2013 SEGeT SIMPÓSIO DE EXCELÊNCIA EM GESTÃO E TECNOLOGIA
Gestão e Tecnologia para a Competitividade
23.24.25 de Outubro de 2013

Análise de Desempenho de Algoritmos em Ambientes Virtualizados em Cloud Computing

Ricardo Soares Bôaventura
ricardoboaventura@iftm.edu.br
IFTM

Keiji Yamanaka
keiji@ufu.br
UFU

Bruno Queiroz Pinto
bruno.queiroz@iftm.edu.br
IFTM

Adilmar Coelho Dantas
akanehar@gmail.com
IFTM

Resumo: As nuvens de computadores permitem utilizar recursos ociosos (processamento, armazenamento, conectividade, plataforma, aplicações e serviços) via Internet sem a necessidade de se preocupar com as suas localizações físicas independente de sistema operacional e hardware. Esses recursos podem ser comprados por clientes para reforçar a capacidade de processamento em um determinado período de tempo, ou substituir a sua infraestrutura. Contudo, os usuários pagam somente por o que será utilizado (processamento, memória, armazenamento, etc). O presente artigo apresenta os resultados iniciais dos experimentos realizados em um ambiente físico e virtualizado em nuvem de computadores usando a ferramenta OpenQRM que é uma plataforma de gerenciamento e integração de recursos para datacenters e nuvens. Esses experimentos servirão de base para desenvolver um sistema de gerenciamento em nuvem, que auxiliará os clientes a descobrir qual a melhor configuração de um ambiente evitando o desperdício de recursos computacionais e custos mensais.

Palavras Chave: cloud computing - virtualização - planej. experimental - performance - complexidade

1. INTRODUÇÃO

Nos dias atuais a capacidade de processamento dos computadores pessoais vem aumentando substancialmente devido ao desenvolvimento de novos hardwares e às novas técnicas de programação (BARUCHI, 2008). Entretanto, toda a capacidade adquirida não tem sido muito bem aproveitada. Com isso, em boa parte do tempo, ocorre um grande desperdício de recursos computacionais por não serem utilizados de forma correta e desejada.

Baseado nesse problema, a virtualização surgiu com um novo conceito de utilização de máquinas de grande porte, definindo uma técnica que possibilita a divisão de um mesmo hardware em diversas máquinas virtuais (POPEC e GOLDBERG, 1974; SOUZA, 2006). A virtualização foi utilizada nos mainframes e, atualmente, com a evolução de recursos computacionais (memória, armazenamento, entre outros) permitiu também que os servidores de pequeno porte e computadores pessoais tivessem condições de utilizar dessa tecnologia (SOUZA, 2006).

O objetivo do artigo é mostrar os resultados iniciais dos experimentos realizados em um ambiente físico e virtualizado em cloud computing que servirão de base para desenvolver um sistema de gerenciamento que auxiliará os clientes descobrir a melhor configuração de um ambiente virtualizado evitando o desperdício de recursos e custos mensais.

O presente artigo está estruturado da seguinte forma: na Seção 02 será abordado um referencial teórico sobre virtualização, na Seção 03 será apresentado os materiais e métodos. A Secção 4 serão apresentados os resultados dos experimentos, e na Seção 5 é apresentada as considerações finais.

2. VIRTUALIZAÇÃO

A virtualização pode ser entendida como uma técnica que divide os recursos do computador em múltiplos ambientes de execução com o objetivo de implantar novas tecnologias, criando uma camada de abstração entre dispositivos físicos e aplicações (FIGUEIREDO, 2005; NANDA e CHIUEH, 2005; MENASCÉ, 2005).

Com a virtualização pode-se: executar sistemas operacionais simultaneamente em um mesmo hardware; executar o sistema operacional juntamente com as suas aplicações como se fosse um processo de outro sistema operacional; utilizar os sistemas operacionais e aplicações desenvolvidas para um tipo específico de plataforma em outas; e executar múltiplos sistemas operacionais.

Segundo LAUREANO (2006) e IBM (2007), um sistema de computadores é projetado basicamente com três componentes: hardware que tem como objetivo executar as operações enviadas pelas aplicações; sistemas operacionais que são desenvolvidos para aproveitar o máximo os recursos fornecidos pelo hardware; e aplicações que são basicamente produtos que interagem com o usuário fornecendo dados para o processamento (Figura 01a).

A camada de virtualização (Figura 01b), também chamada de hipervisor ou monitor (VMM – Virtual Machine Monitor), molda todas as interfaces virtuais a partir da configuração da plataforma real proporcionando maior flexibilidade operacional e aumentando taxa de utilização do hardware físico (IBM, 2005).

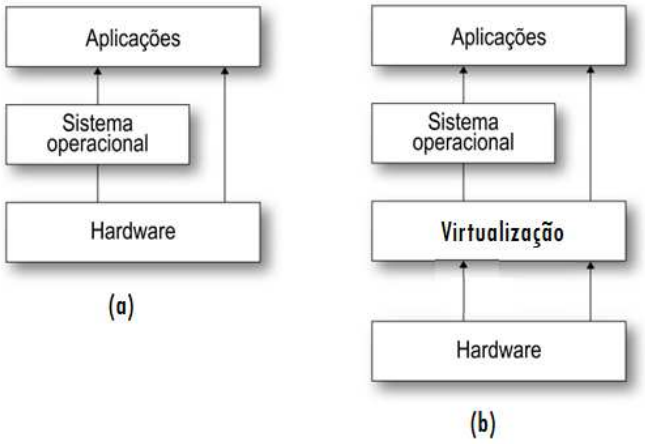

Figura 1. (a) Sistema computacional tradicional. **(b)** Localização da camada de virtualização.

A virtualização pode ser realizada das maneiras: (a) *virtualização de desktop:* é a virtualização em que os sistemas operacionais e as aplicações dos usuários finais (infraestrutura virtual) são executados em um datacenter, sob a forma de uma máquina virtual. Os usuários podem customizar seu ambiente e acessá-lo de diferentes locais, utilizando a rede (IBM, 2007); (b) *virtualização de armazenamento*: esse tipo de virtualização faz com que diversos discos físicos sejam vistos como um conjunto homogêneo de recursos de armazenamento (REGO, 2012); (c) *virtualização de rede*: é a virtualização que permite a criação de ambientes de rede separados para cada grupo de usuários, apesar de compartilhar a mesma rede física. Neste tipo de virtualização é possível criar diversos dispositivos de rede, como switches virtuais (ZHOU, 2010; CHOWDHURY, 2012); e (d) *virtualização de servidores*: tipo de virtualização que permite uma estrutura física seja dividida para executar diversos ambientes virtuais (VERAS, 2009);

2.1. CATEGORIAS DE VIRTUALIZAÇÃO

Segundo ROSENBLUM e GARFINKEL (2005), os softwares que permite realizar a virtualização podem ser classificados em três tipos, porém tem objetivo de oferecer compatibilidade de software, permitir o isolamento de máquinas virtuais e o encapsulamento da camada de virtualização. As três categorias de virtualização são:

1. *nível de hardware*: a camada de virtualização é colocada diretamente sobre a máquina física apresentando as camadas superiores como hardware virtual, igual ao original (Figura 2);

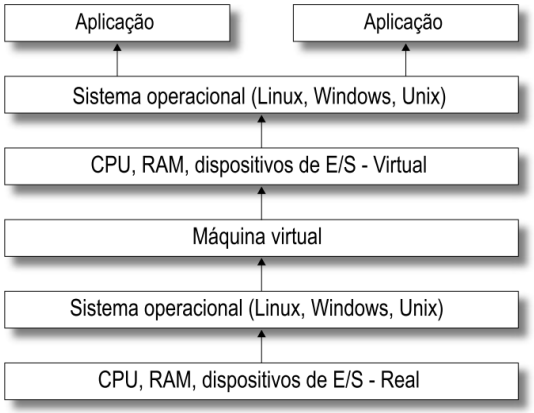

Figura 2: Virtualização no nível de hardware

Fonte: LAUREANO, 2006

2. *nível de sistema operacional*: a camada de virtualização está inserida entre o sistema operacional e a aplicação permitindo que seja criado partições lógicas de forma que cada partição é apresentada de forma isolada compartilhando o mesmo sistema operacional (Figura 3);

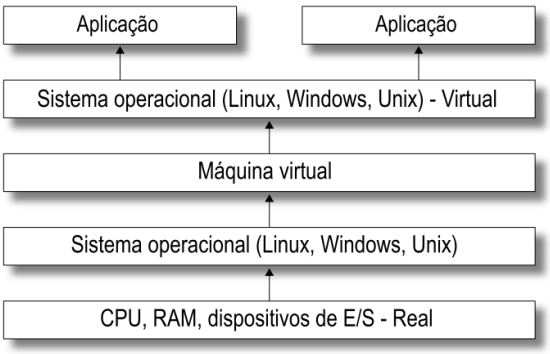

Figura 3: Virtualização no nível de sistema operacional

Fonte: LAUREANO, 2006

3. *nível de linguagem de programação*: a camada de virtualização é um programa de aplicação do sistema operacional (Figura 4);

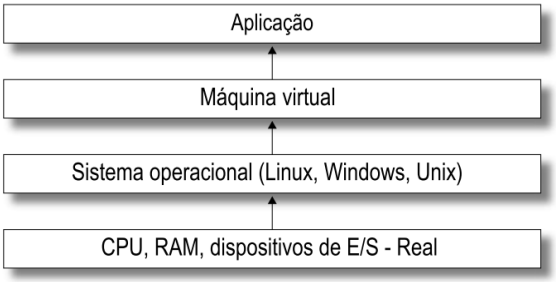

Figura 4: Virtualização no nível de linguagem de programação

Fonte: LAUREANO, 2006

2.2. TÉCNICAS DE VIRTUALIZAÇÃO

Segundo LAUREANO (2006), as técnicas virtualização são a paravirtualização e virtualização total: na virtualização total, a estrutura completa do hardware é virtualizado por completo, contudo, o sistema convidado não precisará sofrer nenhum tipo de alteração. Porém, o sistema real executará de forma mais lenta e o VMM necessitará de implementar alternativas para que operações sejam executadas em processadores que não implementa a virtualização nativa (Figura 5).

Já a paravirtualização, o sistema a ser virtualizado sofre modificações para que a iteração com o VMM seja eficiente. Essa modificação acarretará problemas de portabilidade do sistema. A paravirtualização reduz a complexidade de desenvolvimento de VM, já que a maioria dos processadores não suporta a virtualização nativa (Figura 6).

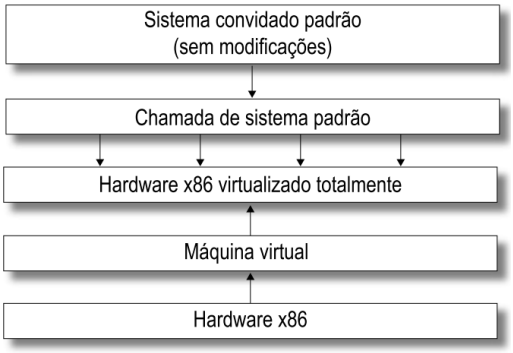

Figura 5: Estrutura de um sistema que utiliza a tecnologia de virtualização total.

Fonte: LAUREANO, 2006

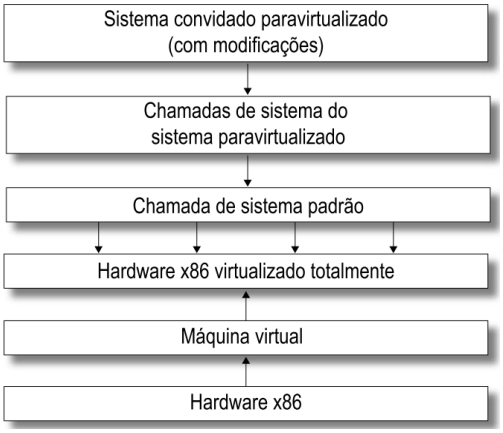

Figura 6: Estrutura de um sistema que utiliza a tecnologia de paravirtualização.

Fonte: LAUREANO, 2006

3. MATERIAIS E MÉTODOS

3.1 ÁREA DE ESTUDO

Segundo ZIVIANI, (2005), os algoritmos servem para resolver um determinado problema e todos os problemas possuem como ponto inicial para o seu processamento, uma entrada de dados. Portanto, o tamanho da entrada de dados está diretamente ligado no tempo de resposta de um algoritmo.

Para analisar a eficiência de um algoritmo deve-se sempre levar em consideração a quantidade de recursos utilizados e a quantidade de tempo. O custo de utilização de um algoritmo pode ser medido de várias maneiras. Uma delas é através da execução do programa em um computador real e/ou virtual, sendo o tempo de execução medido diretamente. As medidas de tempo obtidas desta forma são bastante inadequadas e os resultados jamais devem ser generalizados. As principais objeções são: (i) os resultados são dependentes do compilador que pode favorecer algumas construções em detrimento de outras; (ii) os resultados dependem do hardware; (iii) quando grandes quantidades de memória são utilizadas, as medidas de tempo podem depender deste aspecto (ZIVIANI, 2005).

A maioria dos algoritmos possuem parâmetros que podem afetar o tempo de execução de forma mais significativa. Estes parâmetros podem ser o número de registros de um arquivo a ser ordenado, ou o número de nós de um grafo. Portanto, os algoritmos computacionais podem ser classificados segundo a Tabela 1:

Tabela 1. Classes dos algoritmos.

Taxa de crescimento	Nome	Estrutura do código	Descrição
1	Constante	`if (a mod 2)`	Atribuição de Valores (verificar se número é par)

Log N	Logarítmica	`while( N > 1 ) {`
		`  N = N / 2;`
		`  ...`
		`}`
			Divisão do problema (busca binária)

N	Linear	`for(i = 0; I < N; i++){`
		`  ...`
		`}`
			Laço (buscar um elemento)

N log N	Linearítmica	`MergeSort`	Divisão e conquista (mergesort)

N^2	Quadrática	`for(i = 0; i < N; i++){`
		`  for(j = 0; j < N; j++){`
		`  ...`
		`  }`
		`}`
			Laço duplo (checar todos os pares – ordenação)

N^3	Cúbica	`for(i = 0; i < N; i++){`
		`  for(j = 0; j < N; j++){`
		`  for(k = 0; k < N; k++){`
		`  ...`
		`  }`
		`  }`
		`}`
			Laço triplo (checar todas as triplas)

2^N	Exponencial	`Caixeiro viajante`	Busca exaustiva (chegar todas as possibilidades)

3.2 AMBIENTE DE ESTUDO

O ambiente físico em que os algoritmos estão sendo os testes de desempenho é um computador QuadCore Intel Core 2 Quad Q8400, 2.66 GHz. Possuindo 4 GB de memória, Placa de Vídeo NVIDIA GeForce 7300 SE/ 7200 GS (256 MB) e disco rígido de 320 GB, 7200 RPM, SATA-II (Samsung HD322HJ ATA Device). Com o sistema operacional utilizado é o Ubuntu 10.04 Lucid e o compilador G++ 4.4.3.

Para realizar a medição de tempo de execução dos algoritmos foi utilizado a biblioteca desenvolvida em C++, denominada Boost.Chrono 1.53.0 (HINNANT, 2012).

Os testes virtualizados estão sendo realizados na plataforma OpenQRM 5.0. O OpneQRM é uma plataforma aberta de gestão de infraestrutura do Data-Center completamente integrada, como enorme abrangência e flexibilidade. A arquitetura totalmente conectável e com enfoque na completa automatização de operações do Data-Center de forma rápida é baseada em aplicações de implementação e acompanhamento, de alta disponibilidade, orientadas para a computação em nuvem (OPENQRM, 2012).

Dentro do OpenQRM podem ser criados ambientes virtuais utilizando máquinas virtuais como: Citrix XenServer, VMWare ESX/ESXi, Xen, Linux KVM, VirtualBox, Linux LXC Containers e OpenVZ (OPENQRM, 2012).

3.3 METODOLOGIA

Para cada classe de algoritmo apresentados na Seção 3.1 foram implementados algoritmos clássicos e esses foram testados nos ambientes modelados. Para cada algoritmo foram realizados 30 ensaios. E no final foi apresentado o tempo médio de execução.

Para os algoritmos de cada classe estão sendo realizados testes em diversos ambientes em nuvem, para descobrir a melhor configuração para um determinado algoritmo.

Inicialmente foram determinados quais seriam os fatores de controle que poderão interferir no resultado final do experimento. E segundo RODRIGUES, (2009), esses fatores podem ser classificados em quantitativos e qualitativos: (i) fatores quantitativos que representam valores numéricos que descrevem quantidades. Os fatores quantitativos podem ser de dois tipos: discretos que podem assumir um número determinado de valores e contínuos que representam contagens sequenciais; e (ii) fatores qualitativos que representam qualidades e que não usam números para descrevê-las. Os fatores qualitativos podem ser de dois tipos: ordinais onde os valores agregam a ideia de ordem e nominais, caso contrário.

Os fatores que estão sendo utilizados para a construção da matriz de experimentos que poderão interferir no tempo esperado de execução dos algoritmos são:

- Fatores quantitativos discretos: quantidade memória, números de processadores e tamanho da entrada do problema;
- Fatores qualitativos nominais: sistema operacional (Ubuntu, CentOS, OpenSUSE, Gentoo Linux e Arch Linux), máquinas virtuais (VMWare ESX/ESXi, Xen, Linux KVM, VirtualBox, Linux LXC Containers e OpenVZ.

Os níveis dos fatores Sistemas Operacionais e Máquinas Virtuais foram baseadas nas especificações e limitações da ferramenta OpenQRM (OPENQRM, 2012).

Segundo a teoria de Planejamento Experimental (MONTGOMERY, 2009; RODRIGUES, 2009) em processos complexos, com diversos fatores influentes, não se deve partir de um conjunto extenso de experimentos, que envolva um grande número de fatores com diversos níveis. É mais produtivo estabelecer um conjunto inicial com número reduzido de ensaios (poucos fatores, poucos níveis) e ir aprendendo sobre o processo e aos poucos acrescentando novos fatores e níveis e eliminando fatores que não se apresentem influentes.

Portanto, nos primeiros testes serão analisados para cada máquina virtual os seguintes fatores: (i) quantidade de núcleos de processamento (1 ou 4 núcleos de processamento); (ii) o número de quantidade de memória RAM (1 ou 2 GB); (iii) tamanho da entrada de dados (múltiplos de 2); e (iv) sistema operacional Ubuntu.

4. RESULTADOS E DISCUSSÃO

Para verificar o desempenho dos algoritmos na plataforma OpenQRM usando as máquinas virtuais foram inicialmente realizados testes em um ambiente físico e estão apresentados na Tabela 2. Os resultados apresentados mostram que a alteração da memória não interferiu nos tempos de execução dos algoritmos. Pode-se verificar em cada classe, que os algoritmos respeitaram os pontos destacado na literatura, como por exemplo:

- Para algoritmos pertencentes a classe de algoritmos com complexidade constante o tamanho da entrada não alterou o resultado das execuções dos algoritmos.
- Para algoritmos pertencentes a classe de algoritmos com complexidade linear quando duplica-se o tamanho da entrada, o tempo de execução dos algoritmos são multiplicados por 2;

- Para algoritmos pertencentes a classe de algoritmos com complexidade logarítmica o tempo de execução pode ser considerado como sendo menor do que uma constante grande. Quando n é mil e a base do logaritmo é 2, $\log_2 n \approx 10$, quando n e um milhão, $\log_2 n \approx 20$;

- Para algoritmos pertencentes a classe de algoritmos com complexidade quadrática quando duplica-se o tamanho da entrada, o tempo de execução dos algoritmos são multiplicados por 4.

Os testes apresentados na Tabela 3 foram realizados na plataforma OpenQRM usando a máquina virtual VirtualBox. Os resultados apresentados mostram que a alteração da memória não interferiu nos tempos de execução dos algoritmos. Pode-se verificar em cada classe, que os algoritmos também respeitaram os pontos destacado na literatura (ZIVIANI, 2005).

Baseado nos dados apresentados nas Tabelas 2 e 3 os dados foram plotados de acordo para uma melhor visualização. Os dados do eixo Y foram plotados de forma logarítmica para melhor visualização. A Linha Preta representa a linha de tendência. As Figuras de 7 a 10 apresentam a representação dos dados realizados em um ambiente físico. A Figura 7 mostra uma representação do comportamento do algoritmo da classe Constante. Pode-se observar que a razão entre um tempo posterior e um tempo o atual ficou próximo de 1 unidade.

Tabela 2. Resultados referentes ao ambiente físico.

Tamanho da entrada de dados	Classe Constate		Classe Linear		Classe Logaritmica		Classe Quadrática	
	4 núcleos e 1GB	4 núcleos e 2GB	4 núcleos e 1GB	4 núcleos e 2GB	4 núcleos e 1GB	4 núcleos e 2GB	4 núcleos e 1GB	4 núcleos e 2GB
2	476.7	437.7	452.2	453.9	506.425	495.9	490.6	509.9
4	455.7	449.3	452.275	458.6	464.4	463.3	523.9	537.8
8	448.8	456.4	474.9	467.9	476.65	484.2	972.525	747.2
16	459.2	428.4	508.125	488.8	487.05	484.3	1517.3	1562.1
32	441.8	444.6	548.275	551.8	494.075	488.8	3591.6	3927.5
64	447.1	449.4	670.45	665.8	495.85	491.2	10856.8	12201.2
128	450.4	440.1	887	884.7	520.3	498.2	38885.8	44176.9
256	447.0	447.0	1321.72	1324.7	506.375	507.5	148512	169409.4
512	448.8	437.6	2186.05	2193.1	529.075	519.2	579115	589374.2
1024	452.3	461.0	3923.38	3915.7	532.475	523.9	2.29E+06	2.29E+06
2048	450.4	437.7	7390.95	7389.2	543.025	530.7	9.12E+06	9.12E+06
4096	544.8	423.7	14403.1	14405.9	548.275	540.1	3.64E+07	3.64E+07
8192	455.7	433.1	28626.1	36810.9	548.3	551.8	1.46E+08	1.46E+08
16384	443.5	444.7	56878.8	75500.7	558.8	549.3	5.85E+08	5.85E+08
32768	454.0	440.0	113766	150864.1	569.275	563.4	2.34E+09	2.34E+09
65536	441.7	437.6	226136	227207.7	567.425	563.4	9.36E+09	9.35E+09
131072	448.8	428.3	455320	454471.2	574.425	591.3	3.75E+10	3.74E+10
262144	441.7	440.0	936053	931203.2	586.725	591.3	1.51E+11	1.51E+11
524288	438.2	442.3	1.90E+06	1.89E+06	619.9	612.3	6.17E+11	6.14E+11
1048576	443.6	684.3	3.81E+06	3.77E+06	618.125	612.2	2.49E+12	2.47E+12

Tabela 3. Resultados referentes ao ambiente virtual.

Tamanho da entrada de dados	Classe Constate		Classe Linear		Classe Logaritmica		Classe Quadrática	
	4 núcleos e 1GB	4 núcleos e 2GB	4 núcleos e 1GB	4 núcleos e 2GB	4 núcleos e 1GB	4 núcleos e 2GB	4 núcleos e 1GB	4 núcleos e 2GB
2	6136.63	3156.87	3007.83	2905.4	3017.13	3240.57	2914.63	3827.3
4	2951.97	3035.77	3454.77	2998.53	2942.63	3110.23	2700.53	2737.73
8	2961.3	2905.4	3156.83	2951.97	2924.03	3278.03	2840.17	2896.1
16	2924.07	2933.37	3035.7	3091.63	2933.3	2747.03	3799.33	3240.77
32	2942.67	2961.27	3370.93	3007.9	2942.67	2719.1	4441.8	4572.3
64	2952.03	2942.63	3128.9	5373.17	3259.27	3035.83	9088.67	9870.87
128	2942.67	2886.8	6872.4	3333.7	2989.2	2728.53	28448.7	28206.5
256	2933.37	2886.87	3715.57	3696.9	2924.03	3519.97	101503	97880.2
512	2905.4	2933.33	4423.27	4386.1	3231.37	2914.7	515485	395153
1024	3324.47	3017.13	5857.27	5913.07	2989.3	2998.5	1.94E+06	2.12E+06
2048	2933.3	2924.03	8883.77	9377.33	3007.9	2933.3	6.53E+06	6.57E+06
4096	2905.33	3063.77	15374.3	14890.2	3389.6	3091.67	2.88E+07	2.58E+07
8192	2905.37	3501.43	26958.7	27238.1	3035.8	3994.93	1.06E+08	1.01E+08
16384	2896.07	2933.33	50146	56720.4	3017.13	3091.63	4.55E+08	4.20E+08
32768	2775.03	2914.73	143789	101447	3017.2	3128.83	1.61E+09	1.71E+09
65536	2970.6	3575.87	225634	217709	3026.4	3212.7	6.48E+09	7.14E+09
131072	2700.53	2886.77	426088	748354	3370.93	4982.07	2.70E+10	2.82E+10
262144	3464.03	2905.4	1.09E+06	1.07E+06	3063.7	6183.23	1.08E+11	1.13E+11
524288	3277.83	2905.4	1.98E+06	2.11E+06	3100.97	4283.5	4.32E+11	4.51E+11
1048576	2886.87	2933.37	3.55E+06	4.25E+06	3138.2	4358.2	1.73E+12	1.80E+12

A Figura 8 mostra uma representação do comportamento do algoritmo da classe Linear. Pode-se observar que a razão entre um tempo posterior e um tempo o atual ficou próximo de 2 unidades.

A Figura 9 mostra uma representação do comportamento do algoritmo da classe Logarítmica. Pode-se observar que a razão entre um tempo posterior e um tempo o atual ficou próximo de 1 unidade. Pois como, por exemplo, a razão para de 128 elementos e posteriormente um problema de 256 elementos é de log2256/ log2128 ≈ 1 unidade, como pode ser apresentado no gráfico pelas linhas de diferenças.

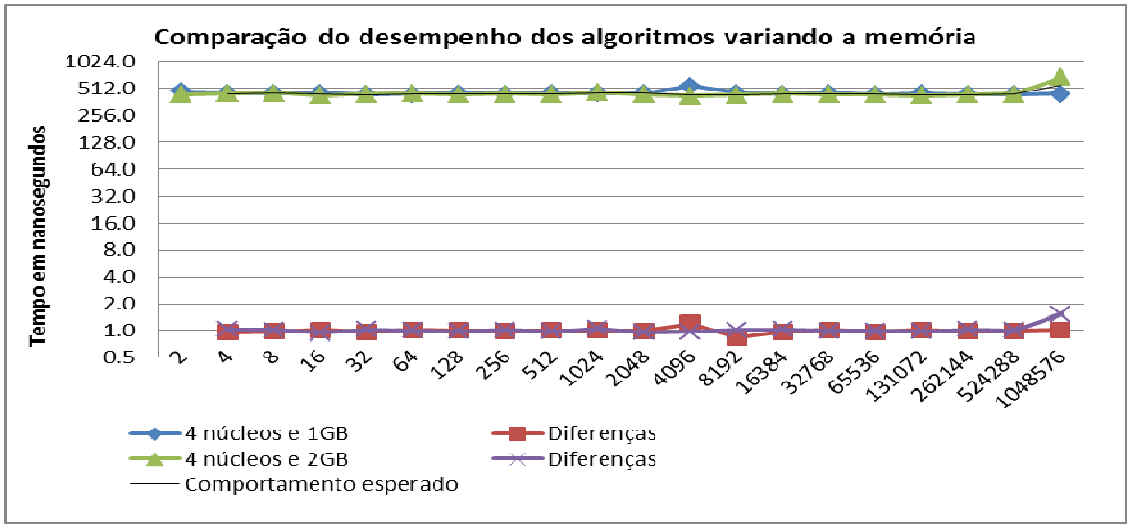

Figura 7: Comparação entre os tempos de algoritmos constantes variando a quantidade memória.

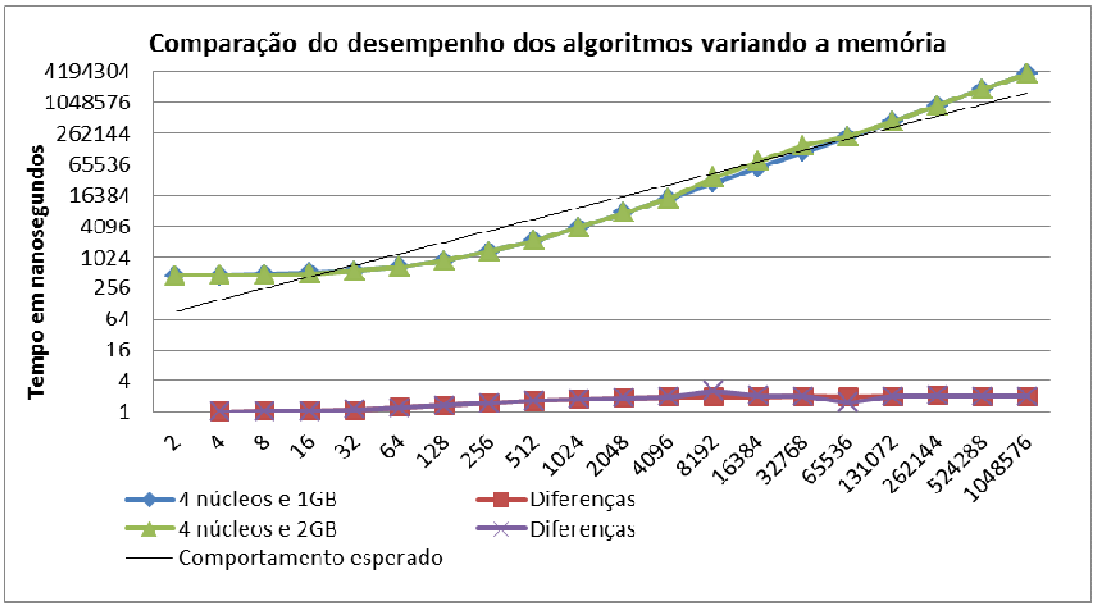

Figura 8: Comparação entre os tempos de algoritmos lineares variando a quantidade memória.

A Figura 10 apresenta uma representação do comportamento do algoritmo da classe Linear. Pode-se observar que a razão entre um tempo posterior e um tempo o atual ficou próximo de 4 unidades.

As Figuras 11 e 12 mostram que: (i) para a faixa de tamanho de entrada o algoritmo que foi executado em um ambiente físico foi sempre superior em relação ao mesmo algoritmo executado em um ambiente virtual; (ii) os testes realizados no ambiente virtual apresentaram uma alternância; e (iii) as razões entre os tempos posteriores e atuais foram também respeitados conforme a literatura.

As Figuras 13 e 14 mostram comportamentos diferentes em relação as Figuras 11 e 12. Os Algoritmos executados em um ambiente virtual para grande quantidade de entrada de dados apresentaram um comportamento aproximado em relação aos algoritmos executados em um ambiente físico.

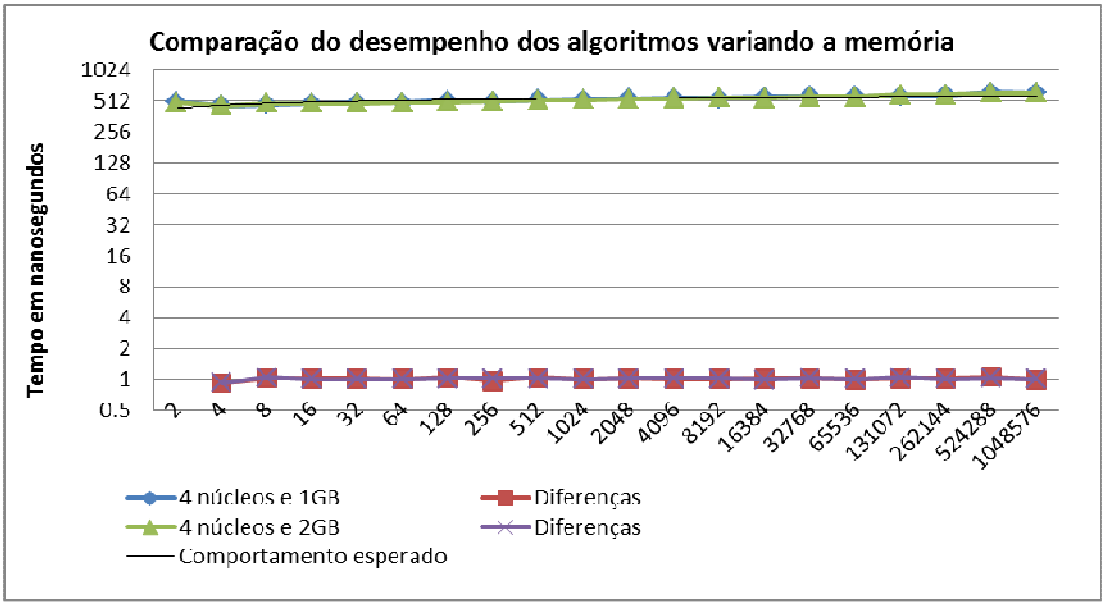

Figura 9: Comparação entre os tempos de algoritmos logarítmicos variando a quantidade memória.

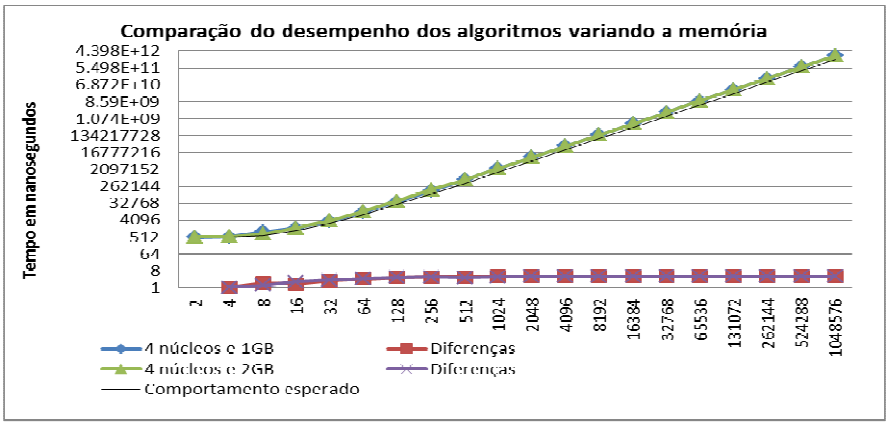

Figura 10: Comparação entre os tempos de algoritmos quadráticos variando a quantidade memória.

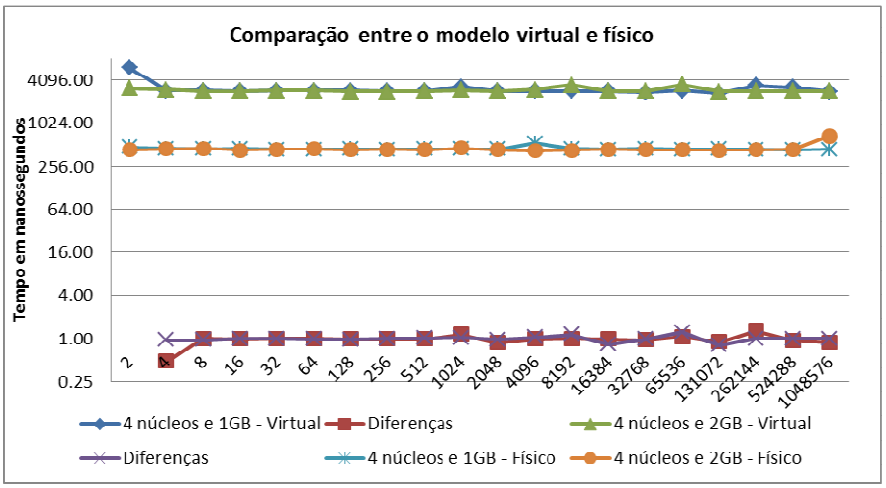

Figura 11: Comportamento do algoritmo pertencente à classe constante executado em um ambiente virtual em relação ao mesmo algoritmo executado em um ambiente físico.

Os testes deverão serem executados novamente em ambiente físico e na máquina virtual VirtualBox para verificar se as mesmas informações serão satisfeitas. Mas inicialmente pode-se concluir que o VirtualBox: (i) adapta melhor a algoritmos com grandes quantidades de entrada de dados; (ii) a memória virtual não interferiu nos resultados apresentados; e (iii) possuem melhor desempenho para algoritmos pertencentes a classe linear e a classe quadrática.

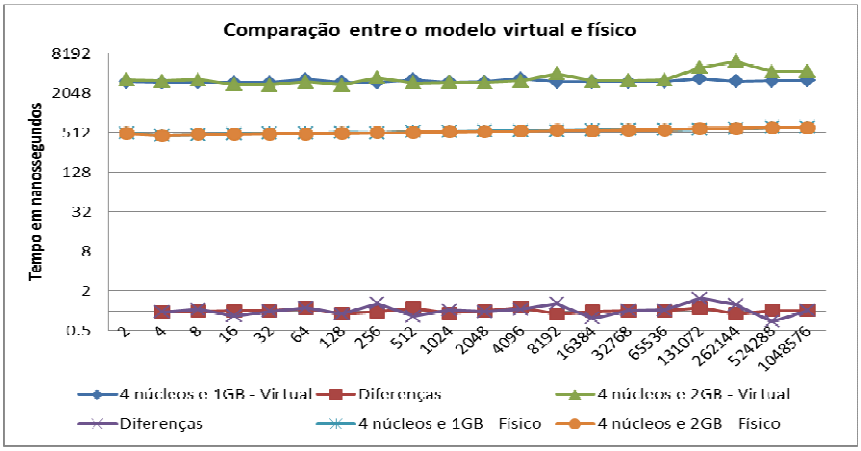

Figura 12: Comportamento do algoritmo pertencente à classe logarítmica executado em um ambiente virtual em relação ao mesmo algoritmo executado em um ambiente físico.

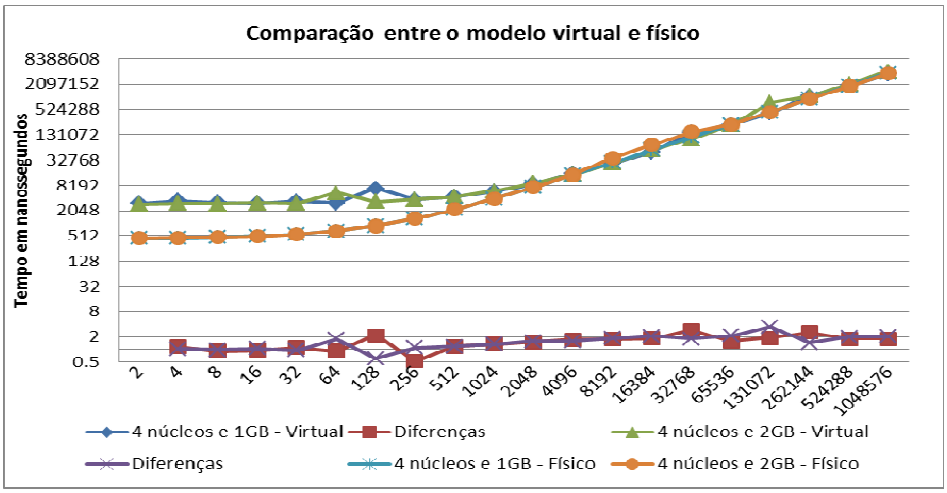

Figura 13: **Comportamento do algoritmo pertencente à classe lineares executado em um ambiente virtual em relação ao mesmo algoritmo executado em um ambiente físico.**

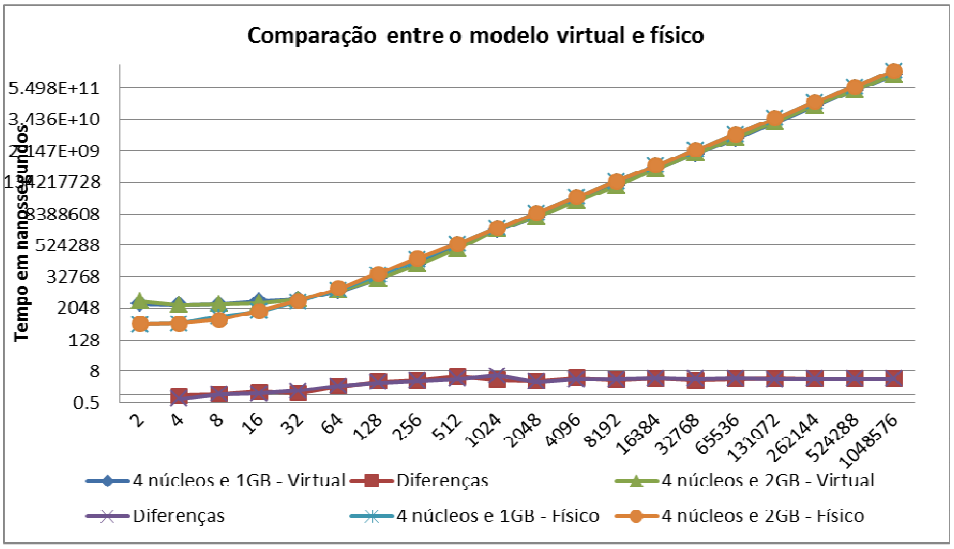

Figura 14: Comportamento do algoritmo pertencente à classe quadrática executado em um ambiente virtual em relação ao mesmo algoritmo executado em um ambiente físico.

5. CONSIDERAÇÕES FINAIS

Os testes mostram que ambientes virtualizados apresentam um comportamento diferenciado em relação a ambientes físicos para algumas classes de algoritmos. Outros testes estão sendo realizados em outras máquinas virtuais como: KVM, VMWare, Xen, e LXC para verificar o comportamento dos algoritmos. Futuramente com todos os dados coletados será necessários descobrir qual a melhor configuração de um ambiente virtual em cloud computing.

Testes serão realizados variando o número de máquinas virtuais no mesmo hardware para avaliar como que o desempenho decresce com o aumento de máquinas. Repare que todas as máquinas deverão executar os algoritmos ao mesmo tempo, competindo assim pelos recursos do hardware. E neste caso, poderá avaliar o desempenho de diferentes máquinas virtuais, como VirtualBox, KVM, VMWare.

Outro aspecto relevante: não houve diferenças entre 1GB e 2GB porque as entradas avaliadas foram todas muito menores do que 1GB. A maior entrada avaliada teve tamanho 2^20 ~ 10^6 que é muito menor do que 1GB ~ 10^9. Então como próximo passo serão avaliadas entradas maiores, como 10^9, para avaliar a diferença entre 1GB e 2GB.

6. REFERÊNCIAS

BARUCHI, J. H. Comparativo entre ferramentas de Virtualização. Faculdade de Jaguariúna, Novembro de 2008.

HINNANT, H., DAWES, B., and ESCRIBA V. J. B. Chapter 5. Boost.Chrono 2.0.0. online: http://www.boost.org/doc/libs/1_53_0/doc/html/chrono.html, 2012.

CHOWDHURY, N. M. M. K. and BOUTABA, R. A Survey of Network Virtualization. Computer Network. Vol.: 54. Iss: 5. pp: 862-876, 2012. Disponível em: http://dx.doi.org/10.1016/j.comnet.2009.10.017

FIGUEIREDO, R., DINDA, P.A. and FORTES, J. Introdution: Resource Virtualization Renaissance. In Proc. of IEEE Internet Computing, 38:5. pp: 28:31, Maio 2005.

IBM, "Virtualization in education," IBM Global Education, White Paper, October 2007. {Online}. Available: http://www-07.ibm.com/solutions/in/education/download/-Virtualization%20in%20Education.pdf.

LAUREANO, M., In Novatec (Editors), Máquinas Virtuais e Emuladores. Conceitos, Técnicas e Aplicações, Novatec, 2006.

MENASCÉ, D. Virtualization: Concepts, Applications, and Performance. In Proc. Of Computer Measurement Group's 2005 International Conference (CMG), Orlando, FL, Dezembro 2005.

MONTGOMERY, D. Design and Analysis of Experiments. John Wiley and Sons, 7th edition, 2009.

NANDA, S. e CHIUEH, T. A Survey on Virtualization Technologies. . Technical Report ECSL–TR–179, SUNY at Stony Brook, Feb. 2005.

POPEK, G. J., GOLDBERG, R. P. "Formal requirements for virtualizable third generation architectures", Commun. ACM, v. 17, n. 7, pp. 412–421, 1974. ISSN: 0001-0782. doi: http://doi.acm.org/10.1145/361011.361073.

REGO, P.A.L. FAIRCPU: uma arquitetura para provisionamento de máquinas virtuais utilizando características de processamento. Dissertação de Mestrado. Programa de Pós-Graduação em Ciência da Computação da Universidade Federal do Ceará. 2012.

RODRIGUES, M. I. and LEMMA A. F. Planejamento de experimentos e otimização de processos. Casa do Espírito Amigo Fraternidade Fé e Amor, Campinas, SP, 2009.

ROSENBLUM, M. and GARFINKEL, T. Virtual machine monitors: current tecnology and future trends. IEEE Computer Magazine, 38:5 pp: 39–47. 2005.

OPENQRM. openQRM Adminstration Guide. Online: http://www.openqrm-enterprise.com/news/article/article/the-administrator-guide-for-openqrm-5.html. 2012

SOUZA, F. B. Uma arquitetura para Monitoramento e Medição de Desempenho para Ambientes Virtuais. Ph. D. Thesis, Universidade Federal de Minas Gerais, 2006.

VERAS, M. DATACENTER: Componente central da infraestrutura de TI. Rio de Janeiro: Brasport, 2009.

ZHOU, S. Virtual networking. SIGOPS Oper. Syst. Rev., ACM, New York, NY, USA, v. 44, p. 80–85, dez. 2010. ISSN 0163-5980. Disponível em: <http://doi.acm.org/10.1145/1899928.1899938>

ZIVIANI, N. Projeto de Algoritmos com implementação em Pascal e C. 2 ed. São Paulo: Pioneira Thomson Learning, 2005.