RECONFIGURAÇÃO DE AMBIENTES VIRTUALIZADOS

RECONFIGURATION OF VIRTUALIZED ENVIRONMENTS

Avelino Francisco Zorzo [*]
Ana T. Winck [**]
Elder Macedo Rodrigues [**]
Leandro Teodoro Costa [***]
Duncan Dubugrás Ruiz [****]

Resumo

Este trabalho propõe a realocacão dinâmica de recursos em ambientes virtualizados a partir de requisitos derivados de Acordos de Nível de Serviço (*Service Level Agreements* - SLAs). A realocação utiliza algoritmos de mineração de dados sobre os resultados da execução de benchmarks. Estes algoritmos produzem modelos preditivos que sugerem, a partir de uma determinada configuração das máquinas virtuais, qual o melhor conjunto de parâmetros a ser modificado para melhorar o desempenho de todo o ambiente virtualizado. Estes modelos preditivos são utilizados por um subsistema de reconfiguração que combina os mesmos com as políticas estabelecidas nos SLAs.

Palavras-chave: Acordo de Nível de Serviço. Máquinas Virtuais. Xen.

Abstract

This work proposes a strategy to dynamically allocate resources on virtualized environments. This allocation is based on Service Level Agreements (SLAs) and uses data mining algorithms executed over a set of results generated by benchmarks. The data mining algorithms produce a predictive model that indicates, from the current virtual machines configuration, the best set of parameters to be changed in order to improve the performance of the whole virtualized system. This paper also presents a subsystem that uses the predictive model and the SLAs to reconfigure the virtualized environment.

Keywords: Service Level Agreement. Virtual Machines. Xen.

* Doutor em Ciência da Computação pela Universidade de Newcastle upon Tyne, Inglaterra. Bolsista de produtividade em pesquisa do CNPq. Diretor da Faculdade de Informática da PUCRS. avelino.zorzo@pucrs.br
** Doutorando em Ciência da Computação pelo Programa de Pós-graduação em Ciência da Computação na PUCRS.
*** Mestrando em Ciência da Computação pelo Programa de Pós-graduação em Ciência da Computação na PUCRS.
**** Doutor em Ciência da Computação pela Universidade Federal do Rio Grande do Sul. Professor da Faculdade de Informática da PUCRS.

1. Introdução

O uso de sistemas computacionais tem se tornado algo comum no cotidiano da sociedade. Cada vez mais vemos sistemas computacionais em diversos setores da economia. É difícil, senão impossível, imaginarmos a sociedade atual sem apoio de sistemas computacionais. Um exercício interessante que cada pessoa pode fazer é o de imaginar quanto de computação existe desde que saímos de casa até chegar ao trabalho. Se pegarmos um elevador, com certeza tem algum sistema computacional controlando o mesmo. Se abastecermos, com certeza a bomba de combustível será controlada por um sistema computacional. Ao pararmos em um semáforo, existe um sistema computacional controlando o mesmo. E assim por diante.

Diante de uma sociedade dependente de sistemas computacionais como a que vivemos, cada vez mais, temos diversos computadores, milhares deles, executando estes sistemas. Muitas vezes, estes computadores não são utilizados em todo seu potencial computacional, mas mesmo assim consomem recursos (espaço físico, energia, pessoas, entre outros). O ideal seria podermos diminuir o número de computadores, e, portanto, os recursos necessários, e utilizar o máximo de seu poder computacional. Uma técnica que tem auxiliado para atingirmos este objetivo é o uso de virtualização.

A virtualização é uma prática que tem sido explorada para permitir a execução simultânea de múltiplos Sistemas Operacionais (SOs) em um único computador (MERGEN, 2006). Diversas soluções de virtualização foram apresentadas nos últimos anos. Entre elas, o Qemu (BELLARD, 2005), Oracle VM (SINGH, 2009), Xen (BARHAMAL, 2003), Virtual Server (O'ROURKE, 2001) e o VMware (SUGERMAN, 2001) são algumas das ferramentas de virtualização utilizadas por grande parte das empresas. Apesar do VMWare ser um dos sistemas mais utilizados comercialmente, o Xen tem se destacado por ser um sistema de código aberto e portanto com possibilidade de melhorias pela comunidade científica de maneira muito mais rápida. O Xen é um paravirtualizador com uma arquitetura representada como uma camada de abstração sobre o *hardware* e que permite a execução de diversas máquinas virtuais (VMs), cada uma com seu próprio SO. Sua empregabilidade tem se destacado em *datacenters*, pois o conjunto de serviços oferecido aos clientes pode utilizar uma mesma infraestrutura computacional (CUNHA, 2007), reduzindo custos e melhorando a rentabilidade do negócio. Entretanto, como estes ambientes são altamente dinâmicos, eles precisam que os recursos compartilhados sejam constantemente ajus-

tados, para uma melhor utilização, sem prejudicar o serviço fornecido aos clientes.

Em geral, os *datacenters* estabelecem acordos de níveis de serviço (*Service Level Agreements* – SLAs) com seus clientes. Um SLA é a declaração de expectativas e obrigações que existem no relacionamento de negócio entre duas organizações: o provedor do serviço e seu consumidor (GUPTA, 2006). Esta declaração especifica os níveis de qualidade de serviço que o fornecedor se compromete em disponibilizar, bem como as cláusulas legais e as consequências para cada parte se houver descumprimento destes deveres. Dessa forma, o SLA estabelece o nível de serviço requisitado, sendo utilizado para estabelecer uma compreensão comum sobre serviços, prioridades e responsabilidades entre provedores e clientes.

Para propor melhorias no que se refere ao desempenho do Xen, respeitando políticas de SLA, a seguinte questão pode ser explorada: em qualquer momento, qual a melhor alocação de recursos para uma máquina Xen quando várias VMs estão sendo executadas em um mesmo computador?

A Figura 1 exemplifica o problema em um ambiente virtualizado composto por quatro VMs (1, 2, 3, 4) executando em um mesmo computador. Cada uma dessas VMs tem necessidades diferentes de consumo, por exemplo, CPU e memória. Na Figura 1, as necessidades das VMs são ilustradas pelo tamanho do retângulo que representa cada VM. As linhas tracejadas representam a quantidade de recursos disponibilizada para cada VM. A área hachurada equivale aos recursos ainda disponíveis no computador e que poderiam ser alocados às VMs. Na Figura 1 temos a representação de uma situação típica em um ambiente virtualizado (a) e a situação ideal (b). A Figura 1a representa uma alocação de recursos sem a preocupação com SLAs, ou melhor, uso dos recursos de acordo com as necessidades de cada VM. Analisando-se a Figura 1a percebe-se que a VM 2 está subutilizando os recursos a ela disponibilizados, enquanto que as necessidades das VMs 1 e 3 é maior do que os recursos que elas têm à disposição. Desta forma, a VM 1 e 3 terão seu desempenho afetado, pois não terão os recursos necessários para a demanda que possuem, ao mesmo tempo que existem recursos disponíveis no computador. Para melhorar a utilização dos recursos disponíveis e também o desempenho global do Xen, espera-se que os recursos possam ser adequadamente redistribuídos, respeitando políticas de SLA previamente definidas. Este modelo ideal está representado na Figura 1b, onde é possível notar, pela linha tracejada, que os recursos computacionais estão distribuídos proporcionalmente à demanda de cada VM, sem excesso ou escassez, respeitando a disponibilidade global do ambiente.

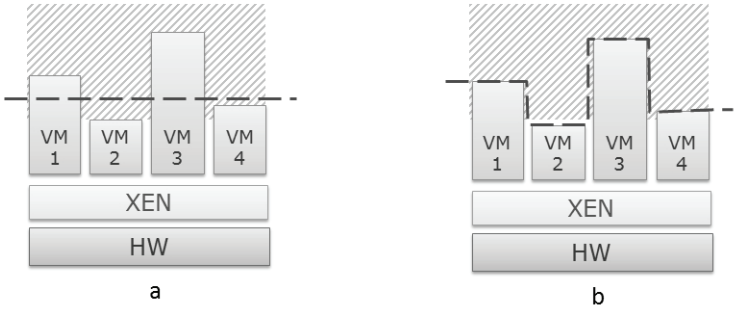

Figura 1. Realocação de recursos no *Xen*

Para que a distribuição de recursos seja efetuada corretamente, é importante ter conhecimento sobre a demanda de cada VM e a disponibilidade do ambiente como um todo. Uma possível estratégia é fazer uso de resultados de testes de desempenho, obtidos através da execução de *benchmarks* sobre cada sistema operacional virtualizado. Através de *benchmarks* é possível obter dados sobre a real capacidade computacional à disposição em cada VM. Em geral, *benchmarks* geram resultados na forma de um relatório tabular, contendo métricas de desempenho de aplicações, sistemas operacionais ou equipamentos. Seus resultados servem como base para que sejam mapeados os níveis de recursos consumidos por uma aplicação sob determinada carga de trabalho. Contudo, devido à diversidade de configurações vigentes, bem como ao grande volume de métricas retornadas pelos *benchmarks*, torna-se difícil examinar, interpretar e aferir seus resultados manualmente. Uma das formas de minimizar o tempo de análise dos resultados de um grande conjunto de dados é através da utilização de algoritmos de mineração de dados. Estes algoritmos podem gerar informações relevantes a partir dos dados coletados.

Trabalhos recentes, (JUNG, 2006); (PAREKH, 2006); (UDUPI, 2007) e (CUNHA, 2007), têm aplicado técnicas de mineração de dados para análise de desempenho em aplicações multicamadas. Por exemplo, (JUNG, 2006) e (PAREKH, 2006) buscam apontar gargalos e identificar, em que situações e com quais configurações pode ocorrer queda de desempenho nas aplicações. Outros dois trabalhos, (UDUPI, 2007) e (CUNHA, 2007), fazem uso de mineração de dados para definir políticas de SLA. Neste último trabalho, é citada a utilização do Xen em um *data-center* como cenário. Apesar dos trabalhos supracitados proporem métodos para garantir determinada qualidade de serviço em ambientes virtualizados, nenhum deles possibilita manter este nível em situações de sobrecarga das VMs da estrutura, nem a possibilidade de utilizar recursos ociosos, estando as VMs sobrecarregadas ou não.

Assim, este artigo apresenta uma nova forma de distribuir recursos de um computador entre diversas máquinas virtuais. A proposta utiliza um conjunto de recomendações de novas configurações existentes em um modelo preditivo gerado por um algoritmo de mineração de dados. O modelo preditivo é gerado a partir de um conjunto de dados produzidos pelos *benchmarks* TPC-W (TPC-W, 2008) e Unixbench (UNIXBENCH, 2007).

Este artigo está estruturado da seguinte forma: a Seção 2 apresenta o modelo do processo de realocação de recursos proposto, detalhando suas etapas. Na Seção 3 é mostrado como o processo proposto é validado. A Seção 4 apresenta os resultados obtidos com o modelo proposto. E, por fim, a Seção 5 relata algumas conclusões.

2. Modelo do Processo de Realocação de Recursos Proposto

Esta seção apresenta o processo proposto para realocação de recursos para máquinas virtuais (VMs). O processo como um todo pode ser visualizado na Figura 2. Primeiramente são executados *benchmarks* sobre distintas configurações do ambiente virtualizado pelo Xen (a). Com estas execuções é possível definir SLAs através da decomposição de métricas de alto nível em métricas de baixo nível (b), e obter dados de desempenho de cada configuração (c). Sobre estes últimos dados são aplicados algoritmos de mineração de dados (d) que buscam identificar padrões e apontar, através de modelos preditivos (e), características quanto ao desempenho computacional do ambiente virtualizado. Estes modelos (e), em conjunto com a configuração atual das VMs sendo executadas pelo Xen (g) e com os SLAs (h), são utilizados por um subsistema de reconfiguração (f) para gerar uma nova configuração para as VMs executando no Xen (i).

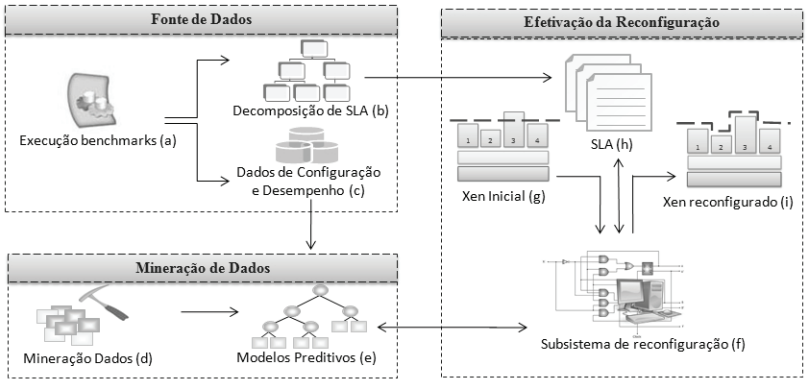

Figura 2. Processo proposto para a reconfiguração de máquinas virtuais no Xen

O restante desta seção descreve cada um dos blocos da Figura 1 de maneira detalhada, ou seja, como é feita a coleta e organização de dados gerados pelos benchmarks; como é feita a geração do modelo preditivo a partir de algoritmos de mineração de dados; e, como é realizada a reconfiguração das VMs, se necessário.

2.1 Fonte de Dados

Um dos aspectos mais importantes da estratégia apresentada neste artigo é a construção de um modelo preditivo contendo as possíveis reconfigurações para as VMs e dos SLAs possíveis para o ambiente existente. Neste artigo utilizamos a execução de benchmarks como ponto inicial para a geração de dados que serão utilizados para a construção do modelo preditivo e dos SLAs. Os *benchmarks* utilizados, TPC-W (TPC-W, 2008) e Unixbench (UNIXBENCH, 2007), são executados sobre cada VM que compõe os distintos ambientes virtualizados. Os resultados do TPC-W são utilizados para a decomposição de SLAs, e do Unixbench são utilizados para compor um banco de dados com o desempenho de cada uma das configurações previstas.

2.1.1 Decomposição de SLA

Um SLA pode especificar várias necessidades de um serviço, que inclui métricas como disponibilidade, tempo de resposta e processamento (UDUPI, 2007). Estes valores de níveis de serviços, quando expressos em um SLA, tomam a forma de Objetivos de Nível de Serviço (*Service Level Objectives* - SLOs) e são aplicados nas métricas de desempenho, conhecidas como Indicadores de Nível de Serviço (*Service Level Indicators* - SLIs). "Disponibilidade" e "tempo médio de resposta do serviço" são exemplos de SLIs utilizados em um SLA, enquanto que os valores 95% de disponibilidade ou 1,3 segundos de tempo de resposta são exemplos de valores de um SLOs (SAUVE, 2005). Apesar de SLOs e SLIs serem os mais importantes parâmetros, um SLA pode ainda possuir diversos outros, tais como penalidades a serem pagas pelo provedor do serviço quando este falhar em cumprir o nível de serviço prometido e recompensas que serão pagas pelo cliente do serviço quando o provedor do serviço superar as expectativas do nível de serviço acordado (MARQUES, 2006).

Para uma definição precisa dos níveis de recursos que uma aplicação neces-

sita para atender as especificações de um SLA, alguns trabalhos (GUPTA, 2006), (CUNHA, 2007) propõem a decomposição das métricas de alto nível em métricas de baixo nível. Desse modo, uma métrica como tempo de resposta de uma aplicação é decomposta em métricas de baixo nível, com uso, por exemplo, dos seguintes atributos: largura de banda (lb), utilização do processador (up) e memória disponível (md). Para cada um desses são definidos valores mínimos a serem atingidos, onde vmlb, vmup e vmmd representam, respectivamente, valores mínimos para os atributos citados.

Para atingirmos um indicador de nível de serviço (SLI) é necessário que, por exemplo, lb ≥ vmlb, up ≥ vmup e md ≥ vmmd. Para conhecermos estes níveis, precisamos submeter à aplicação a um conjunto de testes através da execução de *benchmarks* sob diversas configurações, com o objetivo de definir a quantidade de recursos, métricas de baixo nível, do sistema que cada componente monitorado consome. Deste modo, é possível determinar o volume de recursos consumidos por cada componente do sistema para suprir de maneira precisa cada SLI de um SLA.

2.1.2 Dados de Configuração e Desempenho

Os dados extraídos das execuções de *benchmarks* servem para apoiar a tomada de decisão para a reconfiguração a partir da construção de um modelo preditivo. Para a construção deste modelo é necessário possuir uma grande quantidade de dados para que o algoritmo de mineração de dados utilizado consiga gerar um modelo o mais preciso possível. A execução dos *benchmarks* é planejada de forma a coletar informações que definem (WINCK, 2008):

- Configurações de ambientes. O tipo de paravirtualizador, de sistema operacional e de *hardware* (destacando-se a memória disponível);
- Limites de consumo de CPU (em percentual) disponível para o ambiente. Por conveniência, definimos limites que variam de 70% a 100%, com intervalos de 5% em 5%;
- Número total de VMs utilizadas. Os testes foram realizados com 4 VMs, mas esse número pode ser alterado de acordo com as necessidades do usuário e disponibilidade do ambiente;
- Limites de consumo para cada VM. Cada VM recebe uma fatia (CAP) do percentual de CPU disponível. O critério empregado define que o valor mínimo de CAP é 10 e suas variações são múltiplas de 5;

- Para cada combinação de CAP são atribuídas diferentes alocações de memória em MB. A soma dos valores de memória das VMs resulta no total de memória disponível para o ambiente. O valor mínimo alocado é de 40 MB.

Para fins de ilustração, a Figura 3 mostra duas situações com 7 configurações cada (configurações número 239 a 245 e 281 a 287). Cada configuração é composta por: (1) ambiente, contendo a memória disponível, o escalonador utilizado, tipo de *hardware*, entre outros; (2) CPU, representando o limite máximo de consumo de CPU disponível para as VMs; (3) VMs utilizadas; (4) quantidade de CAP disponível para cada VM; (5) memória (em MB) alocada para cada VM em cada uma das configurações.

Ambiente		Escalonador: Credit / Xen: 3.0.4 / Maquina: Xeon Kernel: 2.6.16.33 / Memória Disponível: 280MB			
CPU (%)		85			
VMs		VM 1	VM 2	VM 3	VM 4
CAP		10	10	20	45
		Memória (MB)			
Configuração	239	70	70	70	70
	240	80	70	70	60
	241	90	70	70	50
	242	100	70	70	40
	243	110	70	60	40
	244	120	70	50	40
	245	130	70	40	40

Ambiente		Escalonador: Credit / Xen: 3.0.4 / Maquina: Xeon Kernel: 2.6.16.33 / Memória Disponível: 280MB			
CPU (%)		100			
VMs		VM 1	VM 2	VM 3	VM 4
CAP		25	15	25	35
		Memória (MB)			
Configuração	281	70	70	70	70
	282	80	70	70	60
	283	90	70	70	50
	284	100	70	70	40
	285	110	70	60	40
	286	120	70	50	40
	287	130	70	40	40

Figura 3. Planejamento de execuções de *benchmarks*

Desta forma, para cada relação entre configuração e quantidade de memória utilizada por uma máquina virtual, existe uma execução de *benchmark*. Para o exemplo que mostramos neste trabalho, foram planejadas 539 configurações distintas, com 4 VMs cada uma, o que resultou em 2.156 execuções de *benchmarks.*

2.2 Mineração de Dados

A mineração de dados é uma técnica que busca converter dados brutos em informação. Neste trabalho ela é empregada para identificar se é possível melhorar o desempenho de máquinas virtuais que estão executando no Xen. Neste sentido, optou-se em utilizar tarefas preditivas de mineração de dados, focalizando em algoritmos de classificação (TAN, 2006). Essas técnicas buscam construir modelos preditivos que apresentem a melhor combinação entre um conjunto de atributos, denominados atributos preditivos, e um dado atributo de interesse, denominado atributo alvo. A classificação utilizada descreve e distingue o atributo preditivo alvo (HAN, 2006), tal que os modelos resultantes possam ser utilizados para predizer a classe cujos atributos preditivos pertencem. Vale ressaltar que,

como sistema de apoio, optou-se em utilizar o algoritmo J48 (implementação de árvore de decisão C4.5) (QUILAN,1996) do ambiente Weka (WITTEN, 2008).

Os modelos preditivos produzidos indicam, a partir da configuração vigente de uma máquina Xen, quais conjuntos de configurações, se alterados, podem fornecer ganho no desempenho. Para que esses resultados pudessem ser produzidos satisfatoriamente, os dados de configuração e desempenho, mencionados na Seção 2.4, foram adequadamente pré-processados, conforme descrito em Winck, 2008. Esta preparação é feita em duas etapas: a primeira para definir o atributo alvo, e a segunda para definir os atributos preditivos.

O atributo alvo foi planejado, convenientemente, como binário, e sua definição é dada como segue. Primeiramente foram coletados os resultados de desempenho de cada VM (rdvm), em cada configuração. Tais resultados foram definidos por $rdvm_i$, onde i é o número da VM em questão. Para obter o desempenho da configuração (dc) como um todo, foi efetuado o somatório do desempenho de cada VM em cada configuração, chegando-se a $dc_n = \sum_{i=1}^{t} rdvm_i$, onde n é o número da configuração e t é o número total de VMs para uma dada configuração.

Para verificar se há benefício em reconfigurar, é definido um valor para o custo de reconfiguração de tal ambiente. De posse desse valor é verificado o resultado da diferença entre o desempenho de uma configuração inicial e o de uma configuração alvo. O resultado da diferença determinará o atributo alvo. Se a diferença for maior do que o custo, então significa que a configuração é benéfica e o atributo alvo é 1; se a diferença for menor ou igual ao custo, então não há benefício em reconfigurar, e o atributo alvo é 0. De posse desses resultados, é elaborada uma matriz quadrada, cujo tamanho corresponde ao total de configurações analisadas (no caso, 539), contendo os valores do atributo alvo. O valor de cada célula da matriz indica se mudança da configuração inicial para a configuração alvo é benéfica. Como efeito, essa matriz deve ser capaz de produzir um grafo dirigido, necessariamente acíclico, com as mudanças de configurações benéficas.

Como a definição do atributo alvo é elaborada a partir da comparação da configuração inicial com as configurações alvo, é apropriado que o pré-processamento dos atributos preditivos siga o mesmo princípio. Esses atributos, que combinados com o atributo alvo correspondente, são utilizados pela mineração de dados. Para a seleção desses, foi obedecido o critério de que devem ser empregados apenas aqueles que correspondem aos parâmetros de configuração que se deseja alterar.

Nesse sentido, optou-se por utilizar os atributos referentes ao percentual de CPU, CAP e memória. Os modelos preditivos produzidos sugerem reconfigurações para esses parâmetros. Vale ressaltar que esses modelos não são gerados dinamicamente.

2.3 Monitoração e Efetivação da Reconfiguração

O subsistema de realocação de recursos proposto busca melhorar a utilização do *hardware* e possibilitar a utilização de SLAs em ambientes virtualizados. Esse processo é realizado através da realocação de recursos entre VMs, tendo como base os modelos preditivos. O modelo preditivo indica, a partir da configuração atual das VMs, qual a melhor nova configuração para atender suas necessidades. Estas necessidades são determinadas por um sistema de monitoramento que analisa o nível de recursos utilizados por cada uma das máquinas virtuais que compartilham a estrutura. Quando uma máquina virtual necessita de um nível de processamento maior do que os limites definidos pelo SLA e existe a disponibilidade de recursos no ambiente, então o subsistema de realocação pode distribuir recursos para além do que foi definido no SLA. Quando isso acontece, o provedor do ambiente fica com crédito perante o cliente. Naturalmente, este tipo de situação deve ser acordado entre o fornecedor dos recursos e o cliente no SLA.

A realocação de recursos consiste em mover os recursos não utilizados pelas demais VMs para a VM que esteja utilizando todos os recursos atribuídos a ela no SLA e que, ainda assim, necessite de mais recursos (área escura da VM 3 na Figura 4a). No entanto, a quantidade de recursos que o subsistema pode realocar para uma VM nunca pode provocar a quebra do SLA das outras VMs. Deste modo, mesmo que existam recursos não utilizados dentro dos limites definidos no SLA de uma VM, estes não serão realocados (área quadriculada da VM 1 na Figura 4). Ou seja, o subsistema executa a realocação apenas dos recursos que se encontram disponíveis no ambiente (área hachurada na Figura 4). Após a realocação (Figura 4b), o subsistema continua monitorando os níveis de recursos utilizados pelas VMs. Quando o processamento da VM que recebeu os recursos retorna ao nível descrito em seu SLA, os recursos tornam-se novamente disponíveis para serem utilizados pelas demais VMs.

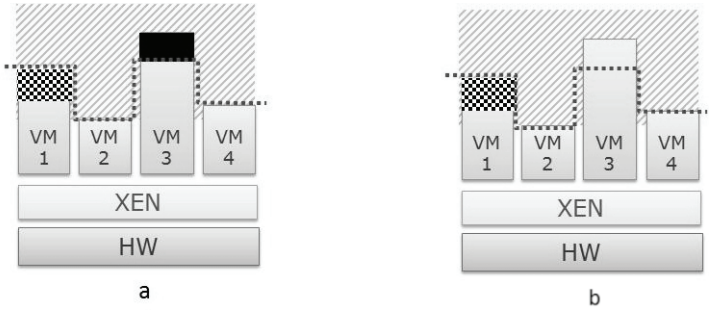

Figura 4. Realocação de recursos

3. Ambiente de Teste

Para verificar o ganho de desempenho obtido com a proposta descrita neste artigo, foram executadas três aplicações de *software* livre rodando em um ambiente virtualizado. Este ambiente é composto por quatro máquinas virtuais idênticas. Cada VM hospeda um servidor *web* Apache Tomcat 5.5 (TOMCAT, 2008) e um servidor de banco de dados MySQL 5.0 (MySQL, 2008). Para realizar as cargas de trabalho utilizamos a ferramenta TPC-W (TPC-W, 2008). O TPC-W é um *benchmark* padrão da indústria para aplicações de comércio eletrônico. O tamanho do banco de dados foi configurado em 10.000 itens e 200.000 usuários. Cada conjunto de teste é executado através de uma aplicação cliente que emula o acesso concorrente dos usuários ao servidor. O intervalo que cada cliente emulado espera antes de iniciar a interação seguinte (*think time*) foi configurado como um valor randômico entre 1 e 7 segundos. Definimos 100 usuários como número máximo de conexões simultâneas no Tomcat e no MySQL.

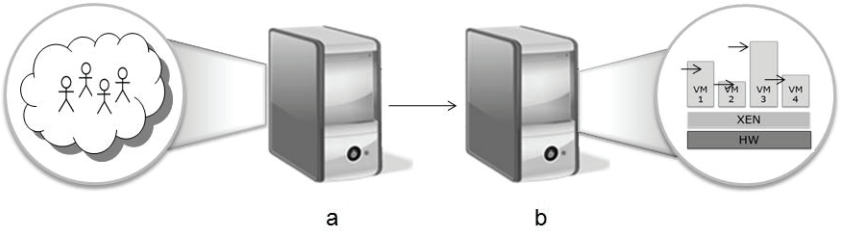

Figura 5. Estrutura utilizada

O ambiente de teste busca simular um *datacenter* virtualizado, com múltiplas aplicações compartilhando um conjunto comum de recursos. Este ambiente é composto por duas máquinas físicas. Uma é utilizada como hospedeira de quatro VMs no Xen (Figura 5 (b)). A outra máquina (Figura 5 (a)) é utilizada como gerador de carga dos

clientes (RBE - *Remote Browsers Emulator*). A ligação entre os clientes (RBEs) e o servidor é realizada através de uma rede rápida Gigabit Ethernet. A estrutura virtualizada descrita foi hospedada em um servidor com um processador Athlon(tm) 64 X2 Dual Core de 3.8 GHz com 2.0 GB de memória RAM e com um disco rígido IDE com capacidade de 160 GB. O sistema operacional utilizado foi o Ubuntu 7.04 server i386, Kernel 2.6.19-4-server. Já a estrutura que hospeda os RBEs consiste em um servidor com um processador Athlon(tm) 64 X2 Dual Core de 3.8 GHz com 2.0 GB de memória RAM.

4. Resultados

Para avaliar as funcionalidades e o desempenho do subsistema desenvolvido, foram aplicadas cargas de trabalhos sintéticas sobre a estrutura descrita na Seção 3. Em um primeiro momento foi definido um SLA que foi aplicado a todas as VMs da estrutura. Este SLA possui um SLO que especifica que, com uma carga de até 25 usuários simultâneos no sistema, o tempo de resposta da aplicação deve ser inferior a 1 segundo. A partir deste ponto, foi realizada a decomposição e configuração dos recursos. Estes foram disponibilizados de maneira uniforme para todas as VMs da estrutura. Através da decomposição, foram definidos os limites de recursos alocados para cada VM, que foram de 20% do processador (CAP) e 400 MB de memória.

Na execução dos testes, busca-se retratar dois diferentes cenários. No primeiro, todas as VMs são submetidas a uma mesma carga de trabalho, com exceção de uma VM, que recebe uma carga maior de usuários do que a definida em seu SLA. No segundo, duas VMs recebem a carga de usuários definidas em seus SLAs e as duas restantes recebem uma carga de usuários maior.

No primeiro cenário, as aplicações hospedadas em cada uma das VMs foram submetidas a uma carga de teste que simula o acesso simultâneo de 25 usuários. No entanto, a VM 1 teve seu número de usuários simultâneos alterado para 40. Este acréscimo de usuários busca retratar um cenário onde o SLA seria quebrado temporariamente por parte do cliente, motivado por um pico de acessos de usuários ao sistema. Os testes foram realizados em dois ambientes idênticos. Em um ambiente utilizamos apenas a decomposição de SLA para definir o limite de recursos utilizados por cada VM, e no outro utilizamos a decomposição de SLA e o subsistema de realocação de recursos proposto.

A Figura 6 apresenta os tempos de resposta das aplicações em cada VM. Foram medidos os tempos com e sem o subsistema de realocação de recursos. Como se pode

observar na Figura 6, quando o subsistema não está executando, o tempo de resposta das VMs 2, 3 e 4 se encontra dentro dos limites definidos em seus SLAs. Entretanto, a VM 1 apresenta um tempo de resposta maior que o estipulado. Este resultado se deve ao número de usuários excedentes aos definidos no SLA. Embora a estrutura que hospeda as VMs ainda possua 20% dos seus recursos livres, o escalonador do Xen não é capaz de realizar a realocação dos recursos não utilizados em ambientes com SLAs. Com a utilização do subsistema de realocação, estes recursos tornam-se disponíveis para qualquer uma das VMs da estrutura, possibilitando suprir uma demanda maior por recursos que as VMs tenham. Comparando o tempo de resposta da aplicação, quando executada apenas com o escalonador do Xen, com o tempo de resposta quando se utiliza o subsistema, nota-se uma acentuada redução do mesmo no segundo caso. Outro resultado importante mostra que a utilização do subsistema possibilitou um ganho de desempenho global para as VMs da estrutura. Este ganho é proporcionado pelo fato do subsistema realizar a realocação para a VM que necessitar primeiro, neste caso a VM 1. No entanto, se o processamento da VM 1 retornar aos níveis descritos em seu SLA, o subsistema retira os recursos adicionais e realoca os mesmos para próxima VM que apresentar um pico de processamento. Deste modo, os recursos não são monopolizados pela VM com sobrecarga e tornam-se disponíveis para todas as VMs da estrutura.

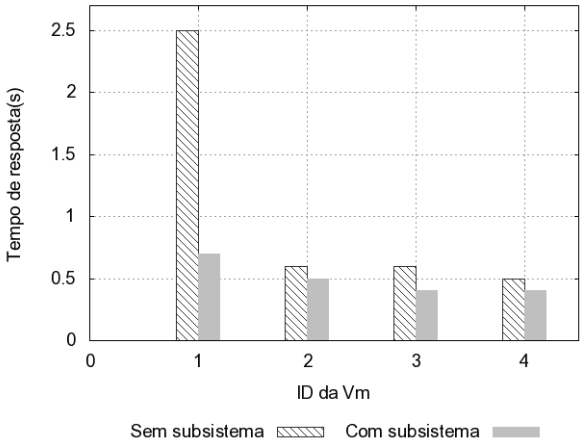

Figura 6. Sobrecarga em uma VM

No segundo cenário de teste, de modo análogo ao anterior, as aplicações hospedadas na VM3 e na VM4 foram submetidas a uma carga que simula o acesso simultâneo de 25 usuários. No entanto, as VMs 1 e 2 tiveram o número de usuários simultâneos alterado para 35. O acréscimo de usuários busca retratar um cenário onde o SLA de duas VMs é quebrado temporariamente e as VMs disputam os recursos.

Como resultado (Figura 7), podemos observar que, quando sem o subsistema de realocação de recursos, as VMs 1 e 2 apresentaram um tempo de resposta maior do que o valor definido nos SLOs dos respectivos SLAs. Este resultado é motivado pelo maior número de usuários no sistema do que aquele utilizado para decomposição do SLA e consequente definição dos níveis de recursos da VM. Ao realizar os mesmos testes, mas utilizando o subsistema de realocação, obtivemos uma considerável redução do tempo de resposta das VMs 1 e 2.

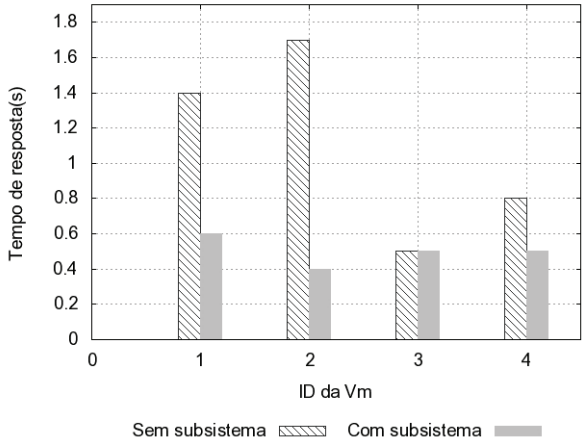

Figura 7. Sobrecarga em duas VMs

Conforme os resultados apresentados, a utilização do subsistema de realocação permite atender uma maior demanda por recursos do que aqueles definidos através da decomposição de SLA e apresenta um desempenho melhor do que o obtido com o escalonador do Xen. Também podemos observar que a utilização do subsistema propiciou uma melhora no desempenho de todas as VMs da estrutura, e não apenas naquelas que possuíam uma sobrecarga de usuários.

5. Conclusões

Este trabalho apresentou um modelo para realocação de recursos em ambientes virtualizados. Tal modelo conta com duas principais estratégias. A primeira está em utilizar mineração de dados para verificar se determinada máquina Xen merece ser reconfigurada, e assim indicar um melhor conjunto de parâmetros a ser modificado. A segunda contribuição está em estabelecer acordos de níveis de serviço (SLA) para que seja possível obter uma melhor utilização dos recursos existentes. Ambas as estratégias utilizam os resultados obtidos com as execuções de *benchmarks*. A

mineração de dados usa estes como fonte de dados para, a partir do desempenho apresentado em cada execução, construir os modelos preditivos utilizados para reconfiguração. Já para definir os SLAs, os resultados de *benchmarks* são utilizados para que seja possível fazer a decomposição das métricas de alto nível em métricas de baixo nível.

Para efetuar a reconfiguração, foi desenvolvido um subsistema que usa os modelos preditivos gerados pela mineração e as políticas de SLA definidas. Os resultados apresentados mostraram que, com a utilização do subsistema desenvolvido, o ambiente consegue manter o serviço com os níveis definidos no SLA, ou, em algumas situações, atender o serviço quando existe sobrecarga de trabalho. Deste modo ele garante recompensas da parte do cliente, pelo fato de o serviço superar as expectativas do nível de serviço acordado. É importante ressaltar que os modelos preditivos utilizados para a reconfiguração dizem respeito a um dado ambiente. Caso o ambiente seja alterado, um novo conjunto de *benchmarks* deve ser executado para que novos modelos preditivos sejam gerados.

6. Agradecimentos

Avelino F. Zorzo possui bolsa de produtividade CNPq. Elder M. Rodrigues possui bolsa de doutorado da CAPES/MEC associada ao Instituto Nacional de Ciência e Tecnologia em Sistemas Embarcados Críticos. Leandro T. Costa possui bolsa de mestrado no projeto PDTI em cooperação com a Dell Computadores. Os autores agradecem a Fábio Rossi pelo trabalho realizado em uma versão preliminar deste artigo apresentado no Workshop de Sistemas Operacionais em 2008. Ana Winck possui bolsa de doutorado do CNPq.

Referências

BARHAM, P.; DRAGOVIC, B.; FRAER, K.; HAND, S.; HARRIS, T.; HO, A.; NEUGEBAUER, R.; PRATT, I.; WARFIELD, A. Xen and the art of virtualization. *Em Symposium on Operating Systems Principles,* p. 164–177, 2003.

CUNHA, I.; ALMEIDA, J.; ALMEIDA, V.; SANTOS, M. Self-adaptive capacity man-agement for multi-tier virtualized environments. *Em International Symposium on Integrated Network Management*, p. 129–138, 2007.

GUPTA, D.; LUDMILA, C.; ROB, G.; AMIN, V. Enforcing performance isolation across virtual machines in Xen. *Technical Report,* HP Laboratories Palo Alto.

HAN, J. *Data mining*: concepts and techniques. Morgan Kaufmann, 2006.

JUNG, G.; PAREKH, J.; PU, C.; SAHAI, A. Detecting Bottleneck in n-tier IT Applications Through Analysis. *Em IFIP/IEEE Distributed Systems:* Operations and Management, p. 149–160, 2006.

MARQUES, F. T. *Projeto de infra-estrutura de TI pela perspectiva de negócio.* Dissertação de Mestrado, Universidade Federal de Campina Grande, Brasil, 2006.

MERGEN, F. M.; UHLIG, V.; KRIEGER, O.; XENIDIS, J. Virtualization for high-performance computing. *Operation System Review,* v. 40, p. 8–11, Abr. 2006.

MySQL. The world's most popular open source database. Disponível em: <http://www.mysql.com>. Acesso em: 10 de fev. 2008.

PAREKH, J.; JUNG, G.; SWINT, G.; PU, C.; SAHAI, A. Comparison of performance analysis approaches for bottleneck detection in multi-tier enterprise applications. *Em IFIP/IEEE Distributed Systems Operation and Management,* p. 149–160, 2006.

QUINLAN, R. J. *C4.5:* Programs for Machine Learning. Morgan Kaufmann, 1996.

SAUVE, J.; MARQUES, F.; MOURA, J.; SAMPAIO, M.; JORNADA, J.; RADZIUK, E. Optimal choice of service level objectives from a business perspective. *Em Workshop of HP Openview University Association*, p. 15–27, 2005.

TAN, N.; STEINBACK, M.; KUMAR, V. *Introduction to Data Mining.* Addison Wesley, 2006.

TOMCAT. Apache Tomcat. Disponível em: <http://tomcat.apache.org>. Acesso em: 12 de jan. 2008.

TPC-W. Transactional Processing Performance Council. Disponível em: <http://www.tpc.org/tpcw/>. Acesso em: 10 de fev. 2008.

UDUPI, B.; SAHAI, A.; SINGHAL, S. A classification-based approach to policy refinement. *Em IFIP/IEEE International Symposium on Integrated Network Management*, p. 758–788, 2007.

UNIXBENCH. Linux benchmark suite homepage. Disponível em: <http://lbs.sourceforge.net/>. Acesso em: 06 de nov. 2007.

WINCK, A.; RUIZ, D. Processo de KDD para auxílio à reconfiguração de ambientes virtualizados. *Em Simpósio Brasileiro de Sistemas de Informação,* p. 211–222, 2007.

WITTEN, I., FRANK, E. *Data mining:* practical machine learning tools and techniques. Morgan & Kaufmann, San Francisco, 2005.