Technical Report p5, ARM.

CPUFreq (2015). CPU Frequency Scaling. `https://wiki.archlinux.org/index.php/CPU_frequency_scaling`. Accessed: 2015-07-09.

Cubieboard (2015). Cubieboard Open-Source Main-Boards. `http://docs.cubieboard.org/products/start`. Accessed: 2015-05-30.

Dupros, F., Aochi, H., and Ducellier, A. (2008). Exploiting Intensive Multithreading for the Efficient Simulation of 3D Seismic Wave Propagation. *11th IEEE International Conference on Computational Science and Engineering*, pages 253–260.

Dupros, F., Martina, F. D., Foerster, E., Komatitsch, D., and Romand, J. (2010). High-performance finite-element simulations of seismic wave propagation in three-dimensional nonlinear inelastic geological media. *Parallel Computing - Parallel Matrix Algorithms and Applications*, 36(5–5):308–325.

Göddeke, D., Rajovic, N., Komatitsch, D., Puzovic, N., Geveler, M., and Ramirez, A. (2014). Energy efficiency vs. performance of the numerical solution of pdes: An application study on a low-power arm-based cluster. *Journal of Computational Physics*, 237(12):132–150.

Keller, V. (2012). Energy-to-solution: a today's metric for tomorrows' concerns. `http://www.ig.fpms.ac.be/sites/default/files/FGPS175-Keller.pdf`. Accessed: 2015-07-05.

Laurenzano, M. A., Tiwari, A., Jundt, A., Peraza, J., Jr, W. A. W., Campbell, R., and Carrington, L. (2014). Characterizing the performance-energy tradeoff of small arm cores in hpc computation. In *Euro-Par 2014 Parallel Processing*, pages 124–137. Springer.

Ou, Z., Pang, B., Deng, Y., Nurminen, J. K., Ylä-Jääski, A., and Hui, P. (2012). Energy- and cost-efficiency analysis of arm-based clusters. In *2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, pages 115–123. IEEE/ACM.

Padoin, E. L., de Oliveira, D. A. G., Velho, P., and Navaux, P. O. A. (2012a). Evaluating Performance and Energy on ARM-based Clusters for High Performance Computing. In *41st International Conference on Parallel Processing Workshops Evaluating*, pages 165–172.

Padoin, E. L., de Oliveira, D. A. G., Velho, P., and Navaux, P. O. A. (2012b). Time-to-Solution and Energy-to-Solution: A Comparison Between ARM anx Xeon. In *2012 Third Workshop on Applications for Multi-Core Architecture*, pages 48–53. IEEE.

Padoin, E. L., Velho, P., de Oliveira, D. A. G., Navaux, P. O. A., and Mehaut, J.-F. (2014). Tuning Performance and Energy Consumption of HPC Applications on ARM MP-SoCs. In *5th Workshop on Applications for Multi-Core Architectures*, pages 1–6.

Reed, D. A. and Dongarra, J. (2015). Exascale Computing and Big Data. *Communications of the ACM*, 58(7):56–68.

Tesser, R. K., Pilla, L. L., Dupros, F., Navaux, P. O. A., Mehau, J.-F., and Mendes, C. (2014). Improving the Performance of Seismic Wave Simulations with Dynamic Load Balancing. *Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pages 196–203.