Revista de Informática Teórica e Aplicada - RITA - ISSN 2175-2745
Vol. 25, Num. 02 (2018) 73-84

RESEARCH ARTICLE

Models for capacity oriented availability evaluation of a private cloud

Modelos para avaliação de disponibilidade orientada a capacidade de uma nuvem privada

Matheus Torquato[1]*, Lucas Torquato[2], Paulo Maciel[3]

Abstract: High availability is a significant requirement of applications hosted in clouds. It is possible to apply hardware and software redundancy to achieve higher levels of system availability. However, besides system availability, we also want to know system capacity to deal with the workload. We can use the Capacity Oriented Availability (COA) to obtain this information. The COA gives an estimate of resources available when the system is running. This paper presents a comprehensive analytical model to evaluate COA of private Clouds. We propose six scenarios with different architectures. All the architectures have the same three essential components: Front-End, PM, and VM. The presented results cover availability, COA and sensitivity analysis of model parameters. By using the model's results, it is possible to point out what components are more important for each studied metric.

Keywords: Availability — Capacity Oriented Availability — Cloud Computing — Analytical modeling

Resumo: Alta disponibilidade é um dos principais requisitos das aplicações que utilizam computação em nuvem. É possível aplicar redundâncias em hardware e software para alcançar melhores níveis de disponibilidade do sistema. Porém, além da preocupação com a disponibilidade do serviço, é necessário mensurar a capacidade do sistema em lidar com a carga de trabalho apresentada. Uma métrica que pode ser utilizada para mensurar essa capacidade é a *disponibilidade orientada a capacidade*. A partir dessa métrica, é possível obter estimativas dos recursos computacionais disponíveis para utilização quando o sistema está em funcionamento. Esse trabalho apresenta um conjunto de modelos analíticos para avaliação de disponibilidade orientada a capacidade considerando ambientes de nuvem privada. Para verificar diferentes situações, esse trabalho apresenta seis diferentes arquiteturas de nuvem privada. Os componentes fundamentais de cada arquitetura são Front-End, PM e VMs. O conjunto de resultados apresentados compreende a avaliação de disponibilidade, avaliação de disponibilidade orientada a capacidade e análise de sensibilidade dos dos parâmetros utilizados nos modelos. A partir dos resultados é possível inferir quais componentes são mais importantes para cada uma das métricas estudadas.

Palavras-Chave: Disponibilidade — Disponibilidade Orientada a Capacidade — Computação em Nuvem — Modelagem analítica

[1] *Instituto Federal de Alagoas (IFAL), Campus Arapiraca, Arapiraca-AL, Brazil*
[2] *Instituto Federal de Alagoas (IFAL), Reitoria, Maceió-AL, Brazil*
[3] *Centro de Informática, Universidade Federal de Pernambuco (CIn-UFPE), Recife-PE, Brazil*
*__Corresponding author__: matheustor4.professor@gmail.com, matheus.torquato@ifal.edu.br
DOI: http://dx.doi.org/10.22456/2175-2745.79158 • **Received:** 24/12/2017 • **Accepted:** 29/05/2018

1. Introdução

A alta disponibilidade é um dos principais requisitos para sistemas que utilizam computação em nuvem [1][2]. Eventos de falhas e paralisações nos serviços virtualizados trazem prejuízos significativos para empresas e organizações que utilizam computação em nuvem. Alguns trabalhos mostram caminhos para evitar problemas com interrupções de serviços e aumentar a disponibilidade do sistema [3] [4].

Dentre os mecanismos comumente adotados para melhorar a disponibilidade de um sistema, temos a redundância. Em sistemas redundantes, quando uma falha ocorre num sistema

principal, um sistema reserva assume a responsabilidade de responder à demanda solicitada [5]. Em *datacenters*, podemos obter redundância adicionando máquinas a um sistema principal.

Porém, além da preocupação com a disponibilidade do sistema, é necessário estimar a quantidade de recursos disponível para atender à carga de trabalho submetida. Nesse contexto, a disponibilidade orientada a capacidade é uma métrica que serve para mensurar a quantidade de recursos disponíveis quando o sistema está rodando [6]. Em relação a disponibilidade, algumas métricas de interesse são: o número de noves e o *downtime* anual. O número de noves é uma métrica comumente utilizada para categorizar a disponibilidade de um sistema. O cálculo feito para obtenção do número de noves é o seguinte $N = 2 - \log(100 - D)$, onde N é o número de noves e D é a disponibilidade. Sistemas com 5 noves são considerados altamente disponíveis [5]. O *downtime* anual mostra o tempo de inatividade do sistema no período de um ano.

Do ponto de vista de implantação, algumas universidades e organizações preferem utilizar ambientes de nuvem privados para suportar suas aplicações. Nos ambientes de computação em nuvem privados, a implantação da plataforma de nuvem computacional é feita no parque computacional da empresa/organização [7]. Geralmente, as organizações optam por esse método de implantação para tentar garantir maior nível de segurança e privacidade dos dados. Com a disseminação do paradigma, já possível implantar um ambiente de computação em nuvem privado em parques computacionais limitados, como o *OpenStack Conjure-up*[1], por exemplo. Assim, empresas e organizações que necessitam do modelo de implantação privado, poderão usufruir do paradigma de computação em nuvem.

Entretanto, mesmo em parques computacionais limitados, os serviços de nuvem disponibilizados em nuvens privadas podem possuir requisitos de alta disponibilidade e capacidade para atender às demandas apresentadas. O intuito desse trabalho é apresentar modelos analíticos para avaliação da disponibilidade de diferentes arquiteturas de nuvens privadas. A partir do modelo e dos cenários propostos é possível observar como as mudanças arquiteturais (adição e remoção de máquinas) afeta a disponibilidade e capacidade do sistema.

Esse trabalho apresenta uma avaliação de disponibilidade e de disponibilidade orientada a capacidade de seis diferentes arquiteturas de nuvens privadas. Seis arquiteturas foram consideradas. Cada uma delas possui três componentes principais: i) *Front-End* - responsável por gerenciar o ambiente da nuvem e repassar as mensagens dos usuários; ii) *PM* - máquina física responsável por hospedar as máquinas virtuais e iii) VM - máquina virtual que roda a aplicação desejada. A arquitetura padrão, denominada aqui de arquitetura *baseline*, possui um Front-End, uma PM e quatro VMs. Essa arquitetutura é utilizada como base para as demais arquiteturas. Maiores detalhes sobre as arquiteturas consideradas estão na Seção 2.

[1] https://www.ubuntu.com/download/cloud/try-openstack

A avaliação realizada é baseada em modelos analíticos. Esses modelos utilizam o formalismo de Redes de Petri Estocásticas [8]. Os modelos construídos levam em consideração os comportamentos de falha e reparo de cada um dos componentes da nuvem. Maiores detalhes sobre os modelos estão na Seção 3. O principal intuito das avaliações realizadas é de perceber como as mudanças arquiteturais (inclusão ou remoção de máquinas) afeta a disponibilidade e a disponibilidade orientada a capacidade do sistema. Por exemplo, a partir da observação dos resultados dos modelos é possível concluir quais componentes da arquitetura possuem maior impacto na disponibilidade ou na disponibilidade orientada a capacidade no ambiente.

Os cálculos para perceber como as alterações podem afetar o sistema são baseados na análise de sensibilidade [9]. De modo simples, a análise de sensibilidade permite observar o comportamento da métrica de interesse (no caso desse trabalho, disponibilidade e disponibilidade orientada a capacidade) mediante a alteração nos parâmetros. Existem diversos métodos para conduzir a análise de sensibilidade. Este artigo considera o método baseado em índices de sensibilidade. Basicamente, os índices de sensibilidade revelam, atráves de uma classificação, quais parâmetros causam maior impacto na métrica de interesse. Logo, conclui-se que parâmetros com maior valor de índice de sensibilidade causam impacto maior que os com índices de sensibilidade com valor menor [10].

Os estudos de caso apresentados estão na Seção 4. O estudo de caso 1 apresenta a avaliação de disponibilidade do sistema. O estudo de caso 2 possui a análise de sensibilidade de cada um dos parâmetros na disponibilidade do sistema. Optou-se por dividir a análise de sensibilidade em duas seções, uma sobre os parâmetros de falha e outra sobre os de reparo. Assim, é possível observar melhor como as políticas de tolerância a falhas (aumentar o tempo de falha) e de manutenção (reduzir o tempo de reparo) podem afetar o ambiente. Além disso, a variação dos parâmetros de falha e reparo são diferentes, utilizar apenas uma análise de sensibilidade poderia prejudicar os resultados. O estudo de caso 3 apresenta a avaliação de disponibilidade orientada a capacidade do sistema. E o estudo de caso 4 apresenta a análise de sensibilidade dos parâmetros sobre a disponibilidade orientada a capacidade. Todos os estudos de caso com análise de sensibilidade contemplam apenas a arquitetura *baseline*. Não há problema em projetar os resultados para as demais arquiteturas, já que são expansões da arquitetura *baseline*. Cada estudo de caso apresenta uma breve discussão dos resultados.

Os resultados apresentados nos estudos de caso mostram que o Front-End é o componente mais importante para a disponibilidade do sistema. E, a PM é o componente mais importante para a disponibilidade orientada a capacidade. A análise de sensibilidade, em ambas as métricas, mostrou que o parâmetro com maior relevância é o tempo de reparo do Front-End. Ou seja, para maximizar a disponibilidade e a disponibilidade orientada a capacidade na arquitetura *baseline*, a melhor opção é reduzir o tempo de reparo do Front-End. As

demais conclusões do trabalho são apresentadas na Seção 6

Comparado aos artigos já publicados (maiores detalhes na Seção 5), o presente trabalho visa contribuir com a área acrescentando modelos que contemplem diversas arquiteturas diferentes de nuvens privadas. De modo específico as contribuições desse trabalho são as seguintes:

- Um modelo genérico para avaliação de disponibilidade e de disponibilidade orientada a capacidade em ambientes de nuvem ou virtualizados;
- Um conjunto abrangente de estudos de caso, apresentando resultados relativos a disponibilidade, disponibilidade orientada a capacidade e análise de sensibilidade dos parâmetros.

O restante do trabalho está organizado da seguinte maneira. A Seção 2 possui a descrição das arquiteturas consideradas nessa pesquisa. A Seção 3 possui os modelos analíticos propostos. A Seção 4 apresenta os estudos de caso e seus resultados. A Seção 5 possui um levantamento dos trabalhos relacionados. Por fim, a Seção 6 apresenta as conclusões e trabalhos futuros.

2. Arquiteturas consideradas

A Figura 1 contém cada uma das arquiteturas consideradas nesse trabalho. Todas as arquiteturas são baseadas em três componentes fundamentais. São eles: i) **Front-End** - que é responsável por encaminhar as requisições enviadas pelos usuários e gerenciar o ambiente de nuvem; ii) **PM** - máquina física que disponibiliza seus recursos computacionais para as máquinas virtuais; e iii) **VM** - máquina virtual que executa os serviços desejados.

A arquitetura inicial (*baseline*) é a `A0`, que contém apenas um Front-End e uma PM. As demais arquiteturas são baseadas na arquitetura *baseline*. Ou seja, todas as arquiteturas possuem, minimamente, um Front-End e uma PM. Por exemplo, se adicionarmos um Front-End à arquitetura *baseline* teremos a arquitetura `A3`.

Para o escopo desse trabalho, cada máquina adicional opera em conjunto com as máquinas presentes na arquitetura *baseline*. Além de atuar dividindo a carga de trabalho, as máquinas atuam num esquema de redundância dinâmica em *hot-standby*. Nesse tipo de redundância um dos componentes assume a responsabilidade da tarefa no caso da falha do outro [11]. Assim sendo, esse trabalho considera que sistema estará disponível quando, ao menos uma instância de cada um dos componentes fundamentais estiver disponível. Ou seja, o sistema está disponível quando ao menos um Front-End, uma PM e uma VM estiverem funcionando simultaneamente.

Em todas as arquiteturas consideradas, a quantidade de máquinas virtuais por máquina física ficou fixada em quatro. Essa escolha foi fundamentada em estudos prévios que mostraram que é razoável alocar 20% de capacidade para cada VM (no caso 80% de capacidade para VMs) e deixar 20% para utilização do próprio hospedeiro [12].

3. Modelos propostos

Além de considerar as arquiteturas mostradas na seção anterior, os modelos também possuem um conjunto adicional de premissas, são elas:

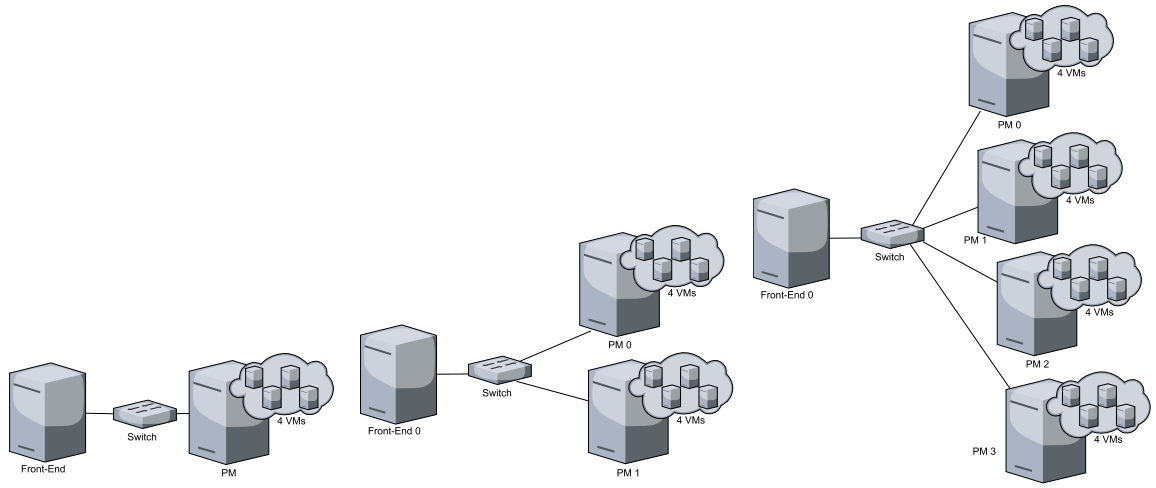
- Possíveis falhas nos equipamentos de rede e enlaces de comunicação são desconsideradas;
- O *reboot* necessário para reativar VMs após uma falha da PM é realizado em paralelo para todas as VMs naquela PM. Ou seja, todas as VMs daquela PM são reiniciadas simultaneamente;
- As falhas nas VMs podem acontecer simultaneamente.

O modelo principal utilizado para as avaliações apresentadas nos estudos de caso da Seção 4 está na Figura 2. Esse modelo principal foi dividido em sub-modelos para cada um dos componentes de hardware da arquitetura. Para avaliar as arquiteturas que não utilizam todos os componentes basta remover os *tokens* dos lugares correspondentes. O lugar `FE_UP` possui uma variável (`NFE`) que indica a quantidade de Front-Ends para a arquitetura avaliada. Ou seja, `NFE` possuirá valor 1 nas arquiteturas `A0`, `A1` e `A2`, e valor 2 nas arquiteturas `A3`, `A4` e `A5`.

A Figura 3 apresenta o sub-modelo capaz de avaliar a arquitetura *baseline*. Como o modelo principal representa as expansões da arquitetura *baseline*, será possível compreender os pontos principais apenas a partir do modelo da arquitetura *baseline*.

No instante inicial considera-se que todos os componentes da arquitetura estão funcionando normalmente. Isso é representado pelos *tokens* nos lugares `FE_UP` (representando que um Front-End está ativo), `PM0_UP` (representando que a PM está ativa) e `VM0_UP` (representam que as quatro VMs estão ativas). A partir desse ponto alguns eventos são possíveis como: falha no Front-End, falha na PM e falha nas VMs. O primeiro evento, falha no Front-End é representado pelo disparo da transição `FE_f`. Esse disparo deposita um token no lugar `FE_DW`. A presença de *tokens* nesse lugar representa que o Front-End está inativo. É importante lembrar que, no contexto desse trabalho, isso significa dizer que o sistema está indisponível. Para retornar à atividade, o sistema necessita de reparo no Front-End. O evento de reparo do Front-End é representado pelo disparo da transição `FE_r`. Após o disparo dessa transição, o *token* é recolocado no lugar `FE_UP`.

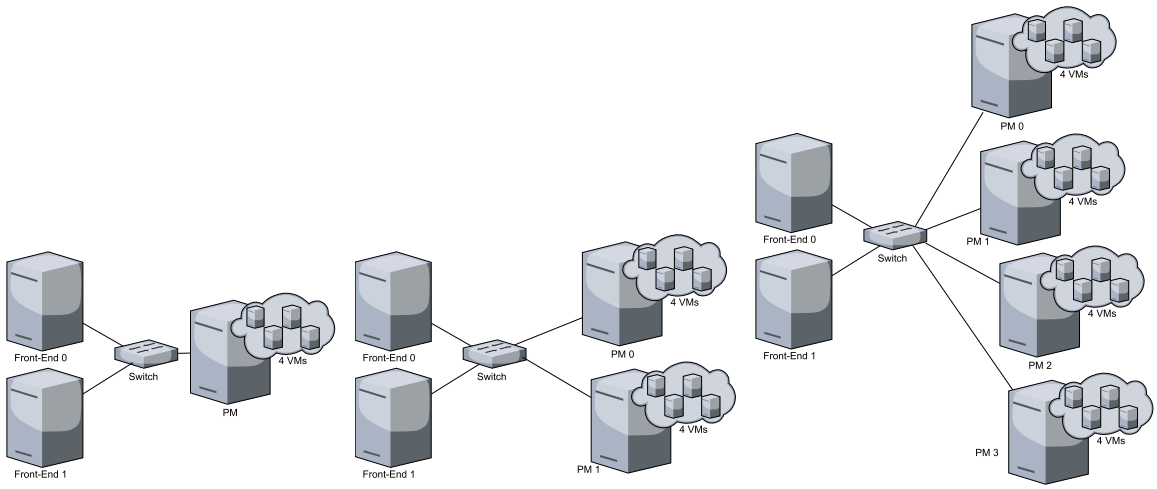
O comportamento de falha e reparo das VMs é similar ao adotado para o Front-End. A quantidade de VMs funcionando é indicada pela quantidade de *tokens* que estão no lugar `VM_UP`. O evento de falha das VMs é representado pelo disparo da transição `VM0_f`. O disparo dessa transição deposita um *token* no lugar `VM_DW`. A quantidade de *tokens* no lugar `VM_DW` indica a quantidade de VMs que está inativa. Vale a pena lembrar que o sistema é considerado disponível se ao menos uma VM estiver funcionando. O reparo da VM



(a) A0 - Um Front-End e uma PM (Baseline)

(b) A1 - Um Front-End e duas PMs

(c) A2 - Um Front-End e quatro PMs



(d) A3 - Dois Front-Ends e uma PM

(e) A4 - Dois Front-Ends e duas PMs

(f) A5 - Dois Front-Ends e quatro PMs

Figura 1. Arquiteturas consideradas

é representado pelo disparo da transição VM0_r. O disparo dessa transição deposita o *token* de volta no lugar VM_UP.

O comportamento de falha na PM segue o mesmo padrão anterior. O evento de falha na PM é representado pelo disparo da transição PM0_f. O disparo dessa transição remove o *token* do lugar PM0_UP (que serve para representar que a PM está funcionando) para o lugar PM0_DW, que representa que a PM está inativa. Como a PM hospeda as máquinas virtuais, as falhas ocorridas nesse componente acarretam paralisação dos serviços da nuvem (o sistema ficará indisponível). Quando uma falha ocorre na PM todas as máquinas virtuais irão parar de funcionar. Isso é representado pelo disparo das transições imediatas PM0-f2 e PM0-f3. A presença do arco inibidor[2] indica que a transição só poderá ser disparada na ausência de tokens nos lugares de entrada. O disparo dessas transições recolhe os *tokens* nos lugares relativos às VMs e os deposita no lugar VM0_DW2. Para a recuperação completa pós falha da PM são necessários dois passos: i) reparo na PM e ii) *reboot* das VMs. O primeiro passo é representado pelo disparo da transição PM0_r. Esse disparo deposita um *token* no lugar VM0_OFF. O *reboot* das VMs é representado pelo disparo da transição VM0_rb. Depois desse disparo um token é depositado no lugar PM0_UP, representando que a PM voltou a funcionar. Além disso, a transição imediata VM0-rb2 recolhe os *tokens* no lugar VM0_DW2 e os coloca no lugar VM0_UP. Isso significa dizer que, no cenário considerado nesse trabalho, o reparo da PM compreende o reparo em VMs que estiverem em estado de falha. É importante notar que a transição imediata VM0_rb2 só estará habilitada se não houver *tokens* nos lugares PM0_DW e VM0_OFF. Isso significa que, as VMs só retornam à atividade após o reparo completo na PM.

Os valores utilizados nos parâmetros utilizados no modelo estão na Tabela 1. Os valores desses parâmetros foram obtidos de trabalhos publicados previamente [13] [14]

[2]Representado por um arco com círculo no final

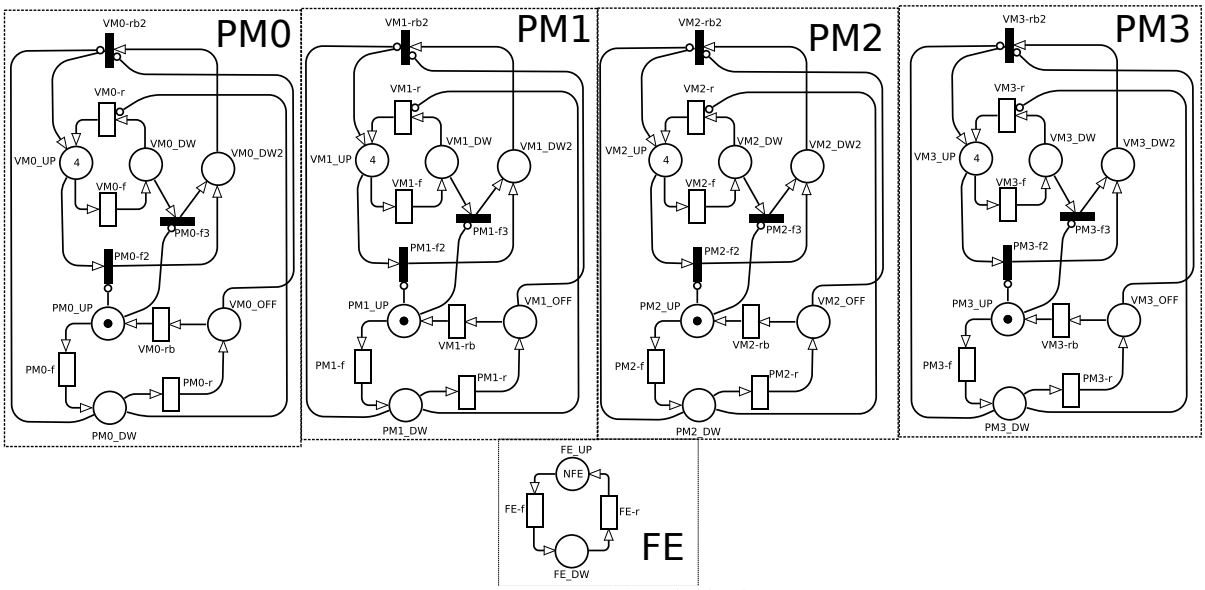


Figura 2. Modelo principal

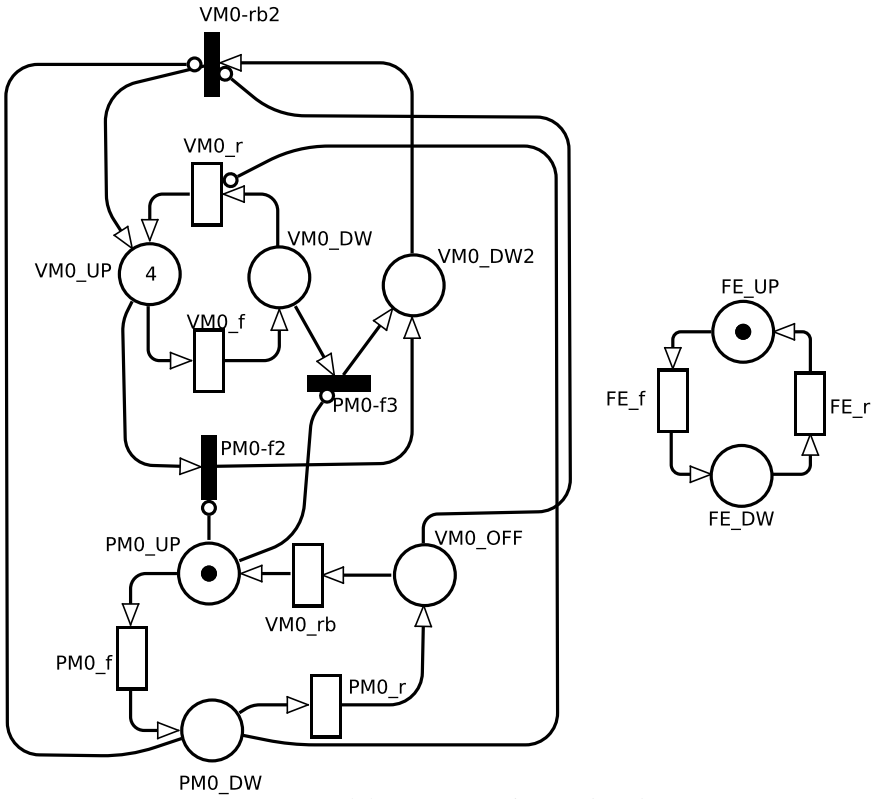


Figura 3. Modelo para a arquitetura *baseline*

e [15]. Os valores de taxas de falha de hardware são baseados em informações disponibilizadas pelos fabricantes de equipamentos. Os valores de taxas de falha relacionadas aos softwares foram obtidos através de estimativas, haja visto que esses dados são confidenciais ou muito difíceis de obter via experimentação num ambiente real. A experimentação de sistemas para obtenção de valores de taxas como MTTF pode levar muito tempo para ser realizada. Além disso, tais dados podem refletir uma situação específica de um sistema, possuindo assim problemas com generalizações [16].

A ferramenta Mercury foi utilizada para a avaliação dos modelos apresentados [17]. Nós selecionamos essa ferramenta por possuir funcionalidade embutida para cálculo de análise de sensibilidade.

Tabela 1. Parâmetros utilizados nos modelos

Parâmetro	Descrição	Tempo
FE_f	Tempo médio para falha do Front-End	481,5 hr
FE_r	Tempo médio para reparo do Front-End	1,03 hr
PM_f	Tempo médio para falha da PM	1236,7 hr
PM_r	Tempo médio para reparo da PM	1,09 hr
VM_f	Tempo médio para falha da VM	2880 hr
VM_r	Tempo médio para reparo da VM	30 min
VM_rb	Tempo médio para *reboot* da VM	5 min

4. Estudos de caso

Estudo de caso 1 - Avaliação de disponibilidade

O primeiro estudo de caso apresenta a avaliação de disponibilidade do sistema. Essa avaliação consiste em mensurar a probabilidade do sistema possuir ao menos uma instância de cada um dos componentes fundamentais da arquitetura funcionando. Por exemplo, na arquitetura *baseline* essa métrica é calculada observando a probabilidade da presença de *tokens* nos lugares FE_UP, PM0_UP e VM0_UP simultaneamente. Considere $P\{\}$ como probabilidade e $\#L$ como a quantidade de tokens no lugar L. O cálculo da disponibilidade da arquitetura *baseline* pode ser obtido através da expressão lógica a seguir: $Disponibilidade_{baseline} = P\{\#VM0\_UP > 0\, AND\, \#FE\_UP > 0\, AND\, \#PM0\_UP > 0\}$.

Os resultados da avaliação de disponibilidade das arquiteturas consideradas estão na Tabela 2 abaixo.

Tabela 2. Resultados da avaliação de disponibilidade

Arquitetura	Disponibilidade	Número de noves	Downtime anual
A0 (*baseline*)	0,996919581	2,511390274	26,9845 hr
A1	0,997864521	2,670504730	18,7068 hr
A2	0,997865418	2,670687095	18,6989 hr
A3	0,999047579	3,021170881	8,3432 hr
A4	0,999994545	5,263215197	2,8671 min
A5	0,999995444	5,341374154	2,3949 min

Discussão

Os resultados desse estudo de caso servem para determinar os impactos que as mudanças na arquitetura causam na disponibilidade do sistema. É interessante notar que, para maximizar a disponibilidade do sistema é mais eficaz adicionar máquinas como Front-End, em vez de novas PMs. De fato, analisando apenas as arquiteturas A4 e A5 nota-se que a adição de duas PMs numa arquitetura que já possui dois Front-Ends ocasiona uma redução de apenas aproximadamente 0,5 minuto de *downtime* anual.

Estudo de caso 2 - Análise de sensibilidade aplicada a avaliação da disponibilidade

Ainda no contexto de disponibilidade do sistema, suponha a seguinte situação. O administrador do ambiente de nuvem deseja maximizar a disponibilidade do sistema, porém, não dispõe de recursos financeiros para a aquisição de novos equipamentos. Algumas outras ações podem ser tomadas, por exemplo, a implantação de mecanismos de redundância em nível de software [18]. Além disso, é possível elaborar rotinas mais frequentes de manutenção para evitar falhas no sistema, ou ainda realizar o reparo de modo mais rápido. Assim sendo, esse estudo de caso serve para mensurar o impacto que cada um dos parâmetros de falha e reparo tem sobre a disponibilidade do sistema.

O cenário deste estudo de caso considera a arquitetura *baseline* como padrão e os valores dos parâmetros são alterados conforme a análise de sensibilidade proposta.

Falha

Os parâmetros foram variados de 200 a 3000 utilizando 2500 pontos de amostragem. A escolha desses valores se deu devido às taxas comumente encontradas na literatura. Os tempos médios de falha costumam ser maiores que os tempos de reparo de um componente. A faixa específica de 200 a 3000 visa uma análise de sensibilidade mais abrangente, em vez de utilizar apenas os valores-padrão da Tabela 1. Ou seja, a utilização de taxas de falha mais baixas representa situações na quais o sistema está mais vulnerável a falhas, como em ambientes com limitações de refrigeração, ou com alta interferência eletromagnética. As taxas de falha mais altas visam representar sistemas que são mais robustos a falhas. Por exemplo, ambientes com redundâncias e maior tolerância a falhas. Para ter uma maior fidelidade nos resultados, tentamos usar o máximo de pontos de amostragem possível. Devido à limitações de processamento, o limite de pontos de amostragem para uma avaliação em tempo razoável foi de 2500.

O índice de sensibilidade de cada parâmetro pode ser visualizado na Tabela 3. Esses resultados mostram que a aplicação de mecanismos de tolerância a falhas (ou seja, aumentar o tempo médio de falha) na PM geram melhores resultados que quando aplicados em outros componentes da arquitetura.

Para observar melhor a importância de cada parâmetro em relação à disponibilidade, a Figura 4 apresenta um gráfico com os resultados completos da análise de sensibilidade. Existe uma linha para cada componente da arquitetura. A linha tracejada *baseline* indica o valor padrão da disponibilidade, utilizando apenas os valores da Tabela 1.

A partir da observação desses resultados (principalmente a Figura 4) é possível perceber que, a partir de um certo ponto a aplicação de mecanismos de tolerância a falhas não fornece alterações significativas na disponibilidade. Observando especialmente o comportamento da VM, é possível perceber que, dentro da alteração proposta nesse estudo de caso, a alteração do tempo médio de falha da VM não afetou a disponibilidade do sistema. Isso significa dizer que, utilizar políticas para aumentar o tempo de falha das VMs, no escopo desse trabalho, não seria eficaz para aumentar a disponibilidade do sistema.

Reparo

Os parâmetros de reparo foram variados de 0,001 a 100 utilizando 100 pontos de amostragem. Assim como explicado anteriormente, variamos as taxas observando as taxas utilizadas na literatura. O reparo em tempo mínimo pode ser obtido por sistemas que possuem redundância e/ou utilizam mecanismos de autogerenciamento do ambiente. Tempos de reparo mais longos podem ser ocasionados por falta de equipe de

Tabela 3. Análise de sensibilidade - Disponibilidade - MTTFs

Parâmetro	Descrição	Índice de sensibilidade
PMFail	Tempo médio para falha da PM	0,005443604460711194
FEFail	Tempo médio para falha do Front-End	0,004782038819413688
VMFail	Tempo médio para falha na VM	4,1856477104994284E-8

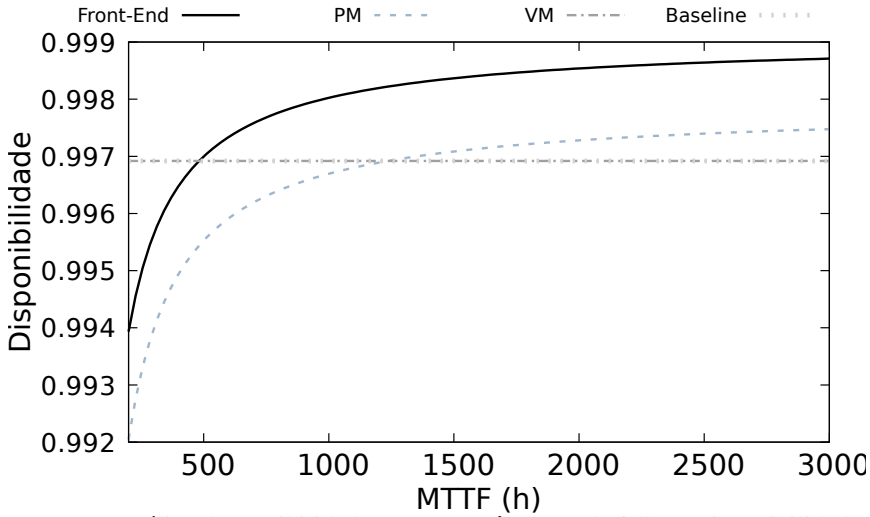


Figura 4. Análise de sensibilidade para os parâmetros de falha - Disponibilidade

manutenção ou dificuldades em detectar problemas.

O índice de sensibilidade dos parâmetros está na Tabela 4. Os resultados indicam que, uma manutenção mais eficaz para resolução de problemas (tempo de reparo mais curto) no Front-End, resulta em melhores resultados quando comparada com os outros componentes da arquitetura. Os resultados completos estão na Figura 5. Assim como visto nos resultado anterior, a alteração do tempo de reparo da VM não ocasiona mudanças na disponibilidade (obedecendo ao escopo desse trabalho).

Discussão

A partir dos resultados apresentados nas Tabelas 4 e 3 e Figuras 4 5, observou-se que, em relação ao tempo médio de reparo, o Front-End é o mais importante. Numa visão mais abrangente, incluindo tempos de falha e reparo, é possível notar que o tempo de reparo do Front-End é o parâmetro mais importante, pois ele possui o maior índice de sensibilidade. Assim, é possível concluir que, uma abordagem adequada para maximizar a disponibilidade da arquitetura *baseline* é encurtar o tempo de reparo do Front-End. Isso pode ser alcançado com rotinas mais frequentes de manutenção ou uma equipe de manutenção com maior disponibilidade.

Estudo de caso 3 - Avaliação de disponibilidade orientada a capacidade

Esse estudo de caso está relacionado com a avaliação de disponibilidade orientada a capacidade do sistema. A partir da avaliação da disponibilidade orientada a capacidade desse estudo de caso, é possível perceber como a adição de máquinas na arquitetura afeta sua capacidade em lidar com carga de trabalho. É importante salientar que esse estudo de caso considera que a capacidade padrão do sistema é de 4 VMs ativas e funcionando simultaneamente. Ou seja, a arquitetura *baseline* é tomada como referência para cálculo dessa métrica.

Considere $E\{\#L\}$ como o número esperado de *tokens* no lugar L, $P\{\}$ como probabilidade. O cálculo da disponibilidade orientada a capacidade da arquitetura *baseline* pode ser obtido através da expressão lógica a seguir: $COA_{baseline} = (E\{\#VM0\_UP > 0\}/4) \times (P\{\#FE\_UP > 0\})$.

Essa expressão considera o número de *tokens* no lugar VM0_UP dividido por 4, que é o número de VMs. Como a disponibilidade do Front-End afeta a disponibilidade do sistema, deve-se levar em consideração no cálculo da disponibilidade orientada a capacidade. Por isso, utiliza-se o trecho $(P\{\#FE\_UP > 0)$ nessa expressão. Para calcular a disponibilidade orientada a capacidade nas demais arquiteturas, deve-se considerar os demais lugares que denotam o número de VMs ativas no sistema. Por exemplo, a disponibilidade orientada a capacidade da arquitetura A1 pode ser obtida através da expressão a seguir $COA_{A1} = ((E\{\#VM0\_UP\} + E\{\#VM1\_UP\})/4) \times (P\{\#FE\_UP > 0\})$.

Os resultados da avaliação estão na Tabela 5. É possível notar que, a adição de uma PM na arquitetura gera um ganho aproximado de 100% na disponibilidade orientada a capacidade. Na arquitetura A5 onde há dois Front-Ends e quatro PMs rodando, o sistema consegue rodar com quase 400% da capacidade.

Fazendo um contraponto com os estudos de caso anteriores, nota-se que a adição de uma PM gera resultados melhores que a adição de um Front-End. Isso acontece porque o Front-

Tabela 4. Análise de sensibilidade - Disponibilidade - MTTRs

Parâmetro	Descrição	Índice de sensibilidade
FErepair	Tempo médio para reparo do Front-End	0,1719673510432318
PMrepair	Tempo médio para reparo da PM	0,07480567995929226
VMrepair	Tempo médio para reparo da VM	2,2082227598260767E-5

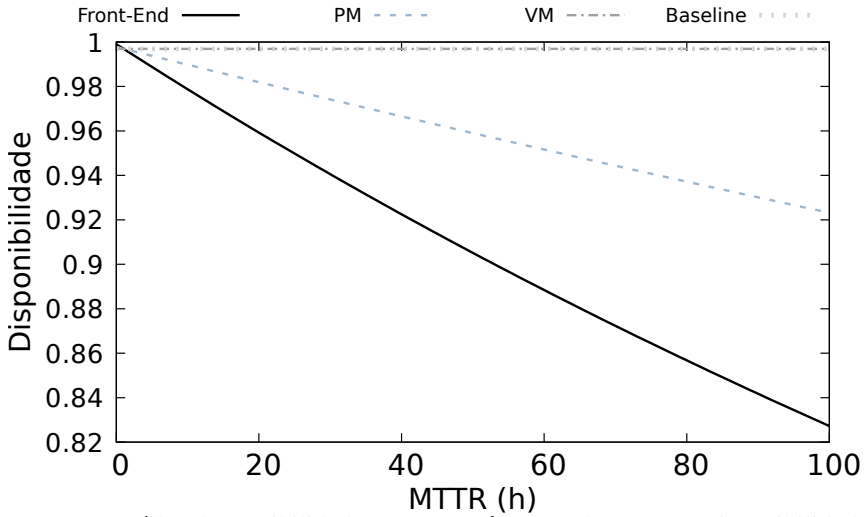


Figura 5. Análise de sensibilidade para os parâmetros de reparo - Disponibilidade

End não roda VMs no ambiente, ele apenas o gerencia e encaminha as mensagens vindas dos usuários. Assim, adicionar um Front-End melhora a disponibilidade do sistema, mas o impacto sobre a disponibilidade orientada a capacidade é reduzido quando comparado ao causado por uma adição de uma máquina física.

Tabela 5. Resultados da avaliação de disponibilidade orientada a capacidade

Arquitetura	Disponibilidade Orientada a Capacidade
A0	0,996876322
A1	1,993493030
A2	3,986986060
A3	0,999004227
A4	1,997748305
A5	3,995496610

Discussão

Esse estudo de caso apresentou a avaliação da disponibilidade orientada a capacidade para as arquiteturas apresentadas na Seção 2. Diferentes componentes na arquitetura causam diferentes impactos nessa métrica. Chega-se a conclusão que, a inclusão de PMs (e conseqüentemente VMs) no ambiente gera melhores resultados quando comparada à inclusão de Front-Ends.

Estudo de caso 4 - Análise de sensibilidade aplicada a avaliação da disponibilidade orientada a capacidade

O último estudo de caso proposto nesse artigo apresenta a análise de sensibilidade dos parâmetros de falha e reparo dos modelos em relação à disponibilidade orientada a capacidade. O cenário considerado aqui utiliza a arquitetura *baseline* como padrão e altera-se os valores dos parâmetros.

Falha

Assim como no estudo de caso 2, os parâmetros foram variados de 200 a 3000 com 2500 pontos de amostragem. A justificativa para a escolha desses parâmetros a de seguir o mesmo padrão estudado na avaliação anterior. Assim, podemos ter uma visão que compreende disponibilidade e disponibilidade orientada a capacidade do sistema. Os índices de sensibilidade dos parâmetros de falha utilizados no modelo estão na Tabela 6. O resultado completo da análise de sensibilidade está na Figura 6.

O parâmetro de falha mais importante para a disponibilidade orientada a capacidade é o tempo de falha da máquina física. A partir do gráfico na Figura 6 nota-se que a variação do tempo de falha na VM também afeta a disponibilidade orientada a capacidade, diferentemente dos resultados mostrados nos estudos de caso sobre a disponibilidade.

Reparo

Assim como no estudo de caso 2, os parâmetros de reparo foram variados de 0,001 a 100 utilizando 100 pontos de amostragem. O índice de sensibilidade dos parâmetros de reparo está presente na Tabela 7. É possível notar que de reparo mais mais importante no sistema é o do Front-End. A Figura 7 possui o resultado completo da análise de sensibilidade realizada.

Discussão

Tabela 6. Análise de sensibilidade - Disponibilidade orientada a capacidade (COA) - MTTFs

Parâmetro	Descrição	Índice de sensibilidade
PMFail	Tempo médio para falha da PM	0,00544320210931872
FEFail	Tempo médio para falha do Front-End	0,004782038596213683
VMFail	Tempo médio para falha na VM	0,0023451804466523155

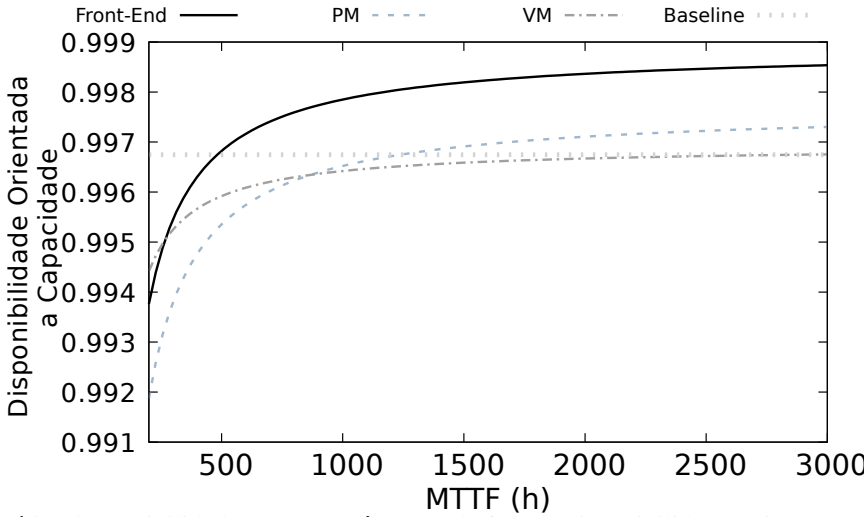


Figura 6. Análise de sensibilidade para os parâmetros de falha - Disponibilidade Orientada a Capacidade

Tabela 7. Análise de sensibilidade - Disponibilidade orientada a capacidade (COA) - MTTRs

Parâmetro	Descrição	Índice de sensibilidade
FErepair	Tempo médio para reparo do Front-End	0,17196735130626295
PMrepair	Tempo médio para reparo da PM	0,07480567999773383
VMrepair	Tempo médio para reparo da VM	0,0338501150018357

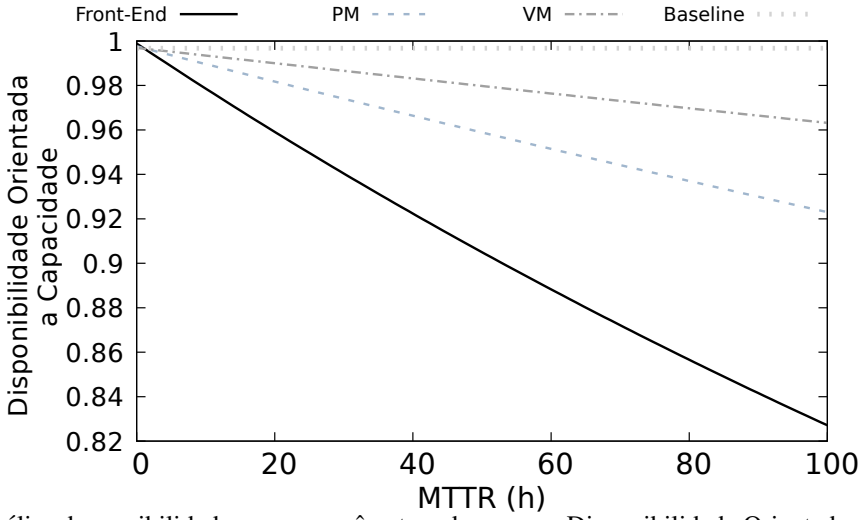


Figura 7. Análise de sensibilidade para os parâmetros de reparo - Disponibilidade Orientada a Capacidade

Os resultados da análise de sensibilidade em relação a disponibilidade orientada a capacidade se mostram semelhantes aos do estudo de caso 2. A partir dos resultados apresentados nas Tabelas 6 e 7 e Figuras 6 7 é possível chegar a conclusão que, considerando a arquitetura *baseline*, a estratégia mais adequada para maximizar a disponibilidade e a disponibilidade orientada a capacidade é reduzir o tempo de reparo do Front-End.

5. Trabalhos relacionados

O artigo [15] possui uma avaliação abrangente de disponibilidade em ambientes virtualizados. Esse é um dos principais artigos da área. As taxas utilizadas nos modelos do nosso trabalho foram baseadas nesse trabalho. Diferente do proposto nele, nosso trabalho é focado na disponibilidade orientada a capacidade e considera diferentes arquiteturas de nuvens privadas.

O artigo [19] apresenta modelos para avaliação de disponibilidade orientada a capacidade em ambientes de nuvem. O trabalho [20] apresenta um estudo focado em análise de sensibilidade. O artigo [21] que apresenta um estudo contemplando outras métricas como consumo de energia e custo financeiro. Diferente do proposto nesses artigos, o nosso trabalho compreende diferentes arquiteturas de nuvens privadas.

O artigo [22] apresenta um modelo para avaliação de disponibilidade, desempenho e consumo de energia de uma nuvem IaaS. Os modelos propostos cobrem diferentes cenários incluindo tempos de provisionamento e processamento de serviços. Os autores também comparam os resultados obtidos com resultados de simulação oriundos do framework CloudSim [23]. Diferente desse trabalho, o nosso trabalho foca em disponibilidade orientada a capacidade.

O artigo [24] apresenta análise de sensibilidade de diferentes parâmetros na disponibilidade de uma nuvem computacional. Os autores usam Redes de Petri para a avaliação. Os modelos propostos compreendem cenários com diferentes políticas de reparo e diferentes arquiteturas do sistema. Esse artigo também negligencia a avaliação de disponibilidade orientada a capacidade.

O artigo [25] apresenta um modelo hierárquico para avaliação de disponibilidade de uma nuvem *Eucalyptus* [26]. Os modelos propostos também utilizam Redes de Petri. Porém, há uma composição hierárquica com outros paradigmas como cadeias de markov e diagrama de blocos de confiabilidade. Os autores apresentam uma análise de sensibilidade extensa considerando diferentes técnicas de redundância. Assim como alguns artigos anteriores, esse também não apresenta avaliação de disponibilidade orientada a capacidade.

Outros artigos relevantes como [27] [28] [29] [13] [14] [30] [31] consideram a avaliação de disponibilidade em ambientes de computação em nuvem. Porém, assim citado anteriormente, o nosso artigo possui um enfoque diferente, contemplando também a disponibilidade orientada a capacidade.

6. Conclusões e trabalhos futuros

Esse trabalho apresentou um conjunto de modelos analíticos para avaliação de ambientes de computação em nuvem privados. A avaliação foi feita em termos de disponibilidade e disponibilidade orientada a capacidade. Foram consideradas seis arquiteturas para cobrir diferentes situações num ambiente de nuvem. Esse trabalho quatro estudos de caso que consideram a avaliação de disponibilidade, avaliação de disponibilidade orientada a capacidade e a análise de sensibilidade dos parâmetros utilizados nos modelos.

A partir dos resultados obtidos é possível traçar diretrizes para maximizar cada uma das métricas. Para maximizar a disponibilidade, quando é possível adicionar novas máquinas, a melhor opção é adicionar mais Front-Ends. Em ambientes onde é inviável adquirir novas máquinas para a infraestrutura, a opção mais viável é aplicar políticas para reduzir o tempo de reparo do Front-End. O mesmo acontece em relação à disponibilidade orientada a capacidade. Em ambientes onde a arquitetura de hardware é escalável, a melhor opção para maximizar a disponibilidade orientada a capacidade é a adição de novas máquinas físicas para rodar VMs. Em alguns casos estudados, os ganhos chegaram a aproximadamente 300%.

Dentre os pontos fortes do trabalho apresentado, podem ser destacados os seguintes. A abrangência dos resultados apresentados e a simplicidade dos modelos analíticos utilizados. Os modelos apresentados nesse artigo podem ser adaptados para outros tipos de arquiteturas. Assim, é possível elaborar novas avaliações para cada ambiente desejado.

Para trabalhos futuros, almeja-se a expansão dos itens considerados na arquitetura, incluindo dispositivos de rede e fornecimento de energia. Como objetivos secundários, pretende-se elaborar experimentos práticos de injeção de falhas e reparos para comparação com os resultados obtidos a partir dos modelos.

7. Contribuição dos autores

Matheus Torquato foi responsável pela escrita, modelagem e análise dos dados.
Lucas Torquato conduziu as revisões e correções textuais.
Paulo Maciel foi o orientador do trabalho indicando os direcionamentos para a condução da pesquisa. Além disso, ele contribuiu diretamente na construção dos modelos propostos.

Acrônimos

PM → *Physical Machine* (Máquina física)
VM → *Virtual Machine* (Máquina virtual)
MTTF → *Mean Time To Failure* (Tempo estimado para falha)
MTTR → *Mean Time To Repair* (Tempo estimado para reparo)
COA → *Capacity Oriented Availability* (Disponibilidade Orientada a Capacidade)
IaaS → *Infrastructure as a Service* (Infraestrutura como serviço)

References

[1] CISCO. *Cisco Global Cloud Networking Survey Summary and Analysis of ResultsWorldwide Results.* San Jose, USA, 2012.

[2] IDG-ENTERPRISE. *2016 - Cloud Computing Survey - IDG Enterprise*. 2016.

[3] PHAM, C. et al. Toward a high availability cloud: Techniques and challenges. In: SWARZ, R.; KOOPMAN, P.; CUKIER, M. (Ed.). *Dependable Systems and Networks Workshops (DSN-W), 2012 IEEE/IFIP 42nd International Conference on*. Boston, USA: IEEE, 2012. v. 1.

[4] HAGEN, S.; SEIBOLD, M.; KEMPER, A. Efficient verification of it change operations or: How we could have prevented. In: NOMS. *Network Operations and Management Symposium*. Maui, USA: IEEE, 2012. v. 1.

[5] GRAY, J.; SIEWIOREK, D. P. High-availability computer systems. *Comput.*, v. 24, n. 9, p. 39–48, 1991.

[6] HEIMANN, D. I.; MITTAL, N.; TRIVEDI, K. S. Availability and reliability modeling for computer systems. In: YOVITS, M. C. (Ed.). Amsterdam, Netherlands: Elsevier, 1990, (Advances in Computers, v. 31). p. 175 – 233.

[7] JADEJA, Y.; MODI, K. Cloud computing-concepts, architecture and challenges. In: ICCEET. *Computing, Electronics and Electrical Technologies (ICCEET), 2012 International Conference on*. Kumaracoil, India: IEEE, 2012. v. 1.

[8] MARSAN, M. A. et al. *Modelling with Generalized Stochastic Petri Nets*. 1st. ed. New York, NY, USA: John Wiley & Sons, Inc., 1994. v. 1.

[9] SALTELLI, A. et al. *Sensitivity analysis*. 1. ed. Hoboken, USA: John Wiley & Sons, 2000. v. 1. (Wiley Series in Probability and Statistics, v. 1).

[10] SALTELLI, A. et al. *Global sensitivity analysis: the primer*. 1. ed. Hoboken, USA: John Wiley & Sons, 2008. v. 1.

[11] WEBER, T. S. Tolerância a falhas: conceitos e exemplos. *Apostila do PPGC-INF UFGRS*, v. 1, n. 1, p. 1–24, 2003.

[12] BESERRA, D. et al. Utilização de hardware legado para o ensino de cad. *IJCAE*, v. 3, n. 1, p. 17–20, 2014.

[13] MELO, M. et al. Comparative analysis of migration-based rejuvenation schedules on cloud availability. In: LAI, L. L.; YEUNG, D. S. (Ed.). *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*. Manchester, UK: IEEE, 2013. v. 1.

[14] MELO, M. et al. Availability study on cloud computing environments: Live migration as a rejuvenation mechanism. In: PATARICZA, A.; CANDEA, G.; KEMPER, P. (Ed.). *Dependable Systems and Networks (DSN), 2013 43rd Annual IEEE/IFIP International Conference on*. Washington, USA: IEEE, 2013. v. 1.

[15] KIM, D. S.; MACHIDA, F.; TRIVEDI, K. S. Availability modeling and analysis of a virtualized system. In: XU, S.; GAO, J.; XIANG, D. (Ed.). *Dependable Computing, 2009.* Shanghai, China: IEEE, 2009. v. 1.

[16] TRIVEDI, K. S.; BOBBIO, A. *Reliability and Availability Engineering: Modeling, Analysis, and Applications*. 1. ed. Cambridge, UK: Cambridge University Press, 2017. v. 1.

[17] MACIEL, P. et al. Mercury: Performance and dependability evaluation of systems with exponential, expolynomial, and general distributions. In: KIM, D. S.; KITAKAMI, M.; VARADHARAJAN, V. (Ed.). *2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing*. Christchurch, New Zealand: IEEE, 2017. v. 1.

[18] ECKHARDT, D. E. et al. An experimental evaluation of software redundancy as a strategy for improving reliability. *IEEE Trans. Softw. Eng.*, v. 17, n. 7, p. 692–702, 1991.

[19] MELO, C. et al. Capacity-oriented availability model for resources estimation on private cloud infrastructure. In: KIM, D. S.; KITAKAMI, M.; VARADHARAJAN, V. (Ed.). *2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC)*. Christchurch, New Zealand: IEEE, 2017. v. 1.

[20] MATOS, R. d. S. et al. Sensitivity analysis of server virtualized system availability. *IEEE Trans. Rel.*, v. 61, n. 4, p. 994–1006, 2012.

[21] DANTAS, J. et al. Eucalyptus-based private clouds: availability modeling and comparison to the cost of a public cloud. *Computing*, v. 97, n. 11, p. 1121–1140, 2015.

[22] ATAIE, E. et al. Hierarchical stochastic models for performance, availability, and power consumption analysis of iaas clouds. *IEEE TSC*, v. 1, n. 1, p. 1, 2017.

[23] CALHEIROS, R. N. et al. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Softw Pract Exp*, v. 41, n. 1, p. 23–50, 2011.

[24] LIU, B. et al. Model-based sensitivity analysis of iaas cloud availability. *Future Gener Comput Syst.*, v. 83, n. 1, p. 1–13, 2018.

[25] MATOS, R. et al. Redundant eucalyptus private clouds: availability modeling and sensitivity analysis. *J Grid Comput.*, v. 15, n. 1, p. 1–22, 2017.

[26] NURMI, D. et al. The eucalyptus open-source cloud-computing system. In: CCGRID '09. *Cluster Computing and the Grid, 2009.* Washington, USA: IEEE Computer Society, 2009. v. 1.

[27] CHANG, X. et al. Modeling and analysis of high availability techniques in a virtualized system. *Comput J.*, v. 61, n. 2, p. 180–198, 2018.

[28] NGUYEN, T. A.; KIM, D. S.; PARK, J. S. A comprehensive availability modeling and analysis of a virtualized servers system using stochastic reward nets. *Sci. World J*, v. 2014, n. 1, p. 1–1, 2014.

[29] XU, J. et al. Availability modeling and analysis of a single-server virtualized system with rejuvenation. *JSW*, v. 9, n. 1, p. 129–139, 2014.

[30] THEIN, T.; PARK, J. S. Availability analysis of application servers using software rejuvenation and virtualization. *J. Comput. Sci. Tech*, v. 24, n. 2, p. 339–346, 2009.

[31] MACHIDA, F.; KIM, D. S.; TRIVEDI, K. S. Modeling and analysis of software rejuvenation in a server virtualized system with live vm migration. *Perform. Evaluation*, v. 70, n. 3, p. 212–230, 2013.