UNIVERSIDADE DO EXTREMO SUL CATARINENSE – UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

DIEGO SCARDUELLI LANGER

COMPARAÇÃO ENTRE FERRAMENTAS DE VIRTUALIZAÇÃO: ESTUDO DE CASO UTILIZANDO BENCHMARKS

CRICIÚMA, DEZEMBRO DE 2010

DIEGO SCARDUELLI LANGER

COMPARAÇÃO ENTRE FERRAMENTAS DE VIRTUALIZAÇÃO:

ESTUDO DE CASO UTILIZANDO BENCHMARKS

Trabalho de Conclusão de Curso apresentado para obtenção do Grau de Bacharel em Ciência da Computação da Universidade do Extremo Sul Catarinense.

Orientador: Prof. MSc. Paulo João Martins
Co-Orientador: Prof. MSc. Kristian Madeira

CRICIÚMA, DEZEMBRO DE 2010

DIEGO SCARDUELLI LANGER

COMPARAÇÃO ENTRE FERRAMENTAS DE VIRTUALIZAÇÃO: ESTUDO DE CASO UTILIZANDO BENCHMARKS

Submetido ao corpo docente do Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense como um dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Profa. MSc. Ana Claudia Garcia Barbosa
Coordenadora do Curso de Ciência da Computação

Banca Examinadora:

Prof. MSc. Paulo João Martins (UNESC)
Orientador

Prof. MSc. Kristian Madeira (UNESC)
Co-Orientador

Esp. Adjano Scarmagni (UNESC)

Prof. Esp. Fabrício Giordani (UNESC)

AGRADECIMENTOS

Quero expressar todo o meu amor e carinho, a todos que tão amáveis dedicaram o seu tempo, sua compreensão e seu conhecimento, para que fosse possível eu chegar até aqui.

Em primeiro momento a *Deus* aos *Anjos* e *Santo,* pela proteção, guarda, guia e sabedoria.

A meu Orientador MSc Paulo João Martins e Co-Orientador MSc Kristian Madeira pela amizade, afinco, atenção, dedicação e experiência transmitida ao longo desse trabalho.

A meu pai Flávio Langer e minha mãe Iraci Gorete Scarduelli Langer, pela força, incentivo e exemplo de vida.

Ao meu irmão e amigos, pela paciência e compreensão.

"É durante as fases de maior adversidade que surgem as grandes oportunidades de se fazer o bem a si mesmo e aos outros".

Dalai Lama

RESUMO

A presente pesquisa fez uma comparação sobre as ferramentas de virtualização. Analisou-se que com está esta técnica pode-se integrar vários sistemas operacionais sobre um mesmo hardware e consequentemente usufruir mais do potencial dos recursos disponíveis, obtendo um ganho na flexibilidade e segurança. Em seguida, verificou-se que existe um grau de dificuldade em decidir qual ambiente a ser utilizado, ou seja, há muitos modelos, tais como: Xen, VirtualBox, VMware, KVM e Virtual PC. E para finalizar foi realizado uma análise de desempenho experimental com a técnica de benchmark entre quatro hypervisores disponíveis, o VMware Server, VirtualBox, KVM e Virtual PC. Para tal análise, foram utilizados os benchmarks PCmark04, Bonnie++ e Scimark, realizando uma série de testes replicados segundo a Fórmula do Tamanho Mínimo de uma Amostra a partir de um Intervalo de Confiança. Com o PCmark04, avaliando em um PC de 32 bits, quem teve maior destaque foi o VirtualBox. Já com um processador de 64 bits, o VMware sobressaiu nos resultados. Com o Bonnie++, o VirtualBox demonstrou um melhor desempenho segundo os resultados gerados. Ao executar o Scimark em um PC de 32 bits o melhor desempenho foi com o VMware. Entretanto, com um PC de 64 bits, o VirtualBox sobrressaiu nas avaliações. Apesar da análise de desempenho entre as ferramentas de virtualização serem equiparadas, ressalta-se a preferência pelo VirtualBox por sobressair-se nos testes, ter uma interface mais fácil para manusear e maior praticidade na aquisição e instalação.

Palavras-Chave: Sistemas Operacionais; Virtualização; VirtualBox; VMware Server; Virtual PC; KVM; Benchmark.

ABSTRACT

This research made a comparison of the various computer virtualization tools that are on the market. It was analyzed that with a technique you could integrate multiple operating systems on the same hardware and have more potential from the available resources, you could gain more in flexibility and security. It was found that there is a degree of difficulty in deciding which environment to be used, because there are so many softwares and manufacturers. Such as Xen, VirtualBox, VMware, KVM and Virtual PC. For this analysis, it was used the benchmark software PCMark04, Bonnie++ and Scimark, to perform a series of tests according to the Form of Minimum Size of a Sample from a confidence interval. A 32-bits computer with the PCMark04 was used to know which virtualization tools was better. The biggest highlight was with the VirtualBox.With a 64-bits processor, the VMware had excelled in the results. With the Bonnie++, the VirtualBox showed a better performance according to the results. When running Scimark on a 32-bit PC, the performance was better with VMware. However, with a 64-bit PC, VirtualBox did stand out during the evaluations.Although the analysis of performance between the virtualization tools are similar, it emphasizes a preference for VirtualBox because it did stand out during the tests The software have an easier interface and it's more practical to handle, to purchase and it's more easy to install.

Keywords: Operating Systems, Virtualization, VirtualBox, VMware Server;

Virtual PC, KVM; Benchmark.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE EQUAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
CPU	Central Processing Unit
E/S	Entrada/Saída
FFT	Fast Fourier Transform
FPS	Frames Per Second
HTTP	HyperText Transfer Protocol
KB/S	Kilobytes por Segundo
KVM	Kernel-based Virtual Machine
MB/S	Megabytes por segundo
MFLOPS/S	Mega Floating-point Operations Per Second
MIPS	Millions of Instrucions Per Second
MPIXELS/S	Milhões de Pixels por Segundo
PAGES/S	Pages Per Second
PC	Personal Computer
SGBD	Sistema de Gerenciamento de Banco de Dados
SO	Sistema Operacional
TI	Tecnologias da Informação
VDI	Virtual Disk Image
VMDK	Virtual Machine Disk
VMM	Virtual Machine Monitor

SUMÁRIO

1 INTRODUÇÃO

A informação tem um papel de grande relevância atualmente, visto que se aplicá-la conjuntamente com os recursos tecnológicos terar-se-á maior resultado na funcionalidade, tática, estratégica e operacional nas empresas. Vivemos em um mundo de grande competitividade, sendo assim devemos usar meios estratégicos tais como a informação para obter meios de fazer com que a empresa consiga se enquadrar no mercado e beneficiar o melhor andamento da mesma (SILVA, R; 2007).

Este momento que vivenciamos é a era digital, onde o volume de informação está crescendo rapidamente em função dos recursos tecnológicos abrirem novas possibilidades de interação entre a humanidade. Essa informação deve ser armazenada em um local apropriado, surgindo então os *data centers*.

Os *data centers*, os quais armazenam e disseminam dados nas empresas, muitas vezes apresentam máquinas que rodam diferentes sistemas operacionais, por exemplo, uma contendo o sistema de gestão e outra o *firewall*. Essa redundância de equipamento, leva os profissionais de TI a buscar formas de consolidar servidores e uma delas é através da virtualização. Um grande benefício com esta forma de consolidação é a otimização da infraestrutura de TI. A partir de uma infraestrutura virtual, é possível ter diversos servidores virtuais sobre um único servidor físico, aumentando a eficiência energética destas máquinas e simplificando a complexidade do ambiente.

Com a virtualização é possível particionar os recursos do *hardware* de modo que ele execute diversos sistemas operacionais (iguais ou diferentes) juntamente com seus aplicativos simultanemante e totalmente isolados um do outro. Em função disso é gerada uma camada de abstração, possibilitando a geração de múltiplas instâncias lógicas, cada qual

utilizando sua parte dos recursos de uma instância física, ou ainda, a geração de uma única instância lógica a partir de diversas instâncias físicas.

Segundo Nanda e Chiueh (2005) virtualização é uma técnica que disponibiliza um ou mais ambientes operacionais através da combinação ou divisão de recursos computacionais. São denominados máquinas virtuais os ambientes criados por meio dessa técnica. Esses ambientes são criados por softwares de virtualização, os quais os profissionais de TI devem escolher para o uso segundo o desempenho apropriado.

A análise de desempenho é o grupo de métodos que permite a análise temporal de um sistema, na prática é uma ferramenta para auxiliar engenheiros na procura do melhor desempenho juntamente com custos mais baixos. A avaliação de desempenho de aplicações computacionais apareceu em 1960 e é uma mistura de medição, interpretação e comunicação. Pode-se enfatizar algumas metas: comparar alternativas, definir o efeito de uma nova funcionalidade ou reconhecer o desempenho relativo entre dois sistemas.

Este trabalho tem o interesse em realizar um comparativo entre algumas ferramentas de virtualização, utilizando ferramentas de banchmark, para servir como apoio para profissionais de informática em uma situação de escolha de um ambiente virtual.

1.1 OBJETIVO GERAL

Comparação de ferramentas de virtualização por meio de benchmarks.

1.2 OBJETIVOS ESPECÍFICOS

Desta forma os objetivos específicos são:

a) descrever e aplicar as características e conceitos sobre virtualização;

b) estudar os conceitos e utilizar os recursos das ferramentas de benchmark;

c) descrever métricas e ambientes de virtualização, para realizar as comparações, por meio dos softwares de benchmark;

d) descrever alguns cenários para avaliação de softwares de virtualização.

1.3 JUSTIFICATIVA

A virtualização pode ser entendida como uma espécie de máquina virtual contendo uma cópia isolada de uma sistema físico, na qual a mesma está protegida (LAUREANO, 2006).

Segundo Laureano (2006) o seu uso pode trazer muitos benefícios como: maior praticidade e facilidade no processo de aperfeiçoamento e testes de novos sistemas operacionais; fornecer auxílio no ensino prático de sistemas operacionais e programação, onde vários sistemas podem ser executados para realizar comparações no mesmo equipamento; em um mesmo hardware, pode ser executado simultaneamente diferentes sistemas operacionais; configurações e situações diferentes do mundo real podem ser simuladas, como por exemplo, a presença de outros dispositivos de E/S.

A importância desta técnica está cada vez mais crescendo devido a utilização da mesma em data centers, segurança e consolidação de servidores.

A segurança da informação é definida como um conjunto de orientações, normas, procedimentos, políticas e outras questões com a finalidade de proteger a informação (FONTES, 2006).

A segurança tem, também, a função de aumentar o nível de segurança do negócio em relação à dependência constante de recursos de informações para que possa trabalhar (FONTES, 2006).

A utilização de benchmarks é importante pois gera uma série de testes, onde o desempenho em um ambiente virtual pode ser avaliado, baseado em métricas. Algumas utilizadas são: Teste de leitura/escrita sequencial e randômica, medir o desempenho da CPU através de uma série de operações matemáticas com números inteiros e de ponto flutuante, testes de memória, de sistema, e throughput, latência end-to-end e entre outros (ANDRADE, 2006).

O motivo da realização de um comparativo entre algumas ferramentas de virtualização, será para propor ao pessoal de TI uma metodologia de escolha e ferramentas que possam auxiliar na tomada de decisão no momento da escolha do ambiente a ser utilizado, para tanto serão realizadas algumas comparações por meio de softwares de benchmark.

1.4 ESTRUTURA DO TRABALHO

O presente trabalho de conclusão de curso constitui de seis capítulos: introdução; conceitos de virtualização e softwares de virtualização; conceitos de benchmark; trabalhos correlatos; um estudo de caso entre ferramentas de virtualização; e conclusão.

A Introdução é constituída de objetivo geral, objetivos específicos, justificativa e a estrutura do trabalho.

O Capítulo 2 trata de conceitos de virtualização e softwares de virtualização.

Conceitos sobre benchmarks e suas métricas encontram-se no capítulo 3.

No quarto capítulo são comentados alguns trabalhos dentro do tema dessa pesquisa.

O Capítulo 5 descreve um estudo de caso onde foram realizados uma série de testes por meio de ferramentas de benchmarks.

Concluindo a pesquisa, o último capítulo apresenta as conclusões obtidas com esta pesquisa e também sugere temas para trabalhos futuros.

2 AMBIENTES VIRTUALIZADOS

Não é recente a aplicação de máquinas virtuais. A técnica que vem sendo empregada a muito tempo desde o início da computação, tendo como intuito estender o multiprocessamento, multi-programação e multi-acesso, transformando os sistemas em multi-ambiente (GOLDBERG, 2003, tradução nossa).

Novas idéias e funções foram aparecendo com o passar do tempo, onde a definição de máquina virtual foi se adaptando e/ou associando a outros tipos de definições. Um exemplo que ocorreu foi em relação aos sistemas operacionais (SO), onde houve uma ligação com estas. Os primeiros sistemas eram bastante simples, monoprogramados, forneciam um acesso básico ao hardware e podiam conter alguma biblioteca para fornecer auxílio a programação. Com o passar dos anos foram adicionando várias funções e funcionalidades e as fronteiras de o que é, e o que não é, um SO passaram a ficar não tão claras. A definição de ser somente o kernel com a capacidade de agregar um grupo de ferramentas e utilitários passou a ser passado. O que confirma essa afirmação, por exemplo, é o processo em que a empresa Microsoft sofreu quando incluiu o navegador Internet Explorer no sistema (SILBERSCHATZ; GALVIN; GAGNE, 2000).

O conceito de máquina virtual consiste em um computador funcionar como se fossem vários.

A IBM considera uma máquina virtual uma total cópia da máquina subjacente em um ambiente isolado. A IBM desenvolveu seus sistemas virtuais de máquina para que uma aplicação possa rodar da mesma forma que no ambiente nativo (ROSE, 2004, tradução nossa). Na prática, máquina virtual consiste em um ambiente, onde também é muito conhecido como "sistema operacional para sistemas operacionais" ou *hypervisor*, construido por um monitor

de máquina virtual (*Virtual Machine Monitor* – VMM). Um VMM consegue gerar uma ou mais máquinas virtuais que podem operar em uma mesma máquina real (SILVA, F., 2007).

Segundo Rodrigues (2008) o processo de funcionamento de máquinas virtuais acontece por intermédio de um monitor, que tem o papel de gerenciar os ambientes virtuais que estão rodando. Por sua vez, cada máquina virtual, criada pelo monitor, trabalha sem que nenhuma afete no funcionamento da outra e que o sistema base (nativo), sobre o qual a máquina está, não seja afetado pelas máquinas virtuais.

Virtualização difere de emulação visto que um software emulador realiza uma simulação da arquitetura de hardware necessária para a execução do sistema hóspede, ou seja, as instruções do Sistema Operacional (SO) emulado são traduzidas para que sejam executadas no sistema operacional original. São utilizados emuladores com muita freqüência para realizar testes de funcionamento de programas escritos para arquiteturas diferentes das que são utilizadas pelo emulador em sua execução (ANDRADE, 2006).

Um exemplo de emulador de processador muito conhecido é o Bochs, que trabalha com base em processadores x86. O Bochs não possui um desempenho muito bom em função de suas instruções serem interpretadas e executadas uma por uma. Utiliza-se o Bochs frequentemente como simulador, onde é mais apropriado em projetos de software *low level* (como SO) ou de simulação de novas arquiteturas de hardware. Um tipo de emulação que não aplica a tradução instrução-por-instrução é a tradução de blocos de intruções para posterior execução nativa. Nessa emulação, blocos de códigos já traduzidos não necessitam uma retradução. Essa técnica é utilizada em emuladores de SO como o Microsoft Virtual PC e QEMU. Outro emulador de plataforma que utiliza esse funcionamento é o UAE Amiga Emulator. Também aplicam essa técnica os populares emuladores de consoles (video games), como o ePSXe, o SNES9X e o ZSNES (ANDRADE, 2006).

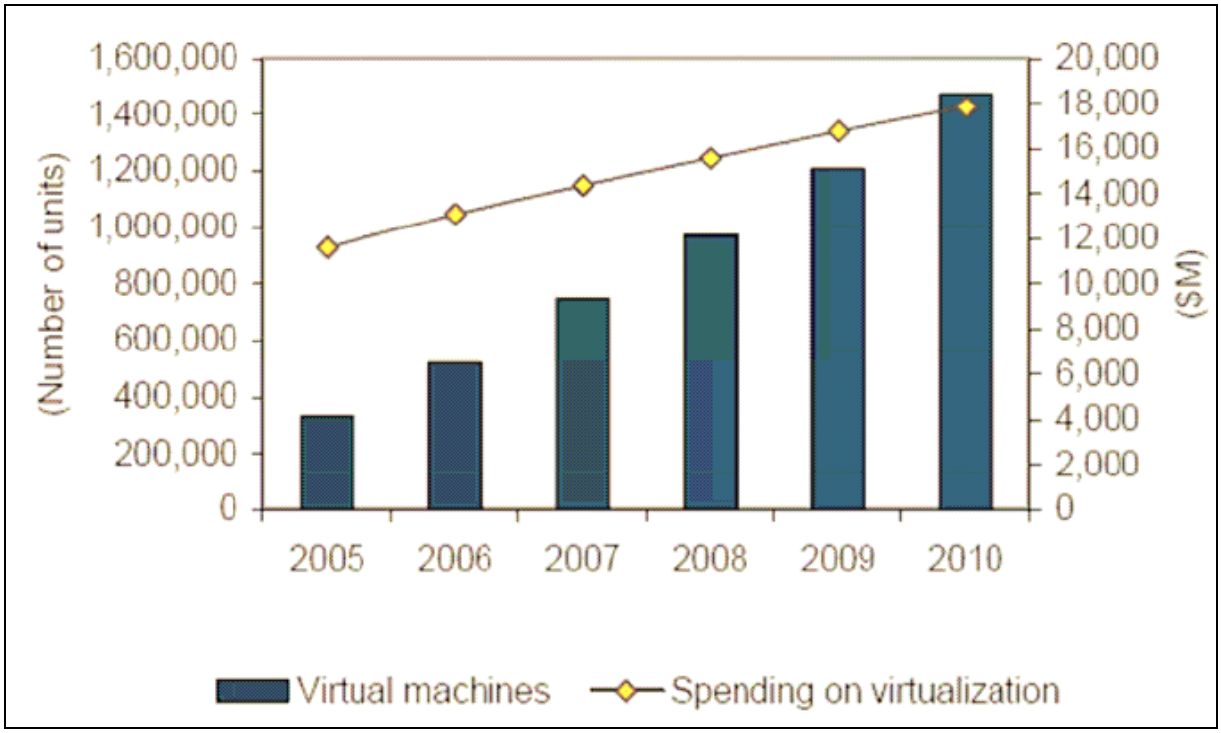

Figura 1. Custo com virtualização
Fonte: HUMPHREYS (2006)

Segundo Humphreys (2006) conforme a Figura 1, a tendência do rápido crescimento das plataformas de hardware para virtualização são de 330.000 unidades em 2005 sobre a 1.400,000 unidades em 2010, onde estima-se que haverá $18 bilhões em despesas com hardware, software e serviços para fins de virtualização dos ambientes.

Uma justificativa para essa forte tendência de uso de máquinas virtuais, é a capacidade de executar vários SO e aplicações de variadas plataformas sobre uma mesma máquina física paralelamente. Pode-se aproveitar os recursos computacionais novos ou existentes de uma melhor maneira, diminuindo a freqüente ociosidade desses recursos em momentos do dia. (SILVA, F; 2007).

2.1 FORMAS DE VIRTUALIZAÇÃO

Segundo Laureano (2006) os softwares são capazes de fazer os recursos transmitir a ilusão de serem outros recursos, ou seja uma falsa camuflagem, esse fenômeno é denominado virtualização. Ela nada mais é do que a interposição do software (Máquina Virtual) em várias camadas do sistema. Ela consiste em dividir os recursos de hardware em múltiplos ambientes de execução.

A virtualização pode ser implementada de três formas: do sistema operacional, do hardware e de linguagens de programação.

2.1.1 Virtualização de Sistemas Operacionais

Nessa forma de virtualização é realizada uma exportação do sistema operacional, o qual é uma abstração de um tipo de sistema exclusivo. Assim sendo, a máquina virtual executa uma aplicação, ou várias, pertinentes a um sistema operacional exclusivo. O FreeBSD ou o User – Mode Linux aplicam essa forma de virtualização (HANSEN; SCHAEFFER, 2010).

2.1.2 Virtualização de Hardware

Este modelo parte do princípio de gerar uma máquina virtual, por meio de um software, capaz de emular o hardware de uma máquina. Dessa forma, é gerado um ambiente de SO exclusivo, separado logicamente do servidor host. O VMware, Virtual PC e o VirtualBox aplicam essa forma de virtualização (HANSEN; SCHAEFFER, 2010).

2.1.3 Virtualização de Linguagens de Programação

Nesta técnica, a camada de virtualização gera uma aplicação no ápice do sistema operacional. As máquinas virtuais dessa modalidade são concebidas para computadores fictícios construídos para um objetivo específico. A camada realiza a tarefa de exportar uma abstração para que aplicativos escritos para essa forma de virtualização rodem. Java e Smalltalk aplicam essa forma de virtualização (HANSEN; SCHAEFFER, 2010).

2.2 TÉCNICAS DE VIRTUALIZAÇÃO

São utilizadas uma variedade de técnicas na virtualização. Serão descritas algumas dessas técnicas nos próximos tópicos, tais como: virtualização completa, paravirtualização e recompilação dinâmica.

2.2.1 Virtualização completa

Na técnica de virtualização completa, o Gerenciador de Máquinas Virtuais realiza um intermédio entre o hardware e as máquinas virtuais, dessa forma ocorre uma simulação completa do hardware da máquina permitindo rodar quaisquer sistema operacional sem realizar alterações (OKANO; ANDRADE, 2008).

A virtualização completa se comparada com a emulação de hardware apresenta um nível maior de eficácia, em função de não precisar representar os estados de execução do hardware, proporcionando mais facilidade na instalação e configuração dos sistemas virtualizados por meio da simulação de dispositivos padrões de mercado (OKANO; ANDRADE, 2008).

Nesse tipo de técnica, a idéia é montar uma arquitetura de hardware na máquina host para que se execute tanto SO quanto aplicativos juntamente. Desse modo, um sistema *guest*[1] é capaz de executar suas instruções no hardware bruto de forma direta. Existe uma camada de software denominada hipervisor, que realiza um papel de vigilante, onde fornece uma ilusão a cada sistema guest de possuir sua cópia de hardware exclusiva (INOCENTE, 2009).

A Figura 2, apresenta o hardware para cada sistema operacional instalado sendo o hardware físico, exercendo controle apenas sobre o hipervisor (INOCENTE, 2009).

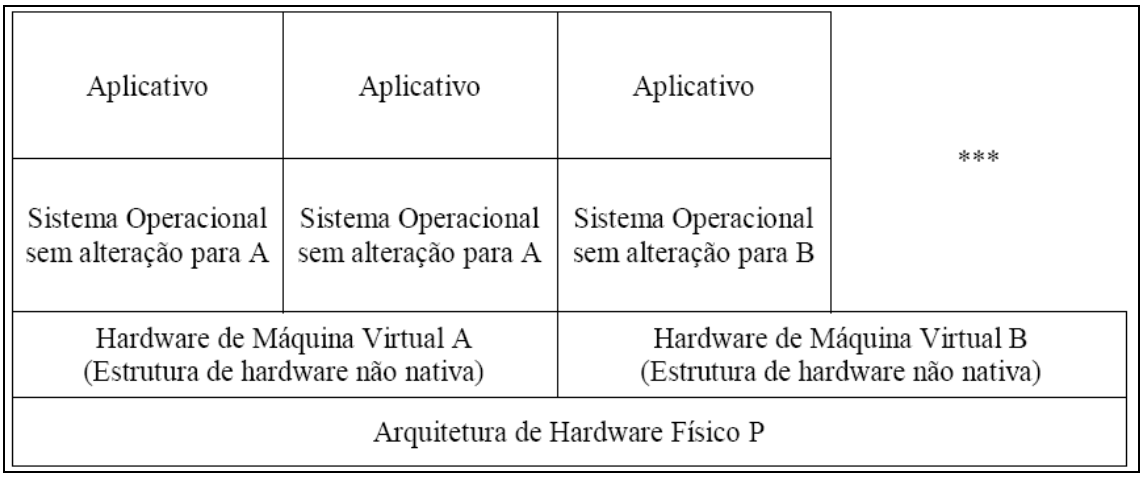

Figura 2. Modelo de Virtualização via emuladores
Fonte : INOCENTE, E. (2009)

Entre as máquinas virtuais que aplicam a virtualização total temos o VMware, o VirtualBox e o KVM.

[1] "Máquina Virtual executada no Sistema host. Cada sistema Guest tem a ilusão de ter uma máquina física exclusiva para ele" (GONÇALVES; VAHL JUNIOR, 2010, p. 1).

2.2.1.1 VMware

O VMware foi desenvolvido em 1999 e está incluido na lista dos aplicativos mais conhecidos de virtualização para plataforma x86. Considerado o pioneiro na solução de virtualização para a arquitetura x86, prove para um SO convidado uma implementação total desta arquitetura. A empresa que desenvolve o aplicativo é a VMware Inc. e contém vários tipos de aplicativos para virtualização. A empresa dispõe algumas versões de seus aplicativos sem custo de licença, onde somente algumas funções estão disponíveis para os usuários (QUEVEDO JUNIOR, 2008).

O monitor, nesse tipo de software de virtualização, tem a função de emular certas instruções, chamadas sensíveis, com a finalidade de representar para cada máquina o seu processador virtual da melhor forma. O mecanismo chamado trap, do processador, que permite executar as intruções sensíveis. Essas instruções não são muitas vezes capturadas pelos processadores de arquitetura x86. Quando isso ocorre, o VMware aplica a reescrita binária, que é uma técnica que realiza uma análise prévia das instruções antes que elas sejam executadas. Quando ocorre essa situação, o monitor realiza a troca das instruções sensíveis por pontos de parada, os quais podem ser capturados e executados pelo processador (QUEVEDO JUNIOR, 2008).

O gerenciamento de memória no VMware é realizado pelo sistema hóspede, onde a máquina virtual deixa alocado para seu uso particular uma região da memória para que não haja o risco do sistema hóspede e o sistema hospedeiro tentarem utilizar o mesmo endereço. Nessa memória, previamente alocada, que o sistema hóspede irá trabalhar (QUEVEDO JUNIOR, 2008).

A meta da VMware Inc. era dispor aos computadores em geral a tecnologia de máquinas virtuais presente nos mainframes. Dentre os principais programas para uso comercial destaca-se o VMware ESX Server (CASTRO, 2006).

2.2.1.1.1 VMware ESX Server

Um componente chave do VMware ESX Server é o *hypervisor*, o qual se encarrega de fazer a virtualização, as partições e o gerenciamento do hardware. A Figura 3 ilustra a arquitetura básica do VMware ESX Server (CASTRO, 2006).

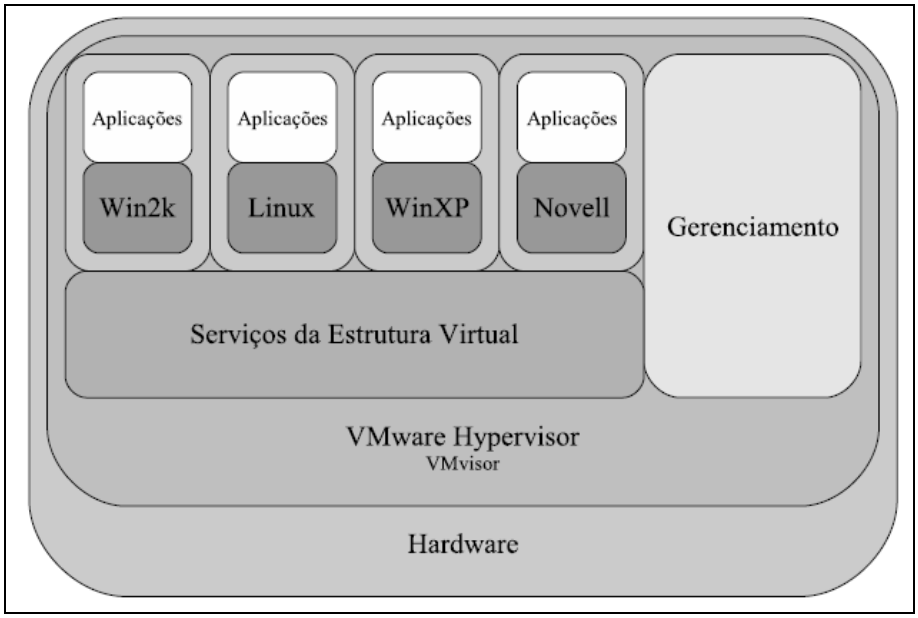

Figura 3. Arquitetura do VMware ESC Server
Fonte: CASTRO, A. (2006)

O VMware ESX Server dispõe uma interface de grande utilidade denominada *Hypercall* (VMware Hypercall Interface), a qual pode ser utilizada por SOs, ao efetuar determinadas alterações nestes. Esse processo está presente na técnica de paravirtualização, que será mencionada adiante (CASTRO, 2006).

2.2.1.2 VirtualBox

Um outro software de virtualização muito popular. Ele é muito fácil de usar e gerenciar e tem uma versão de código aberto (*VirtualBox Open Source Edition*). Possui uma documentação de fácil entendimento, do mesmo modo as suas ferramentas, as quais são frequentemente modificadas e atualizadas, permitindo gerenciar suas máquinas virtuais com eficácia. Dentre as ferramentas mais populares destaca-se o VirtualBox Gui, que é uma interface gráfica administrativa (SANTOS, 2008).

Uma das característica marcante é o suporte aos tipos de arquivos de armazenamento muito optados atualmente: o VDI e VMDK. Este último é utilizado pelo VMware (SANTOS, 2008).

É uma máquina virtual do tipo II e torna-se um processo de Sistema Operacional host para poder executar, podendo ser Linux, Windows 32 e 64 bits, ou Mac OS X. Dentre os sistemas convidados que suporta, estão o DOS, FreeBSD e Linux (SILVA, R; 2007).

O software funciona baseado na técnica de virtualização total, aplicando a emulação em alguns principais componentes de hardware. Consequentemente, não é preciso alterar SO convidados para rodarem em uma máquina virtual (SILVA, R; 2007).

Este software separa segmentos de código dos sistemas virtuais, onde tenta executá-los direto no processador. Tenta colocar o sistema operacional do ring 0 para o nível de ring 1, para rodar instruções privilegiadas. Difilmente é aplicado o nível de ring 1 na arquitetura x86. Se houver falhas no processo, o VirtualBox também pode aplicar a técnica de recompilação dinâmica. Seu recompilador funciona com base no *open source* QEMU (Descrito num tópico adiante) (VIRTUALBOX, 2010).

2.2.1.3 KVM

Kernel-based Virtual Machine (KVM) é um projeto incluso na última geração de ferramentas de virtualização de código aberto. A intenção era gerar um hypervisor que funcionasse de acordo com a experiência de tecnologias anteriores e aplique no hardware robusto da atualidade (REDHAT, 2010, tradução nossa).

É implementada como uma modificação do núcleo do Linux que consequentemente lhe faz virar um hypervisor ao incluir um módulo extra (REDHAT, 2010, tradução nossa).

No funcionamento do KVM, todo hóspede é executado no ambiente de usuários do sistema hospedeiro. Este procedimento faz com que toda instância (um kernel hóspede juntamente com seu ambiente de usuários) funcione como um processo padrão para o kernel hospedeiro no qual atua (MATHEWS; DOW, 2009).

O modulo KVM aplica um controlador de hardware contendo novas propriedades para propor hardware virtual para as instâncias hóspedes por meio do caminho /dev/kvm nos diretórios do hipervisor. Os hóspedes KVM entram em contato com o hardware virtual por meio de um processo QEMU alterado. Ao executar sobre hardware atual com extensões voltadas para virtualização, pode-se obter hóspedes Linux (32 bits e 64) e Windows (32 bits) (MATHEWS; DOW, 2009).

2.2.2 Paravirtualização

A paravirtualização foi criada com o propósito de tentar oferecer um melhor desempenho na execução de máquinas virtuais na arquitetura x86,diferente das utilizadas até o momento. A arquitetura x86 não possui suporte a virtualização nativamente. A idéia seria

oferecer, ao sistema operacional a ser utilizado, outro tipo de ambiente onde tenha um conjunto diferente de instruções, permitindo que diminua o nível de carga de tratamento de instruções sensíveis (CASTRO, 2006).

Esta técnica de virtualização dispõe para as máquinas virtuais uma *Application Programming Interface* (API), a qual assemelha-se com o hardware real. A paravirtualização exige modificações no sistema operacional para que o mesmo possa funcionar. Assim sendo, o sistema operacional que executa dentro da máquina virtual tem a falsa impressão de estar rodando sobre o hardware nativo da máquina e não em um simulado (SAFT, 2008).

Em relação a virtualização completa, a paravirtualização apresenta um nível de desempenho superior devido ao fato de que neste modelo são utilizados drivers reais para o funcionamento enquanto na virtualização completa são utilizados os drivers emulados (SAFT, 2008).

Duas máquinas que utilizam esse tipo de técnica são o Denali e o Xen. Apesar do fato de ser necessário realizar alterações no SO para que o mesmo possa executar em um ambiente paravirtualizado, esse tipo de técnica possui melhor performance e uma VMM mais simples do que as outras técnicas existentes (CASTRO, 2006).

Entre as máquinas virtuais que aplicam a paravirtualização temos o Xen.

2.2.2.1 Xen

Segundo Linux-magazine(2009):

> O Xen é um hypervisor que "roda" diretamente entre o sistema operacional real e o hardware. Outros sistemas operacionais, virtualizados sobre o sistema operacional real, "conversam" como esse hypervisor quando precisam de acesso ao hardware do sistema, que intermedia esse acesso. O projeto foi originalmente desenvolvido como um projeto de pesquisa na Universidade de Cambridge, liderado por Ian Pratt, e conta com suporte da Red Hat e da Novell, além de ser utilizado pela Sun Microsystems para virtualização de servidores.

Assim como o VMware e o Virtual PC, o Xen possibilita executar vários SO em máquinas virtuais sobre um mesmo servidor. É semelhante ao VMware, porém sua implementação ocorre de uma outra maneira (WINXLINUX, 2009).

O Xen tem seu funcionamento baseado na paravirtualização, onde o SO tem a falsa ideia de estar executando diretamente sobre o hardware. Ele realiza o papel de analisar e encaminhar as requisições feitas pela máquina virtual para o sistema principal. Ao contrário de um emulador, encaminha as instruções sem interpretar, consequentemente perdendo um pouco de desempenho (KLEIN, 2008).

Geralmente os SO operam em nível 0 em função de poder executar instruções privilegiadas (NEVES; PACHECO, 2010). Ele executa diretamente em nível 0 em função de ele mesmo realizar alterações no kernel do nível 0 para o nível 1 (ROSSI, 2008).

Este software assemelhasse com um microkernel, que caracteriza-se por poder haver diversos SO, que fazem o trabalho de gerenciar todo o acesso à memória e dispositivos (POTRICH, 2008). Ele realiza uma troca com o hardware, na qual fornece uma camada denominada hypervisor, possibilitando ao mesmo portar seus SO e em troca disso tem a capacidade de rodar uma variedade de instâncias de SO. A Figura 4 mostra a arquitetura (ROSSI, 2008).

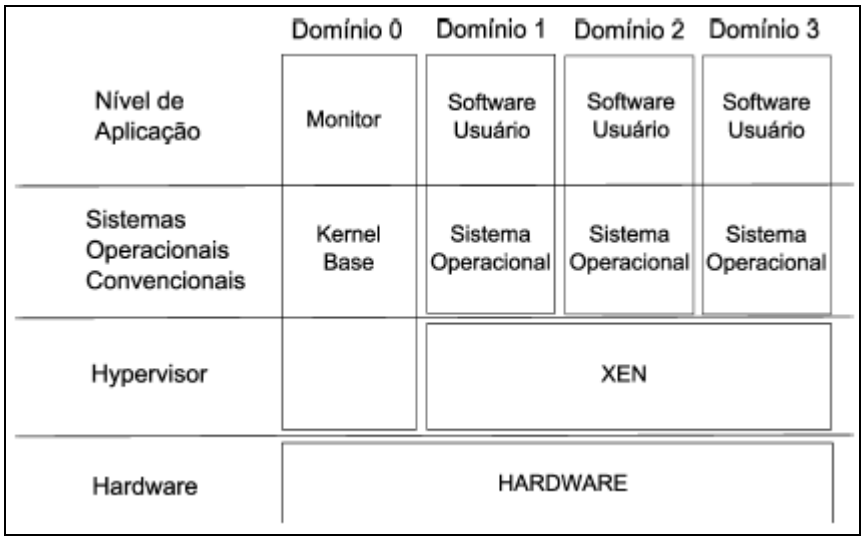

Figura 4. Estrutura do Xen
Fonte: ROSSI, F. (2008)

A Figura 4, mostra os quatro níveis do Xen. No nível 1 (nível mais superior), estão as aplicações que executam dentro das máquinas virtuais e támbem alguns softwares com o papel de monitorar o funcionamento do Xen. No nível 2 está o sistema operacional hospedeiro com seu kernel alterado, possibilitando então a virtualização, e os SO convidados. No nível 3 está o hypervisor (uma pequena camada de software que gerencia as chamadas de sistema que ocorrem entre as máquinas virtuais e o hardware). E no último nível, está o hardware (ROSSI, 2008).

2.2.3 Recompilação dinâmica

Esta técnica realiza compilação, no momento de execução, de forma que o código obtido represente instruções em linguagem de máquina. Também é utilizada para fins de representação portátil de um aplicativo, como os bytecodes de Java (LAUREANO, 2006).

Segundo Laureano (2006) a recompilação dinâmica possui sete etapas:

a) agrupamento de bits: um aplicativo ao ser compilado e convertido em um executável, através da linkedição, irá conter informações de manipulação dos registradores e memória. Um emulador pode utilizar as informações do executável e aplicar técnicas heurísiticas para obter os grupos de bits do executável e reorganizá-los;

b) desmontagem (*disassembling*): os bits são desagrupados e colocados em uma sequência de intruções e operadores em pares;

c) geração intermediária do código: as instruções são convertidas para um formato de máquina;

d) decompilação: o formato obtido é convertido em uma linguagem de alto nível;

e) compilação: o código obtido é convertido para a nova plataforma;

f) montagem (*assembling*): as instruções geradas na etapa anterior são linkeditados para gerar um novo executável;

g) armazenagem dos bits: são unificados para a obtenção de um novo executável.

O QEMU é um emulador que aplica a recompilação dinâmica para elevar o nível de desempenho (BELLARD, 2005, tradução nossa). O VMware Workstation também funciona com base nesta técnica, onde recompila pedaços de código, sendo possível que seja executado nativamente uma certa porcentagem do mesmo. Dessa forma, somente instruções que não rodem diretamente passam por recompilação (LAUREANO, 2006).

Houve um projeto da DEC (DIGITAL FX! 32) que mesclava emulação e tradução binária para rodar em microcomputadores Alpha aplicações de 32-bit que funcionavam no SO Windows NT 4.0. Paralelamente, o emulador conseguia obter um perfil de execução, no momento em que as aplicações rodavam, que seria utilizado por um tradutor binário para traduzir instruções das aplicações que rodaram diretamente no código de máquina do processador Alpha (CHERNOFF; HOOKWAY, 2010, tradução nossa).

O interpretador de linguagem java conhecido como Sun *hotspot*, também aplica a recompilação dinâmica em seu funcionamento. Este interpretador faz com que as instruções pertencentes a máquina virtual e localizadas em classes Java sejam traduzidas e possam rodar no *hardware* que se encontram (ANDRADE, 2009).

Entre os softwares que aplicam a recompilação dinâmica temos o Qemu e o Virtual PC.

2.2.3.1 Qemu

O Qemu é um emulador popular e de código aberto que apresenta uma ótima performance por meio da tradução dinâmica. Consegue emular um SO inteiro, inclusive os componentes de hardware. Ao ver um fragmento de código, o transforma em código da máquina hospedeira. O objetivo é transformar toda instrução em outras novas de fácil interpretação (CAMPOS; KASSALIAS, 2006).

É possível rodar o Qemu em sistemas x86, x86_64 e PowerPC e também emular esses sistemas no mesmo. Um SO não precisa alterações para executar na ferramenta, pois o mesmo emula o hardware por completo. Por dispensar alterações e/ou módulos no sistema principal, o emulador perde em desempenho ao ser comparado com sistemas que não dispensam tais alterações (CASTRO, 2006).

O Software opera em 2 modalidades, na emulação completa e emulação no modo usuário. No primeira, é realizada uma emulação de um sistema inteiro, inclusive o precessador e outros dispositivos e hardware. Nesse modo, o Qemu consegue rodar vários SO diferentes. A segunda modalidade, modo usuário, é possível executar só um sistema hospedeiro Linux, podendo executar processos compilados para rodar em um tipo de CPU em

outra CPU. É possível rodar programas do SO compilados para um processador em um outro, como de um processador Pentium em um Duron (SANTOS, 2005).

2.2.3.2 Virtual PC

É um software de virtualização que funciona de acordo com uma tecnologia que a Microsoft adquiriu, em 2003, da empresa Connectix, que iniciou essa fase de criação de máquinas virtuais para computadores pessoais (SANTOS, 2005).

A Figura 5, mostra como é possível, através do Virtual PC, construir um computador virtual que contenha seu hardware virtual exclusivo até mesmo um processador só para si. Esse computador é criado em um ambiente auto-suficiente e de uma forma que permaneça independente dos demais computadores virtuais (SANTOS, 2005).

Figura 5. Arquitetura do Virtual PC
Fonte: SANTOS, G. (2005)

Segundo Davis (2009, tradução nossa), o Virtual PC foi projetado para trabalhar com SO desktop e aplicativos. Algumas ocasiões favoráveis para seu uso são:

a) suporte à execução de aplicativos de desktop legado: se na máquina estiver instalado o windows XP e precisar rodar algum aplicativo para windows 98, o mesmo pode ser rodado no devido SO em uma máquina virtual;

b) help desk: técnicos podem optar por utilizar este software para criar ambientes de trabalho, diminuindo a necessidade de mais equipamento. A partir disso, os ambientes de trabalhos de clientes podem dobrar, facilitando então a solução de problemas atuais;

c) testes de aplicativos: é ideal para desenvolvedores, o qual possibilita que sejam testados softwares desktop em diversos SO em máquinas virtuais;

d) treinamento: para trainers, facilita muito o trabalho pois permite configurar programas de treinamento em máquinas virtuais. Assim, pode-se proporcionar uma diversidade de classes diferentes sem que seja cada uma utilizada em máquina diferente. Pode-se, inclusive, adaptar as máquinas de forma que alterações feitas em outra aula sejam descartadas.

2.3 APLICAÇÃO DAS MÁQUINAS VIRTUAIS

A seguir segue algumas típicas aplicações de máquinas virtuais:

a) consolidação de servidores: praticamente uma das aplicações mais utilizadas nas empresas. Dessa forma, diminui-se a variedade de máquinas, onde cada uma possui seu sistema operacional, centralizando em apenas uma. Dentre as vantagens de consolidar servidores destaca-se o gerenciamento centralizado e a diminuição de ociosidade de processamento (SANTOS, 2005);

b) ambientes de alta disponibilidade: apesar de haver um certo custo de hardware e software para aplicar, a ideia é oferecer um ambiente que mantenha o tempo de *downtime* (ociosidade devido a problemas técnicos ou manutenção preventiva) próximo a zero. Um caso muito comum é o uso de cluster em máquinas virtuais localizadas em diferentes servidores físicos. Assim, caso ocorra alguma falha em uma das máquinas, a outra assume a responsabilidade de manter as máquinas virtuais funcionando para que não haja indisponibilidade (SANTOS, 2005);

c) isolamento: conseguem isolar o que rodam, gerando um ambiente que mantenha a propagação de erros e anomalia de softwares e diminuir a possibilidade de uma aplicação afetar o trabalho de outras aplicações que estejam executando no mesmo equipamento (SOUZA, 2006);

d) segurança: podem criar um ambiente separado para executar aplicações não seguras (SOUZA, 2006);

e) hardware virtual: a virtualização é capaz de proporcionar a ilusão de um hardware que não exista fisicamente como discos SCSI virtuais, placas de rede virtuais, entre outros (SANTOS, 2005);

f) d*ebugging*: um desenvolvedor pode ter a sua disposição *device drivers* ou um sistema operacional que gerencie uma aplicação rodando em um hardware (SANTOS, 2005).

2.4 VANTAGENS DA VIRTUALIZAÇÃO

Dentre elas existem as seguintes:

a) permite diminuir os gastos com hardware, pois faz um melhor aproveitamento da capacidade de processamento do mesmo com vários SO rodando em uma mesma máquina

física. Dessa forma, diminui-se a quantidade de servidores e os gastos para manutenção (INOCENTE, 2009);

b) quem trabalha com programação de sistemas ganha muito com o uso de hipervisores, pois para testar uma aplicação em um sistema operacional diferente, não é preciso reiniciar a máquina física (INOCENTE, 2009);

c) incompatibilidade entre aplicações em uma mesma máquina física ocorrem frequentemente. Com ambientes virtualizados, é possível isolar cada aplicação em uma máquina virtual exclusiva (SANTOS, 2005);

d) simula a existência de um hardware que não existe fisicamente, como por exemplo discos SCSI e processadores (SANTOS, 2005);

e) instalações ficam mais facéis, os backups mais simples de realizar e mais praticidade em aperfeiçoar e testar novos SO (HANSEN; SCHAEFFER, 2010).

2.5 DESVANTAGENS DA VIRTUALIZAÇÃO

Algumas desvantangens são:

a) se o servidor consolidado apresentar uma falha e parar seu funcionamento, todos aplicativos e serviços que estavam rodando irão parar também. Os servidores não centralizados possuem vantagem nesse aspecto (GOMES; FRACALOSSI, 2007);

b) uma máquina perde um pouco em desempenho devido a estar compartilhando o hardware com a máquina hospede. Porém a perda de desempenho é mínima em relação a vantagem de uma máquina apenas fazer o serviço de muitas (GOMES; FRACALOSSI, 2007);

c) Laureano (2006) destaca:

> A principal desvantagem do uso de máquinas virtuais é o custo adicional de execução dos processos em comparação com a máquina real. Esse custo é muito variável, podendo chegar a 50% ou mais em plataformas sem suporte de hardware à

virtualização, como os PCs de plataforma Intel. Esse problema inexiste em ambientes de hardware com suporte à virtualização, como é o caso de mainframes.

3 CONCEITO DE MEDIÇÃO

Compreender o estado no qual um sistema se encontra, exige estar ciente de algumas definições de extrema importância. A seguir elas serão descritas e mais adiante serão aplicadas (INSTRON, 2010).

Segundo Instron (2010) para avaliar qualidade em uma pesquisa, três aspectos relacionados aos dados de medição devem ser compreendidos:

a) exatidão: representa a proximidade de uma medição em relação ao seu valor real;

b) precisão: um ponto pode ser acertado diversas vezes sobre um dado limite de erro. É uma medida que representa os diversos dados alcançados em uma medição;

c) resolução: a menor alteração do dado que um dispositivo pode discernir.

Ao efetuar medições, vários fatores podem ocasionar erros nos resultados obtidos, porém, tais erros podem ser delimitados. Assim, ao se estar ciente da sua existência, pode-se obter dados aceitáveis decorrentes de um sistema de medição, entretanto, a ordem de grandeza, também a natureza, do erro devem ser conhecidas (FRANCO; OLIVEIRA, 2010).

Os erros de medição classificam-se nos seguintes grupos (ARAÚJO, 2010):

a) aleatório: muda de uma maneira inesparada ao realizar diversas medições com a mesma dimensão;

b) sistemático: invariável ou que varia de uma maneira previsível ao realizar diversar medições com a mesma dimensão. Esse tipo de erro e suas causas podem, ou não, ser conhecidos.

3.1 INTERVALOS DE CONFIANÇA

A média dos resultados, ao se encaixar em um certo intervalo, apresenta um certo grau de confiança. Caso contrário, uma amostra, com tal média, dificilmente seria observada sem tal aspecto (RAMOS, 2008).

Segundo Ramos (2008) tendo uma função densidade de probabilidade f(x) juntamente com uma variável aleatória x, defini-se a equação 1:

$$F(b) - F(a) = P(a \leq X \leq b) = \int_a^b f(x)dx \tag{1}$$

Onde:

b = intervalo; a = intervalo.

Conforme Bernardinelli (2010) uma váriavel aleatória contínua X apresenta distribuição normal tendo μ e σ^2 como parâmetros, que representam média e variância da distribuição, $-\infty < \mu < +\infty$ e $0 < \sigma^2 < +\infty$, caso a função densidade de probabilidade seje:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{2}$$

Onde:

μ = valor esperado (média) de X com $-\infty < \mu < \infty$;

σ^2 = a variância de X com $\sigma^2 \geq 0$.

Existe a possibilidade da distribuição normal se encontrar na forma reduzida caso $\mu = 0$ e $\sigma = 1$, dessa forma, aplicando os cálculos, define-se a equação 3 da seguinte maneira (RAMOS, 2008):

$$f(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{z^2}{2}\right) \tag{3}$$

Onde:

$z = (x - \mu) / \sigma$;

z = valor padronizado de x (número de desvios padrão com relação à média);

x = valor da V. A. Normal X;

σ = desvio padrão da V. A. Normal X;

μ = média da V. A. Normal X.

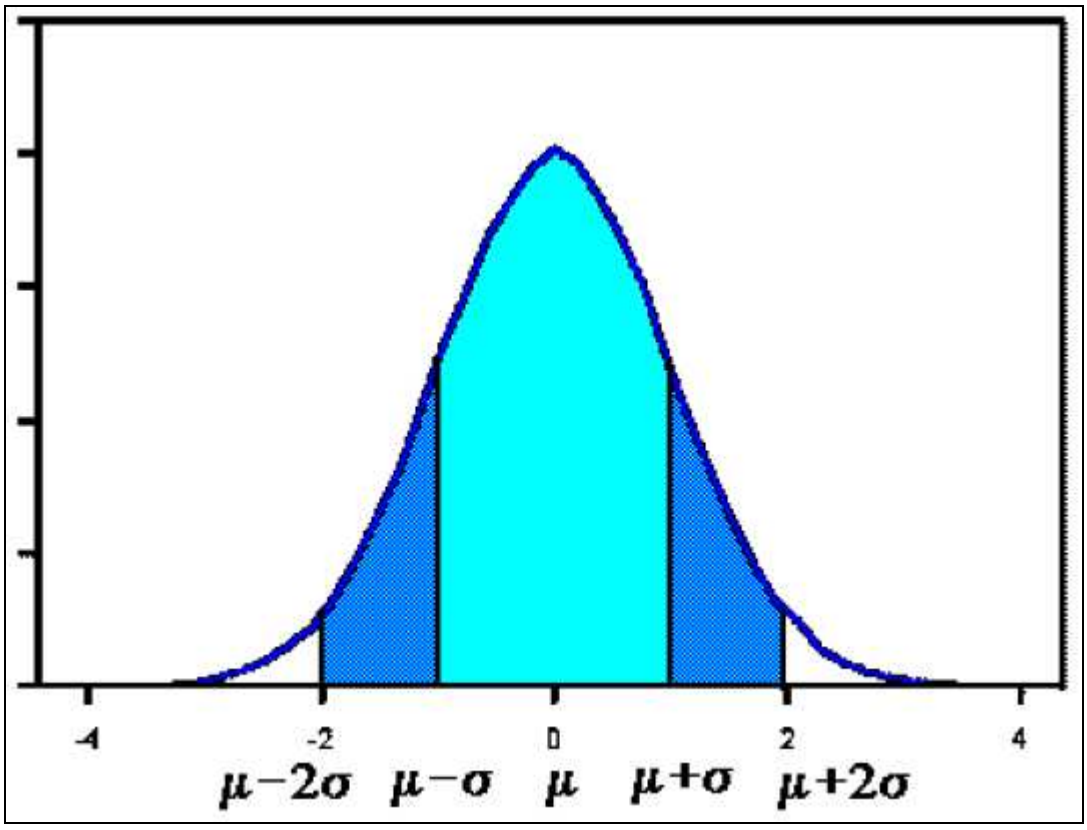

Figura 6. Função distribuição de probabilidade normal reduzida
Fonte: RAMOS, N. (2008)

Em relação ao modelo normal, o cálculo das probabilidades é realizado com o auxílio de tabelas, e para poupar a multiplicação de tabelas para todo par de valores (μ , σ^2),

aplica-se uma transformação de forma que encaminhe sempre a conter uma varíavel de parâmetros (0, 1), ou seja, $\mu = 0$ (sendo a média) e $\sigma^2 = 1$ (como a variância) (BERNARDINELLI, 2010).

Assim, caso $X \sim N(\mu,\sigma^2)$, surge outra nova variável $Z = (X - \mu) / \sigma$, para qual explana-se que $\mu(Z) = 0$ e $\sigma^2(Z) = 1$. Então, $Z \sim N(\mu,\sigma^2)$ é chamada de Normal Padrão ou Normal Reduzida (BERNARDINELLI, 2010).

Conforme Bernardinelli (2010) o calculo de $P(a \leq X \leq b)$, requer a transformação:

$$P(a \leq X \leq b) = P((a-\mu)/\sigma \leq Y \leq (b-\mu)/\sigma)$$

São tabelados os valores em $P(0 \leq Z \leq z)$, $z \geq 0$.

O teorema central do limite assegura que a distribuição da média amostral está propensa à uma distribuição normal conforme o volume da amostra n tende para infinito. O encaminhamento para a normalidade fica mais ágil caso a distribuição dos dados é de forma simétrica; caso a distribuição ser assimétrica ou bimodal, a convergência demora mais tempo (CARBONARI et al, 2010).

Segundo Piana, Machado e Selau (2010) a técnica de intervalo de confiança pode ser aplicada para estimar a média μ de uma população X. Por meio de um estimador $\overline{X}$, se $X \sim N(\mu, \sigma^2)$, então, $\overline{X} \sim N(\mu, \sigma^2/n)$. Em relação a $\overline{X}$, tem-se:

$$Z = \frac{\overline{X}-\mu}{\sigma_{\overline{X}}}()/^{\sigma}\overline{X} = \frac{(\overline{X}-\mu)}{\sqrt{V(\overline{X})}} = \frac{(\overline{X}-\mu)}{\sqrt{\frac{\sigma^2}{n}}} = \frac{(\overline{X}-\mu)}{\frac{\sigma}{\sqrt{n}}}$$

Conforme a equação 4, Z possui distribuição normal, média zero e variância um:

$$Z = \frac{\overline{X}-\mu}{\frac{\sigma}{\sqrt{n}}} \sim N(0,1) \qquad (4)$$

Onde:

$\overline{X}$ = é a média amostral;

μ = média de uma certa população Normal;

$\frac{\sigma}{\sqrt{n}}$ = desvio padrão da média amostral.

Para obter um intervalo de confiança em relação a média da população, μ deve ser isolado procedendo da seguinte maneira (PIANA; MACHADO; SELAU, 2010):

$$P\left(-Z_{\alpha/2} < \frac{\overline{X}-\mu}{\frac{\sigma}{\sqrt{n}}} < Z_{\alpha/2}\right) = 1-\alpha$$

$$P\left(-Z_{\alpha/2}\frac{\sigma}{\sqrt{n}} < \overline{X}-\mu < Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right) = 1-\alpha$$

$$P\left(-\overline{X}-Z_{\alpha/2}\frac{\sigma}{\sqrt{n}} < -\mu < -\overline{X}+Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right) = 1-\alpha$$

$$P\left[\left(-\overline{X}-Z_{\alpha/2}\frac{\sigma}{\sqrt{n}} < -\mu < -\overline{X}+Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right)\times(-1)\right] = 1-\alpha$$

$$P\left(\overline{X}+Z_{\alpha/2}\frac{\sigma}{\sqrt{n}} > \mu > \overline{X}-Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right) = 1-\alpha$$

Exemplo (MEYER, 2000): Suponha que X represente a duração da vida de uma peça de equipamento. Admita-se que 100 peças sejam ensaiadas, fornecendo uma duração de vida média $\overline{X}$ = 501,2 horas. Suponha-se que σ seja conhecido e igual a 4 horas, e que se deseje obter um intervalo de confiança de 95% para a média μ, teremos:

$$2\Phi(z) - 1 = P\left(-z \leq \frac{\overline{X}-\mu}{\frac{\sigma}{\sqrt{n}}} \leq z\right) = (1-\sigma) = 0{,}95$$

$\Phi(z) = (0{,}95 + 1) / 2$

$\Phi(z) = 0{,}975$

Consultando a tabela de distribuição normal reduzida, chegamos à z = 1,96. O intervalo de confiança será:

I.C = (501,2 – 4/10 * 1,96; 501,2 + 4/10 * 1,96)

I.C = (500,4; 502,0)

Ao afirmar que (500,4; 502,0) constitui um intervalo de confiança de 95% para μ, não quer dizer que 95% das vezes a média amostral cairá nesse intervalo. Estamos afirmando que 95% das vezes a média estará contida no invervalo ($\overline{X} - \frac{z\sigma}{\sqrt{n}} ; \overline{X} + \frac{z\sigma}{\sqrt{n}}$).

Conforme Barbetta, Reis e Bornia (2008) a partir de um intervalo de confiança, é possível calcular o tamanho mínimo de uma amostra da seguinte maneira:

$$n \geq \frac{z_y^2 \sigma^2}{E_0^2} \tag{5}$$

Onde:

n= Tamanho mínimo a amostra;

z_y^2 = Z tabelado;

σ^2 = Variância;

E_0^2 =Erro amostral máximo tolerado.

3.2 ANÁLISE DE DESEMPENHO E MÉTRICAS DE DESEMPENHO

A análise de desempenho é uma das técnicas mais utilizadas, na informática, para medir e comparar software ou hardware. Avalia o grau de eficiência de produtos em relação a uma determinada funcionalidade. O grau de eficiência é representado por medidas escalares como, por exemplo, o tempo e a distância (CIFERRI, 1995).

Na análise de desempenho, os modelos a seguir são aplicados (CIFERRI, 1995):

a) modelo analítico: consiste em obter equações matemáticas com algoritmos para calcula-las, os quais relacionam parâmetros do sistema com medidas de desempenho. Geram uma série de suposições teóricas em relação ao sistema e, consequentemente, os resultados tendem ser imprecisos. Este modelo é aplicado mais frequentemente em fase de projeto.

b) modelo de simulação: possui aspectos de simulação quantitativa e/ou qualitativa. A simulação quantitativa combina parâmetros, variáveis independentes e variáveis depentendes em um modelo numérico. A simulação qualitativa não apresenta um modelo totalmente numérico, onde os componentes do modelo e suas relações são representados simbolicamente ou estruturalmente. Este modelo classifica a entrada e saída de dados como pertencentes ao mundo real, onde devem ser modelados. Dessa forma, produz resultados de saída a partir de cálculos ou inferência sobre os dados de entrada. Assim, o modelo de simulação gera as atividades de um sistema a partir de um conjunto de condições e hipóteses, não sendo necessário a experimentação.

c) modelo experimental: o próprio sistema é utilizado para gerar os resultados de desempenho. Assim, os resultados na análise de desempenho de sistema são considerados confiáveis. As técnicas de monitoração e *benchmark* pertencem a este modelo. A monitoração utiliza ferramentas de avaliação estatística presentes no sistema que está sendo avaliado. Por haver falta de padronização para tais ferramentas, o uso fica limitado e específico. A técnica de *benchmark* executa uma série de testes em um sistema para avaliar o seu desempenho. Em sistemas computacionais, a análise de desempenho caracteriza-se pela execução de uma série de programas sobre os mesmos. *Benchmarks* são aplicados em diferentes sistemas, gerando resultados confiáveis e precisos.

Na análise de desempenho aplica-se métricas, que são medidas que representam o estado de execução de alguma estrutura. Além disso, são empregadas em comparação de sistemas (SILVA, 2004).

Toda métrica possui uma relação de valores que é julgada como a faixa de normalidade. É preciso que uma métrica-base seja selecionada para realizar comparações e tomar decisões entre um sistema ou outro. Em geral elas ajudam em optar por um ou outro sistema por meio de requisitos pré-estabelecidos (SILVA, 2004).

A taxa de *clock*, bastante aplicada para representar desempenho, considera um processador de 250 MHz mais ágil do que um de 200 MHz para executar um dado processamento. Mais o melhor desempenho de uma aplicação não é provado somente com esta taxa, pois ela não considera todo o trabalho dentro dos ciclos de *clock* e o processo de interação que ocorre entre o processador e memória (RAMOS, 2008).

Conforme a equação 6, uma métrica voltada à medir velocidade é a Millions of Instrucions Per Second (MIPS). Ao utilizá-la, existe o problema de processadores de arquiteturas distintas efetuarem um número de operações diferentes para executar uma mesma instrução. Por exemplo, ao calcular uma equação grande, um processador pode realizar somente uma instrução enquanto outro requer duas ou mais (RAMOS, 2008).

$$MIPS = \frac{n}{t_e \times 10^6} \tag{6}$$

Onde:

n = número de instruções executadas;

t = tempo, em segundos, necessário para executar as n instruções.

O tempo de resposta é entendido como o tempo que leva para um usuário receber um retorno do sistema a partir de uma requisição. Durante uma requisição existe o momento t_0, que indica que o retorno do servidor chegou até o cliente. Entre t_0 e t_1 (momento conhecido como tempo de pensar), o usuário está determinando a próxima ação e se organizando para mandar a próxima requisição. No momento t_1, o cliente manda ao servidor uma nova

requisição e só começa a receber o retorno no momento t_2 e finaliza em t_3. Os intervalos t_2-t_1 e t_3-t_1 indicam o tempo de resposta (SILVA, 2004).

O *throughput* é uma métrica que indica quantas requisições são executadas em uma unidade de tempo. Cada tipo de requisição mandada ao servidor possui seu tipo de unidade. Em servidores web, por exemplo, pode-se calcular o número de requisições HTTP atendidas por segundo (SILVA, 2004).

O *speedup* é a aptidão que um sistema tem de realizar um trabalho em menos tempo, sendo este proporcional aos recursos computacionais adicionados. Quando não houver mudança na razão entre recursos adicionados e tempo, tal aceleração é definida como linear. Na equação 7, mostra-se como calcular (SOARES, 2010):

$$aceleração = \frac{T_1}{T_N} \tag{7}$$

Onde:

T_1: tempo de execução serial de um trabalho;

T_N: tempo de execução de um trabalho de forma paralela.

3.3 CONCEITO DE BENCHMARK

Há anos atrás, utilizava-se uma régua enterrada no solo para que fosse verificado o nível da maré. Assim sendo, a ciência estabeleceu que qualquer utensílio fixado, onde se pode comparar um novo registro (“mark”: uma marca) a um já existente como base , através de um local de análise (“bench”: Um banco onde os pesquisadores permaneciam para avaliar o nível da maré). Entretanto, as empresas passaram a utilizar a palavra com o significado de modelo ao invés do que ela representava a um tempo atrás (SANTOS, 2005).

Segundo Silva (2006) para medir o desempenho de aplicativos para computadores emprega-se os benchmarks, que possuem rotinas padronizadas para avaliar performance. Para que um benchmark seja classificado como apropriado, é necessário que o mesmo apresente os seguintes aspectos:

a) representatividade: são considerados somente benchmarks aplicados em sistemas reais, descartando os aplicados em sistemas emulados;

b) portabilidade: é necessário que os benchmarks sejem multi-plataformas, possibilitando avaliar o desempenho de aplicativos de vários fabricantes;

c) repetibilidade: se o benchmark for executado, duas ou mais vezes, em um mesmo ambiente, é preciso que gere resultados similares;

d) escalabilidade: o benchmark deve atuar sobre ambientes com diferentes recursos e potenciais;

e) não intrusividade: ao avaliar um ambiente, o benchmark não deve realizar alterações no mesmo;

f) simplicidade: os benchmarks devem apresentar simples compreensão aos usuários que irão utilizá-lo.

Segundo Silva (2006) para que um benchmark seje considerado confiável, deve-se constar em seu manual os seguintes elementos:

a) medidas: este elemento representa numericamente os resultados das examinações e subdividi-se em duas classes: condicionais e incondicionais. As medidas condicionais apresentam todo o funcionamento dos aplicativos e realizam comparações entre os mesmos. Estas são alcançadas por meio dos resultados das avaliações. Já as medidas incondicionais apontam a confiabilidade do sistema, tais como disponibilidade e integridade;

b) worload: este componente avalia o SGBD, analisando sua performance;

c) faultload: atua sobre a confiabilidade nos aplicativos. Este componente relata as falhas, identificando as possíveis causas das mesmas;

d) ambiente experimental: uma relação geral das características do ambiente e plataforma no qual o benchmark atua.

Uma estratégia que as empresas vem adotando nesses últimos anos, é adquirir um conjunto de aplicativos para testes de benchmark. Dessa forma, é possivel avaliar o desempenho de sistemas informatizados com diversas ferramentas. Um ganho obtido nessa idéia é que quando um dos softwares dessa coleção falha, um outro pode assumir o trabalho (HENNESSY; PATTERSON, 2003).

3.3.1 Ferramentas de Benchmark

A Seguir, serão descritos os benchmarks utilizados nesse trabalho e em seguida os testes realizados com os mesmos. As ferramentas são open source e os resultados são fornecidos de forma completa, sem restrições.

3.3.1.1 Scimark

Segundo Scimark (2010) esta é uma ferramenta de benchmark que verifica, por meio de cálculos, a performance de códigos numéricos presentes em aplicações científicas ou de engenharia. Consiste em cinco etapas: FFT, Jacobi Successive Over-relaxation, Sparse matrix-multiply, Monte Carlo integration, e dense LU factorization.

a) Fast Fourier Transform (FFT): realiza transformações de Fourier sobre um grupo numérico de 4kb. Esse kernel lida com operações aritméticas complexas, misturando endereços de memória não constantes e funções trigonométricas;

b) Monte Carlo Integration: Aproxima o valor de pi por meio da resolução da integral y=sqrt(1-x^2);

c) Jacobi Successive Over-relaxation (SOR): Sobre uma matriz 100x100 realiza uma variedade de padrões de acesso comum em aplicações de diferenças finitas;

d) Sparse matrix multiply: Trabalha com uma matriz esparsa estruturada, sendo disposta em um formato comprimido de linha e uma estrutura de esparsidade prescrita;

e) Dense LU factorization: Através de uma série de cálculos, obtém a fatoração LU de uma matriz 100x100.

3.3.1.2 Bonnie++

O Bonnie++ é uma ferramenta que realiza uma variedade de avaliações em discos rígidos e em sistemas de arquivos. Os sistemas de arquivos operam em diversas formas e utilizam aplicações diferentes. A ferramenta testa algumas dessas operações, onde apresenta um valor para a quantidade de trabalho realizada por segundo e a porcentagem de CPU utilizada neste tempo (COKER, 2010).

Segundo Coker (2010) os testes realizados são:

a) saída sequencial: Subdivide-se em 3 partes. Primeiro realiza a escrita de uma sequencia de caracteres, sendo um por vez. O laço que realiza a escrita deve ser pequeno para que qualquer I-cache o suporte. Em seguida faz uma escrita de um bloco de caracteres, no qual a sobrecarga de CPU funciona como um sistema operacional de alocação de espaço para arquivo. E por último, é realizado uma reescrita de blocos de caracteres;

b) entrada sequencial: Subdivide-se em 2 partes. Primeiro realiza a leitura de uma sequencia de caracteres, sendo um por vez. Está etapa deve realizar entrada somente

sequencial. E em seguida realiza a leitura de blocos de caracteres, testando o desempenho de entrada sequencial;

c) buscas randômicas: Processos em paralelo são testados realizando escritas e leituras em partes aleatórias do arquivo.

3.3.1.3 PCMark04

É um *benchmark* constituído por partes de aplicativos reais e seu código pode ser livremente analisado por qualquer usuário. Possui testes pré-definidos que avaliam a memória, unidade de disco rígido, CPU e gráficos. Aplica Multithreading em função de aumentar o desempenho e o uso de recursos (CORPORATION, 2010, tradução nossa).

Segundo Corporation (2010) o PCMark04 realiza os seguintes testes, sendo cada um durante 20 segundos:

a) compressão / decompressão de arquivo: utiliza a bilioteca Zlib 1.1.4 para a compressão de arquivo. Para tal teste, são utilizados 2 mb de arquivo executável, 2 mb de documento do Microsoft Word, 7.4 mb de arquivo de vídeo .avi e 2.7 mb de arquivo de textura .dds. O resultado é descrito em Megabytes processados por segundo;

b) checagem da gramática: realiza uma análise automática de um texto para efetuar uma correção gramatical. A biblioteca Link Grammar Parsing 4.2 é usada. O arquivo de entrada é um documento de texto de 130 Kilobyte. O resultado descrito é em Kilobytes processados por segundo;

c) renderização de página web: utiliza o Internet Explorer 6 no processo, onde 3 arquivos HTML de 200 Kilobyte com 4 imagens relacionadas são testados. O resultados descrito está nas páginas que são processadas por segundo;

d) processamento de imagem: os arquivos usados são um arquivo de 130 Kilobyte, dois arquivos de 900 Kilobyte e um arquivo de 1,1 megabytes. O resultado é descrito em milhões de pixels (MPixels) por segundo;

e) conversão de áudio: utiliza cálculos em ponto flutuantes e testa principalmente o processador. As bibliotecas Ogg Vorbis libraries libogg 1.0 e libvorbis 1.0 são aplicadas. Neste teste é codificado um arquivo descompactado WAV de áudio de 1.8 Megabyte para o formato Ogg Vorbis. O resultado é descrito em Kilobytes por segundo;

f) compressão de vídeo: neste teste o formato WMV e DivX são utilizados. Consiste em codificar um dado vídeo de formato MPEG para WMV e um DV para DivX. Enfatiza principalmente a CPU com cálculos de ponto flutuante. Tanto para compactação para WMV e DivX, os resultados são descritos em quadros processados por segundo;

g) memória gráfica: cria uma superfície primária bufferizada de 1024 x 768 resolução no modo DX exclusivas. Em seguida cria uma segunda superfície fora da tela sendo o dobro. Onde em cada quadro, a superfície de trabalho é modificada por conversão dos dados a partir do barramento AGP com velocidade de deslocação de 64 linhas por segundo. O resultado é descrito em frames atualizados por segundo;

h) cálculo físico e gráficos 3d: nesta etapa, a camada gráfica 3D trabalha o subsistema de gráficos 3D a partir de cálculos inteiros e a parte física destaca a CPU a partir de cálculos em ponto flutuante. Os resultados são descritos em frames renderizados por segundo;

i) examinação de vírus: utiliza a ferramenta F-Secure Anti-virus. Neste teste, o motor de scanner contém duas DLLs e um pequeno arquivo de banco de dados de vírus. Vinte e um arquivos de vários formatos, onde em média 23 Megabytes são examinados. O resultado é descrito em Megabytes examinados por segundo;

j) criptografia / descriptografia de arquivo: o algoritmo de Blowfish é utilizado nesta etapa e a biblioteca Crypto++ 5.0. Os arquivos utilizados são 2 mb de executável, 2 mb de documento do Microsoft Word, 1.1 mb de arquivo JPEG e 1.8 mb de arquivo de áudio WAV.

3.3.1.4 Httperf

Suporta dois protocolos bastante utilizados, o HTTP e o HTTPS. Permite que um fluxo constante de solicitações HTTPS ,vindo de uma máquina cliente, seja gerado e processado em uma máquina servidora, a qual realiza uma série de medições de desempenho enviados pelo httperf (FERNANDÉZ, 2010) .

A ferramenta possui um parâmetro responsável por informar quantos clientes novos por segundo iniciam uma interação com o servidor. A cada interação, onde fica ativa por um determinado tempo, denominado tempo da sessão, e fecha a conexão ao expirá-lo. Toda sessão iniciada é uma conexão HTTPS persistente com o servidor. Com o uso desta, o cliente faz requisições ao servidor, analisa sua resposta fornecida e segue um link juntamente com esta. O httperf permite estabelecer um tempo limite por cliente, onde caso este for ultrapassado e o servidor não enviar nenhuma resposta, a conexão atual é encerrada e um novo cliente emulado assume o lugar (FERNÁNDEZ,2010).

Este benchmark utiliza pouca memória, sendo inferior a 2mb. Para que suas requisições sejam atendidas, são frequentemente disparadas solicitações do servidor web, as quais utilizam uma maior parte da memória enquanto estão em execução (CASTRO, 2010).

Cada solicitação disparada, consome em média 3mb de memória virtual, entretanto as instâncias do mesmo executável compartilham uma mesma região dessa memória (CASTRO, 2010).

3.3.1.5 Autobench

É um script Pearl desenvolvido com o propósito de automatizar o benchmark de um servidor web ou efetuar uma análise comparativa entre dois servidores web. Atua como um complemento ao *httperf*. O autobench dispara diversas instâncias *httperf* contra cada máquina, possibilitando, a cada uma destas, aumentar a quantidade de requisições por segundo. A ferramenta gera gráficos para análise (XENOCLAST, 2010).

3.3.1.6 Unixbench

O unixbench foi criado a bastante tempo, em 1983, e teve algumas melhorias que foram realizadas pela revista Byte Magazine. Esta ferramenta realiza uma avaliação básica do desempenho do sistema (SANTOS, 2010).

Segundo Santos (2010) o unixbench é composto de alguns micro benchmark:

a) cópia de arquivos, o qual calcula a taxa de dados que um arquivo transfere para o outro, apresentando variados tamanhos de buffers;

b) cálculo de quantas vezes um processo tem permissão para ler ou escrever em um pipe;

c) calcula o tempo que um processo consome para criar um outro processo usando a função fork;

d) calcula quantas vezes um processo tem permissão para iniciar cópias de um script shell.

Esta ferramenta inclusive realiza medições de desempenho gráfico em 2D e 3D. As primeiras versões também mediam o desempenho da memória e thrasing do sistema, tais funcionalidades não estão mais presentes nas versões atuais (SANTOS, 2010).

4 TRABALHOS CORRELATOS

Atualmente, encontram-se no mercado uma variedade de softwares de virtualização, onde os mesmos possuem desempenho diferentes em relação a determinadas tarefas. Estudos e pesquisas são frequentemente realizados para oferecer noções de vantagens e desvantagens entre tais softwares, com base em certas métricas.

Neste capítulo são relacionados alguns trabalhos que realizaram pesquisas relacionadas à comparação de máquinas virtuais.

4.1 MÁQUINAS VIRTUAIS: AVALIAÇÃO DE DESEMPENHO E CONSOLIDAÇÃO DE SERVIDORES

Trabalho de Conclusão de Curso de Ciência da Computação, realizado em 2005, na Universidade Luterana do Brasil, Gravataí, Rio Grande do Sul.

Este trabalho realiza um estudo comparativo de desempenho entre as principais máquinas virtuais existentes. Também é apresentado uma metodologia desenvolvida para consolidar servidores físicos em máquinas virtuais (SANTOS, 2005).

4.2 UM ESTUDO COMPARATIVO SOBRE AS PRINCIPAIS FERRAMENTAS DE VIRTUALIZAÇÃO

Trabalho de Conclusão de Curso de Ciência da Computação, realizado em 2006, na Universidade Federal de Pernambuco, Recife, Pernambuco.

O objetivo deste trabalho é explanar os conceitos fundamentais de máquinas virtuais e realizar uma comparação de produtos de virtualização presentes no mercado em relação ao desempenho sob vários aspectos (ANDRADE, 2006).

4.3 UMA ARQUITETURA PARA MONITORAMENTO E MEDIÇÃO DE DESEMPENHO PARA AMBIENTES VIRTUAIS

Dissertação de Pós-Graduação em Ciência da Computação, realizado em 2006, na Universidade Federal de Minas Gerais, Belo Horizonte, Minas Gerais.

Este trabalho apresenta duas técnicas, para aplicar em ambientes virtuais, que avaliam e monitoram desempenho. Além disso, consta um teste de desempenho de softwares executando no Xen (SOUZA, 2006).

4.4 VIRTUALIZAÇÃO: CONCEITOS, TÉCNICAS APLICADAS E UM COMPARATIVO DE DESEMPENHO ENTRE AS PRINCIPAIS FERRAMENTAS SEM CUSTO DE LICENCIAMENTO

Trabalho de Conclusão de Curso de Sistemas de Informação, realizado em 2008, na Sociedade Educacional de Santa Catarina, Joinville, Santa Catarina.

O trabalho é constituído de uma explanação das técnicas fundamentais de virtualização e sua aplicação em produtos presentes no mercado. Foi realizado um experimento com máquinas virtuais, avaliando o desempenho relacionado à acesso a disco (QUEVEDO JUNIOR, 2008).

5 AVALIAÇÃO DAS MÁQUINAS VIRTUAIS

A virtualização é importante por proporcionar proteção aos recursos do sistema. Hypervisores monitoram as máquinas virtuais para que nenhuma afete no funcionamento da outra para que o sistema base (nativo), sobre o qual a máquina está não seja afetado. Testar estes ambientes é essencial para verificar o grau de estabilidade em executar determinadas tarefas. Este estudo de caso consiste em duas etapas para avaliar as máquinas virtuais, onde primeiramente foi definida a metodologia de avaliação e em seguida a análise dos resultados obtidos.

5.1 METODOLOGIA DE AVALIAÇÃO

Esta pesquisa descreve uma análise de desempenho experimental com a técnica de *benchmark*. Nesta etapa são descritas as configurações do equipamento, dos sistemas operacionais, dos softwares de comparação e das máquinas virtuais presentes no estudo. Também serão comentados os passos nos quais foi realizado a avaliação e as métricas em que se basearam.

5.1.1 Equipamento Utilizado

Para a realização do experimento, foram utilizados dois computadores, um com processador de 32 bits e outro com 64 bits, com a capacidade para executar todos os aplicativos nos testes.

O computador com processador de 32 bits possui a seguinte configuração:

- Processador: AMD Athlon (tm) XP 1500+ 1.33 GHz;

- Placa-mãe: PCCHIPS M810 DLU;

- Memória: 4 GB – DDR 400 ;

- Disco: Samsung ATA – 300 GB.

O segundo computador possui a seguinte configuração:

- Processador: Intel(R) Core(TM) 2 Quad Q800 2.33 GHz;

- Placa-mãe: ASUSTEK Computer INC. P5KPL-AM;

- Memória: 4 GB – DDR 2 ;

- Disco: Samsung SATA – 300 GB.

5.1.2 Sistemas Operacionais Utilizados

Foram escolhidos no experimento, SO que rodassem na máquina real e nas máquinas virtuais. Tanto para os sistemas hospedeiros e nas máquinas virtuais foram utilizados os SO Windows XP *Professional* com *Service Pack* 2 e Linux Ubuntu 9.10.

5.1.3 Máquinas Virtuais Utilizadas na Avaliação

Foram utilizadas nesta pesquisa, a técnica de virtualização completa (conceito mencionado no capítulo 2), onde o VMM realiza um intermédio entre o hardware e as máquinas virtuais, dessa forma ocorre uma simulação completa do hardware. Tal uso deve-se ao fato de comparar máquinas virtuais semelhantes. Foram, então, utilizados o VMware Server 2.0.2, VirtualBox 3 e KVM. Foi utilizado, também, o Virtual PC, que aplica a técnica de recompilação dinâmica, apenas para verificar o diferencial existente.

.

5.1.4 Benchmarks Utilizados

Foram escolhidos *benchmarks* que realizam avaliações nas plataformas, Linux e Windows. Na plataforma Windows, o *benchmark* utilizado foi o PCMark04 e na Linux foram utilizados o Scimark e o Bonnie++. As características básicas dos mesmos são as seguintes:

- PCMark04: possui testes pré-definidos que avaliam memória, unidade de disco rígido, CPU e gráficos. Para que o PCMark04 execute corretamente seus testes, os seguintes aplicativos devem estar instalados na máquina:

- Windows Media Encoder 9;

- Windows Media Player 9;

- Microsoft Excel;

- Microsoft DirectX 9c.

Aplica Multithreading em função de aumentar o desempenho e o uso de recursos (CORPORATION, 2010, tradução nossa);

- Scimark: verifica, por meio de cálculos, a performance de códigos numéricos presentes em aplicações científicas ou de engenharia. Consiste em cinco etapas: FFT, Jacobi Successive Over-relaxation, Sparse matrix-multiply, Monte Carlo integration, e dense LU factorization (SCIMARK, 2010). Requer o pacote openjdk-6-jdk para que possa inicializar e efetuar as avaliações;

- Bonnie++: é uma ferramenta que realiza uma variedade de avaliações em discos rígidos e em sistemas de arquivos. Os sistemas de arquivos operam em diversas formas e utilizam aplicações diferentes. A ferramenta testa algumas dessas operações, onde apresenta um valor para a quantidade de trabalho realizada por segundo e a porcentagem de CPU utilizada neste tempo (COKER, 2010).

5.1.5 Etapas do processo de avaliação

A avaliação de desempenho foi aplicada no ambiente nativo hospedeiro e nas máquinas virtuais na plataforma Linux e Windows. No ambiente nativo, foram aplicados os testes apenas para fornecer uma noção dos resultados. A avaliação de desempenho, nessa pesquisa, é focada nas máquinas virtuais.

Os seguintes cenários foram utilizados para efetuar as avaliações:

a) computadores com processadores de 32 e 64 bits;

b) computadores com processador Intel e AMD;

c) computadores com 4 GB de RAM e 300 GB de espaço em disco;

d) máquinas virtuais com 1 GB de RAM e 3 GB de espaço em disco;

e) máquinas virtuais com 2 GB de RAM e 3 GB de espaço em disco;

f) avaliação das máquinas virtuais com o Bonnie++;

g) avaliação das máquinas virtuais com o PCmark04;

h) avaliação das máquinas virtuais com o Scimark.

O primeiro passo foi instalar o SO Windows XP em ambas as máquinas físicas, cujas configurações já foram mencionadas. Em seguida foi instalado o PCMark04 nos computadores e executado 10 vezes consecutivas para calcular a quantidade de amostras mínimas necessárias de acordo com a fórmula do Tamanho Mínimo de uma Amostra a partir de um Intervalo de Confiança . Os demais *benchmarks* também foram executados o mesmo número de vezes, inicialmente, para obter o número de amostras mínimas e calcular a média. Estimou-se que o erro máximo tolerável foi de 50% do desvio padrão. Em seguida, foi instalado o Linux em ambas as máquinas físicas e também o pacote openjdk-6-jdk (necessário para executar o Scimark) e o Bonnie++. Após as instalações, as 2 ferramentas de avaliação foram utilizadas.

O próximo passo foi instalar, no PC com processador de 64 bits, o VMware, o VirtualBox e o KVM no sistema hospedeiro Linux e o Virtual PC no sistema hospedeiro Windows criando duas máquinas virtuais em cada ferramenta. Somente o VirtualBox e o VMware foram instalados e testados em ambos os computadores para verificar o desempenho com diferentes processadores.

Cada VMM, enfim, ficou com uma máquina virtual com SO Windows e uma com Linux.

A etapa seguinte foi instalar e rodar os *benchmarks* nas máquinas virtuais presentes e salvar os resultados gerados.

Foi determinado uma nomenclatura para identificar as máquinas virtuais, onde os dois primeiros dígitos na descrição representam o processador da máquina hospedeira, os três caracteres seguintes representam o sistema de virtualização e as duas últimas identificam o SO da máquina virtual. Para as máquinas físicas, foi adotado uma nomenclatura diferente, onde os dois primeiros dígitos representam o processador da máquina as três primeiras letras indicam que é um sistema físico e as demais representam o SO utilizado.

As máquinas virtuais criadas, foram configuradas com 1 GB de memória RAM e 3 GB de espaço em disco. Primeiramente, foram testadas as máquinas virtuais com SO Windows e em seguida as com o Linux em ambos os computadores. Posteriormente, foi configurado a memória RAM para ter 2 GB nas duas máquinas virtuais do VirtualBox presentes no computador com processador de 64 bits. Tal aumento foi realizado para verificar o desempenho dos ambientes virtuais com mais memória RAM.

Ao efetuar os testes, foram desativadas as conexões de rede, descansos de tela e recursos de economia de energia nos ambientes virtuais e físicos para todos os benchmarks. As conexões de rede ficaram ativas ao executar o Scimark, em função de ser um *applet* executado *online*.

O número de ambientes virtuais avaliados, segundo constam na Tabela 1, foram doze. Porém, na máquina com processador de 64 bits, os dois primeiros ambientes, relativos ao VirtualBox, foram avaliados duas vezes em função do aumento de memória RAM para 2 GB. Dessa forma, o total de ambientes totaliza quatorze. Como em cada máquina virtual que contém SO Linux foi executado dois *benchmarks*, o total de testes realizados foram vinte e um. Somando esse resultado com os dos ambientes nativos hospedeiros dos dois computadores, sendo seis ambientes, totaliza-se em vinte e sete avaliações.

Tabela 1. Ambientes virtuais avaliados

PROCESSADOR DA MÁQUINA	MÁQUINA VIRTUAL	SO	NOMENCLATURA
64 bits	VirtualBox	Windows	64VBXWI
		Linux	64VBXLI
	Vmware	Windows	64VMWWI
		Linux	64VMWLI
	KVM	Windows	64KVMWI
		Linux	64KVMLI
	Virtual PC	Windows	64VPCWI
		Linux	64VPCLI
32 bits	VirtualBox	Windows	32VBXWI
		Linux	32VBXLI
	VMware	Windows	32VMWWI
		Linux	32VMWLI

5.2 ANÁLISE DOS RESULTADOS OBTIDOS

Na execução dos testes de desempenho, para cada ambiente virtual e físico, a quantidade de amostras necessária, segundo a Fórmula do Tamanho Mínimo de uma Amostra

a partir de um Intervalo de Confiança, coincidiu em 16. Ou seja, primeiramente foram executados 10 vezes cada benchmark para estimar o número mínimo de amostras necessárias, onde, coincidentemente, resultou em 16.

A seguir segue o cálculo efetuado para o teste *File Compression* do *benchmark* PCMark04, segundo a fórmula tamanho mínimo de uma amostra, que resultou em 16 amostras para as avaliações. Os dez primeiros resultados obtidos foram: 1936247,70; 1948401,20; 1943130,00; 1940669,90; 1951803,90; 1952215,20; 1950445,40; 1951784,20; 1953015,70 e 1950668,40.

(Z tabelado² * variância) / Erro²

(1,96² * 30027818,30) / 2739,89²

(3,8416 * 30027818,30) / 7506997,22

115354866,79 / 7506997,22

15,37 ≅ 16

Para obter a variância, inicialmente calculou-se a média, que foi 1947838,16. Em seguida calculou-se a variância, a qual é a soma dos quadrados das diferenças entre cada elemento do conjunto e a média, dividido pelo número de elementos.

variância = ((1936247,70-1947838,16)² + (1948401,20-1947838,16)² + (1943130,00-1947838,16)² + (1940669,90-1947838,16)² + (1951803,90- 1947838,16)² + (1952215,20-1947838,16)² + (1950445,40-1947838,16)² + (1951784,20-1947838,16)² + (1953015,70 -1947838,16)² + (1950668,40-1947838,16)²) / 10

variância = (134338763,01 + 317014,04 + 22166770,59 + 51383951,43 + 15727093,75 + 19158479,16 + 6797700,42 + 15571231,68 + 26806920,45 + 8010258,46) / 10

variância = 30027818,30

O próximo passo foi obter o erro, onde calculou-se primeiramente o desvio padrão e em seguida dividiu-se o resultado por dois. O desvio padrão é dado pela raiz quadrada da variância, sendo 5479,77. Dividindo por dois, obteve-se 2739,89.

Para um intervalo de confiança de 95% para a média μ, obteve-se:

$$2\Phi(z) - 1 = P\left(-z \leq \frac{\overline{X} - \mu}{\frac{\sigma}{\sqrt{n}}} \leq z \right) = (1 - \sigma) = 0{,}95$$

$\Phi(z) = (0{,}95 + 1) / 2$

$\Phi(z) = 0{,}975$

Consultando a tabela de distribuição normal reduzida, chegamos à z tabelado= 1,96.

O *benchmark* PCMark04, foi utilizado para avaliar ambientes que utilizaram o SO Windows XP. Foram descartados os testes Virus Scanning e Physics Calculation and 3D, pois em alguns casos os resultados obtidos eram zerados.

A Figura 7 apresenta os resultados obtidos com o PCMark04. Pode-se observar que no PC com processador de 64 bits, o VMware apresentou um melhor desempenho em relação ao demais ambientes em 63,6 % dos testes efetuados com o benchmark. Em segundo lugar ficou o Virtual PC, em terceiro o VirtualBox e em quarto o KVM.

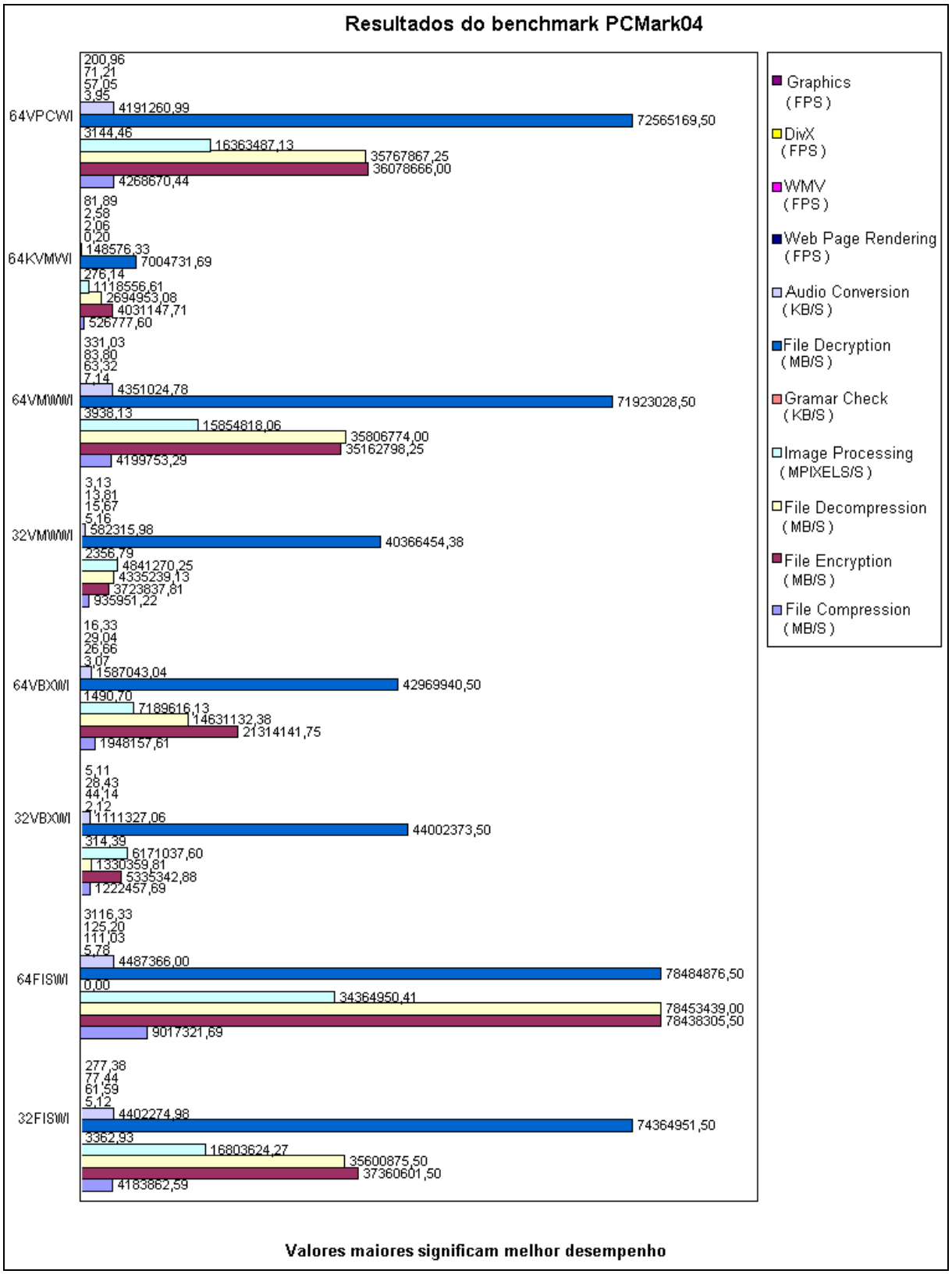

Figura 7. Resultados do PCMark04

Em relação aos ambientes virtuais presentes no PC com processador de 32 bits, o VirtualBox apresentou melhor desempenho na maioria dos testes com exceção do Web Page Rendering, Grammar Check, File Decompression.

Foi avaliado o desempenho do VirtualBox, presente no PC com processador de 64 bits, com o PCMark04, ao ser aumentado a memória RAM da máquina virtual para 2 GB. Conforme a Figura 8, em todos os testes pode ser observado um ganho de desempenho, exceto no Web Page Rendering.

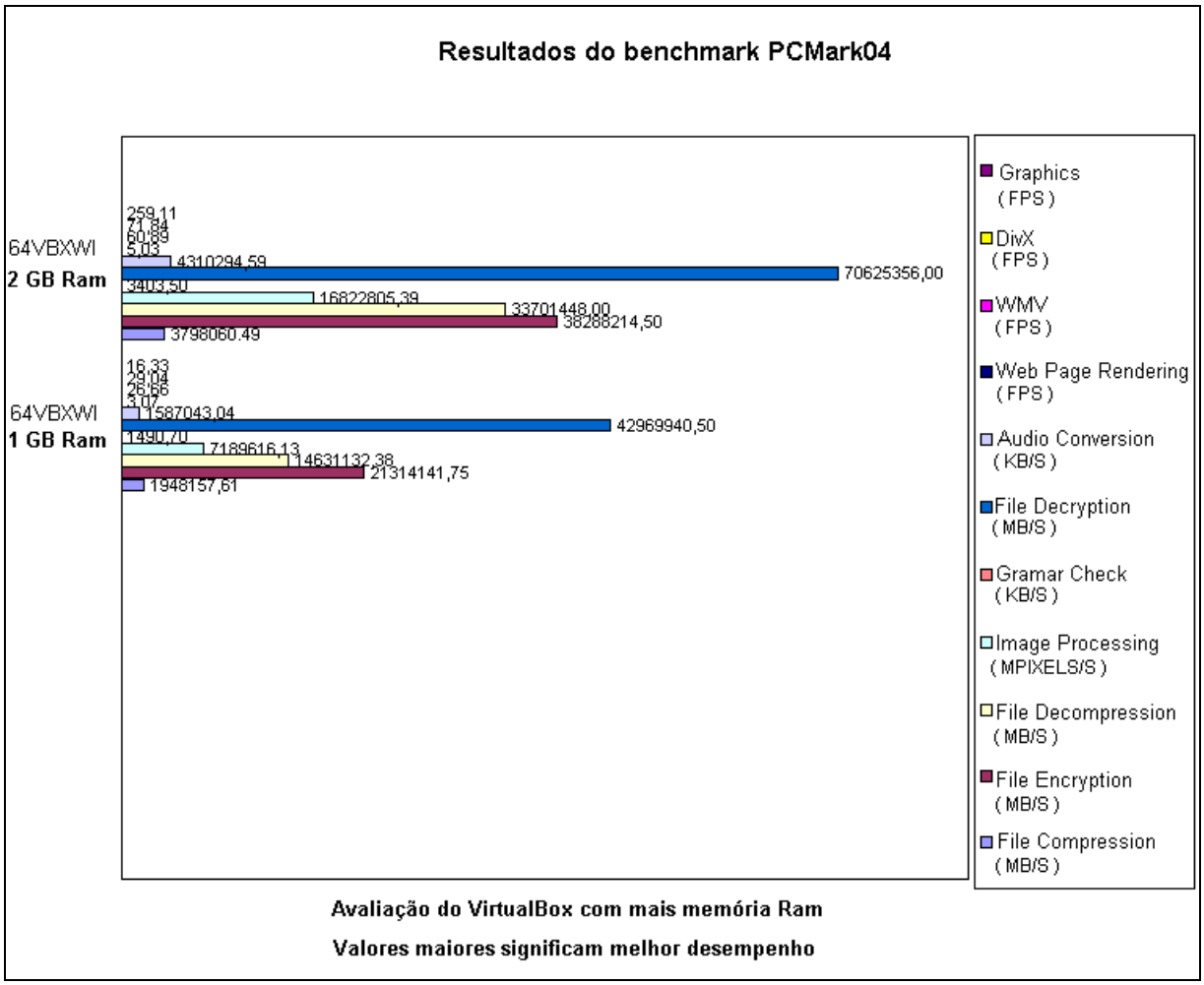

Figura 8. Resultados do PCMark04 com aumento de memória no VirtualBox

Em seguida, o Bonnie++ foi utilizado para avaliar ambientes que utilizaram o SO Linux Ubuntu 9.10. Dentre os testes deste benchmark, conforme já mencionados no capítulo

3, foram utilizados *Sequencial Output* com seus subtestes *Per Char, Block* e *rewrite;* e *Sequencial Input* com seus subtestes *Per Char* e *Block*. .

A Figura 9 apresenta os resultados obtidos com o Bonnie++. O pior desempenho nesse caso foi no ambiente físico do PC com processador de 32 bits. O ambiente físico no PC com processador de 64 bits, em relação aos dois ambientes virtuais que possui, obteve sucesso em 60% dos testes. Obteve melhores resultados nos testes *Sequencial Output Block*, *Sequencial Output Rewrite* e *Sequencial Input block*.

Em relação aos ambientes virtuais presentes no PC com processador de 32 bits, o VirtualBox apresentou melhor desempenho em 60% dos testes, dentre os quais estão o *Sequencial Output Per Char*, *Sequencial Output Rewrite* e *Sequencial Input Per Char*.

No PC com processador de 64 bits, o VirtualBox também apresentou melhor desempenho, destacando-se em 60% dos testes. Os testes foram os mesmos com esta máquina virtual no PC com processador de 32 bits. Em segundo lugar ficou o VMware, em terceiro o Virtual PC e em quarto o KVM.

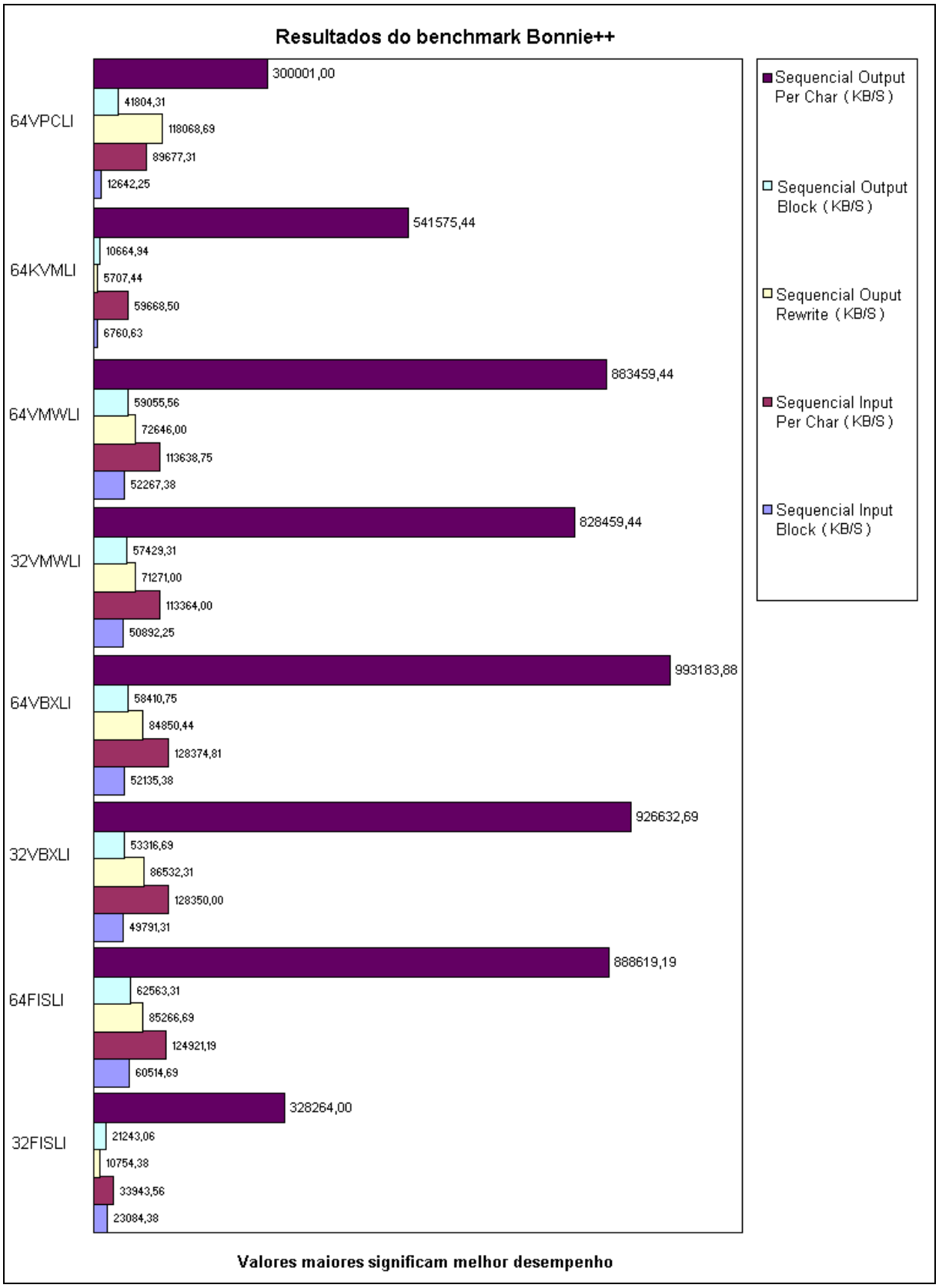

Figura 9. Resultados do Bonnie++

Enfim, do mesmo modo realizado no PCMark04, aumentou-se a memória Ram do VirtualBox, no PC com processador de 64 bits, para 2 GB. Inesperadamente, o desempenho obtido decaiu em 80 % dos testes. O único teste em que não houve perda foi o *Sequencial Input Block*. Os resultados podem ser conferidos na Figura 10.

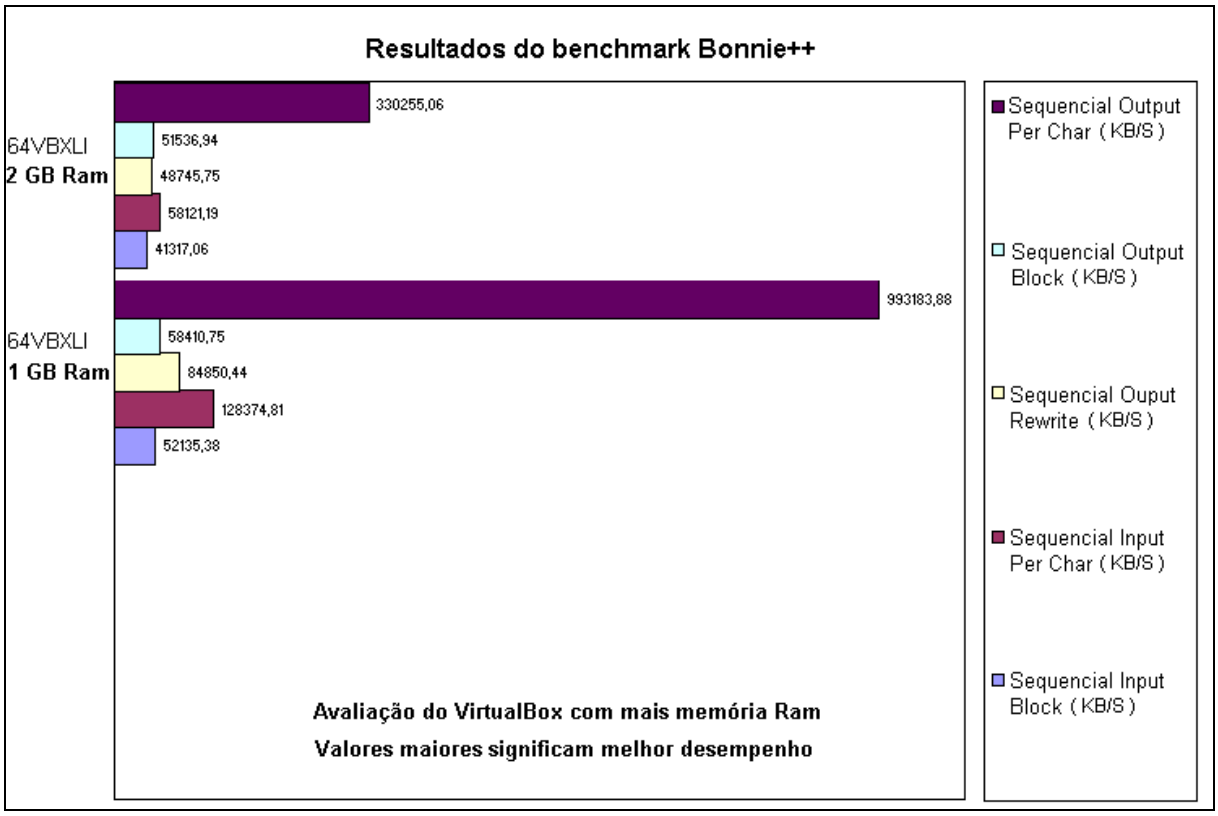

Figura 10. Resultados do Bonnie++ com aumento de memória no VirtualBox

Concluído os testes com o Bonnie++, o Scimark foi utilizado, também, para avaliar ambientes que utilizaram o SO Linux Ubuntu 9.10. Dentre os testes deste benchmark, conforme já mencionados no capítulo 3, foram utilizados *FFT*, *Jacobi Successive Over-relaxation*, *Sparse matrix-multiply*, *Monte Carlo integration*, e *dense LU factorization*.

A Figura 11 apresenta os resultados obtidos com o Scimark. O ambiente virtual que obteve melhor desempenho no computador com PC de 32 bits, foi o VMware. Com 66.7% de destaque nos testes, obteve desempenho mais elevado no *LU*, *Sparse*, *FFT* e *Scimark*.

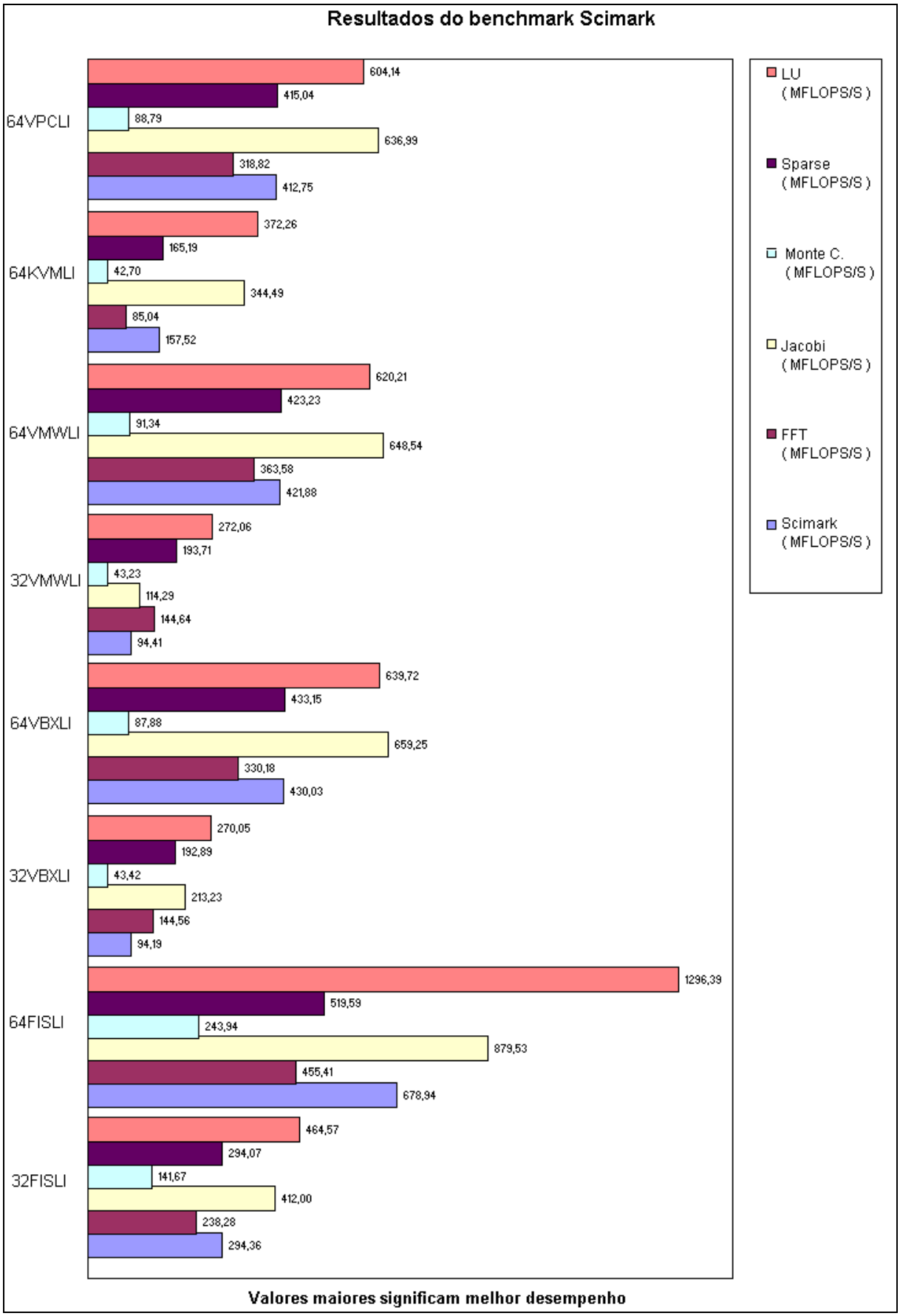

Figura 11. Resultados do Scimark

No outro PC, o que sobressaiu nas avaliações foi o VirtualBox, também 66.7% de sucesso e foram nos testes *LU*, *Sparse*, *Jacobi* e *Scimark.* Em segundo lugar ficou o VMware, em terceiro o Virtual PC e em quarto o KVM.

Seguindo a metodologia definida, foi aumentado a memória RAM nesta máquina virtual, no PC com processador de 64 bits, para 2 GB. Conseqüentemente, o desempenho obtido decaiu em 100 % dos testes. Os resultados podem ser conferidos na Figura 12.

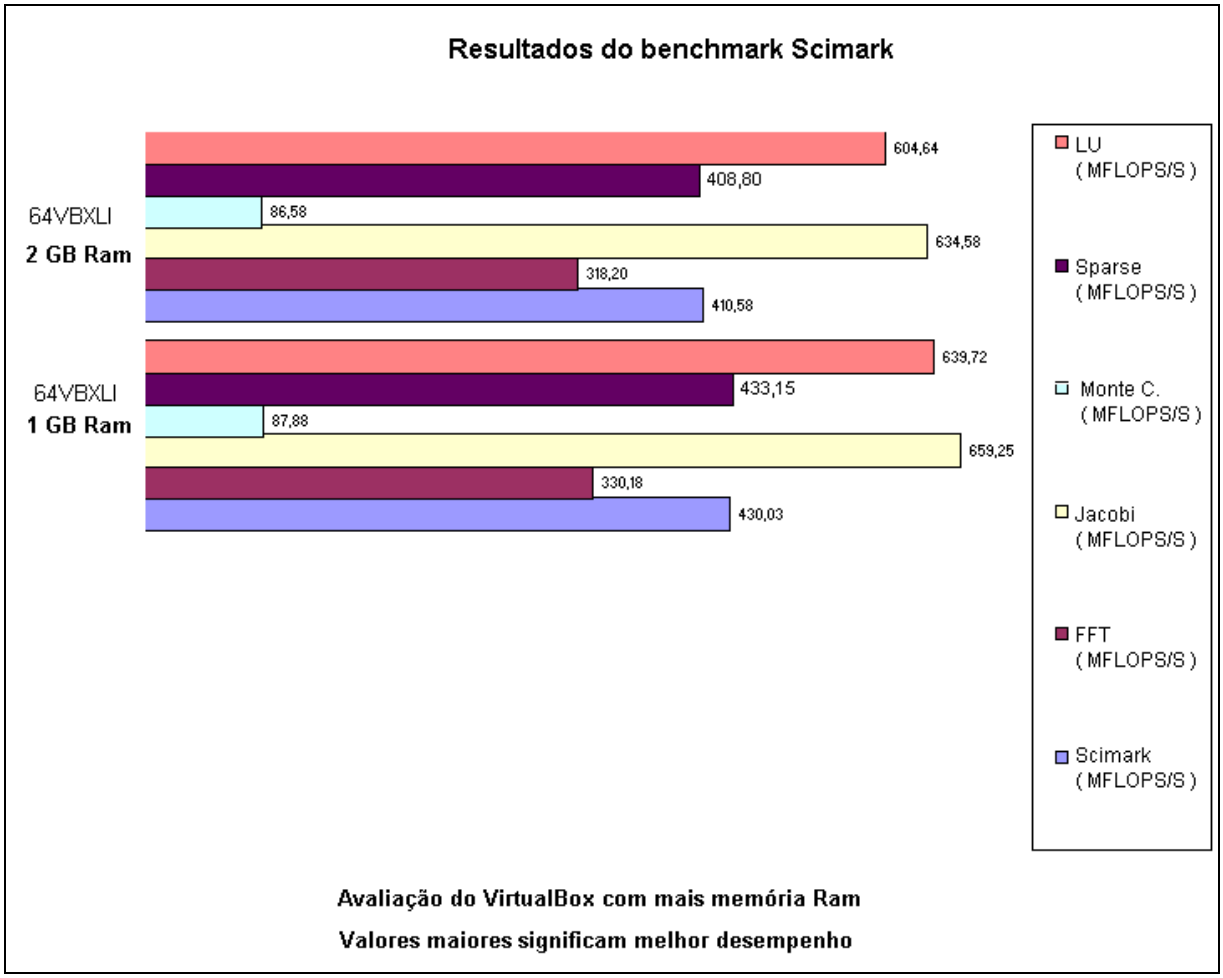

Figura 12. Resultados do Scimark com aumento de memória no VirtualBox

CONCLUSÃO

Analisou-se que a virtualização está sendo bastante utilizada hoje em dia, em função de agregar diversos sistemas operacionais em um único hardware, ocasionando melhor aproveitamento dos recursos disponíveis, proporcionando economia, flexibilidade, segurança, gerenciabilidade de aplicações e um ótimo isolamento de falhas.

Seu uso auxilia na proteção de recursos do sistema, proporcionando um ganho robusto em termos de segurança e também aumentando a possibilidade de desenvolver aplicativos sem atrapalhar as operações normais do sistema, além disso, aproveita-se mais o potencial do equipamento, onde pode-se utilizar os 85% de ociosidade do mesmo e diminuir a necessidade da organização em adquirir mais servidores. Com o uso do mesmo equipamento, a organização economizará espaço físico, tempo, dinheiro e facilita a estrutura de suporte de TI.

A importância de comparar máquinas virtuais é de extrema valia para determinar a melhor opção no uso para um fim específico, sendo que, uma das maneiras de realizar comparações é através de softwares de *benchmark.*

Nesta pesquisa, foi realizado uma análise de desempenho experimental com a técnica de *benchmark* com o VirtualBox, VMware Server, Virtual PC e KVM cada um com suas peculiaridades, seja em termos de desempenho, facilidade de instalação ou flexibilidade de configuração.

Para tal análise, foram utilizados os *benchmarks* PCmark04, Bonnie++ e Scimark, realizando uma série de testes replicados segundo a Fórmula do Tamanho Mínimo de uma Amostra a partir de um Intervalo de Confiança.

Com o PCmark04, os resultados demonstram que uma máquina virtual pode se comportar diferente em processadores de 32 e 64 bits. Executando em um processador de 32

bits, o melhor desempenho foi com o VirtualBox. Já com um processador de 64 bits, o VMware sobressaiu nos resultados. Com o Bonnie++, o VirtualBox demonstrou um melhor desempenho nos resultados gerados. Os testes com o Scimark em um processador de 32 bits o melhor desempenho foi com o VMware. Já com um processador de 64 bits, o VirtualBox obteve melhor performance.

Verificou-se que há inúmeros trabalhos que versam sobre o tema em comento, ou seja, sobre comparações de ferramentas de virtualização, dentre os analisados pode-se citar alguns autores que trataram do tema com grande precisão, tais como Santos (SANTOS, 2005), Quevedo Junior (QUEVEDO JUNIOR, 2008) e Ramos (RAMOS, 2008).

Como sugestão para trabalhos futuros, há a possibilidade de avaliar o desempenho de máquinas virtuais que utilizam a técnica de Paravirtualização ou Recompilação Dinâmica, juntamente com outros SO hospedeiros e convidados e outras ferramentas de benchmark.

REFERÊNCIAS

ANDRADE, Marcos Tadeu: **Um estudo comparativo sobre as principais ferramentas de virtualização.** Disponível em: < http://www.cin.ufpe.br/~tg/2006-2/mta.pdf >. Acesso em: 05/11/2009.

ARAÚJO, Márcio Valério de. **Sistemas de Medida e instrumentação.** Disponível em: <http://www.dca.ufrn.br/~acari/Sistemas%20de%20Medida/Material%20de%20sala%20de%20aula/Sistemas%20de%20Medidas%20e%20instrumenta%E7%E3o%20-%20parte%201.pdf>. Acesso em: 28 abr. 2010.

BARBETTA, Pedro Alberto; REIS, Marcelo Menezes; BORNIA, Antonio Cezar. . **Estatística: para cursos de engenharia e informática.** 2. ed São Paulo: Atlas, 2008. 410 p.

BELLARD, Fabrice. **QEMU, a Fast and Portable Dynamic Translator.** Disponível em: <http://www.usenix.org/events/usenix05/tech/freenix/full_papers/bellard/bellard_html/>. Acesso em: 22 abr. 2010.

BERNARDINELLI, Regina Maria Sigolo. **Estatística.** Disponível em: <http://teleduc.unisa.br/~teleduc/cursos/diretorio/apoio_880_79///apostila.pdf>. Acesso em: 29 abr. 2010.

CAMPOS, Cassio P. de; KASSALIAS, Nicolas. **Atividades Práticas no Ensino Introdutório de Sistemas Operacionais.** In: WORKSHOP DE SISTEMAS OPERACIONAIS, 3., 2006, Campo Grande. Anais... . Campo Grande: Wso, 2006. v. 1, p. 1 - 8. Disponível em: <http://www.lisha.ufsc.br/wso/papers/2006/campos2006.pdf>. Acesso em: 22 abr. 2010.

CARBONARI, Maria Elisa Ehrhardt et al. **Auto-Avaliação de IES - Base estatística para Índices de Desempenho.** Disponível em: <http://sare.unianhanguera.edu.br/index.php/rcext/article/viewFile/396/396>. Acesso em: 30 abr. 2010.

CASTRO, Arthur Bispo de. **Máquinas virtuais em ambientes seguros.** 2006. 85 f. Dissertação (Mestrado) - Curso de Ciência da Computação, Instituto de Computação Universidade Estadual de Campinas, Campinas Sp, 2006. Disponível em: <http://www.las.ic.unicamp.br/paulo/teses/20060210-MSc-Arthur.Bispo.de.Castro-Maquinas.virtuais.em.ambientes.seguros.pdf>. Acesso em: 05 dez. 2009.

CASTRO, Rodrigo Souza de. **Cache comprimido adaptativo: projeto, estudo e implementação.** 2003. 147 f. Dissertação (Mestrado) - Curso de Ciência da Computação, Instituto de Matemática e Estatística da Universidade de São Paulo, São Paulo, 2003. Disponível em: <http://linuxcompressed.sourceforge.net/linux24-cc/files/docs/dissertacao.pdf>. Acesso em: 23 fev. 2010.

CHERNOFF, Anton; HOOKWAY, Ray. **DIGITAL FX!32.** Disponível em: <http://www.usenix.org/publications/library/proceedings/usenix-nt97/full_papers/chernoff/chernoff.pdf>. Acesso em: 22 abr. 2010.

CIFERRI, Ricardo Rodrigues. **Um benchmark voltado à análise de desempenho de sistemas de informações geográficas.** 1995. 192 f. Dissertação (Mestrado) - Curso de Ciência da Computação, Universidade Estadual de Campinas, Campinas, 1995. Disponível em: <http://biblioteca.universia.net/html_bura/ficha/params/id/51034730.html>. Acesso em: 12 nov. 2010.

COKER. **Bonnie++.** Disponível em: <http://www.coker.com.au/bonnie++/readme.html>. Acesso em: 23 fev. 2010.

CORPORATION, Futuremark. **PCMark04: PC Performance Analysis.** Disponível em: <http://www.futuremark.com/pressroom/companypdfs/PCMark04_Whitepaper>. Acesso em: 12 set. 2010.

COSTA, Giovani Glaucio de Oliveira. **Um Procedimento Inferencial para Análise Fatorial Utilizando as Técnicas Bootstrap e Jackknife: Construção de Intervalos de Confiança e Testes de Hipóteses.** 2006. 1 v. Tese (Mestre) - Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 2006. Disponível em: <http://www2.dbd.puc-rio.br/pergamum/tesesabertas/0310444_06_cap_05.pdf>. Acesso em: 04 maio 2010.

DAVIS, Megan. **Virtual PC vs. Virtual Server: Comparing Features and Uses.** Disponível em: <http://download.microsoft.com/download/1/4/d/14d17804-1659-435d-bc11-657a6da308c0/VSvsVPC.doc>. Acesso em: 03 fev. 2009.

FERNÁNDEZ, Jordi Guitart. **Performance Improvement of multithreaded java applications execution on multiprocessor systems.** Disponível em: <http://www.tesisenxarxa.net/TESIS_UPC/AVAILABLE/TDX-0303106-123000//01Jgf01de01.pdf>. Acesso em: 23 fev. 2010.

FONTES, Edison. **Segurança da informação:** o usuário faz a diferença. São Paulo: Saraiva, 2006. 172p

FRANCO, Samuel Mendes; OLIVEIRA, Claudio Lopes de. **AVALIAÇÃO DA INFLUÊNCIA DOS ERROS SISTEMÁTICOS E DAS PARCELAS DE INCERTEZAS DE MEDIÇÃO NAS COMBINAÇÕES DE BLOCOS PADRÃO.** Disponível em: <http://www.abepro.org.br/biblioteca/ENEGEP1999_A0322.PDF>. Acesso em: 28 abr. 2010.

GOLDBERG, R. P.. **ARCHITECTURE OF VIRTUAL MACHINES.** Disponível em: <http://www.cse.psu.edu/~bhuvan/teaching/spring06/papers/goldberg.pdf>. Acesso em: 19 mar. 2010.

GOMES, Marcelo Marques; FRACALOSSI, Weverson Fantin. **Máquinas virtuais.** Disponível em: <http://www.gfsolucoes.net/trabalhos/maquinas_virtuais.pdf>. Acesso em: 2007.

GONÇALVES, Danilo Brandão; VAHL JUNIOR, José Cláudio. **White Paper – Virtualização.** Disponível em: <http://www.sensedia.com/br/anexos/wp_virtualizacao.pdf>. Acesso em: 19 abr. 2010.

HANSEN, Jean Carlos; SCHAEFFER, Carlos Adriani Lara. **Estudo e Aplicação de Virtualização na Criação de Ambientes para Ensino de Redes de Computadores.** Disponível em: <http://www.upf.br/computacao/images/stories/TCs/arquivos_20092/Jean_Carlos_Hansen.pdf>. Acesso em: 22 mar. 2010.

HUMPHREYS, John. System Virtualization: Sun Microsystems Enables Choice. IDC, USA, p.1-12, out. 2006. Disponível em: <http://www.sun.com/servers/virtualization/idc_systemvirtualization06.pdf>. Acesso em: 16 mar. 2010.

INOCENTE, Emerson Meneses. **Green It – Processo de Redução de Consumo de energia elétrica baseado na virtualização.** Disponível em: <http://tconline.feevale.br/tc/files/0002_1966.pdf>. Acesso em: dez. 2009.

INSTRON. **7 DICAS PARA TESTES DE MATERIAIS.** Disponível em: <http://www.instron.com.br/wa/library/StreamFile.aspx?doc=1467>. Acesso em: 28 abr. 2010.

KLEIN, Jandir. **Análise de desempenho de rede de servidores virtuais.** 2008. 1 v. Tese (Bacharel) - Centro Universitário Feevale Instituto de Ciências Exatas e Tecnológicas, Novo Hamburgo, 2008. Disponível em: <http://tconline.feevale.br/tc/files/0001_1505.pdf>. Acesso em: 12 dez. 2009.

HENNESSY, John L.; PATTERSON, David A. **Arquitetura de Computadores: uma abordagem quantitativa.** 3. ed. Rio de Janeiro: Campus, 2003. 827 p.

LAUREANO, Marcos. **Máquinas virtuais e emuladores:** conceitos, técnicas e aplicações. São Paulo: Novatec, 2006. 184 p.

LINUX-MAGAZINE. **Citrix torna o Xen Server gratuito.** Linux Magazine Online, São Paulo, n. , p.1-1, 23 fev. 2009. Disponível em: <http://www.linux-magazine.com.br/noticia/citrix_torna_o_xen_server_gratuito>. Acesso em: 20 abr. 2010.

LUCAS, Bianka Capelato. **A segurança da infomação em organizações de Salvador.** Disponível em: <http://www.bibliotecadigital.ufba.br/tde_busca/arquivo.php?codArquivo=1160>. Acesso em: dez. 2009.

MATHEWS, Jeanna N.; DOW, Eli M.. **Executando o XEN.** São Paulo: Alta Books, 2009.

MEYER, Paul L.. **Probabilidade Aplicações Á Estatistica.** 2ª São Paulo: Ltc, 2000.
NANDA, Susanta; CHIUEH, Tzi-cker. **A Survey on Virtualization Technologies.** Disponível em: <http://www.ecsl.cs.sunysb.edu/tr/TR179.pdf>. Acesso em: 17 mar. 2010

NEVES, Glauco; PACHECO, Guilherme. **Xen Hypervisor.** Disponível em: <www.lisha.ufsc.br/teaching/os/ine5412-2008-2/work/xen.pdf>. Acesso em: 05 fev. 2010.

OKANO, Marcelo Tsuguio; ANDRADE, Fernanda Favero de. O IMPACTO DA VIRTUALIZAÇÃO NAS EMPRESAS. In: CONGRESSO NACIONAL DE EXCELÊNCIA EM GESTÃO, 4., 2008, Niterói Rj. **Anais... .** Niterói RJ: CNEG, 2008. p. 31/07/2008 - 02/08/2008. Disponível em: <http://www.excelenciaemgestao.org/Portals/2/documents/cneg4/anais/T7_0014_0102.pdf>. Acesso em: 31 jan. 2009.

PEREIRA JUNIOR, Deoclides: **VIRTUALIZAÇÃO: CONCEITOS, TÉCNICAS APLICADAS E UM COMPARATIVO DE DESEMPENHO ENTRE AS PRINCIPAIS FERRAMENTAS SEM CUSTO DE LICENCIAMENTO.** Disponível em: < http://ist.sociesc.org.br/cursos/bsi/TrabalhoDeDiplomacao/TD-DeoclidesQuevedoJunior-2008-2.pdf > Acesso em: 17/11/2009.

PIANA, Clause Fátima de Brum; MACHADO, Amauri de Almeida; SELAU, Lisiane Priscila Roldão. **Estatística Básica.** Disponível em: <http://minerva.ufpel.edu.br/~markus.stein/Apostila_EB.pdf>. Acesso em: 10 maio 2010.

POTRICH, Juliano. **APRIMORAMENTO DO ESCALONADOR CREDIT.** 2008. 1 v. Dissertação (Bacharel) - Curso de Sistemas de Informação, Universidade Católica do Rio Grande do Sul, Porto Alegre, 2008. Disponível em: <http://caioba.pucrs.br/teo/ojs/index.php/graduacao/article/viewFile/2858/3152>. Acesso em: 05 fev. 2010.

QUEVEDO JUNIOR, Deoclides Pereira. **Virtualização: Conceitos, técnicas aplicadas e um comparativo de desempenho entre as principais ferramentas sem custo de licenciamento.** 2008. 1 v. Tese (Bacharel) - Sociedade Educacional De Santa Catarina, Joinvile, 2008. Disponível em: <http://ist.sociesc.org.br/cursos/bsi/TrabalhoDeDiplomacao/TD-DeoclidesQuevedoJunior-2008-2.pdf>. Acesso em: 21 jan. 2010.

RAMOS, Nelson Azoubel. **Avaliação e Comparação de Desempenho de Computadores: Metodologia e Estudo de Caso.** 2008. 1 v. Dissertação (Bacharel) - Curso de Engenharia da Computação, Universidade Federal de Pernambuco, Recife, 2008. Disponível em: <http://www.google.com.br/url?sa=t&source=web&ct=res&cd=2&ved=0CA0QFjAB&url=http%3A%2F%2Fwww.cin.ufpe.br%2F~tg%2F2008-2%2Fnar.pdf&ei=r5nZS4DZOcSyuAfDxoSpDw&usg=AFQjCNEamiKwV5aFLBsMSNST7cNJLAUQjg&sig2=ug7bfSyAh_dX_7irqlKLPQ>. Acesso em: 29 abr. 2010.

REDHAT. **KVM – KERNEL BASED VIRTUAL MACHINE.** Disponível em: <http://www.redhat.com/f/pdf/rhev/DOC-KVM.pdf>. Acesso em: 27 abr. 2010.

RIZZO, Ana Lucia Tucci; CYMROT, Raquel. **UTILIZAÇÃO DA TÉCNICA DE REAMOSTRAGEM BOOTSTRAP EM APLICAÇÂO NA ENGENHARIA DE PRODUÇÃO.** Disponível em: <http://www.inicepg.univap.br/cd/INIC_2006/inic/inic/07/INIC0000347ok.pdf>. Acesso em: 04 maio 2010.

RODRIGUES, Guilherme da Cunha. **VMIB: Uma MIB Genérica para Gerenciamento de Recursos Virtuais.** Disponível em: <http://tede.pucrs.br/tde_busca/arquivo.php?codArquivo=1239>. Acesso em: 05 jan 2009.

ROSE, Robert. **Survey of System Virtualization Techniques.** Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=511164D5184345848FBDB5926534EE47?doi=10.1.1.58.9811&rep=rep1&type=pdf>. Acesso em: 05 dez. 2010.

ROSSI, Fábio Diniz. **Alocação dinâmica de recursos no Xen.** 2008. 1 v. Dissertação (Mestre) - Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, 2008. Disponível em: <http://tede.pucrs.br/tde_arquivos/4/TDE-2009-02-20T124758Z-1694/Publico/409163.pdf>. Acesso em: 05 fev. 2010.

SAFT, Luana Sandrini. **Virtualização de Software.** 2008. 1 v. Tese (Bacharel) - Curso de Sistemas de Informação, Universidade Federal de Santa Catarina, Florianópolis Sc, 2008. Disponível em: <http://projetos.inf.ufsc.br/arquivos_projetos/projeto_764/TCC_projetoI.pdf>. Acesso em: 08 jan. 2009.

SANTOS, António Manuel Rodrigues Carvalhos Dos. **Segurança nos Sistemas de informação Hospitalares: Políticas, Práticas e Avaliação.** 2007. 1 v. Tese (Doutorado) - Universidade do Minho, Guimarães, 2007. Disponível em: <http://repositorium.sdum.uminho.pt/bitstream/1822/7282/4/Tese_Integral_H.pdf>. Acesso em: 04 dez. 2009.

SANTOS, Douglas. **Avaliação do Comportamento de Sistemas Operacionais em Situação de Thrashing. 2009.** 1 v. Dissertação (Mestrado) - Pontifícia Universidade Católica do Paraná, Curitiba, 2009. Disponível em: <http://www.ppgia.pucpr.br/lib/exe/fetch.php?id=teses&cache=cache&media=dissertacoes:2009:2009_douglas_santos.pdf>. Acesso em: 23 fev. 2010.

SANTOS, Garilan Maia Dos. **Máquinas virtuais: avaliação de desempenho e consolidação de servidores.** 2005. 58 f. Tese (Bacharel) - Curso de Ciência da Computação, Universidade Luterana do Brasil, Gravataí, 2005. Disponível em: <http://gravatai.ulbra.tche.br/~roland/tcc-gr/monografias/2005-1-tc2-Garilan_Maia_dos_Santos.pdf>. Acesso em: 01 fev. 2010.

SANTOS, Alexandro Klein Dos. **Automatização na configuração de rede em virtual appliances para virtualbox.** 2008. 1 v. Trabalho (Bacharel) - Universidade Federal de Santa Maria, Santa Maria, 2008. Disponível em: <http://www-app.inf.ufsm.br/bdtg/arquivo.php?id=96&download=1>. Acesso em: 08 fev. 2010.

SCIMARK. **About SciMark 2.0.** Disponível em: <http://math.nist.gov/scimark2/about.html>. Acesso em: 23 fev. 2010.

SEO, Carlos Eduardo: **Virtualização – Problemas e Desafios.** Disponível em: <http://www.ic.unicamp.br/~ducatte/mo401/1s2009/T2/008278-t2.pdf> Acesso em: 17/11/2009

SILBERSCHATZ, Abraham; GALVIN, Peter B.; GAGNE, Greg. **Sistemas operacionais: conceitos e aplicações.** Rio de Janeiro: Ed. Campus, 2000. 585 p.

SILVA, Ernesto Cleyton Linhares da. **Análise da Interdependência de Métricas de Desempenho na Composição de Clusters de Servidores Web.** 2004. 1 v. Dissertação (Mestrado) - Universidade Federal do Paraná, Curitiba, 2004. Disponível em: <http://dspace.c3sl.ufpr.br/dspace/bitstream/1884/1897/1/dissertacao.pdf>. Acesso em: 04 maio 2010.

SILVA, Ricardo Czelusniak da. **Benchmark em banco de dados multimídia: Análise de desempenho em recuperação de objetos multimídia.** 2006. 1 v. Dissertação (Mestre) - Universidade Federal do Paraná, Curitiba, 2006. Disponível em: <http://dspace.c3sl.ufpr.br:8080/dspace/bitstream/1884/4683/1/disserta_ricardo.pdf>. Acesso em: 24 fev. 2010.

SILVA, Fábio Rodrigo Albuquerque da. **Vantagens e desvantagens na utilização de software de virtualização em servidores de empresas de pequeno e médio porte: Estudo de casos em faculdades particulares no Recife.** 2007. 1 v. Tese (Bacharel) - Faculdade Santa Maria, Recife PE, 2007. Disponível em: <http://www.nogueira.eti.br/profmarcio/obras/Fabio%20-%20Virtualizacao.pdf>. Acesso em: 05 nov. 2009.

SILVA, Rodrigo Ferreira da. **Virtualização de Sistemas Operacionais.** 2007. 1 v. Tese (Graduado) - Instituto Superior de Tecnologia em Ciências da Computação de Petrópolis, Petrópolis, 2007. Disponível em: <http://www.lncc.br/~borges/doc/Virtualizacao%20de%20Sistemas%20Operacionais.TCC.pdf>. Acesso em: 17 nov. 2009.

SOARES, Jorge de Abreu. **Sistemas de Bancos de Dados Paralelos.** Disponível em: <http://www.jsoares.net/CEFET/BD/sbdp.pdf>. Acesso em: 05 maio 2010.

SOUZA, Fabrício Benevenuto de. **Uma arquitetura para monitoramento e medição de desempenho para ambientes virtuais.** 2006. 1 v. Dissertação (Mestrado) - Curso de Ciência da Computação, Universidade Federal de Minas Gerais, Belo Horizonte, 2006. Disponível em: <http://www.bibliotecadigital.ufmg.br/dspace/bitstream/1843/RVMR-6QHGTR/1/dissertacao.pdf>. Acesso em: 23 mar. 2010.

TANENBAUM, Andrew S.: **Sistemas operacionais modernos.** Rio de Janeiro: LTC, 1999. 493 p.

VIRTUALBOX. Technical Documentations, Innotek GmbH. Disponível em: <http://www.virtualbox.org>. Acesso em: 20 abr. 2010.

XENOCLAST. **Autobench.** Disponível em: **http://www.xenoclast.org/autobench/**>. Acesso em: 23 fev. 2010.

WINCK, Ana Trindade. Processo de KDD para auxílio à Reconfiguração de ambientes virtualizados. 2007. 1 v. Dissertação (Mestrado) - Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, 2007. Disponível em: <http://tede.pucrs.br/tde_busca/arquivo.php?codArquivo=1114>. Acesso em: 13 nov. 2009.

WINXLINUX. **Como instalar o XEN no Ubuntu – Virtualizador tipo VMware.** Disponível em: <http://winxlinux.com/instalar-xen-ubuntu-virtualizador-vmware/>. Acesso em: 12 dez. 2009.

APÊNDICE 1. ARTIGO

Comparação Entre Ferramentas de Virtualização: Estudo de Caso Utilizando Benchmarks

Diego Scarduelli Langer[1], Kristian Madeira[2], Paulo João Martins[2]

[1]Acadêmico do curso de Ciência da Computação – Unidade Acadêmica de Ciências, Engenharias e Tecnologias - Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC - Brasil

[2]Professor(a) do curso de Ciência da Computação - Unidade Acadêmica de Ciências, Engenharias e Tecnologias - Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC - Brasil

diegosl1207@hotmail.com, {pjm,kma}@unesc.net

Abstract. *This research made a comparison of the various computer virtualization tools that are on the market. It was analyzed that with a technique you could integrate multiple operating systems on the same hardware and have more potential from the available resources, you could gain more in flexibility and security. It was found that there is a degree of difficulty in deciding which environment to be used, because there are so many softwares and manufacturers. Such as Xen, VirtualBox, VMware, KVM and Virtual PC. For this analysis, it was used the benchmark software PCMark04, Bonnie++ and Scimark, to perform a series of tests according to the Form of Minimum Size of a Sample from a confidence interval.A 32-bits computer with the PCMark04 was used to know which virtualization tools was better. The biggest highlight was with the VirtualBox.With a 64-bits processor, the VMware had excelled in the results. With the Bonnie++, the VirtualBox showed a better performance according to the results. When running Scimark on a 32-bit PC, the performance was better with VMware. However, with a 64-bit PC, VirtualBox did stand out during the evaluations.Although the analysis of performance between the virtualization tools are similar, it emphasizes a preference for VirtualBox because it did stand out during the tests The software have an easier interface and it's more practical to handle, to purchase and it's more easy to install.*

Resumo. *A presente pesquisa fez uma comparação sobre as ferramentas de virtualização. Analisou-se que com está esta técnica pode-se integrar vários sistemas operacionais sobre um mesmo hardware e consequentemente usufruir mais do potencial dos recursos disponíveis, obtendo um ganho na flexibilidade e segurança. Em seguida, verificou-se que existe um grau de dificuldade em decidir qual ambiente a ser utilizado, ou seja, há muitos modelos, tais como: Xen, VirtualBox, VMware, KVM e Virtual PC. E para finalizar foi realizado uma análise de desempenho experimental com a técnica de benchmark entre quatro hypervisores disponíveis, o VMware Server, VirtualBox, KVM e Virtual PC. Para tal análise, foram utilizados os benchmarks PCmark04, Bonnie++ e Scimark, realizando uma série de testes replicados segundo a Fórmula do Tamanho Mínimo de uma Amostra a partir de um Intervalo de Confiança. Com o PCmark04, avaliando em um PC de 32 bits, quem teve maior destaque foi o VirtualBox. Já com um processador de 64*

bits, o VMware sobressaiu nos resultados. Com o Bonnie++, o VirtualBox demonstrou um melhor desempenho segundo os resultados gerados. Ao executar o Scimark em um PC de 32 bits o melhor desempenho foi com o VMware. Entretanto, com um PC de 64 bits, o VirtualBox sobrressaiu nas avaliações. Apesar da análise de desempenho entre as ferramentas de virtualização serem equiparadas, ressalta-se a preferência pelo VirtualBox por sobressair-se nos testes, ter uma interface mais fácil para manusear e maior praticidade na aquisição e instalação.

1. Introdução

A informação tem um papel de grande relevância atualmente, visto que se aplicá-la conjuntamente com os recursos tecnológicos terar-se-á maior resultado na funcionalidade, tática, estratégica e operacional nas empresas. (SILVA, R; 2009).

Com a virtualização é possível particionar os recursos do hardware de modo que ele execute diversos sistemas operacionais (iguais ou diferentes) juntamente com seus aplicativos simultanemante e totalmente isolados um do outro.

A análise de desempenho é o grupo de métodos que permite a análise temporal de um sistema, na prática é uma ferramenta para auxiliar engenheiros na procura do melhor desempenho juntamente com custos mais baixos.

Este trabalho tem o interesse em realizar um comparativo entre algumas ferramentas de virtualização, utilizando ferramentas de banchmark, para servir como apoio para profissionais de informática em uma situação de escolha de um ambiente virtual.

2. Ambientes Virtualizados

Não é recente a aplicação de máquinas virtuais. A técnica que vem sendo empregada a muito tempo desde o início da computação, tendo como intuito estender o multiprocessamento, multi-programação e multi-acesso, transformando os sistemas em multi-ambiente (GOLDBERG, 2003, tradução nossa).

O conceito de máquina virtual consiste em um computador funcionar como se fossem vários.

A IBM considera uma máquina virtual uma total cópia da máquina subjacente em um ambiente isolado. A IBM desenvolveu seus sistemas virtuais de máquina para que uma aplicação possa rodar da mesma forma que no ambiente nativo (ROSE, 2004, tradução nossa). Na prática, máquina virtual consiste em um ambiente, onde também é muito conhecido como "sistema operacional para sistemas operacionais" ou hypervisor, construido por um monitor de máquina virtual (Virtual Machine Monitor – VMM). Um VMM consegue gerar uma ou mais máquinas virtuais que podem operar em uma mesma máquina real (SILVA, F., 2007).

2.1. Técnicas de Virtualização

São utilizadas uma variedade de técnicas na virtualização. Serão descritas algumas dessas técnicas nos próximos tópicos, tais como: virtualização completa, paravirtualização e recompilação dinâmica.

2.1.1. Virtualização completa

Nesse tipo de técnica, a idéia é montar uma arquitetura de hardware na máquina host para que se execute tanto SO quanto aplicativos juntamente. Desse modo, um sistema guest é capaz de executar suas instruções no hardware bruto de forma direta. Existe uma camada de software denominada hipervisor, que realiza um papel de vigilante, onde fornece uma ilusão a cada sistema guest de possuir sua cópia de hardware exclusiva (INOCENTE, 2009).

Entre as máquinas virtuais que aplicam a virtualização total temos o VMware, o VirtualBox e o KVM.

2.1.2. Paravirtualização

Esta técnica de virtualização dispõe para as máquinas virtuais uma Application Programming Interface (API), a qual assemelha-se com o hardware real. A paravirtualização exige modificações no sistema operacional para que o mesmo possa funcionar. Assim sendo, o sistema operacional que executa dentro da máquina virtual tem a falsa impressão de estar rodando sobre o hardware nativo da máquina e não em um simulado (SAFT, 2008).

Duas máquinas que utilizam esse tipo de técnica são o Denali e o Xen. (CASTRO, 2006).

2.1.3. Recompilação dinâmica

Esta técnica realiza compilação, no momento de execução, de forma que o código obtido represente instruções em linguagem de máquina. Também é utilizada para fins de representação portátil de um aplicativo, como os bytecodes de Java (LAUREANO, 2006).

Entre os softwares que aplicam a recompilação dinâmica temos o Qemu e o Virtual PC.

3. Ferramentas de Benchmark

Segundo Silva (2006) para medir o desempenho de aplicativos para computadores emprega-se os benchmarks, que possuem rotinas padronizadas para avaliar performance.

A seguir, serão descritos os benchmarks utilizados nesse trabalho e em seguida os testes realizados com os mesmos. As ferramentas são open source e os resultados são fornecidos de forma completa, sem restrições.

3.1.1. Scimark

Segundo Scimark (2010) esta é uma ferramenta de benchmark que verifica, por meio de cálculos, a performance de códigos numéricos presentes em aplicações científicas ou de engenharia. Consiste em cinco etapas: FFT, Jacobi Successive Over-relaxation, Sparse matrix-multiply, Monte Carlo integration, e dense LU factorization.

3.1.2. Bonnie++

O Bonnie++ é uma ferramenta que realiza uma variedade de avaliações em discos rígidos e em sistemas de arquivos. Os sistemas de arquivos operam em diversas formas e utilizam aplicações diferentes. A ferramenta testa algumas dessas operações, onde apresenta um valor para a quantidade de trabalho realizada por segundo e a porcentagem de CPU utilizada neste tempo (COKER, 2010).

Segundo Coker (2010) os testes realizados são saída sequencial, entrada sequencial e buscas randômicas.

3.1.3. PCMark04

É um benchmark constituído por partes de aplicativos reais e seu código pode ser livremente analisado por qualquer usuário. Possui testes pré-definidos que avaliam a memória, unidade de disco rígido, CPU e gráficos. Aplica Multithreading em função de aumentar o desempenho e o uso de recursos (CORPORATION, 2010, tradução nossa).

3.2. Intervalos de Confiança

A média dos resultados, ao se encaixar em um certo intervalo, apresenta um certo grau de confiança. Caso contrário, uma amostra, com tal média, dificilmente seria observada sem tal aspecto (RAMOS, 2008).

Segundo Ramos (2008) tendo uma função densidade de probabilidade f(x) juntamente com uma variável aleatória x, defini-se a seguinte a equação de Distribuição Contínua de Probabilidade:

$$F(b) - F(a) = P(a \leq X \leq b) = \int_a^b f(x)dx$$

Onde:

b = intervalo; a = intervalo.

Conforme Bernardinelli (2010) uma váriavel aleatória contínua X apresenta distribuição normal tendo μ e σ^2 como parâmetros, que representam média e variância da distribuição, $-\infty < \mu < +\infty$ e $0 < \sigma^2 < +\infty$, caso a função densidade de probabilidade seje:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}$$

Onde:

μ = valor esperado (média) de X com $-\infty < \mu < \infty$;

σ^2 = a variância de X com $\sigma^2 \geq 0$.

Existe a possibilidade da distribuição normal se encontrar na forma reduzida caso $\mu = 0$ e $\sigma = 1$, dessa forma, aplicando os cálculos, define-se a equação de Distribuição Normal Reduzida da seguinte maneira (RAMOS, 2008):

$$f(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{z^2}{2}\right)$$

Onde:

$z = (x - \mu) / \sigma$;

z = valor padronizado de x (número de desvios padrão com relação à média);

x = valor da V. A. Normal X;

σ = desvio padrão da V. A. Normal X;

μ = média da V. A. Normal X.

Fonte: RAMOS, N. (2008)

Figura 1. Função distribuição de probabilidade normal reduzida

Em relação ao modelo normal, o cálculo das probabilidades é realizado com o auxílio de tabelas, e para poupar a multiplicação de tabelas para todo par de valores (μ , σ^2), aplica-se uma transformação de forma que encaminhe sempre a conter uma varíavel de parâmetros (0, 1), ou seja, $\mu = 0$ (sendo a média) e $\sigma^2 = 1$ (como a variância) (BERNARDINELLI, 2010).

Assim, caso $X \sim N(\mu,\sigma^2)$, surge outra nova variável $Z = (X - \mu) / \sigma$, para qual explana-se que $\mu(Z) = 0$ e $\sigma^2(Z) = 1$. Então, $Z \sim N(\mu,\sigma^2)$ é chamada de Normal Padrão ou Normal Reduzida (BERNARDINELLI, 2010).

Conforme Bernardinelli (2010) o calculo de $P(a \leq X \leq b)$, requer a transformação:

$$P(a \leq X \leq b) = P((a-\mu)/\sigma \leq Y \leq (b-\mu)/\sigma)$$

São tabelados os valores em $P(0 \leq Z \leq z)$, $z \geq 0$.

O teorema central do limite assegura que a distribuição da média amostral está propensa à uma distribuição normal conforme o volume da amostra n tende para infinito. O encaminhamento para a normalidade fica mais ágil caso a distribuição dos dados é de forma simétrica; caso a distribuição ser assimétrica ou bimodal, a convergência demora mais tempo (CARBONARI et al, 2010).

Segundo Piana, Machado e Selau (2010) a técnica de intervalo de confiança pode ser aplicada para estimar a média μ de uma população X. Por meio de um estimador $\overline{X}$, se $X \sim N(\mu, \sigma^2)$, então, $\overline{X} \sim N(\mu, \sigma^2/n)$. Em relação a $\overline{X}$, tem-se:

$$Z = \frac{\overline{X}-\mu}{\sigma_{\overline{X}}}()/^{\sigma}\overline{X} = \frac{(\overline{X}-\mu)}{\sqrt{V(\overline{X})}} = \frac{(\overline{X}-\mu)}{\sqrt{\frac{\sigma^2}{n}}} = \frac{(\overline{X}-\mu)}{\frac{\sigma}{\sqrt{n}}}$$

Conforme a equação a seguir, Z possui distribuição normal, média zero e variância um:

$$Z = \frac{\overline{X}-\mu}{\frac{\sigma}{\sqrt{n}}} \sim N(0,1)$$

Onde:

$\overline{X}$ = é a média amostral;

μ = média de uma certa população Normal;

$\frac{\sigma}{\sqrt{n}}$ = desvio padrão da média amostral.

Para obter um intervalo de confiança em relação a média da população, μ deve ser isolado procedendo da seguinte maneira (PIANA; MACHADO; SELAU, 2010):

$$P\left(-Z_{\alpha/2} < \frac{\overline{X}-\mu}{\frac{\sigma}{\sqrt{n}}} < Z_{\alpha/2}\right) = 1-\alpha$$

$$P\left(-Z_{\alpha/2}\frac{\sigma}{\sqrt{n}} < \overline{X}-\mu < Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right) = 1-\alpha$$

$$P\left(-\overline{X}-Z_{\alpha/2}\frac{\sigma}{\sqrt{n}} < -\mu < -\overline{X}+Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right) = 1-\alpha$$

$$P\left[\left(-\overline{X}-Z_{\alpha/2}\frac{\sigma}{\sqrt{n}} < -\mu < -\overline{X}+Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right)\times(-1)\right] = 1-\alpha$$

$$P\left(\overline{X}+Z_{\alpha/2}\frac{\sigma}{\sqrt{n}} > \mu > \overline{X}-Z_{\alpha/2}\frac{\sigma}{\sqrt{n}}\right) = 1-\alpha$$

Exemplo (MEYER, 2000): Suponha que X represente a duração da vida de uma peça de equipamento. Admita-se que 100 peças sejam ensaiadas, fornecendo uma duração de vida média = 501,2 horas. Suponha-se que σ seja conhecido e igual a 4 horas, e que se deseje obter um intervalo de confiança de 95% para a média μ, teremos:

$2\Phi(z) - 1 =$ $= (1- \sigma) = 0,95$

$\Phi(z) = (0,95 + 1) / 2$

$\Phi(z) = 0,975$

Consultando a tabela de distribuição normal reduzida, chegamos à z = 1,96. O intervalo de confiança será:

I.C = (501,2 – 4/10 * 1,96; 501,2 + 4/10 * 1,96)

I.C = (500,4; 502,0)

Ao afirmar que (500,4; 502,0) constitui um intervalo de confiança de 95% para μ, não quer dizer que 95% das vezes a média amostral cairá nesse intervalo. Estamos afirmando que 95% das vezes a média estará contida no invervalo ($\overline{X}-\frac{z\sigma}{\sqrt{n}}$; $\overline{X}+\frac{z\sigma}{\sqrt{n}}$).

Conforme Barbetta, Reis e Bornia (2008) a partir de um intervalo de confiança, é possível calcular o tamanho mínimo de uma amostra da seguinte maneira:

$$n \geq \frac{z_y^2\sigma^2}{E_0^2}$$

Onde:

n= Tamanho mínimo a amostra;

z_y^2 = Z tabelado;

σ^2 = = Variância;

E_0^2 =Erro amostral máximo tolerado.

4. Avaliação das Máquinas Virtuais

Este estudo de caso consiste em duas etapas para avaliar as máquinas virtuais, onde primeiramente foi definida a metodologia de avaliação e em seguida a análise dos resultados obtidos.

4.1. Metodologia de Avaliação

Esta pesquisa descreve uma análise de desempenho experimental com a técnica de benchmark. Nesta etapa são descritas as configurações do equipamento, dos sistemas operacionais, dos softwares de comparação e das máquinas virtuais presentes no estudo. Também serão comentados os passos nos quais foi realizado a avaliação e as métricas em que se basearam.

4.1.1. Equipamento Utilizado

Para a realização do experimento, foram utilizados dois computadores, um com processador de 32 bits e outro com 64 bits, com a capacidade para executar todos os aplicativos nos testes.

O computador com processador de 32 bits possui a seguinte configuração:

- Processador: AMD Athlon (tm) XP 1500+ 1.33 GHz;
- Placa-mãe: PCCHIPS M810 DLU;
- Memória: 4 GB – DDR 400 ;
- Disco: Samsung ATA – 300 GB.

O segundo computador possui a seguinte configuração:

- Processador: Intel(R) Core(TM) 2 Quad Q800 2.33 GHz;
- Placa-mãe: ASUSTEK Computer INC. P5KPL-AM;
- Memória: 4 GB – DDR 2 ;
- Disco: Samsung SATA – 300 GB.

4.1.2. Sistemas Operacionais Utilizados

Foram escolhidos no experimento, SO que rodassem na máquina real e nas máquinas virtuais. Tanto para os sistemas hospedeiros e nas máquinas virtuais foram utilizados os SO Windows XP Professional com Service Pack 2 e Linux Ubuntu 9.10.

4.1.3. Máquinas Virtuais Utilizadas na Avaliação

Foram utilizadas nesta pesquisa, a técnica de virtualização completa (conceito mencionado no capítulo 2), onde o VMM realiza um intermédio entre o hardware e as máquinas virtuais, dessa forma ocorre uma simulação completa do hardware. Tal uso deve-se ao fato de comparar máquinas virtuais semelhantes. Foram, então, utilizados o VMware Server 2.0.2, VirtualBox 3 e KVM. Foi utilizado, também, o Virtual PC, que aplica a técnica de recompilação dinâmica, apenas para verificar o diferencial existente.

4.1.4. Benchmarks Utilizados

Foram escolhidos benchmarks que realizam avaliações nas plataformas, Linux e Windows. Na plataforma Windows, o benchmark utilizado foi o PCMark04 e na Linux foram utilizados o Scimark e o Bonnie++.

4.1.5. Etapas do processo de avaliação

A avaliação de desempenho foi aplicada no ambiente nativo hospedeiro e nas máquinas virtuais na plataforma Linux e Windows. No ambiente nativo, foram aplicados os testes apenas para fornecer uma noção dos resultados. A avaliação de desempenho, nessa pesquisa, é focada nas máquinas virtuais.

O primeiro passo foi instalar o SO Windows XP em ambas as máquinas físicas, cujas configurações já foram mencionadas. Em seguida foi instalado o PCMark04 nos computadores e executado 10 vezes consecutivas para calcular a quantidade de amostras mínimas necessárias de acordo com a fórmula do Tamanho Mínimo de uma Amostra a partir de um Intervalo de Confiança . Os demais benchmarks também foram executados o mesmo número de vezes, inicialmente, para obter o número de amostras mínimas e calcular a média. Estimou-se que o erro máximo tolerável foi de 50% do desvio padrão. Em seguida, foi instalado o Linux em ambas as máquinas físicas e também o pacote openjdk-6-jdk (necessário para executar o Scimark) e o Bonnie++. Após as instalações, as 2 ferramentas de avaliação foram utilizadas.

O próximo passo foi instalar, no PC com processador de 64 bits, o VMware, o VirtualBox e o KVM no sistema hospedeiro Linux e o Virtual PC no sistema hospedeiro Windows criando duas máquinas virtuais em cada ferramenta. Somente o VirtualBox e o VMware foram instalados e testados em ambos os computadores para verificar o desempenho com diferentes processadores.Cada VMM, enfim, ficou com uma máquina virtual com SO Windows e uma com Linux.

A etapa seguinte foi instalar e rodar os benchmarks nas máquinas virtuais presentes e salvar os resultados gerados.

Foi determinado uma nomenclatura para identificar as máquinas virtuais, onde os dois primeiros dígitos na descrição representam o processador da máquina hospedeira, os três caracteres seguintes representam o sistema de virtualização e as duas últimas identificam o SO da máquina virtual. Para as máquinas físicas, foi adotado uma nomenclatura diferente, onde os dois primeiros dígitos representam o processador da máquina as três primeiras letras indicam que é um sistema físico e as demais representam o SO utilizado.

As máquinas virtuais criadas, foram configuradas com 1 GB de memória RAM e 3 GB de espaço em disco. Primeiramente, foram testadas as máquinas virtuais com SO Windows e em seguida as com o Linux em ambos os computadores. Posteriormente, foi configurado a memória RAM para ter 2 GB nas duas máquinas virtuais do VirtualBox presentes no computador com processador de 64 bits. Tal aumento foi realizado para verificar o desempenho dos ambientes virtuais com mais memória RAM.

Ao efetuar os testes, foram desativadas as conexões de rede, descansos de tela e recursos de economia de energia nos ambientes virtuais e físicos para todos os benchmarks. As conexões de rede ficaram ativas ao executar o Scimark, em função de ser um applet executado online.

O número de ambientes virtuais avaliados, segundo constam na Tabela 1, foram doze. Porém, na máquina com processador de 64 bits, os dois primeiros ambientes, relativos ao

VirtualBox, foram avaliados duas vezes em função do aumento de memória RAM para 2 GB. Dessa forma, o total de ambientes totaliza quatorze. Como em cada máquina virtual que contém SO Linux foi executado dois benchmarks, o total de testes realizados foram vinte e um. Somando esse resultado com os dos ambientes nativos hospedeiros dos dois computadores, sendo seis ambientes, totaliza-se em vinte e sete avaliações.

PROCESSADOR DA MÁQUINA	MÁQUINA VIRTUAL	SO	NOMENCLATURA
64 bits	VirtualBox	Windows	64VBXWI
		Linux	64VBXLI
	Vmware	Windows	64VMWWI
		Linux	64VMWLI
	KVM	Windows	64KVMWI
		Linux	64KVMLI
	Virtual PC	Windows	64VPCWI
		Linux	64VPCLI
32 bits	VirtualBox	Windows	32VBXWI
		Linux	32VBXLI
	VMware	Windows	32VMWWI
		Linux	32VMWLI

Tabela 1. Ambientes virtuais avaliados

4.1.6. Análise dos Resultados Obtidos

Na execução dos testes de desempenho, para cada ambiente virtual e físico, a quantidade de amostras necessária, segundo a Fórmula do Tamanho Mínimo de uma Amostra a partir de um Intervalo de Confiança, coincidiu em 16. Ou seja, primeiramente foram executados 10 vezes cada benchmark para estimar o número mínimo de amostras necessárias, onde, coincidentemente, resultou em 16.

A seguir segue o cálculo efetuado para o teste File Compression do benchmark PCMark04, segundo a fórmula tamanho mínimo de uma amostra, que resultou em 16 amostras para as avaliações. Os dez primeiros resultados obtidos foram: 1936247,70; 1948401,20; 1943130,00; 1940669,90; 1951803,90; 1952215,20; 1950445,40; 1951784,20; 1953015,70 e 1950668,40.

(Z tabelado² * variância) / Erro²

(1,96² * 30027818,30) / 2739,89²

(3,8416 * 30027818,30) / 7506997,22

115354866,79 / 7506997,22

15,37 ≅ 16

Para obter a variância, inicialmente calculou-se a média, que foi 1947838,16. Em seguida calculou-se a variância, a qual é a soma dos quadrados das diferenças entre cada elemento do conjunto e a média, dividido pelo número de elementos.

variância = ((1936247,70-1947838,16)² + (1948401,20-1947838,16)² + (1943130,00-1947838,16)² + (1940669,90-1947838,16)² + (1951803,90- 1947838,16)² + (1952215,20-1947838,16)² + (1950445,40-1947838,16)² + (1951784,20-1947838,16)² + (1953015,70 - 1947838,16)² + (1950668,40-1947838,16)²) / 10

variância = (134338763,01 + 317014,04 + 22166770,59 + 51383951,43 + 15727093,75 + 19158479,16 + 6797700,42 + 15571231,68 + 26806920,45 + 8010258,46) / 10

variância = 30027818,30

O próximo passo foi obter o erro, onde calculou-se primeiramente o desvio padrão e em seguida dividiu-se o resultado por dois. O desvio padrão é dado pela raiz quadrada da variância, sendo 5479,77. Dividindo por dois, obteve-se 2739,89.

Para um intervalo de confiança de 95% para a média μ, obteve-se:

$2\Phi(z) - 1 = = (1- \sigma) = 0,95$

$\Phi(z) = (0,95 + 1) / 2$

$\Phi(z) = 0,975$

Consultando a tabela de distribuição normal reduzida, chegamos à z tabelado= 1,96.

O benchmark PCMark04, foi utilizado para avaliar ambientes que utilizaram o SO Windows XP. Foram descartados os testes Virus Scanning e Physics Calculation and 3D, pois em alguns casos os resultados obtidos eram zerados.

A Figura 2 apresenta os resultados obtidos com o PCMark04. Pode-se observar que no PC com processador de 64 bits, o VMware apresentou um melhor desempenho em relação ao demais ambientes em 63,6 % dos testes efetuados com o benchmark. Em segundo lugar ficou o Virtual PC, em terceiro o VirtualBox e em quarto o KVM.

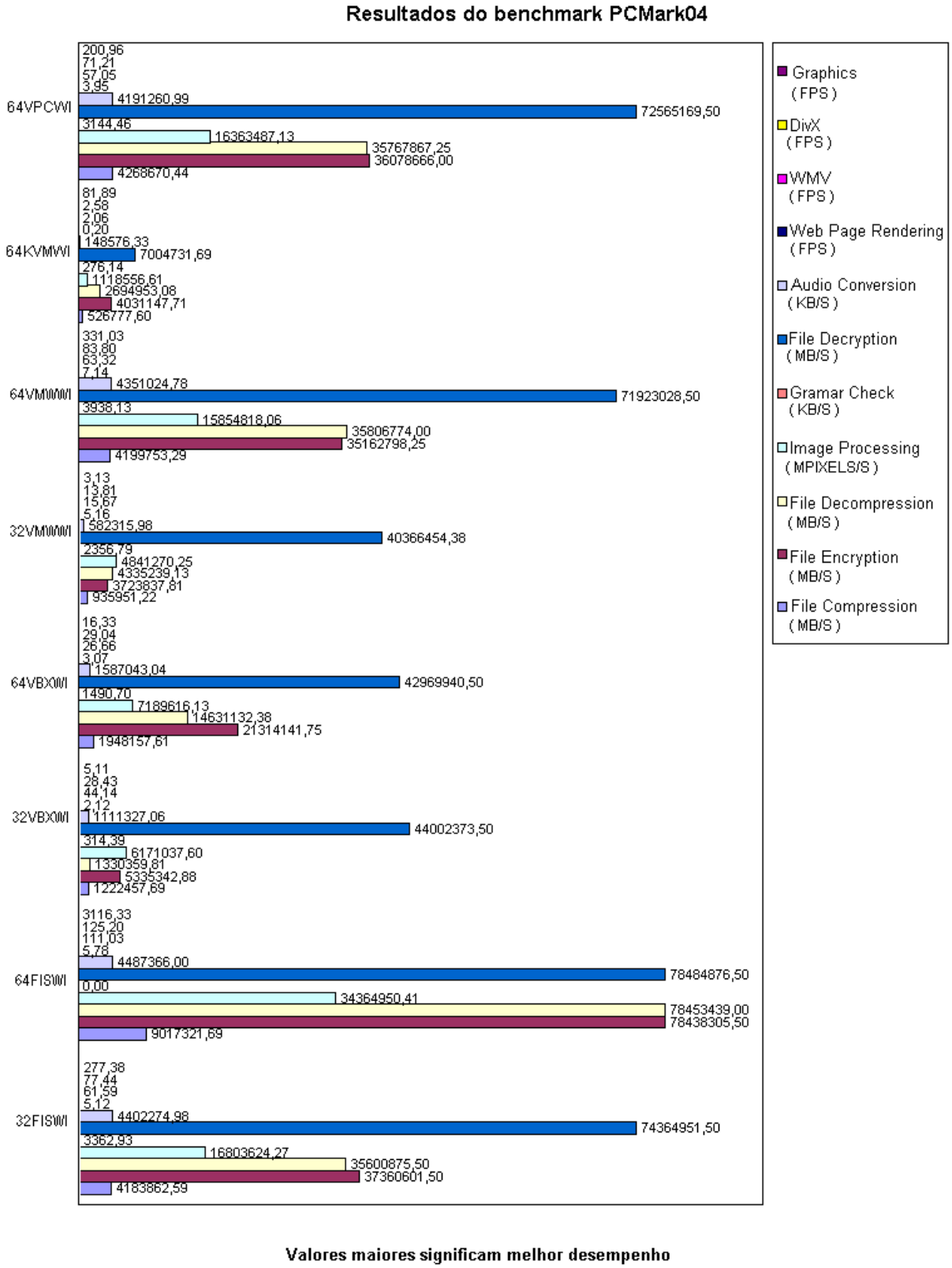

Figura 2. Resultados do PCMark04

Em relação aos ambientes virtuais presentes no PC com processador de 32 bits, o VirtualBox apresentou melhor desempenho na maioria dos testes com exceção do Web Page Rendering, Grammar Check, File Decompression.

Foi avaliado o desempenho do VirtualBox, presente no PC com processador de 64 bits, com o PCMark04, ao ser aumentado a memória RAM da máquina virtual para 2 GB. Conforme a Figura 3, em todos os testes pode ser observado um ganho de desempenho, exceto no Web Page Rendering.

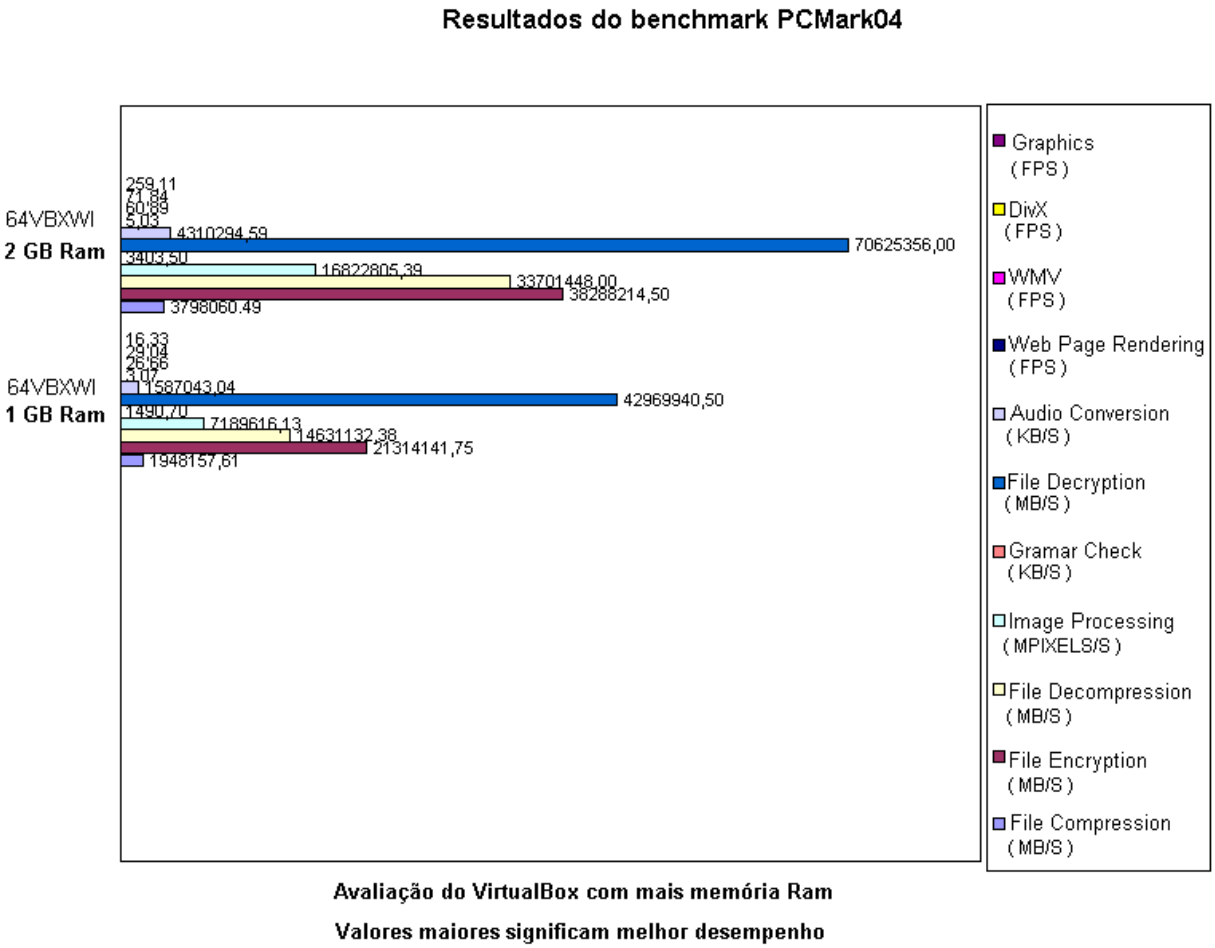

Figura 3. Resultados do PCMark04 com aumento de memória no VirtualBox

Em seguida, o Bonnie++ foi utilizado para avaliar ambientes que utilizaram o SO Linux Ubuntu 9.10. Dentre os testes deste benchmark, conforme já mencionados no capítulo 3, foram utilizados Sequencial Output com seus subtestes Per Char, Block e rewrite; e Sequencial Input com seus subtestes Per Char e Block. .

A Figura 4 apresenta os resultados obtidos com o Bonnie++. O pior desempenho nesse caso foi no ambiente físico do PC com processador de 32 bits. O ambiente físico no PC com processador de 64 bits, em relação aos dois ambientes virtuais que possui, obteve sucesso em 60% dos testes. Obteve melhores resultados nos testes Sequencial Output Block, Sequencial Output Rewrite e Sequencial Input block.

Em relação aos ambientes virtuais presentes no PC com processador de 32 bits, o VirtualBox apresentou melhor desempenho em 60% dos testes, dentre os quais estão o Sequencial Output Per Char, Sequencial Output Rewrite e Sequencial Input Per Char.

No PC com processador de 64 bits, o VirtualBox também apresentou melhor desempenho, destacando-se em 60% dos testes. Os testes foram os mesmos com esta máquina virtual no PC com processador de 32 bits. Em segundo lugar ficou o VMware, em terceiro o Virtual PC e em quarto o KVM.

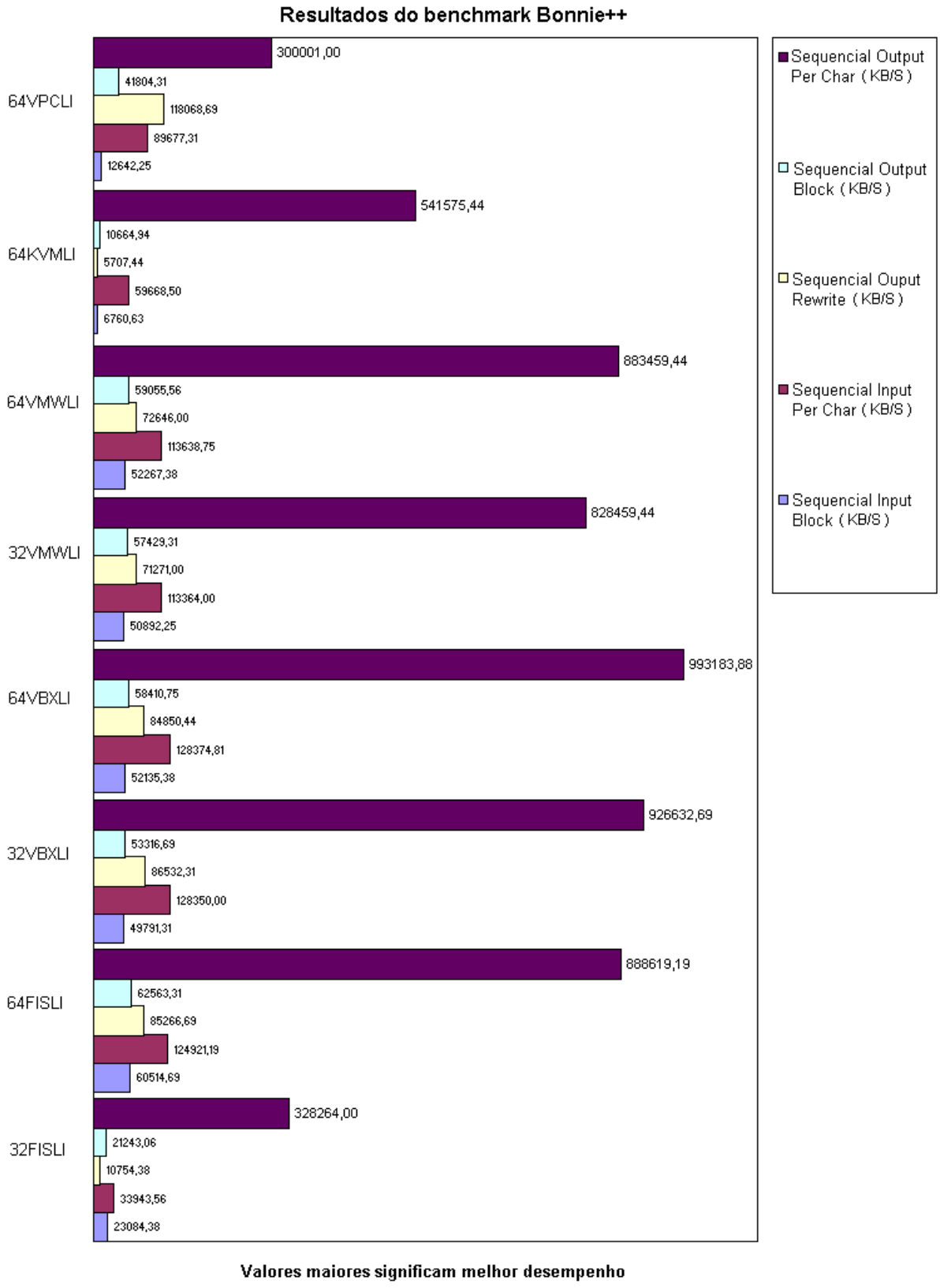

Figura 4. Resultados do Bonnie++

Enfim, do mesmo modo realizado no PCMark04, aumentou-se a memória Ram do VirtualBox, no PC com processador de 64 bits, para 2 GB. Inesperadamente, o desempenho

obtido decaiu em 80 % dos testes. O único teste em que não houve perda foi o Sequencial Input Block. Os resultados podem ser conferidos na Figura 5.

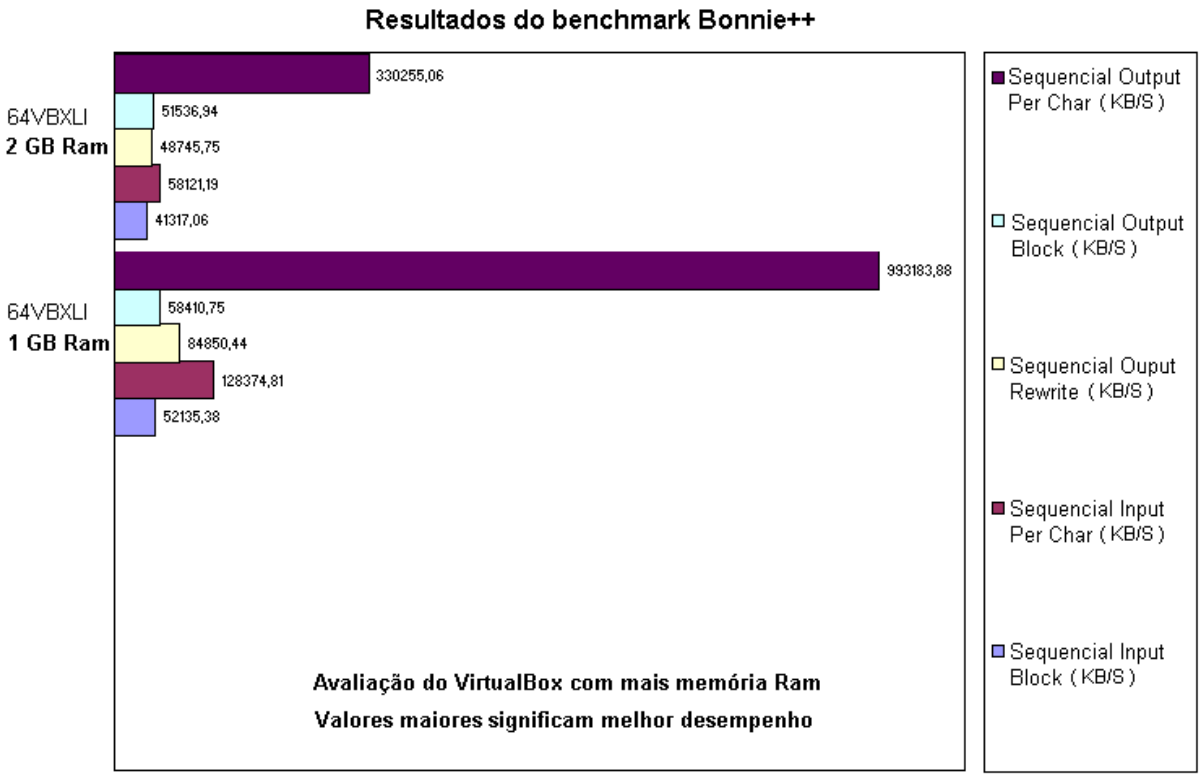

Figura 5. Resultados do Bonnie++ com aumento de memória no VirtualBox

Concluído os testes com o Bonnie++, o Scimark foi utilizado, também, para avaliar ambientes que utilizaram o SO Linux Ubuntu 9.10. Dentre os testes deste benchmark, conforme já mencionados no capítulo 3, foram utilizados FFT, Jacobi Successive Over-relaxation, Sparse matrix-multiply, Monte Carlo integration, e dense LU factorization.

A Figura 6 apresenta os resultados obtidos com o Scimark. O ambiente virtual que obteve melhor desempenho no computador com PC de 32 bits, foi o VMware. Com 66.7% de destaque nos testes, obteve desempenho mais elevado no LU, Sparse, FFT e Scimark.

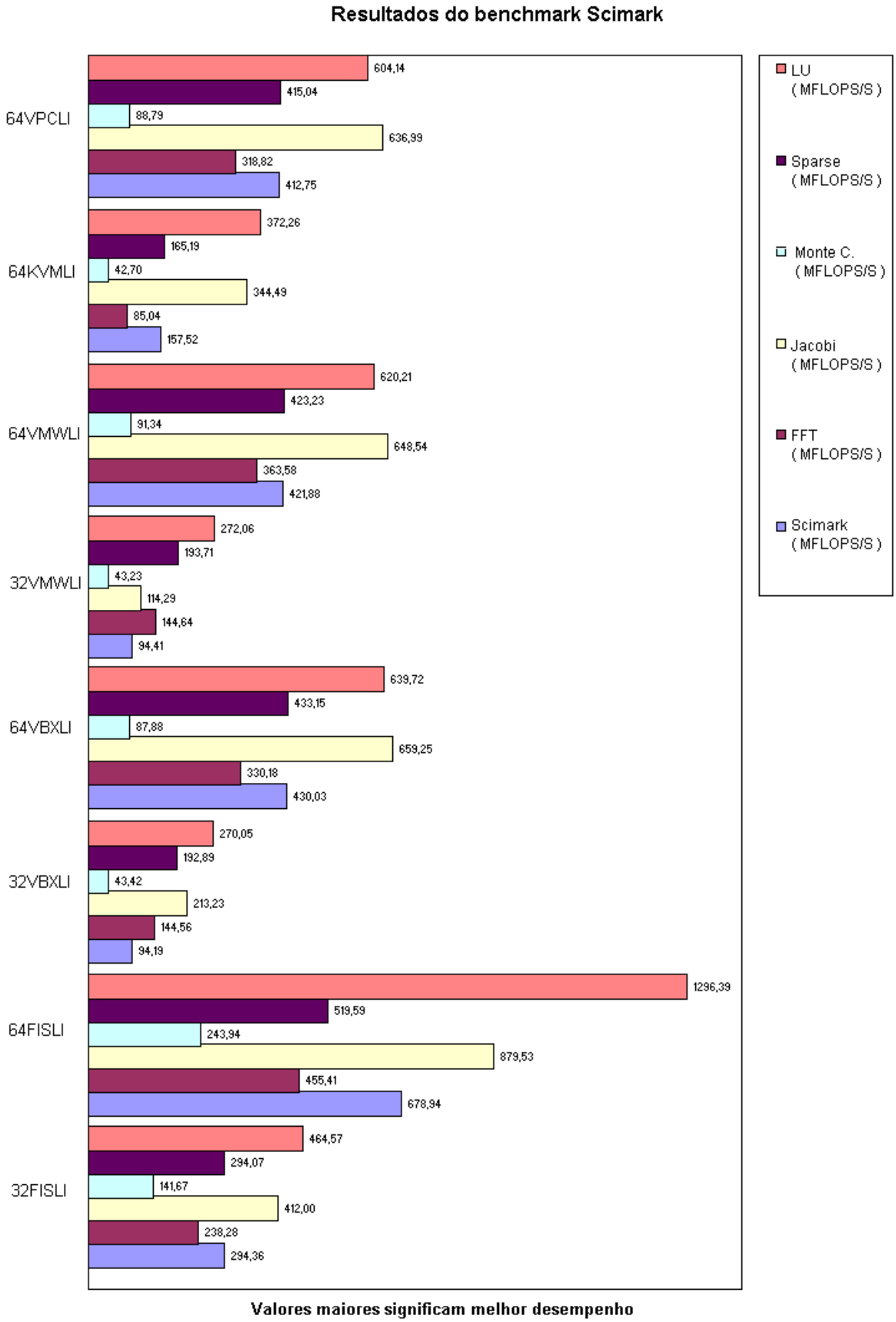

Figura 6. Resultados do Scimark

No outro PC, o que sobressaiu nas avaliações foi o VirtualBox, também 66.7% de sucesso e foram nos testes LU, Sparse, Jacobi e Scimark. Em segundo lugar ficou o VMware, em terceiro o Virtual PC e em quarto o KVM.

Seguindo a metodologia definida, foi aumentado a memória RAM nesta máquina virtual, no PC com processador de 64 bits, para 2 GB. Conseqüentemente, o desempenho obtido decaiu em 100 % dos testes. Os resultados podem ser conferidos na Figura 7.

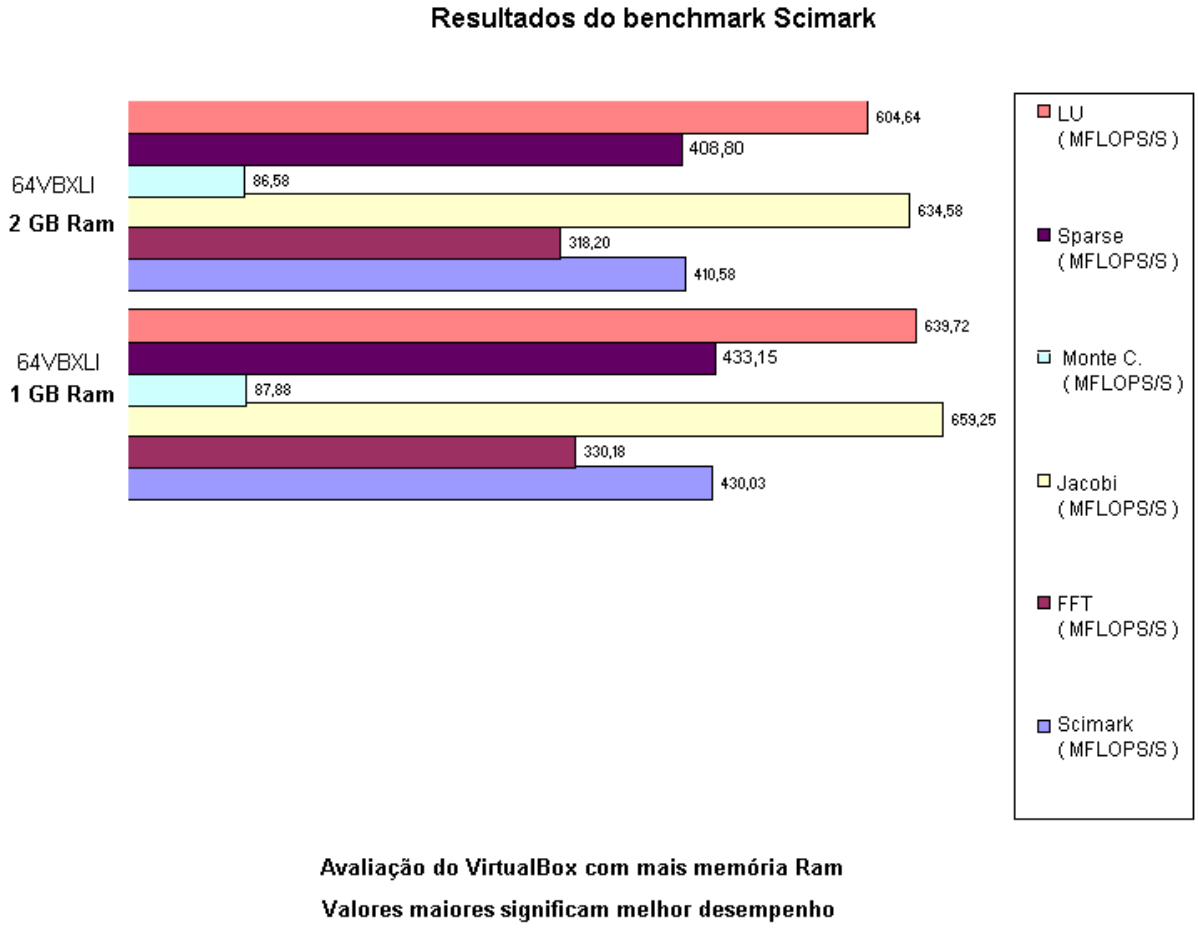

Figura 7. Resultados do Scimark com aumento de memória no VirtualBox

5. Conclusão

Analisou-se que a virtualização está sendo bastante utilizada hoje em dia, em função de agregar diversos sistemas operacionais em um único hardware, ocasionando melhor aproveitamento dos recursos disponíveis, proporcionando economia, flexibilidade, segurança, gerenciabilidade de aplicações e um ótimo isolamento de falhas.

A importância de comparar máquinas virtuais é de extrema valia para determinar a melhor opção no uso para um fim específico, sendo que, uma das maneiras de realizar comparações é através de softwares de benchmark.

Nesta pesquisa, foi realizado uma análise de desempenho experimental com a técnica de benchmark com o VirtualBox, VMware Server, Virtual PC e KVM cada um com suas peculiaridades, seja em termos de desempenho, facilidade de instalação ou flexibilidade de configuração.

Para tal análise, foram utilizados os benchmarks PCmark04, Bonnie++ e Scimark, realizando uma série de testes replicados segundo a Fórmula do Tamanho Mínimo de uma Amostra a partir de um Intervalo de Confiança.

Com o PCmark04, os resultados demonstram que uma máquina virtual pode se comportar diferente em processadores de 32 e 64 bits. Executando em um processador de 32 bits, o melhor desempenho foi com o VirtualBox. Já com um processador de 64 bits, o VMware sobressaiu nos resultados. Com o Bonnie++, o VirtualBox demonstrou um melhor desempenho nos resultados gerados. Os testes com o Scimark em um processador de 32 bits o melhor desempenho foi com o VMware. Já com um processador de 64 bits, o VirtualBox obteve melhor performance.

Como sugestão para trabalhos futuros, há a possibilidade de avaliar o desempenho de máquinas virtuais que utilizam a técnica de Paravirtualização ou Recompilação Dinâmica, juntamente com outros SO hospedeiros e convidados e outras ferramentas de benchmark.

6. Referencias

BARBETTA, Pedro Alberto; REIS, Marcelo Menezes; BORNIA, Antonio Cezar. . **Estatística: para cursos de engenharia e informática.** 2. ed São Paulo: Atlas, 2008. 410 p.

BERNARDINELLI, Regina Maria Sigolo. **Estatística.** Disponível em: <http://teleduc.unisa.br/~teleduc/cursos/diretorio/apoio_880_79///apostila.pdf>. Acesso em: 29 abr. 2010.

CARBONARI, Maria Elisa Ehrhardt et al. **Auto-Avaliação de IES - Base estatística para Índices de Desempenho.** Disponível em: <http://sare.unianhanguera.edu.br/index.php/rcext/article/viewFile/396/396>. Acesso em: 30 abr. 2010.

CASTRO, Arthur Bispo de. **Máquinas virtuais em ambientes seguros.** 2006. 85 f. Dissertação (Mestrado) - Curso de Ciência da Computação, Instituto de Computação Universidade Estadual de Campinas, Campinas Sp, 2006. Disponível em: <http://www.las.ic.unicamp.br/paulo/teses/20060210-MSc-Arthur.Bispo.de.Castro-Maquinas.virtuais.em.ambientes.seguros.pdf>. Acesso em: 05 dez. 2009.

COKER. **Bonnie++.** Disponível em: <http://www.coker.com.au/bonnie++/readme.html>. Acesso em: 23 fev. 2010.

CORPORATION, Futuremark. **PCMark04: PC Performance Analysis.** Disponível em: <http://www.futuremark.com/pressroom/companypdfs/PCMark04_Whitepaper>. Acesso em: 12 set. 2010.

GOLDBERG, R. P.. **ARCHITECTURE OF VIRTUAL MACHINES.** Disponível em: <http://www.cse.psu.edu/~bhuvan/teaching/spring06/papers/goldberg.pdf>. Acesso em: 19 mar. 2010.

INOCENTE, Emerson Meneses. Green It – **Processo de Redução de Consumo de energia elétrica baseado na virtualização.** Disponível em: <http://tconline.feevale.br/tc/files/0002_1966.pdf>. Acesso em: dez. 2009.

LAUREANO, Marcos. **Máquinas virtuais e emuladores: conceitos, técnicas e aplicações.** São Paulo: Novatec, 2006. 184 p.

PIANA, Clause Fátima de Brum; MACHADO, Amauri de Almeida; SELAU, Lisiane Priscila Roldão. **Estatística Básica.** Disponível em: <http://minerva.ufpel.edu.br/~markus.stein/Apostila_EB.pdf>. Acesso em: 10 maio 2010.

MEYER, Paul L.. **Probabilidade Aplicações à Estatistica.** 2ª São Paulo: Ltc, 2000.

NANDA, Susanta; CHIUEH, Tzi-cker. **A Survey on Virtualization Technologies.** Disponível em: <http://www.ecsl.cs.sunysb.edu/tr/TR179.pdf>. Acesso em: 17 mar. 2010

POTRICH, Juliano. **APRIMORAMENTO DO ESCALONADOR CREDIT.** 2008. 1 v. Dissertação (Bacharel) - Curso de Sistemas de Informação, Universidade Católica do Rio Grande do Sul, Porto Alegre, 2008. Disponível em: <http://caioba.pucrs.br/teo/ojs/index.php/graduacao/article/viewFile/2858/3152>. Acesso em: 05 fev. 2010.

RAMOS, Nelson Azoubel. **Avaliação e Comparação de Desempenho de Computadores: Metodologia e Estudo de Caso.** 2008. 1 v. Dissertação (Bacharel) - Curso de Engenharia da Computação, Universidade Federal de Pernambuco, Recife, 2008. Disponível em: <http://www.google.com.br/url?sa=t&source=web&ct=res&cd=2&ved=0CA0QFjAB&url=http%3A%2F%2Fwww.cin.ufpe.br%2F~tg%2F2008-2%2Fnar.pdf&ei=r5nZS4DZOcSyuAfDxoSpDw&usg=AFQjCNEamiKwV5aFLBsMSNST7cNJLAUQjg&sig2=ug7bfSyAh_dX_7irqlKLPQ>. Acesso em: 29 abr. 2010.

ROSE, Robert. **Survey of System Virtualization Techniques.** Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=511164D5184345848FBDB5926534EE47?doi=10.1.1.58.9811&rep=rep1&type=pdf>. Acesso em: 05 dez. 2010.

SAFT, Luana Sandrini. **Virtualização de Software.** 2008. 1 v. Tese (Bacharel) - Curso de Sistemas de Informação, Universidade Federal de Santa Catarina, Florianópolis Sc, 2008. Disponível em: <http://projetos.inf.ufsc.br/arquivos_projetos/projeto_764/TCC_projetoI.pdf>. Acesso em: 08 jan. 2009.

SCIMARK. **About SciMark 2.0.** Disponível em: <http://math.nist.gov/scimark2/about.html>. Acesso em: 23 fev. 2010.

SILVA, Fábio Rodrigo Albuquerque da. **Vantagens e desvantagens na utilização de software de virtualização em servidores de empresas de pequeno e médio porte: Estudo de casos em faculdades particulares no Recife.** 2007. 1 v. Tese (Bacharel) - Faculdade Santa Maria, Recife PE, 2007. Disponível em: <http://www.nogueira.eti.br/profmarcio/obras/Fabio%20-%20Virtualizacao.pdf>. Acesso em: 05 nov. 2009.

SILVA, Ricardo Czelusniak da. **Benchmark em banco de dados multimídia: Análise de desempenho em recuperação de objetos multimídia.** 2006. 1 v. Dissertação (Mestre) - Universidade Federal do Paraná, Curitiba, 2006. Disponível em: <http://dspace.c3sl.ufpr.br:8080/dspace/bitstream/1884/4683/1/disserta_ricardo.pdf>. Acesso em: 24 fev. 2010.

SILVA, Rodrigo Ferreira da. **Virtualização de Sistemas Operacionais.** 2007. 1 v. Tese (Graduado) - Instituto Superior de Tecnologia em Ciências da Computação de Petrópolis, Petrópolis, 2007. Disponível em: <http://www.lncc.br/~borges/doc/Virtualizacao%20de%20Sistemas%20Operacionais.TCC.pdf>. Acesso em: 17 nov. 2009