A utilização da função *affinity* na manutenção de QoS nos serviços de nuvem: uma comparação de desempenho entre os virtualizadores Xen e KVM

Dionisio Leite[1], Maycon Peixoto[2], Bruno Batista [4], Bruno Tardiole[4], Carlos Ferreira[3], Danilo Segura[3]

[1]Universidade Federal do Mato Grosso do Sul
Campus Ponta Porã (UFMS-CPPP)
Ponta Porã - MS - Brasil

[2]Universidade Federal da Bahia (UFBA)
Departamento de Ciência da Computação (DCC-IM)
Salvador - Bahia - Brasil

[3]Instituto de Ciências Matemáticas e Computação
Universidade de São Paulo (ICMC-USP)
São Carlos, SP - Brasil

[4]Universidade Federal de Itajubá
Itajubá - MG - Brasil

dionisio.leite@ufms.br,mayconleone@dcc.ufba.br

{brunoguazzelli,brunokuehne}@unifei.edu.br

{chgferreira,dcmsdancosta}@usp.br

Abstract. *This paper presents a performance comparison between Xen and KVM virtualizers regarding the use of affinity function, which is common to both virtualizers. Currently, virtualization gained prominence for being the basis for the realization of cloud computing. However, as cloud providers omit some information about the infrastructure used is not always possible to know which features provided by virtualizers are actually used by cloud providers. Thus, the purpose of this article is to demonstrate how application performance may be affected by the use of a common specific function to virtualizers Xen and KVM. Through experiments, the affinity function is used. The Xen results presented performance variations with the use of affinity function, since, for the virtualizer KVM, performance was not affected.*

Resumo. *Este artigo apresenta uma comparação de desempenho entre os virtualizadores Xen e KVM com relação à utilização da função affinity, que é comum aos dois virtualizadores. Atualmente a virtualização ganhou destaque por ser uma da principais bases para a realização da computação em nuvem. No entanto, como os provedores de nuvem omitem algumas informações sobre a infraestrutura utilizada, nem sempre é possível saber quais funcionalidades providas pelos virtualizadores são utilizadas de fato. Sendo assim, o objetivo deste artigo é demonstrar como o desempenho de aplicações pode ser afetado com a utilização de uma função específica comum aos virtualizadores Xen e KVM. Por*

meio de experimentos a função affinity foi utilizada. Nos resultados foi possível verificar que o virtualizador Xen apresentou variações de desempenho com a utilização da função affinity, sendo que no virtualizador KVM o desempenho não foi afetado.

1. Introdução

A computação em nuvem é um tipo de computação que vem revolucionando a história da Tecnologia da Informação (TI) [Weingärtner et al. 2015]. A forma como os serviços são oferecidos, por meio da Internet, é o que faz com que a computação em nuvem seja atraente, tanto para os consumidores como para os prestadores de serviços. Essa forma de computação faz com que os serviços sejam acessados de uma maneira mais prática do que as formas convencionais, uma vez que as aplicações estão disponíveis na Internet e o acesso às informações e aplicações depende, na maior parte das vezes, apenas de um navegador Web [Weingärtner et al. 2015].

Com a utilização da nuvem, é necessário considerar diferentes abordagens para a utilização dos recursos computacionais e dos serviços oferecidos, como, por exemplo, a forma como os dados são processados. Nesse contexto, surge a necessidade de oferecer qualidade de serviço para os serviços de infraestrutura de nuvem [Armbrust et al. 2009].

A Qualidade de Serviço (QoS) pode ser definida como a percepção do usuário em relação à eficiência de um dado serviço. Prover QoS significa garantir a um serviço características como: perdas mínimas, desempenho máximo, pequeno tempo de resposta, exatidão e consistência dos dados recebidos [Kumar et al. 2013] [Dubey and Agrawal 2013] [Alhamazani et al. 2014].

Um problema enfrentado na computação em nuvem é a ausência de informações por parte dos provedores de serviço sobre a infraestrutura que hospeda as máquinas virtuais, tornando a avaliação e seleção de um serviço uma tarefa complexa [Manvi and Krishna Shyam 2014]. A presença de vários fatores envolvidos, como tipos de virtualizadores, utilização de função *affinity*, *middlewares* de gerenciamento e otimizações do ambiente tornam ainda mais complexa a determinação do desempenho apresentado pelos provedores de serviços de nuvem.

Como há uma grande quantidade de fatores que podem ser considerados, neste artigo iremos abordar a função CPU-*affinity*[1][2]. Com essa função é possível fazer a relação de um vCPU a um CPU do hospedeiro. Em outras palavras, é possível fazer com que o escalonamento de CPU real e vCPU seja feito de forma que cada vCPU ocupe sempre o mesmo núcleo da CPU que lhe foi configurado.

Sendo assim, o objetivo deste artigo é apresentar como a utilização da função *affinity* pode influenciar na QoS dos serviços de infraestrutura, contribuindo assim com a discussão sobre qual tipo de virtualizador pode promover um melhor desempenho para as aplicações em nuvem. Para avaliar a utilização do recurso *affinity*, foram utilizados os virtualizadores Xen e KVM, por serem duas tecnologias amplamente utilizadas em ambientes de nuvens comerciais. Além disso, dois tipos de *benchmarks* foram utilizados,

[1]Para informações do virtualizador Xen, acesse: http://wiki.xen.org/wiki/Tuning_Xen_for_Performance

[2]Para informações do virtualizador KVM, acesse: http://docs.fedoraproject.org/en-US/Fedora/13/html/Virtualization_Guide/ch25s06.html

e os resultados mostraram que no virtualizador Xen a utilização da função *affinity* pode promover diferenças significativas de desempenho, o que não ocorreu com o virtualizador KVM.

As demais seções deste artigo estão organizadas da seguinte forma: na seção 2 é apresentada a revisão da literatura demonstrando o estado da arte em virtualização; na seção 3 é apresentada a metodologia utilizada para a elaboração dos experimentos; na seção 4 os resultados são apresentados; na seção 5 são apresentadas as discussões pertinetes ao tema e na seção 6 são apresentadas as conclusões deste artigo.

2. Revisão da Literatura

A virtualização é uma das principais tecnologias utilizadas pela computação em nuvem. Essa tecnologia apoia, principalmente, os serviços do tipo IaaS, mas também serve de base para os demais tipos de serviços [Nadeem and Qaiser 2015].

A virtualização pode ser definida como a utilização de um ou vários computadores lógicos em um único computador real, o que implica dizer que um único computador físico pode assumir o papel de vários computadores lógicos também chamados de máquinas virtuais (VMs). Cada VM fornece um ambiente completo aos seus utilizadores, semelhante a uma máquina física [Chieu et al. 2009]. Assim, cada VM pode ter seu próprio sistema operacional, aplicações, serviços de armazenamento e rede [Carissimi 2008].

O trabalho de [Chierici and Veraldi 2010] apresenta uma análise quantitativa entre os virtualizadores KVM e Xen. No caso do Xen, os autores utilizaram as duas formas de virtualizar permitidas pelo Xen: virtualização completa e paravirtualização. A motivação para a avaliação foi perceber que o KVM possui bons resultados com aplicações científicas. Os testes realizados foram relacionados ao uso de CPU, acesso a disco e rede. Os experimentos apresentam o Xen (tanto virtualização completa como paravirtualização) como a melhor solução entre os três testes realizados.

Uma análise de tecnologias de virtualização para ambientes de computação de alto desempenho é apresentada por [Younge et al. 2011]. A avaliação dos autores considerou aspectos funcionais de quatro virtualizadores, como suporte a paravirtualização e virtualização completa, quantidade de memória suportada e quantidade de núcleos suportados. Além dessa avaliação, foi abordada uma avaliação quantitativa entre o Xen, o KVM e o VirtualBox, onde os melhores desempenhos foram obtidos pelo virtualizador KVM.

[Leite et al. 2012] apresentam uma avaliação de desempenho sobre os virtualizadores que podem ser utilizados na nuvem. Os autores avaliaram os virtualizadores Xen e KVM em relação às cargas de trabalho de CPU e sistema. As comparações de desempenho foram realizadas comparando uma VM com a máquina física, com a finalidade de estimar a porcentagem de degradação causada pelo uso da camada de virtualização e em relação às quantidades de máquinas virtuais concorrendo por recursos. Nos resultados o virtualizador KVM foi o que obteve os melhores resultados.

Em [de Carvalho and Bellezi 2014], uma análise sobre os virtualizadores Xen e KVM é apresentada. Nesse estudo os autores avaliaram: escrita e leitura em disco, acesso à memória, CPU, largura de banda de rede e escalabilidade. Os resultados demonstraram

que o Xen foi superior ao KVM, inclusive nos acessos ao disco.

[Huber et al. 2011] apresentam uma avaliação de desempenho sobre recursos virtualizados aplicados à nuvem. Nesse estudo são comparados os virtualizadores Xen e VmWare e vários componentes do sistema são avaliados, entre eles a utilização da função *affinity*. Os resultados apresentados demonstram que a utilização da função *affinity* é util na escalabilidade entre máquinas virtuais.

[Gupta et al. 2013] apresentam um estudo sobre computação de alto desempenho na nuvem. Um dos principais pontos é a utilização da função *affinity* em aplicações concorrentes. Os autores utilizaram em seus estudos apenas o virtualizador KVM e concluem que a função *affinity* causa influência nos resultados obtidos quando a quantidade de instâncias é pequena. Além disso, nos resultados os autores não determinam se a relação foi de uma vCPU para uma CPU ou se há várias vCPUs para uma mesma CPU.

[Nadeem and Qaiser 2015] apresentam uma avaliação sobre três diferentes plataformas de nuvem privada. Nesse estudo cada plataforma possui um virtualizador diferente, sendo: KVM, Xen e Openvz. Foram considerados: o tempo de resposta, *throughput* do processador, desempenho de memória, memória cache e operações de entrada e saída. Nos resultados, os autores indicam que não há uma solução melhor que outra. Na verdade, para cada tipo de situação um virtualizador pode ser melhor que o outro.

Há vários trabalhos que consideram a avaliação ou comparação de desempenho entre virtualizadores. Essa avaliação é necessária uma vez que esses virtualizadores possuem arquiteturas diferentes, como é o caso do Xen e KVM. Esses estudos demonstram que as tecnologias de virtualização estão em constante evolução.

Sendo assim, este trabalho se diferencia por apresentar uma comparação entre os virtualizadores Xen e KVM com relação à utilização da função *affinity* e pela utilização da relação de apenas uma vCPU por uma CPU física.

3. Metodologia

Como a camada de infraestrutura de nuvem possui vários componentes, desenvolver um ambiente de nuvem se torna um desafio. Sendo assim, a ferramenta Cloudstack foi utilizada para desenvolver o protótipo utilizado neste estudo.

A Cloudstack é uma plataforma de código aberto que reúne funcionalidades para o desenvolvimento de nuvens públicas, privadas e híbridas e o fornecimento de serviços de IaaS [Kumar et al. 2014]. Todos os componentes necessários para uma nuvem funcionar, como rede interna, armazenamento e serviços de computação, são gerenciados pelo Cloudstack.

Uma das vantagens na utilização da Cloudstack é a possibilidade de utilização de mais de um virtualizador no processo de gerenciamento do serviço de computação, sendo suportados os virtualizadores: Xen, VMware, KVM e Openvz [Kumar et al. 2014]. Todos os nós envolvidos no desenvolvimento do protótipo possuem a mesma configuração, apresentada na Tabela 1.

Tabela 1. Características físicas dos hospedeiros

Memória	16 GB (2 x 8 GB) 1333 MHz DDR3
Processador	Intel *Core* i7 870 2.93 GHz
Cores	4
Threads	4
Cache	8 MB
Disco	SATA 1 TB 7.200 rpm
Sistema Operacional	Ubuntu *Server* x64 12.04 LTS *Kernel* 3.11

Como apresentado na Tabela 1, todas as máquinas físicas possuem o mesmo tipo de *hardware* e suporte à virtualização. As vCPUS das VMs serão alocadas com relação de 1 para 1, ou seja, cada vCPU terá um núcleo específico.

As avaliações experimentais seguiram a metodologia apresentada por [Jain 1991]. A metodologia utilizada propõe que o ambiente seja conhecido e o experimento seja controlado, possibilitando assim a reprodução dos resultados obtidos e a validação dos dados de forma estatística.

O projeto de experimentos utilizados foi o fatorial completo [Jain 1991]. A Tabela 2 apresenta o modelo de experimentos realizado, demonstrando as variações de níveis para os fatores vCPU, memória, *affinity* e quantidade de VMs.

Tabela 2. Experimentos considerando a presença ou ausência da função *affinity*

Fatores	**vCPU**	**Memória**	***affinity***	**VMs**
Níveis	2	3,5	Sim	2
Níveis	4	7	Não	4

A Tabela 2 é a mesma tanto para o virtualizador Xen quanto para o virtualizador KVM. Além disso, o sistema operacional das VMs é o Ubuntu *server* 12.04. Com relação às variáveis de resposta, ou resultados, foram considerados os *benchmarks* N-queens e Apache, com o **tempo de resposta** e a quantidade de **requisições por segundo**. Essas duas cargas de trabalho podem ser descritas como:

- ***N-queens*** - Essa carga é relacionada ao tempo de solução do problema das N rainhas em um tabuleiro de tamanho 18. É um problema combinatório exponencial que necessita de uma elevada carga de processamento, logo é uma carga de trabalho do tipo CPU-*bound*.
- **Apache 1.6** - Essa carga é relacionada às solicitações por segundo que um determinado sistema pode atender quando são realizados 1.000.000 pedidos, sendo 100 solicitações realizadas simultaneamente. Essa carga de trabalho pode ser considerada de sistema, pois há uma grande quantidade de utilização de memória, disco e processador.

Para todos os conjuntos de experimentos foram consideradas N execuções do *benchmark* (carga de trabalho) selecionado, possibilitando assim realizar as análises estatísticas. As N execuções são utilizadas para a composição das médias e obtenção do desvio padrão e erro obtido nas observações [Jain 1991]. O desvio e o erro são descritos de acordo com as Equações 1 e 2.

$$D_p = \sqrt{\sum_{i=1}^{n} \frac{(M - a_1)^2}{(N - 1)}} \tag{1}$$

$$\left(\frac{D_p * Const}{\sqrt{N}}\right) \tag{2}$$

O desvio padrão (Dp) apresentado na Equação 1 é calculado a partir da observação entre a média obtida (M) e cada amostra ($a1...an$) em um conjunto de N execuções.

Cada conjunto de experimento foi submetido ao teste de normalidade, mais especificamente o Teste *Shapiro-Wilk* [Shapiro and Wilk 1965]. Com isso, foi possível observar que os dados possuíam normalidade, possibilitando escolher o Teste *t-student* para análise comparativa das amostras.

4. Resultados

Nesta seção iremos apresentar os resultados obtidos, separando os virtualizadores para o melhor entendimento dos resultados.

4.1. *Affinity* no virtualizador Xen

O gráfico da Figura 1 apresenta o resultado obtido na execução da carga de trabalho N-*queens*, considerando a execução dos 16 experimentos.

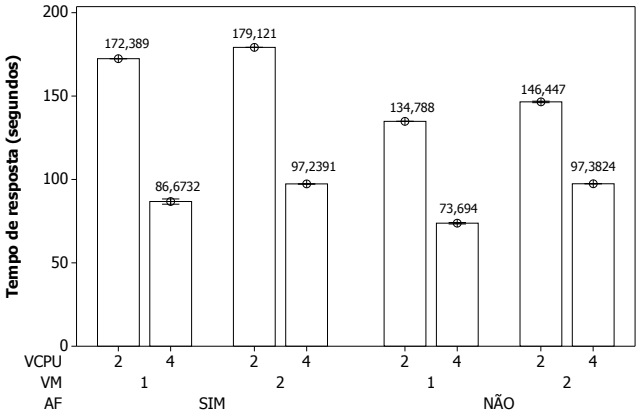

Figura 1. Tempos de resposta observados considerando 5% de erro.

Os resultados apresentados na Figura 1 mostram que os tempos de resposta são mais elevados com a utilização do *affinity*. A Tabela 3 sumariza os resultados obtidos.

Tabela 3. Diferenças de desempenho no uso da função *affinity*

Melhor desempenho	VM	vCPU	AF	Diferença %
134,78s	1	2	Não	27,89%
73,69s	1	4	Não	17,61%
146,44s	2	2	Não	22,31%
97,38s	2	4	Não	0,14%

Na Tabela 3 os melhores tempos de resposta são obtidos com a não utilização da função *affinity*. É importante observar as diferenças de tempo de resposta obtidas no aumento da quantidade de VMs. Na Tabela 4 são apresentadas as diferenças nos tempos de resposta com o aumento no número de VMs.

Tabela 4. Comparações com o aumento no número de máquinas virtuais.

Comparação	Tempo	vCPU	AF	Diferença
1VM x 2VM	6,73s	2	Sim	3,90%
1VM x 2VM	11,65S	2	Não	8,64%
1VM x 2VM	10,56s	4	Sim	12,19%
1VM x 2VM	23,68s	4	Não	32,14%

Para os casos apresentados na Tabela 4, quanto menor a diferença de tempos de resposta entre a quantidade de VMs utilizadas melhor é a qualidade com que os serviços são disponibilizados. Na Figura 2 são apresentadas as influências causadas pelos fatores considerados.

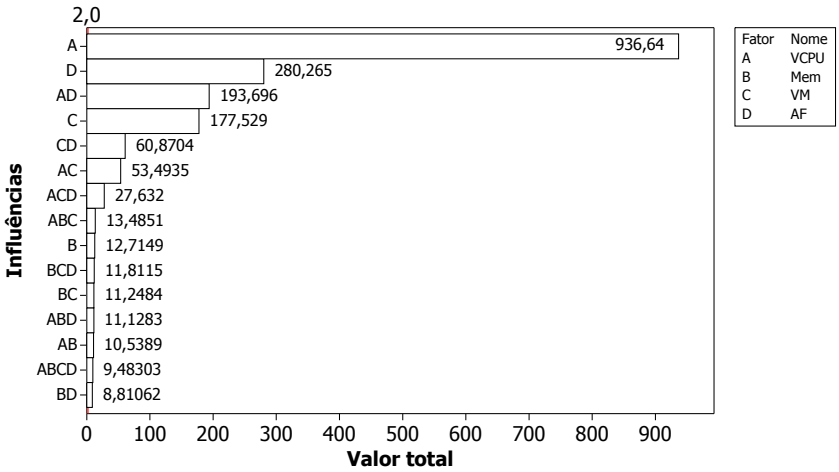

Figura 2. Gráfico de Pareto com os valores de influência absolutos para a carga de trabalho N-*queens* com grau de confiança de 95%.

Na Figura 2 o eixo X representa a quantidade absoluta de influência obtida e o eixo Y apresenta os fatores mais influentes em ordem crescente. O limite de influência são duas unidades, ou seja, todos os valores que estão acima de duas unidades possuem influência nos resultados obtidos. É possível descrever a porcentagem de influência de cada fator através das Equações 3 e 4.

$$SST = SSA + SSB + SSC + SSD + SSAB + SSAC + ... + SSABCD \quad (3)$$

A Equação 3 é utilizada para obter a soma total dos quadrados (SST) e a partir do valor de SST é possível obter a porcentagem de influência dos demais fatores utilizando a Equação 4.

$$Inf_X = \frac{SSX}{SST} \quad (4)$$

Na Equação 4 o valor de X representa o fator analisado. No caso da Figura 2, nota-se que as maiores influências são dadas pelos fatores vCPU; *affinity*; a combinação de vCPU e *affinity*; e quantidade de VMs e são apresentados na Tabela 5.

Tabela 5. Fatores com maior influência.

Fatores mais influentes	Porcentagem
vCPU	51,48%
affinity	15,40%
Combinação vCPU e *affinity*	10,64%
VMs	9,75%

A Figura 3 apresenta os resultados obtidos na execução de experimentos com a carga de trabalho Apache. Para esses resultados são consideradas as requisições atendidas por segundo e, neste caso, quanto mais elevado o resultado, melhor.

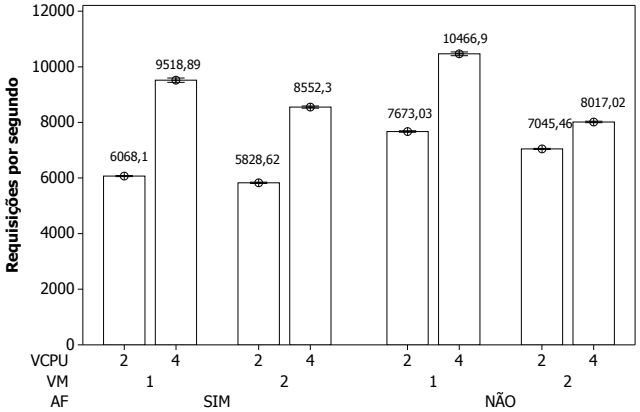

Figura 3. Requisições por segundos atendidas na execução do *benchmark* Apache considerando 5% de erro.

De acordo com a Figura 3, os melhores resultados são obtidos na não utilização da função *affinity*. No entanto, essa superioridade pode esconder características de desempenho importantes. Quando há um acréscimo de máquinas virtuais há também uma queda de desempenho. Com a função *affinity* os valores mudam 4,10% para dois vCPUs e 11,30% para quatro vCPUs. Sem a função *affinity* os valores mudam 8,9% para dois vCPUs e 30,55% para quatro vCPUs.

A função *affinity* é uma boa opção para ambientes que compartilham recursos, como é o caso da computação em nuvem. Com o aumento de recursos utilizados o desempenho teve uma queda de apenas 11,30% enquanto sem a utilização desta função, a queda no desempenho ficou em 30,55%. As influências dos fatores utilizados para este experimento são apresentadas na Figura 4.

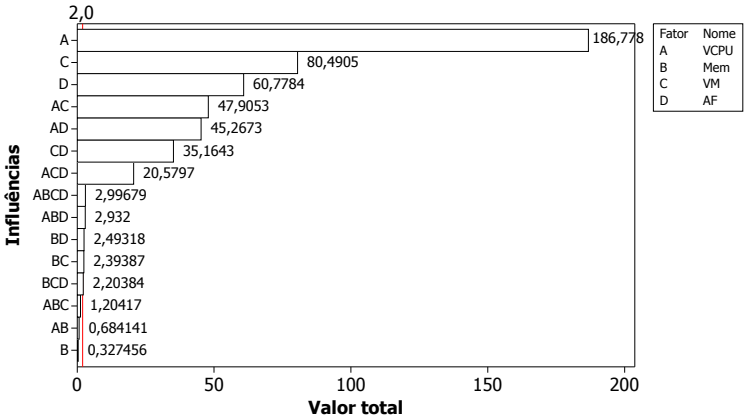

Figura 4. Gráfico de Pareto com os valores de influência absolutos para a carga de trabalho Apache com 5% de erro.

De acordo com o gráfico da Figura 4, os fatores com maior influência são o fator vCPU, VMs e *affinity* e as interações entre esses fatores, sendo as diferenças percentuais apresentadas na Tabela 6.

Tabela 6. Relação de fatores e suas influências.

Fatores mais influentes	Porcentagem
vCPU	37,94%
VMs	16,35%
affinity	12,34%
vCPU e VMs	9,73%
vCPU e *affinity*	9,19%
VM e *affinity*	7,14%
vCPU, VM e *affinity*	4,18%

4.2. *Affinity* no virtualizador KVM

Assim como no virtualizador Xen, um conjunto de experimentos foi realizado. As mesmas cargas de trabalho foram consideradas. A Figura 5 apresenta os resultados obtidos na execução da carga N-*queens*.

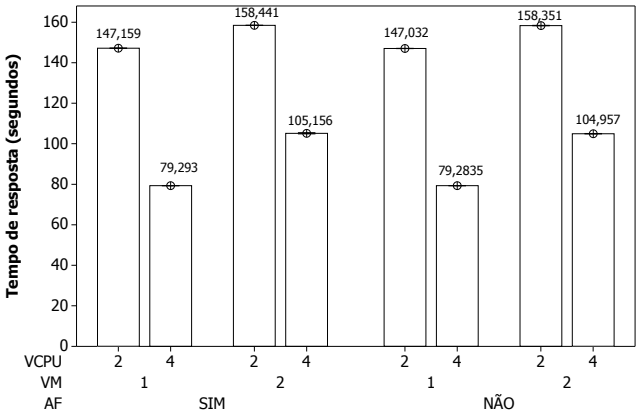

Figura 5. Tempos de resposta observados na execução do *benchmark* N-queens considerando um grau de confiança de 95%.

Nos resultados apresentados na Figura 5 não há diferenças significativas entre o uso ou não da função *affinity*. Esse fato é diretamente relacionado à arquitetura dos virtualizadores utilizados. Nesse caso não foram observados ganhos ou perdas significativas. A Figura 6 apresenta a influência dos fatores.

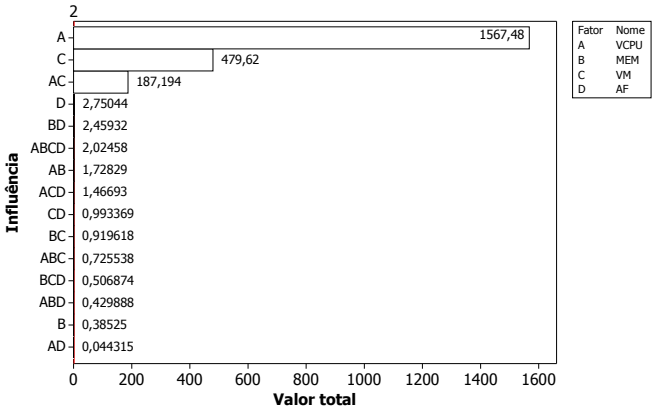

Figura 6. Gráfico de Pareto com os valores de influência absolutos para a carga de trabalho N-*queens* considerando 5% de erro.

De acordo com o gráfico da Figura 6, as maiores influências foram obtidas pelos fatores vCPU, VM e interação vCPU e VM. As porcentagens das influências são apresentados na Tabela 7.

Tabela 7. Relação de fatores e suas influências.

Fatores mais influentes	Porcentagem
vCPU	69,7%
VM	21,32%
vCPU e VM	8,32%

De acordo com a Tabela 7 e diferente dos resultados obtidos no virtualizador Xen, a função *affinity* não foi determinante na obtenção dos resultados. Como no virtualizador Xen, a carga de trabalho Apache também foi utilizada. Os resultados obtidos são apresentados na Figura 7.

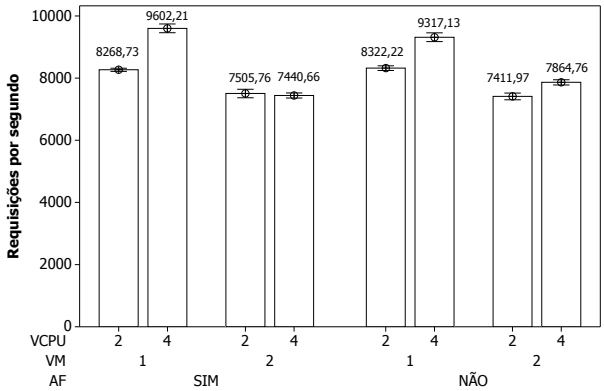

Figura 7. Tempos de resposta observados na execução do *benchmark* Apache considerando um grau de confiança de 95%.

Na carga de trabalho Apache houve diferenças de desempenho na utilização da função *affinity*. Na Figura 7 apenas os resultados com duas VMs e quatro vCPUs tiveram diferenças significativas, sendo a diferença de desempenho em torno de 5,69%. As diferenças de desempenho com o aumento de quantidade de núcleos virtuais ficou em 16,12% com a utilização do *affinity* para uma VM e não houve diferenças para duas VMs. Sem a utilização do *affinity*, o desempenho ficou em torno de 11,95% melhor com o aumento de vCPUs e em torno de 6,10% para duas VMs também com o aumento de vCPUs. O gráfico da Figura 8 apresenta as influências dos fatores.

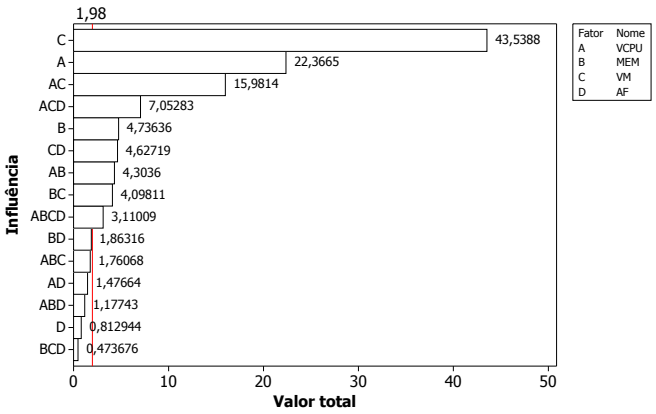

Figura 8. Gráfico de Pareto com os valores de influência absolutos para a carga de trabalho Apache.

De acordo com a Figura 8, as maiores influências são observadas nos fatores VM, vCPU, memória e interações vCPU e VM, vCPU, VM e *affinity* e VM e *affinity*, como é apresentado na Tabela 8.

Tabela 8. Influência dos Fatores.

Fatores mais influentes	Porcentagem
VM	37,11%
vCPU	19,06%
Memória	4,03%
vCPU e VM	13,62%
vCPU, VM e *affinity*	6,0%
VM e *affinity*	3,94%

5. Discussão

Os resultados apresentados sobre diferentes tipos de virtualizadores são importantes para compreender o funcionamento da infraestrutura de nuvem que, apesar de ser formada por uma grande quantidade de componentes, usa a tecnologia de virtualização para o fornecimento de serviços.

Os resultados demonstram que, dependendo do tipo de carga de serviço, um virtualizador pode ser melhor que o outro independente do *hardware* que está hospedando esses virtualizadores.

A função *affinity* foi utilizada para demonstrar a diferença de desempenho que pode ocorrer. Hoje em dia não é possível saber se um provedor usa ou não essa função. Esse fato pode conduzir a problemas de escalabilidade, por exemplo. Nos resultados

apresentados foi demonstrado que, quando a função *affinity* não é utilizada, as mudanças de desempenho são maiores. Esse fato implica na contratação do serviço da seguinte forma: se há apenas um cliente no hospedeiro, seu desempenho será mantido, no entanto, se adicionarmos mais um cliente no mesmo hospedeiro, o desempenho poderá ser afetado. Essas considerações são importantes antes de se fazer um investimento em serviços de nuvem.

Para os serviços do tipo IaaS, espera-se que seu desempenho seja sempre o melhor, independente se há compartilhamento de recursos ou não. Atualmente, o modelo de cobrança contempla o pagamento por hora, dos recursos contratados, e nem sempre os usuários tem certeza de que estão utilizando o que estão contratando.

Outro problema encontrado atualmente é a ausência de informação sobre a infraestrutura que esta sendo contratada. Por se tratar de um serviço alugado (pague pelo que foi contratado) é necessário o conhecimento sobre as características do provedor que será contratado, evitando assim despesas desnecessárias e baixa qualidade nos serviços contratados.

6. Conclusão

A computação em nuvem, apoiada pelo uso de virtualização, pode trazer benefícios na prestação de serviços de infraestrutura. Considerar todas as funcionalidades que os virtualizadores possuem pode ser um diferencial no fornecimento de serviços com qualidade.

De forma geral, os resultados foram apresentados de forma a demonstrar que a preocupação com os serviços de infraestrutura são necessários quando a qualidade de serviço é desejada. O principal atributo utilizado foi o desempenho observado nos virtualizadores Xen e KVM com a utilização da função *affinity*. É possível estender esses estudos para outros tipos de atributos, como: disponibilidade, confiabilidade entre outros. Como o foco deste trabalho foi desempenho, o mesmo foi investigado. E foi demonstrado que o virtualizador Xen é o que fornece maiores diferenças de desempenho com a utilização da função *affinity*.

Agradecimentos

Os autores agradecem ao Grupo de Sistemas Distribuídos e Programação Concorrente, pertencente ao Departamento de Sistemas de Computação do ICMC -USP, bem como à contribuição de pesquisadores do grupo que deram valiosas sugestões. Além de agradecer as agências FAPESP, CNPq e CAPES pelo apoio financeiro.

Referências

Alhamazani, K., Ranjan, R., Mitra, K., Rabhi, F., Jayaraman, P., Khan, S., Guabtni, A., and Bhatnagar, V. (2014). An overview of the commercial cloud monitoring tools: research dimensions, design issues, and state-of-the-art. *Computing*, 96:1–21.

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., Lee, G., Patterson, D. A., Rabkin, A., Stoica, I., and Zaharia, M. (2009). Above the Clouds: A Berkeley View of Cloud Computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley.

Carissimi, A. (2008). Virtualização: da teoria a soluções. In *26 Brazilian Symposium on Computer Networks and Distributed Systems, SBRC 2008, pag. 173-207. May 26th-30th, 2008, Rio de Janeiro, Brazil*.

Chierici, A. and Veraldi, R. (2010). A quantitative comparison between xen and kvm. *Journal of Physics: Conference Series*, 219(4):042005.

Chieu, T. C., Mohindra, A., Karve, A. A., and Segal, A. (2009). Dynamic Scaling of Web Applications in a Virtualized Cloud Computing Environment. In *ICEBE '09: Proceedings of the 2009 IEEE International Conference on e-Business Engineering*, pages 281–286, Washington, DC, USA. IEEE Computer Society.

de Carvalho, F. L. and Bellezi, M. A. (2014). Avaliação de desempenho dos hypervisors xen e kvm utilizando administração simplificada através do libvirt. *Revista TIS*, 3(1).

Dubey, S. and Agrawal, S. (2013). Methods to Ensure Quality of Service in Cloud Computing Environment. *International Journal*, 3(6):406–411.

Gupta, A., Kalé, L. V., Milojicic, D. S., Faraboschi, P., Kaufmann, R., March, V., Gioachin, F., Suen, C. H., and Lee, B.-S. (2013). The who, what, why and how of high performance computing applications in the cloud. In *Proceedings of the 5th IEEE International Conference on Cloud Computing Technology and Science*, CloudCom '13.

Huber, N., von Quast, M., Hauck, M., and Kounev, S. (2011). Evaluating and modeling virtualization performance overhead for cloud environments. In *CLOSER*, pages 563–573.

Jain, R. (1991). *The Art of Computer Systems Performance Analysis: techniques for experimental design, measurement, simulation, and modeling*. Wiley.

Kumar, N., Chilamkurti, N., Zeadally, S., and Jeong, Y.-S. (2013). Achieving Quality of Service (QoS) Using Resource Allocation and Adaptive Scheduling in Cloud Computing with Grid Support. *The Computer Journal*, 57(2):281 – 290.

Kumar, R., Jain, K., Maharwal, H., Jain, N., and Dadhich, A. (2014). Apache cloudstack: Open source infrastructure as a service cloud computing platform. In *International Journal of advancement in Engineering technology, Management and Applied Science*, volume 1, pages 111–116.

Leite, D., Peixoto, M., Santana, M., and Santana, R. (2012). Performance Evaluation of Virtual Machine Monitors for Cloud Computing. In *Computer Systems (WSCAD-SSC), 2012 13th Symposium on*, pages 65–71. IEEE.

Manvi, S. S. and Krishna Shyam, G. (2014). Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey. *Journal of Network and Computer Applications*, 41:424–440.

Nadeem, F. and Qaiser, R. (2015). An early evaluation and comparison of three private cloud computing software platforms. *Journal of Computer Science and Technology*, 30(3):639–654.

Shapiro, S. S. and Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611.

Weingärtner, R., Bräscher, G. B., and Westphall, C. B. (2015). Cloud resource management: A survey on forecasting and profiling models. *Journal of Network and Computer Applications*, 47:99–106.

Younge, A. J., Henschel, R., Brown, J. T., von Laszewski, G., Qiu, J., and Fox, G. C. (2011). Analysis of virtualization technologies for high performance computing environments. In *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pages 9–16. IEEE.