Sistema Automatizado de Gerência de Recursos para Ambientes Virtualizados

Govinda Mohini Gonzalez Bezerra, Diogo Menezes Ferrazani Mattos, Lyno Henrique Gonçalves Ferraz e Otto Carlos Muniz Bandeira Duarte

[1]Grupo de Teleinformática e Automação
Universidade Federal do Rio de Janeiro (UFRJ)
Rio de Janeiro – RJ – Brasil

Resumo. *A gerência de recursos em ambientes virtualizados é uma tarefa complexa e desafiadora, pois deve se adaptar dinamicamente às oscilações de carga de trabalho e garantir o nível de serviço das diferentes aplicações em execução. Este artigo propõe o sistema AMAS (Automatic Migration and Allocation System for Virtual Resource Management) que monitora máquinas físicas e virtuais e constrói perfis de uso de diferentes recursos computacionais, como processador, memória e rede. O sistema proposto é capaz de detectar, com base nos perfis elaborados, a escassez dos recursos monitorados e automaticamente redistribuir as cargas de trabalho, evitando a sobrecarga dos nós físicos e violações de acordos de níveis de serviços. O sistema proposto automatiza migrações ao vivo de máquinas virtuais, que permitem transferir uma máquina virtual de um nó físico origem para um nó físico destino sem a interrupção dos serviços em execução. Um protótipo do sistema AMAS foi desenvolvido e o seu desempenho foi avaliado no Future Internet Testbed with Security (FITS). Os resultados obtidos mostram a eficácia do sistema em distribuir as cargas de trabalho e eliminar sobrecargas de processamento e memória dos nós físicos.*

Abstract. *One of the major challenges in virtualized environments is dynamically managing available resources, adapting them to changes in workload and ensuring the level of service of the running applications. In this paper, we propose AMAS, Automatic Migration and Allocation System for Virtual Resource Management, a system for monitoring physical and virtual machines, and it profiles the use of different computing resources, such as processor, memory and network usage. Based on profiles, the system is able to detect the lack of a monitored resource and automatically redistribute workloads, avoiding the overhead of physical nodes and violations of Service Level Agreements (SLA). The proposed system performs load distribution through live migration primitive of virtual machines that can transfer a virtual machine from a physical source node to another without interruption of running services. The proposed system was developed and we evaluated its performance with the Future Internet Security Testbed (FITS). The results show the effectiveness of the system to distribute workloads and to eliminate overloads of physical nodes.*

1. Introdução

A tecnologia de virtualização provê o compartilhamento de recursos computacionais entre diferentes ambientes virtuais de execução, chamados de máquinas virtuais. As

Este trabalho foi realizado com recursos da CNPq, CAPES, FINEP, FUNTTEL e FAPERJ.

máquinas virtuais acessam os recursos da máquina física de forma isolada com a ilusão de acessarem os recursos computacionais como se lhes fossem dedicados, para a execução de aplicações e sistema operacional próprios. A virtualização está sendo adotada nas empresas para consolidação de servidores, que consiste em reunir diversos servidores virtuais, lógicos, sobre uma mesma máquina física. Além das vantagens econômicas, a virtualização proporciona uma infraestrutura ágil e dinâmica, pois permite realocar facilmente os recursos computacionais, de forma a responder eficazmente às variações de demanda por recursos e melhor atender as necessidades dos diferentes serviços.

O gerenciamento e a alocação dinâmica de recursos em ambientes virtualizados são tarefas bastante complexas. A funcionalidade de migração de máquinas virtuais ao vivo é uma abordagem eficiente para realizar a gerência de recursos. Através da migração é possível reorganizar os domínios nos nós físicos de acordo com a disponibilidade dos recursos. Assim, se houver um aumento significativo na carga de uma máquina virtual, de forma a sobrecarregar um nó físico, esse domínio virtual poderá ser migrado para outro nó que possua recursos disponíveis. Essa abordagem permite realizar a distribuição de cargas de uma forma eficiente, porém também traz alguns desafios. O primeiro consiste em prever o comportamento futuro das máquinas virtuais para estimar a demanda futura de recursos. O segundo consiste em encontrar a melhor combinação de máquinas físicas e virtuais que é um problema de otimização do tipo *bin-packing*, cuja complexidade é *NP-difícil*. Logo, dada a complexidade do problema, os trabalhos desenvolvidos nesta área são baseados em heurísticas e modelos estatísticos.

Este artigo propõe o *Automatic Migration and Allocation System for Virtual Resource Management* (AMAS), um sistema automático de gerência de recursos que é capaz de realocar dinamicamente as máquinas virtuais de acordo com o nível de utilização das máquinas físicas monitoradas. O sistema proposto monitora os recursos computacionais em ambientes virtualizados, tais como capacidade de processamento, memória e banda passante, detecta situações de sobrecarga e toma decisões que eliminam a sobrecarga. O sistema distribui automaticamente as cargas de trabalho entre as máquinas físicas disponíveis, realocando máquinas virtuais críticas em máquinas físicas menos sobrecarregadas. Assim, as principais contribuições deste artigo são: i) o monitoramento e a criação de perfis do uso de recursos das máquinas físicas e virtuais; ii) a proposta de um algoritmo simples capaz de tomar decisões de realocação de máquinas virtuais , baseado nos perfis de uso, que elimine situações de sobrecarga; iii) a implementação de um painel de controle que permita a visualização, em tempo real, da alocação e utilização dos recursos computacionais monitorados. O sistema AMAS é compatível com diversas tecnologias de virtualização, mas é focado no Xen, pois esta é a tecnologia utilizada no testbed *Future Internet Testbed with Security* (FITS)[1].

As principais propostas para prover o balanceamento e a distribuição de cargas em ambientes virtualizados se diferenciam em analisar o estado de cada nó individualmente [Wood et al., 2009a, Carvalho e Duarte, 2012, Khanna et al., 2006] ou o estado de global do conjunto de todos os nós físicos [Arzuaga e Kaeli, 2010]. Algumas propostas focam na análise de afinidade entre máquinas físicas e virtuais [Sonnek et al., 2010, Wood et al., 2009b], ou ainda, consideram a topologia da rede e a banda disponível para realizar a migração de máquinas virutais [Singh et al., 2008]. A proposta deste artigo, no

[1]O FITS é uma rede de testes interuniversitária desenvolvida a partir da parceria de instituições brasileiras e europeias. Maiores informações em http://www.gta.ufrj.br/fits/.

entanto, foca em estabelecer um algoritmo simples e rápido para tomar decisões de migrar máquinas virtuais em tempo real para garantir o nível de serviço prestado por essas máquinas. Um protótipo do sistema foi implementado e avaliado. Os resultados mostram que o sistema AMAS é capaz de identificar gargalos no uso de recursos computacionais e reagir em tempo hábil para evitar a degradação do nível de serviço prestado pelas máquinas virtuais.

O restante deste artigo está organizado da seguinte forma. A Seção 2 apresenta os trabalhos relacionados. A Seção 3 discute a arquitetura do sistema AMAS. O protótipo do sistema e os resultados da avaliação de desempenho do protótipo são discutidos na Seção 4. A Seção 5 conclui o artigo.

2. Trabalhos Relacionados

A alocação de máquinas virtuais em servidores físicos é um problema complexo. Há propostas de heurísticas [Fajjari et al., 2011] e modelagens baseadas em otimização de sistemas [Alkmim et al., 2011, Rodriguez et al., 2013] que buscam o melhor aproveitamento dos recursos físicos. Um fator importante a ser considerado no uso de algoritmos de otimização é o tempo de convergência do algoritmo, que vai influir diretamente na dinamicidade do sistema. Propostas de otimização de uso de recursos físicos são complementares ao sistema proposto, pois podem ser usadas no o algoritmo de migração. Este trabalho não foca na otimização e opta por uma proposta simples para uma rápida tomada de decisão.

Sandpiper [Wood et al., 2009a] é uma ferramenta de gerenciamento de recursos para centros de dados (*data centers*), formado basicamente por três componentes: um mecanismo de criação de perfis, um detector de sobrecarga e um gerenciador de migração. A criação de perfis é feita através do monitoramento e da coleta de estatísticas do uso de recursos das máquinas físicas e virtuais. O detector de sobrecarga monitora estes perfis em busca de nós sobrecarregados que consistem em máquinas físicas cuja utilização de pelo menos um dos recursos ultrapassou um limiar ou se ocorreu a violação do acordo de nível de serviço (*Service Level Agreement* - SLA). O detector de sobrecarga determina quando é necessário realizar uma realocação de recursos, ou seja, quando o gerenciador deverá atuar para eliminar esta sobrecarga. Cabe ao gerenciador decidir qual máquina virtual migrar e qual servidor irá recebê-la. Diferentemente da proposta deste artigo, o Sandpiper utiliza o estado local de cada máquina física e não leva em conta a afinidade entre as máquinas e topologia da rede para realizar as migrações.

Voltaic [Carvalho e Duarte, 2012] é um sistema de gerência voltado para computação em nuvem que visa garantir o cumprimento de acordos de níveis de serviços (SLAs) e otimizar o uso dos recursos computacionais. Assim como o Sandpiper, o Voltaic também é uma solução baseada em perfis de uso. O sistema pode ser dividido em três módulos principais: o primeiro, chamado de coletor de estatísticas, é responsável por coletar informações de uso dos recursos computacionais; o segundo consiste no analisador de perfil que utiliza os dados provenientes do coletor de estatística para gerar perfis de uso; e o terceiro é o orquestrador, módulo central do Voltaic, responsável por gerenciar as máquinas, detectar escassez de recursos e orquestrar a migração. O orquestrador monitora a carga de todas as máquinas físicas e caso a média das últimas amostras de carga de alguma das máquinas ultrapassarem um limiar de segurança, o algoritmo começa o procedimento de realocação de recursos. As máquinas físicas são ordenadas de acordo com a carga do sistema e pelo número de máquinas virtuais críticas que ela possui. Dada a so-

brecarga de algum dos recursos monitorados, a máquina virtual escolhida para a migração será a máquina mais crítica do servidor físico mais sobrecarregado. O servidor de destino será aquele que possuir recursos disponíveis compatíveis com o perfil da máquina virtual. O comportamento do Voltaic foi simulado e o sistema não foi implementado. Quando comparado ao Voltaic, o sistema proposto neste artigo usa uma heurística mais simples que permite reagir às mudanças de cargas de trabalho em tempo real.

Yanagisawa *et al.* propõem um algoritmo de alocação de máquinas virtuais baseado em programação inteira, que considera *a priori* padrões de flutuações da demanda de recursos para alocação ótima e confiável de máquinas virtuais em uma infraestrutura de nuvem [Yanagisawa et al., 2013]. A proposta usa a migração de máquinas virtuais entre servidores físicos somente quando há a necessidade de manutenção ou falhas no servidor. A ideia principal é minimizar as possibilidades de falha e de degradação de desempenho de máquinas virtuais devido às migrações. Na proposta, o requisito mais importante de confiabilidade a ser considerado no planejamento de capacidade é que os recursos computacionais sejam suficientes e estejam disponíveis a todo o momento, mesmo quando houver falhas no servidor. Contudo, essa proposta considera que os usuários sejam altamente conservadores. Por sua vez, o sistema AMAS considera que os usuários podem ter comportamentos bem variáveis e o sistema deve ser capaz de realocar as máquinas virtuais e atender às mudanças de demanda por recursos em tempo real. O sistema AMAS usa a mesma abordagem de Yanagisawa *et al.* ao limitar o número de migrações possíveis, para evitar a degradação de desempenho das máquinas virtuais.

Guo *et al.*, por sua vez, propõem um modelo de alocação de banda entre máquinas virtuais em centros de dados para nuvens [Guo et al., 2013]. A ideia central da proposta é aplicar a teoria de jogos e modelar a disputa por banda entre os diferentes inquilinos da nuvem como um problema de negociação de Nash (*Nash bargaining problem*). Assim, a proposta desenvolve um jogo cooperativo para alocação de banda entre as máquinas virtuais. A proposta alcança uma banda mínima garantida entre pares de máquinas virtuais e garante a justiça na divisão da banda entre todos os pares de máquinas virtuais. No entanto, essa proposta modela somente a alocação de banda entre máquinas virtuais e não considera a alocação de outros recursos, tais como processamento e memória. O modelo de alocação de banda pode ser usado em paralelo ao sistema AMAS para garantir a divisão justa de banda entre máquinas virtuais sobre uma mesma máquina física.

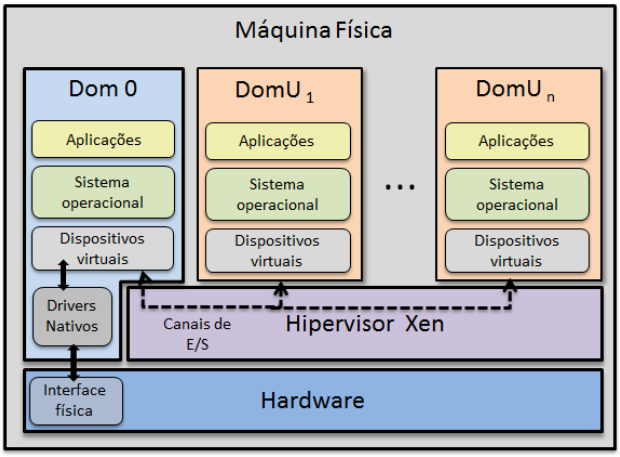


Figura 1. A arquitetura Xen e a centralização das atividade de E/S no Domínio 0.

Este trabalho propõe um sistema de gerenciamento de recursos baseado em perfis e na funcionalidade de migração ao vivo de máquinas virtuais. O sistema proposto usa a biblioteca libvirt para aquisição de dados e um protótipo do sistema foi implementado na plataforma de virtualização Xen. O Xen [Barham et al., 2003, Xen Project, 2009] é uma plataforma de virtualização de código aberto composta por um hipervisor, um Domínio 0 e diversos Domínios Us. O hipervisor Xen é a camada de abstração básica de software localizada logo acima da camada de hardware. É responsável pelo escalonamento de CPU e pelo fatiamento de memória. O Domínio 0 é uma máquina virtual que possui privilégios especiais para acessar os recursos físicos de E/S e para interagir com os outros domínios em execução no sistema. Os Domínios Us são domínios não privilegiados que não têm acesso direto ao hardware da máquina física.

A libvirt [Red Hat, 2013] é uma API para gerenciar, de forma segura, máquinas em ambientes virtualizados. A libvirt permite o gerenciamento completo das máquinas virtuais, tornando possível criar, modificar, controlar, migrar e destruir as máquinas através de sua interface. A ferramenta permite o acesso a hipervisores utilizando conexões autenticadas e criptografadas, através do protocolo TLS (*Transport Layer Security*). A criptografia e a autenticação são feitas usando a abordagem de chave pública (PKI), logo, é necessário que todos os computadores monitorados possuam certificados válidos, além das chaves pública e privada.

3. O Sistema de Gerência e Distribuição de Cargas Proposto

O sistema AMAS (*Automatic Migration and Allocation System for Virtual Resource Management*) monitora as máquinas, físicas e virtuais, com objetivo de gerenciar a alocação e melhorar a utilização dos recursos físicos, além de garantir o cumprimento dos acordos de níveis de serviço (SLAs) dos servidores virtuais. A ideia central da proposta é criar perfis de uso de diferentes recursos para todas as máquinas físicas e virtuais de um aglomerado (*cluster*) e, com base nestes perfis, detectar nós físicos sobrecarregados e distribuir a carga de trabalho através da migração ao vivo.

A Figura 2 mostra a arquitetura do AMAS, que possui três módulos principais: o Gerador de Perfil, responsável por monitorar o uso dos recursos físicos; o Detector de sobrecarga, capaz de detectar escassez de recursos monitorados; e o Orquestrador de migração, responsável por tomar decisões visando distribuir as cargas de trabalho em caso de sobrecarga. Toda a comunicação entre o gerenciador e as máquinas físicas é realizada através da libvirt, o que garante uma maior portabilidade do sistema para outras plataformas.

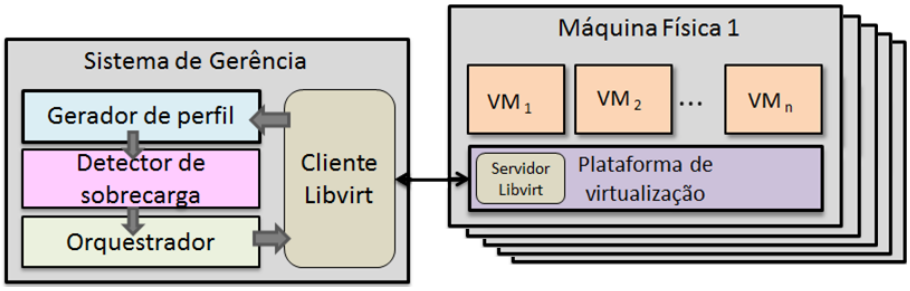


Figura 2. Arquitetura do sistema de gerência AMAS com o gerente e módulos servidores libvirt em cada máquina física.

O mecanismo de monitoramento foi desenvolvido utilizando uma abordagem do tipo caixa preta. Na solução adotada, todas as informações relativas ao uso de recursos das máquinas virtuais são obtidas através da libvirt, sem a instalação de nenhum programa específico nas máquinas virtuais. Essa abordagem foi escolhida por garantir uma menor interferência no funcionamento dos domínios, assegurando maior autonomia e privacidade aos proprietários das máquinas virtuais. Essa propriedade é importante em ambientes virtualizados, pois em muitas soluções comerciais baseadas em virtualização, as máquinas virtuais não pertencem aos proprietários do hardware, não sendo, portanto, viável a instalação de softwares de monitoramento nos domínios de clientes.

Como pode ser observado na Figura 2, o Gerador de Perfil interage com as máquinas físicas através da libvirt. Essa conexão é feita utilizando o protocolo *Transport Layer Security* (TLS), que consiste em uma conexão TCP/IP autenticada e criptografada, garantindo a segurança das informações. A coleta de dados é realizada através da amostragem periódica do uso de CPU, da quantidade de memória alocada e da quantidade de dados de rede enviados e recebidos pela máquina. Com as amostras obtidas, são criados três perfis de uso, representando cada uma das três métricas, que consistem em séries temporais representadas por uma janela deslizante contendo N amostras.

A coleta das amostras de uso dos recursos é feita de modo paralelo através da utilização de processos leves, chamados *threads*. Dessa forma, é possível construir simultaneamente o perfil das diferentes máquinas monitoradas, aproveitando melhor os recursos computacionais da máquina gerenciadora e permitindo maior coerência temporal das amostras obtidas. Cada módulo do sistema possui um *thread* responsável pela coleta de dados e construção dos seus perfis de uso. Os perfis criados são apresentados em um painel de controle através de gráficos que indicam a participação de cada máquina virtual, inclusive do Domínio-0, na utilização dos recursos das máquinas físicas monitoradas.

O monitoramento de CPU das máquinas virtuais é realizado através da amostragem da medida $tempo\ de\ CPU$ obtida diretamente da libvirt, que corresponde ao tempo de processamento utilizado pela máquina virtual desde a sua criação. O perfil de uso do processador dos nós físicos é calculado indiretamente, através da soma dos perfis de todos os domínios que nele executam, pois não é possível obter a medida $tempo\ de\ CPU$ das máquinas físicas através da libvirt. Esta medida possui uma imprecisão associada, pois as amostras das máquinas virtuais podem não corresponder a exatamente o mesmo instante de tempo. Os perfis de memória das máquinas virtuais são construídos a partir da amostragem da quantidade de memória alocada ao domínio. Para os perfis das máquinas físicas, é utilizada a quantidade de memória alocada. Ambas as medidas são obtidas diretamente da libvirt. No Xen, o Domínio-0 realiza as operações de rede pelos outros domínios não privilegiados. Logo, uma forma simples de obter informações que representem a atividade de rede das máquinas virtuais é medindo a quantidade de dados enviados e recebidos através de suas interfaces virtuais, o que também é facilmente obtido através da libvirt. Essa métrica é importante para identificar a contribuição de cada máquina virtual no tráfego total gerenciado pelo Domínio-0. Então, é possível identificar se a causa desse consumo provém da utilização intensiva de rede e qual a participação de cada máquina virtual no tráfego total gerenciado pelo Domínio-0.

No sistema AMAS são considerados dois tipos de sobrecarga, a de processamento (CPU) e a de memória. A sobrecarga de CPU ocorre quando a utilização do processador ultrapassa um determinado limiar durante as K últimas amostras. O conceito de sobre-

carga de memória é similar e ocorre quando a quantidade de memória alocada de um nó físico ultrapassa o limiar definido para memória durante as últimas M amostras. Apesar de o sistema monitorar a intensidade do tráfego de rede, não há uma sobrecarga associada a esse perfil, uma vez que a métrica de rede implementada na versão atual não leva em conta os parâmetros nem a topologia da rede de computadores. A medida é utilizada, então, para verificar o impacto que o tráfego de rede dos domínios não privilegiados causa no uso de processamento do Domínio-0.

Um nó físico é considerado sobrecarregado se pelo menos um dos recursos monitorados estiver escasso naquele nó. A cada intervalo de tempo T, o algoritmo percorre as máquinas físicas verificando se há sobrecarga no sistema. O algoritmo constrói duas listas, uma contendo as máquinas físicas sobrecarregadas e outra contendo as máquinas com recursos disponíveis e ordena-as de acordo com o nível de utilização de cada um dos recursos monitorados. Detectada a escassez de algum dos recursos, o Orquestrador de migração é acionado e deve tomar decisões para eliminar a sobrecarga e distribuir as cargas de trabalho. O algoritmo realiza uma migração a cada iteração, então é preciso selecionar: i) uma máquina física sobrecarregada que terá sua carga diminuída, ii) um domínio virtual deste nó físico que será migrado e iii) uma máquina física com recursos disponíveis para receber esse domínio.

A escolha dos elementos envolvidos na migração é feita da seguinte forma:

Seleção da máquina física sobrecarregada: As máquinas físicas são classificadas de acordo com o nível de utilização de CPU e alocação de memória. Assim, se existirem diversos nós sobrecarregados com os dois tipos de sobrecarga, a máquina física com maior nível de utilização de processamento é selecionada. Caso só haja sobrecarga de memória, a máquina com menos memória disponível é selecionada.

Seleção da máquina física de destino: A máquina física que receberá a máquina virtual migrada é aquela que possuir a maior quantidade de recursos disponíveis, suficientes para alocar a máquina virtual sem ficar sobrecarregada.

Seleção da máquina virtual: Caso haja uma sobrecarga de CPU, é escolhida, dentre os domínios da máquina física sobrecarregada, a máquina virtual mais crítica de acordo com a equação

$$criticidade = \frac{cpu + net \cdot cpuDomain0}{memória}, \tag{1}$$

onde cpu representa o uso de CPU, net indica o tráfego de rede, $cpuDomain0$ representa o uso do processador pelo Domínio-0 e memória a quantidade de memória alocada ao domínio. O termo $net \cdot cpuDomain0$ visa incluir na equação de criticidade a sobrecarga causada no Domínio-0 pelo uso da rede [Cherkasova e Gardner, 2005].

Para realizar a migração ao vivo de uma máquina virtual, é preciso transmitir todas as suas páginas de memória à máquina física de destino, logo o tempo de migração é diretamente influenciado pela quantidade de memória utilizada e pela taxa de atualização dos dados na memória. O tempo de migração também varia de acordo com o nível de utilização dos recursos nas máquinas físicas de origem e destino, pois o processo de migração consome recursos em ambas as máquinas físicas [Wu e Zhao, 2011]. Assim, optou-se pela relação $\frac{cpu}{memória}$ na equação de criticidade.

Caso a sobrecarga seja de memória, é migrada a máquina virtual que possuir a menor quantidade de memória alocada do nó físico sobrecarregado. Com esta solução pode

ser necessário à migração de mais de uma máquina virtual para eliminar a sobrecarga, entretanto ela evita migrações longas e, com isso, tende a diminuir o processamento total a ser realizado pelo Domínio-0 das máquinas físicas envolvidas na migração.

O algoritmo do Orquestrador percorre a lista de máquinas virtuais do nó físico mais sobrecarregado buscando a máquina virtual a ser migrada. É escolhida a máquina virtual mais crítica que consiga ser migrada para a máquina física com mais recursos disponíveis. É possível notar que a sobrecarga de CPU é tratada com prioridade maior do que a sobrecarga de memória. Esta escolha foi feita porque o uso de CPU é mais dinâmico, já que o monitoramento de memória é realizado através da quantidade de memória alocada e não pela quantidade de memória usada. Nota-se também que se todas as máquinas físicas estiverem saturadas, ou se a migração puder causar sobrecarga na máquina de destino, a migração não é realizada.

O sistema AMAS é baseado na funcionalidade de migração ao vivo do Xen, logo, é necessário que as imagens dos discos de todos os domínios virtuais estejam disponíveis para todas as máquinas do sistema. Além disso, é necessário que todas as máquinas físicas possuam a libvirt e os certificados, necessários para autenticar as conexões, instalados. É desejável que as máquinas físicas monitoradas possuam configurações homogêneas de hardware e software, pois, em caso de migração, configurações distintas podem implicar em perda de desempenho das aplicações em execução. Na versão atual do sistema, por simplicidade, a modelagem não inclui verificação e adequação de determinadas variáveis como, por exemplo, modelos e frequências de processadores diferentes. Assim, em um ambiente onde os servidores possuem alto nível de utilização, se uma máquina virtual intensiva em rede for migrada para uma máquina física com configurações de escalonamento diferente da máquina física de origem, as aplicações sensíveis a latência podem ter o seu funcionamento fortemente alterado [Govindan et al., 2007, Lee et al., 2010].

4. Avaliação de Desempenho do Protótipo Desenvolvido

O sistema proposto foi implementado no ambiente de testes FITS do Grupo de Teleinformática e Automação da UFRJ/COPPE/PEE. Foram utilizados computadores padrão de mercado com placas de rede gigabit Ethernet. A implementação de cada um

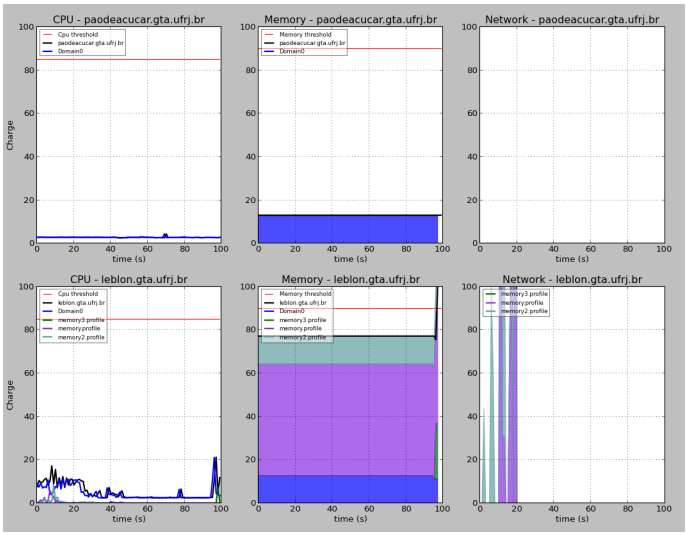
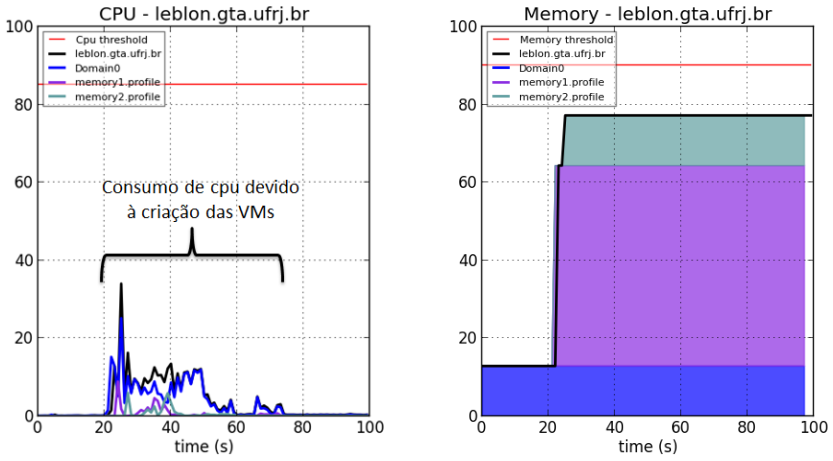


Figura 3. Painel de controle do sistema de gerenciamento AMAS.



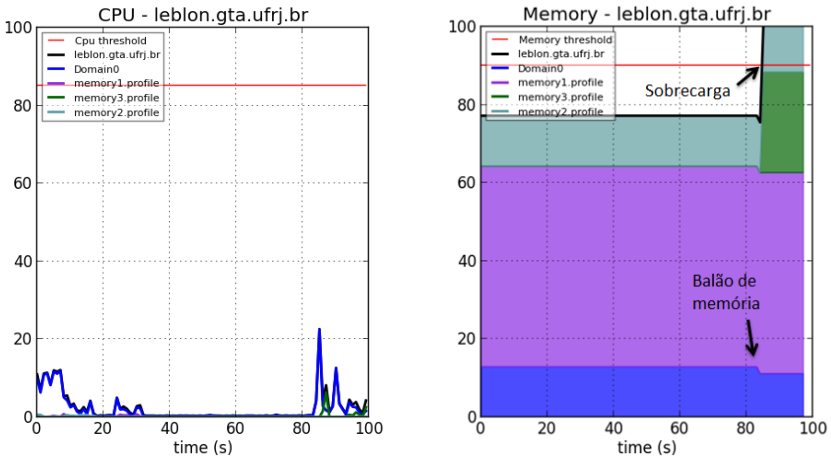
(a) Uso de processador. (b) Memória alocada.

Figura 4. Passo 1 do cenário de testes. Configuração da máquina leblon.gta.ufrj.br após a criação das máquinas virtuais memory1 e memory2. a) A criação das máquinas virtuais consome processamento do Domínio 0. b)A alocação de memória após a criação das máquinas virtuais revela o uso de quase 80% da memória física disponível.

dos módulos desenvolvidos segue as diretrizes da orientação a objetos, na linguagem de programação Python. O escalonador utilizado no Xen foi o Credit Scheduler e o *timeslice* de 30 ms, ambos são configurações padrão do Xen. A memória de inicialização do Domínio-0 foi definida como 2048 MB, a opção de balão foi habilitada e a memória mínima para este domínio foi configurada como 1024 MB. A funcionalidade de balão de memória foi ativada, pois apesar de o sistema operacional `Debian Wheezy` necessitar de mais de 1GB de memória para iniciar o sistema, o total de memória gasto pelo Domínio-0 e pelo hipervisor para gerenciar as máquinas virtuais está na ordem de 1GB. Então, visto que uma parte da memória alocada exclusivamente para o Domínio-0 ficaria ociosa durante todo o funcionamento do nó físico, optou-se por habilitar o balão de memória visando melhor aproveitar a memória disponível. Alguns parâmetros do sistema são configurados de acordo com a necessidade e as características das aplicações e do conjunto de máquinas monitoradas. São eles: o tamanho da janela de observação, ou seja, o número de amostras contidas no perfil das máquinas; os limiares que definem a sobrecarga de CPU e de memória; o período de amostragem; a quantidade de amostras acima do limiar que devem ser consideradas antes de realizar a migração. Foi desenvolvido um painel de controle para permitir a visualização do uso de recursos das máquinas físicas e virtuais em tempo real. O painel é composto de diferentes gráficos, representando cada um dos recursos monitorados de cada uma das máquinas físicas. A Figura 3 mostra o painel de controle em funcionamento com algumas máquinas virtuais de teste, utilizando duas máquinas do testbed FITS.

Para validar o funcionamento do sistema, foi desenvolvido um cenário de teste que permite testar o algoritmo nos diferentes casos de sobrecarga. Através deste cenário de teste também foi possível observar o comportamento do Domínio 0 durante a fase de migração e durante as atividades de rede dos domínios não privilegiados. A carga de CPU foi simulada utilizando o programa Stress [Waterland, 2013], de licença livre, que permite gerar cargas de trabalho de forma controlada.

Dentre os computadores disponíveis na rede de testes, foram selecionadas duas



(a) Uso de processador. (b) Memória alocada.

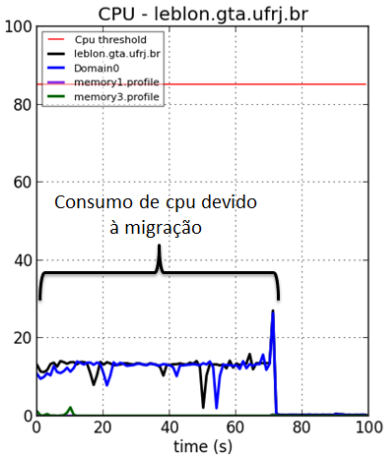
Figura 5. Passo 2 do cenário de testes. Configuração da máquina leblon.gta.ufrj.br após a criação da máquina virtual memory3, gerando uma situação de sobrecarga de memória. a) A criação da máquina virtual consome processamento do Domínio 0. b) A alocação de memória após a criação das máquinas virtuais atinge 100%, ultrapassando o limiar de segurança e levando o domínio0 a ceder memória para o domínio memory3 recém criado.

máquinas equipadas com processador Intel I7 de 3.2 GHz e 16 GB de memória RAM. O sistema operacional instalado nas máquinas é o `Debian Wheezy` e o hipervisor Xen utilizado é a versão 4.1.3. As imagens dos discos das máquinas virtuais e os arquivos de configuração encontram-se em um nó central da arquitetura do FITS e são acessíveis por qualquer máquina da rede de teste. Para criar diferentes cenários de sobrecarga, foram utilizadas máquinas virtuais com diferentes configurações, de modo a consumir cada um dos recursos monitorados. Os domínios virtuais denominados cpu1, cpu2 e cpu3 foram configurados cada um com 8 processadores virtuais e 512 MB de memória RAM. Os domínios denominados memory1, memory2 e memory3 foram configurados cada um com 1 processador virtual e 1024 MB, 2048 MB e 4096 MB de memória RAM, respectivamente.

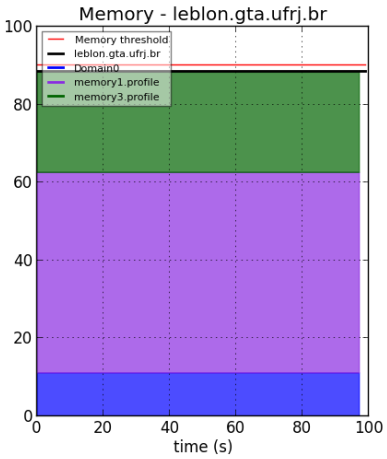
O cenário de teste é dividido nas seguintes passos:

Passo 1 - criação das máquinas memory1, memory2 na máquina física *leblon.gta.ufrj.br*
Passo 2 - criação da máquina memory3 na máquina *leblon.gta.ufrj.br*
Passo 3 - criação das máquinas cpu1 e cpu2 na máquina física *paodeacucar.gta.ufrj.br* e da máquina cpu3 na *leblon.gta.ufrj.br*
Passo 4 - geração de carga em 5 vCPUs da máquina cpu2 e em 2 vCPUs da cpu3

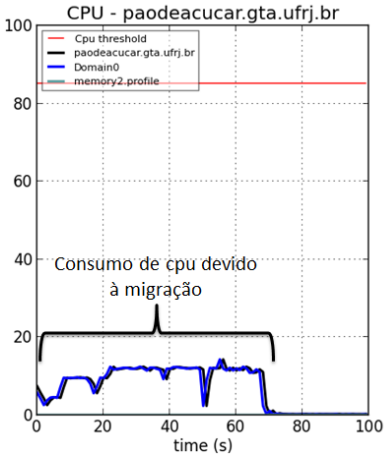
O uso de memória e processador pelas duas máquinas físicas, sem nenhuma máquina virtual em execução além do Domínio 0 é muito pequeno e a sua memória alocada está na ordem de 12% da quantidade de memória total. O primeiro item do cenário de testes visa criar um conjunto inicial de máquinas virtuais na máquina física. A Figura 4 ilustra esta etapa, onde é possível notar o consumo de CPU pelo Domínio-0 para realizar a criação das duas máquinas virtuais. O passo 2 cria a máquina memory3 na máquina *leblon.ufrj.br*, ilustrado na Figura 5, onde é possível notar que a quantidade de memória alocada atinge 100%, levando o Domínio 0 a realizar o balão de memória e ceder uma



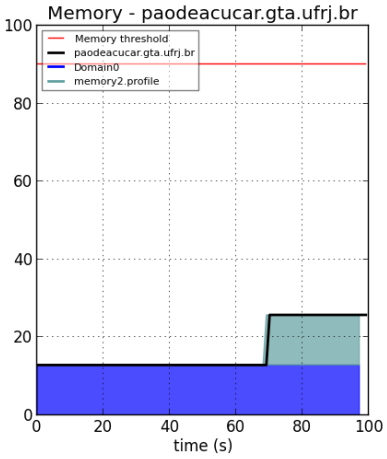
(a) Uso do processador da máquina leblon.gta.ufrj.br.



(b) Memória alocada da máquina leblon.gta.ufrj.br.



(c) Uso do processador da máquina paodeacucar.gta.ufrj.br.

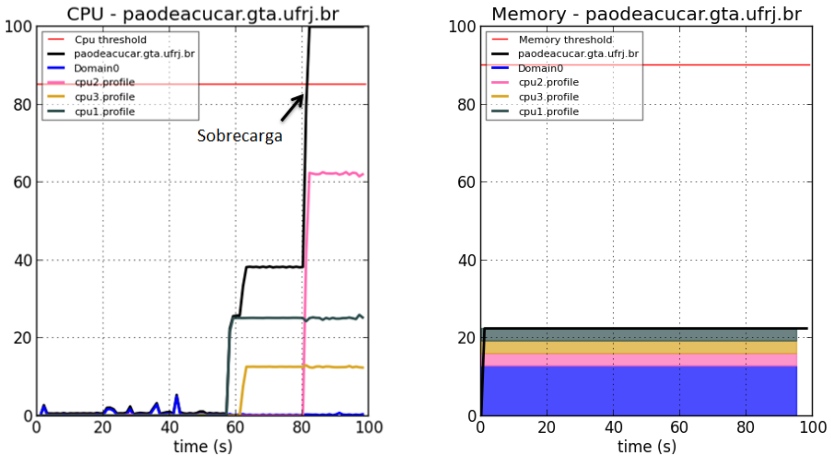


(d) Memória alocada da máquina paodeacucar.gta.ufrj.br.

Figura 6. Configuração do cenário de testes após o Passo 2. O sistema AMAS migrou a máquina memory2 para eliminar a sobrecarga de memória na máquina leblon.gta.ufrj.br. a) A migração da máquina virtual consome processamento do Domínio 0 na máquina de origem. b) A migração da máquina virtual memory2 libera memória alocada, eliminando a sobrecarga de memória. c) A migração da máquina virtual consome processamento do Domínio 0 na máquina de destino. d) Enquanto uma máquina virtual não é completamente migrada, a libvirt não contabiliza sua memória como alocada.

parte de sua memória ao domínio memory3, recém criado. Com a criação desse domínio, o limiar de segurança é ultrapassado e o Orquestrador de migração é acionado para distribuir as cargas. A Figura 6 ilustra o resultado da ação do Orquestrador que consistiu na migração da máquina virtual memory2 da máquina *leblon.ufrj.br* para a máquina *paodeacucar.ufrj.br*. Conforme o esperado, a máquina memory2 foi selecionada por possuir a menor quantidade de memória alocada dentre as 3 máquinas virtuais candidatas. Através da Figura 6 também é possível observar o impacto que a migração de máquinas virtuais causa na utilização do processador pelo Domínio 0 das duas máquinas físicas envolvidas.

No Passo 4 do cenário de testes, é simulada uma carga de trabalho nas máquinas



(a) Uso de processador. (b) Memória alocada.

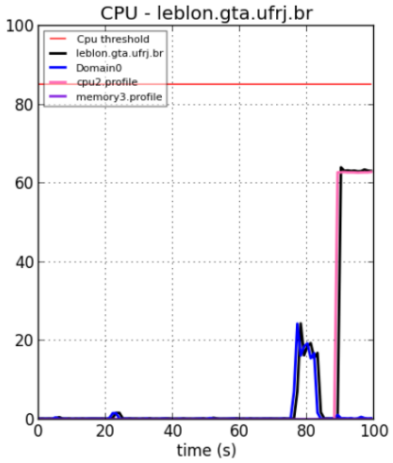
Figura 7. Passo 4 do cenário de testes. Configuração da máquina paodeacucar.gta.ufrj.br durante a execução do `Stress` nas máquinas virtuais cpu1, cpu2 e cpu3. a) A soma do consumo de CPU de todas as máquinas virtuais gera uma situação de sobrecarga de processamento. b) A alocação de memória se mantém abaixo do limiar já que não há a criação de nenhuma nova máquina virtual.

cpu1, cpu2 e cpu3, através do programa Stress. Esse passo é ilustrado na Figura 7, onde é possível observar que a utilização do processador da máquina *paodeacucar.gta.ufrj.br* ultrapassa o limiar de segurança de processamento, acionando o algoritmo de migração. De acordo com o algoritmo de decisão, a máquina selecionada para a migração é a cpu2, uma vez que apresenta a maior relação $\frac{CPU}{\text{memória}}$ dentre as máquinas virtuais candidatas. O resultado da migração pode ser visto na figura 8, onde é possível verificar a diminuição da memória alocada e do processamento da máquina *paodeacucar.gta.ufrj.br* e um aumento na quantidade de memória alocada e do processamento da máquina *leblon.gta.ufrj.br*.

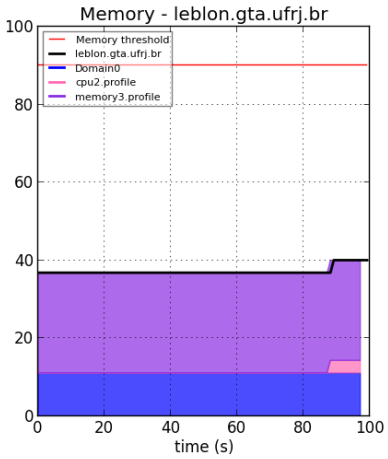
Os resultados obtidos nos testes demonstram que o sistema proposto é capaz de realizar o monitoramento de CPU, memória e rede, detectar a escassez destes recursos e distribuir as cargas de trabalho entre as máquinas físicas monitoradas. O sistema também permite a observação do comportamento do Domínio 0 nas diferentes situações, como criação, migração e destruição de máquinas virtuais. Isto é muito importante para a gerência dos acordos de níveis de serviço para verificação de alguma tentativa maliciosa de ataque negação de serviço.

5. Conclusão

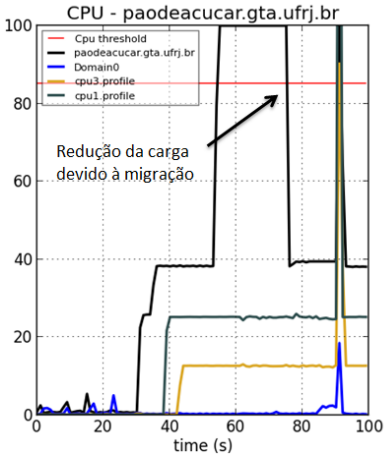
Esse artigo propõe o *Automatic Migration and Allocation System for Virtual Resource Management* (AMAS), um sistema de gerência automatizado de recursos para ambientes virtualizados, no qual é possível realizar o monitoramento, a detecção de sobrecargas e a distribuição de cargas de trabalho. O sistema é capaz de monitorar a utilização do processador, a quantidade de memória alocada e o tráfego de rede de um conjunto de máquinas físicas e virtuais, detectando escassez desses recursos e distribuindo cargas, de forma a evitar perda de desempenho nas aplicações em execução. O mecanismo de distribuição de cargas é baseado na técnica de migração ao vivo fornecida pelo Xen, que permite a migração de máquinas virtuais entre nós físicos, sem a interrupção dos serviços em execução. O sistema foi testado utilizando máquinas da plataforma de testes *Future Internet Testbed with Security* (FITS), permitindo a validação e uma avaliação



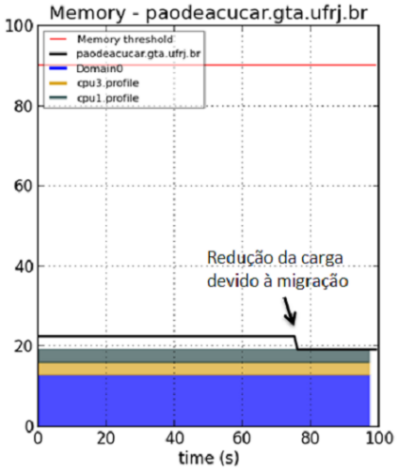
(a) Uso do processador da máquina leblon.gta.ufrj.br.



(b) Memória alocada da máquina leblon.gta.ufrj.br.



(c) Uso do processador da máquina paodeacucar.gta.ufrj.br.



(d) Memória alocada da máquina paodeacucar.gta.ufrj.br.

Figura 8. Configuração do cenário de testes após o Passo 4. O sistema AMAS identifica a sobrecarga de processamento e migra a máquina virtual cpu2 para balancear a carga. a) Após a migração, a máquina virtual cpu2 executa na máquina física leblon.gta.ufrj.br, cuja carga continua abaixo do limiar de segurança definido pelo sistema AMAS. b) Enquanto uma máquina virtual não é completamente migrada, a libvirt não contabiliza sua memória como alocada. c) A migração da máquina cpu2 libera recursos de processamento, eliminando a sobrecarga de cpu da máquina física paodeacucar. d) A alocação de memória após a migração da máquina virtual cpu2 libera memória alocada.

do desempenho da proposta. Diferentes cenários de sobrecarga foram testados e os resultados obtidos mostram que o sistema proposto é capaz de manter o nível de serviço das máquinas virtuais, mesmo em cenários de sobrecarga das máquinas físicas. Também foi implementado um painel de controle que permite a visualização do uso dos recursos computacionais em tempo real.

Como trabalho futuro, pretende-se integrar os mecanismos de monitoramento e distribuição de carga nas máquinas do FITS e disponibilizá-los como serviço na interface

Web da plataforma de teste. O sistema AMAS pode ser adaptado para computação verde através do uso métricas e critérios de migração que minimizem o consumo de energia.

6. Referências

[Alkmim et al., 2011] Alkmim, G., Batista, D. e Fonseca, N. (2011). Mapeamento de redes virtuais em substratos de rede. Em *XXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC'2011*, Campo Grande, MS.

[Arzuaga e Kaeli, 2010] Arzuaga, E. e Kaeli, D. R. (2010). Quantifying load imbalance on virtualized enterprise servers. Em *Proceedings of the first joint WOSP/SIPEW international conference on Performance engineering*, WOSP/SIPEW '10, p. 235–242, New York, NY, USA. ACM.

[Barham et al., 2003] Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T. L., Ho, A., Neugebauer, R., Pratt, I. e Warfield, A. (2003). Xen and the art of virtualization. Em *SOSP*, p. 164–177. ACM.

[Carvalho e Duarte, 2012] Carvalho, H. E. T. e Duarte, O. C. M. B. (2012). Voltaic: volume optimization layer to assign cloud resources. Em *Proceedings of the 3rd International Conference on Information and Communication Systems*, ICICS '12, p. 3:1–3:7.

[Cherkasova e Gardner, 2005] Cherkasova, L. e Gardner, R. (2005). Measuring cpu overhead for i/o processing in the xen virtual machine monitor. Em *Proceedings of the annual conference on USENIX Annual Technical Conference*, ATEC '05, p. 24–24, Berkeley, CA, USA. USENIX Association.

[Fajjari et al., 2011] Fajjari, I., Aitsaadi, N., Pujolle, G. e Zimmermann, H. (2011). Vne-ac: Virtual network embedding algorithm based on ant colony metaheuristic. Em *Communications (ICC), 2011 IEEE International Conference on*, p. 1–6.

[Govindan et al., 2007] Govindan, S., Nath, A. R., Das, A., Urgaonkar, B. e Sivasubramaniam, A. (2007). Xen and co.: Communication-aware cpu scheduling for consolidated xen-based hosting platforms.

[Guo et al., 2013] Guo, J., Liu, F., Zeng, D., Lui, J. e Jin, H. (2013). A cooperative game based allocation for sharing data center networks. Em *INFOCOM, 2013 Proceedings IEEE*, p. 2139–2147.

[Khanna et al., 2006] Khanna, G., Beaty, K., Kar, G. e Kochut, A. (2006). Application Performance Management in Virtualized Server Environments. Em *Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP*, p. 373–381.

[Lee et al., 2010] Lee, M., Krishnakumar, A. S., Krishnan, P., Singh, N. e Yajnik, S. (2010). Xentune: Detecting xen scheduling bottlenecks for media applications. Em *GLOBECOM*, p. 1–6. IEEE.

[Red Hat, 2013] Red Hat (2013). libvirt: The virtualization API. `http://libvirt.org/index.html`. Acessado em dezembro de 2013.

[Rodriguez et al., 2013] Rodriguez, E., Alkmim, G., Batista, D. e da Fonseca, N. (2013). Live migration in green virtualized networks. Em *Communications (ICC), 2013 IEEE International Conference on*, p. 2262–2266.

[Singh et al., 2008] Singh, A., Korupolu, M. e Mohapatra, D. (2008). Server-storage virtualization: Integration and load balancing in data centers. Em *High Performance Computing, Networking, Storage and Analysis, 2008. SC 2008. International Conference for*, p. 1–12.

[Sonnek et al., 2010] Sonnek, J., Greensky, J., Reutiman, R. e Chandra, A. (2010). Starling: Minimizing communication overhead in virtualized computing platforms using decentralized affinity-aware migration. Em *Parallel Processing (ICPP), 2010 39th International Conference on*, p. 228–237.

[Waterland, 2013] Waterland, A. (2013). stress project page. `http://people.seas.harvard.edu/~apw/stress/`. Acessado em dezembro de 2013.

[Wood et al., 2009a] Wood, T., Shenoy, P., Venkataramani, A. e Yousif, M. (2009a). Sandpiper: Black-box and gray-box resource management for virtual machines. *Comput. Netw.*, 53(17):2923–2938.

[Wood et al., 2009b] Wood, T., Tarasuk-Levin, G., Shenoy, P., Desnoyers, P., Cecchet, E. e Corner, M. D. (2009b). Memory buddies: Exploiting page sharing for smart colocation in virtualized data centers. Em *Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments*, VEE '09, p. 31–40, Washington, DC, USA. ACM.

[Wu e Zhao, 2011] Wu, Y. e Zhao, M. (2011). Performance modeling of virtual machine live migration. *2012 IEEE Fifth International Conference on Cloud Computing*, 0:492–499.

[Xen Project, 2009] Xen Project (2009). How Does Xen Work? `http://www-archive.xenproject.org/files/Marketing/HowDoesXenWork.pdf`. Acessado em dezembro de 2013.

[Yanagisawa et al., 2013] Yanagisawa, H., Osogami, T. e Raymond, R. (2013). Dependable virtual machine allocation. Em *INFOCOM, 2013 Proceedings IEEE*, p. 629–637.