Cloud Capacitor: Uma Ferramenta de Apoio ao Planejamento da Capacidade de Aplicações na Nuvem

Matheus Cunha, Marcelo Gonçalves, Nabor C. Mendonça, Américo Sampaio

[1]Programa de Pós-Graduação em Informática Aplicada (PPGIA)
Universidade de Fortaleza (UNIFOR)
Av. Washington Soares, 1321, Edson Queiroz, CEP 60811-905 Fortaleza, CE

{mathcunha,marcelocg}@gmail.com,{nabor,americo.sampaio}@unifor.br

Resumo. *Este trabalho apresenta* Cloud Capacitor*, uma ferramenta para apoiar o planejamento da capacidade de aplicações em nuvens que oferecem infraestrutura-como-serviço (IaaS). A ferramenta é baseada em uma nova abordagem para avaliar o desempenho de aplicações na nuvem, denominada* inferência de desempenho*. Essa abordagem tem como premissa a definição de uma relação de capacidade entre diferentes configurações de recursos de um dado provedor de nuvem IaaS, a partir da qual é possível prever (ou "inferir") o desempenho esperado de uma aplicação para uma ampla variedade de cenários de implantação, sendo que apenas uma pequena parte desses cenários precisa de fato ser implantada e executada na nuvem. A utilização da ferramenta é ilustrada através de um exemplo envolvendo a avaliação da aplicação de* blogging *WordPress no provedor de nuvem Amazon EC2.*

Abstract. *This work presents* Cloud Capacitor*, a tool to support application capacity planning in infrastructure-as-a-service (IaaS) clouds. The tool is based on a novel performance evaluation approach for cloud applications, named* performance inference*. This approach relies on the definition of a capacity relation between different resource configurations offered by a given IaaS cloud provider, enabling one to predict (or "infer") an application's expected performance for a large variety of deployment scenarios, with the benefit that only a small fraction of those scenarios need to be effectively deployed and executed in the cloud. The use of the tool is illustrated through an example evaluation of the WordPress blogging application in the Amazon EC2 cloud.*

1. Introdução

Um dos principais desafios enfrentados pelos usuários de nuvens que oferecem infraestrutura-como-serviço (IaaS) é planejar adequadamente a capacidade dos recursos da nuvem necessários para atender as demandas específicas de suas aplicações [Menascé and Ngo 2009]. Tipicamente, esse problema de planejamento tem sido atacado em duas frentes: através do uso de ferramentas de predição de desempenho na nuvem (por exemplo, [Li et al. 2011, Fittkau et al. 2012, Jung et al. 2013]), ou empiricamente, medindo-se o desempenho real da aplicação na nuvem quando implantada sob diferentes configurações de recursos e submetida a diferentes níveis de carga de trabalho — o que normalmente é feito com o auxílio de ferramentas automatizadas para avaliação de desempenho na nuvem (por exemplo, [Jayasinghe et al. 2012,

Silva et al. 2013, Cunha et al. 2013a]. Por executarem a aplicação no próprio ambiente de nuvem, ferramentas de avaliação de desempenho conseguem resultados muito mais precisos no que diz respeito à seleção das melhores configurações de recursos para demandas específicas. No entanto, uma limitação importante desses trabalhos é a necessidade de se testar exaustivamente a aplicação alvo sob uma grande quantidade de configurações de recursos e de cargas de trabalho, implicando em altos tempo e custo durante a fase de planejamento.

Este trabalho apresenta a ferramenta *Cloud Capacitor*, a qual faz uso de *inferência de desempenho* [Gonçalves et al. 2015a, Gonçalves et al. 2015b] para reduzir o número de configurações de recursos e de cargas de trabalho que precisam ser efetivamente avaliadas na nuvem, reduzindo, assim, o tempo e o custo da fase de planejamento. A abordagem de inferência de desempenho utilizada tem como premissa a definição de uma relação de capacidade entre diferentes configurações de recursos de um dado provedor de nuvem IaaS, a partir da qual é possível prever (ou "inferir"), com alta precisão, o desempenho esperado de uma aplicação para uma ampla variedade de cenários de implantação, sendo que apenas uma pequena parte desses cenários precisa de fato ser implantada e executada na nuvem [Gonçalves et al. 2015a, Gonçalves et al. 2015b].

A próxima seção descreve o projeto e a implementação da ferramenta *Cloud Capacitor* na forma de uma biblioteca para criação de sistemas de apoio ao planejamento de capacidade na nuvem. A Seção 3 apresenta um desses sistemas, *Capacito,Web*, que disponibiliza uma interface web para acesso aos serviços oferecidos pela biblioteca *Cloud Capacitor*. A Seção 4 compara a ferramenta proposta com outros trabalhos relacionados. Por fim, a Seção 5 oferece as conclusões e sugestões para trabalhos futuros.

2. *Cloud Capacitor*

Cloud Capacitor é uma biblioteca para criação de sistemas de avaliação e inferência de desempenho de aplicações em ambientes de nuvem IaaS, implementada como um *gem* (pacote) da linguagem Ruby. A documentação e o código fonte da biblioteca estão disponíveis em `https://github.com/marcelocg/cloud_capacitor`.

2.1. Classes e Responsabilidades

A biblioteca é composta por um conjunto de classes que representam entidades relevantes no domínio da avaliação da capacidade de aplicações em ambientes de nuvem (ver Figura 1). A classe principal, *Capacitor*, implementa o fluxo básico do processo de avaliação e inferência de desempenho, com todos os seus pontos de decisão e de extensão. Maiores detalhes sobre o funcionamento do processo podem ser obtidos em [Gonçalves et al. 2015a].

Entre os pontos de extensão, destaca-se o uso da classe *Strategy*, que pode ser especializada para a criação de diferentes estratégias de avaliação (por exemplo, estratégias com comportamento otimista, pessimista ou conservador em relação à escolha das configurações de recursos e das cargas de trabalho a serem avaliadas [Gonçalves et al. 2015a]). Outro ponto de extensão oferecido pela biblioteca é a classe *DefaultExecutor*, que pode ser especializada para utilizar diferentes ferramentas de implantação e execução de aplicações na nuvem (na Figura 1, a classe *DefaultExecutor* é especializada pela classe *RealExecutor*).

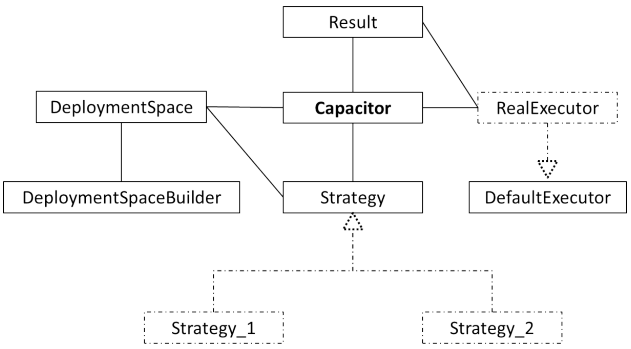


Figura 1. Diagrama de classes da biblioteca *Cloud Capacitor*.

Para que o processo de avaliação de capacidade possa ser efetuado, a classe *Capacitor* deve conhecer o resultado de cada execução da aplicação alvo, de modo que possa tomar as decisões corretas na indicação das configurações que atenderiam ou não os seus requisitos de desempenho, também conhecidos como *Service Level Objectives* (SLOs). No contexto deste trabalho, essas configuração são rotuladas como candidatas e rejeitadas, respectivamente. Os resultados de cada cenário de implantação avaliado são encapsulados na classe *Result*, cujos atributos são fornecidos pela subclasse responsável pela execução dos testes de desempenho da aplicação alvo na nuvem.

Finalmente, durante a execução do processo, a classe *Capacitor* precisa ter conhecimento das configurações de recursos disponibilizados pelo provedor de nuvem nas quais a aplicação alvo será implantada e avaliada, aqui referenciadas como *espaço de implantação*. Essa é a responsabilidade da classe *DeploymentSpace*, que implementa uma estrutura de dados em memória para representar os diversos níveis de capacidade definidos entre as configurações de recursos selecionadas para avaliação. A construção dessa estrutura é responsabilidade da classe *DeploymentSpaceBuilder*, que contém os algoritmos necessários à geração do grafo direcionado usado para navegação pelos níveis de capacidade definidos entre as configurações do espaço de implantação.

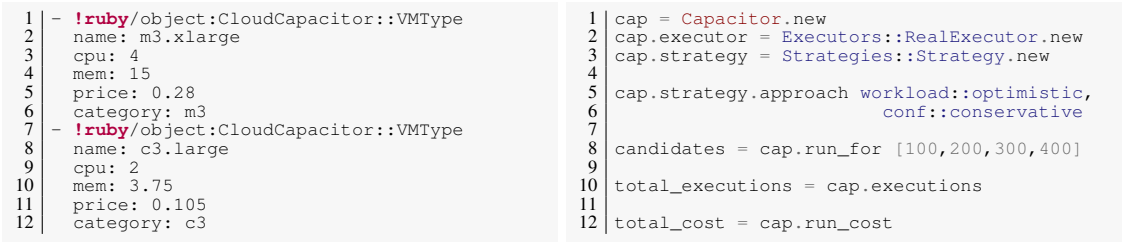
2.2. Fluxo de Utilização

A utilização da biblioteca na criação de um sistema de avaliação de capacidade deve seguir os seguintes passos:

1. Configurar os parâmetros de criação do espaço de implantação.
2. Identificar os tipos de máquinas virtuais do provedor de nuvem que irão compor o espaço de implantação.
3. Instanciar um objeto *Capacitor*.
4. Atribuir ao *Capacitor* um objeto *DefaultExecutor*.
5. Atribuir ao *Capacitor* um objeto *Strategy*.
6. Executar o método *run_for* do *Capacitor*.

A configuração do espaço de implantação é feita através de um arquivo em formato YAML[1] e inclui apenas dois parâmetros: (i) o custo máximo permitido para cada configuração (calculado pela soma dos custos de cada instância que compõe a

[1] `http://www.yaml.org/`



```
- !ruby/object:CloudCapacitor::VMType
  name: m3.xlarge
  cpu: 4
  mem: 15
  price: 0.28
  category: m3
- !ruby/object:CloudCapacitor::VMType
  name: c3.large
  cpu: 2
  mem: 3.75
  price: 0.105
  category: c3
```

(a)

```
1  cap = Capacitor.new
2  cap.executor = Executors::RealExecutor.new
3  cap.strategy = Strategies::Strategy.new
4
5  cap.strategy.approach workload::optimistic,
6                        conf::conservative
7
8  candidates = cap.run_for [100,200,300,400]
9
10 total_executions = cap.executions
11
12 total_cost = cap.run_cost
```

(b)

Figura 2. Exemplos de uso da biblioteca *Cloud Capacitor*: (a) especificação de tipos de máquinas virtuais; (b) código para execução do processo de avaliação.

configuração); e (ii) o número máximo de instâncias permitidas para cada configuração. Por exemplo, se os dois parâmetros acima foram configurados com os valores $7,00$ e 4, respectivamente, então serão criadas configurações com 1, 2, 3 e 4 instâncias para cada tipo de máquina virtual especificada, desde que o custo total da configuração não ultrapasse o valor de $7,00$ unidades monetárias.

O próximo passo na utilização da biblioteca é especificar os tipos de máquinas virtuais a serem utilizados na geração do espaço de implantação. Essa especificação é feita em outro arquivo em formato YAML, discriminando as características de CPU, memória e custo de cada tipo de máquina virtual. Essas informações serão utilizadas pela classe *DeploymentSpaceBuilder* para a criação das configurações de recursos que comporão o espaço de implantação, atendendo às restrições de tamanho e custo impostas no passo anterior, bem como para a definição de relações de capacidade entre elas. A Figura 2(a) mostra um exemplo da especificação em YMAL de dois tipos de máquinas virtuais (quais sejam, *m3.xlarge* e *c3.large*) oferecidos pelo provedor Amazon EC2.[2]

Definidas as configurações necessárias, o desenvolvedor pode, assim, criar um objeto a partir da classe *Capacitor* e, então, atribuir a ele um objeto instanciado a partir de uma subclasse de *DefaultExecutor*, cuja implementação deve fornecer os meios necessários para a implantação e execução dos testes de desempenho da aplicação alvo.

A Figura 2(b) mostra um exemplo de código em Ruby responsável pela criação e execução de um sistema de avaliação de capacidade implementado a partir da biblioteca *Cloud Capacitor*. Na linha 1, vê-se a instanciação do *Capacitor*. Na linha 2, um objeto da classe *RealExecutor* é atribuído ao *Capacitor*. A partir da linha 3, vêem-se os passos seguintes da utilização da biblioteca, com a definição de uma estratégia de avaliação, configurada com uma heurística de comportamento optimista para seleção de cargas de trabalho e conservador para seleção de configurações. A execução do processo de avaliação de capacidade acontece de fato a partir da chamada realizada na linha 8, onde são passados valores que representam uma lista de grandezas de demandas que serão impostas à aplicação alvo quando implantada e executa utilizando as configurações de recursos geradas como parte da construção do espaço de implantação.

Ao final da execução, o processo retorna a lista de configurações candidatas para cada valor de demanda passado como parâmetro. Adicionalmente, o desenvolvedor pode solicitar alguns dados a respeito da própria avaliação, como o número total de execuções da aplicação alvo na nuvem (linha 10) e o custo total dessas execuções (linha 12).

[2] `http://aws.amazon.com/ec2`

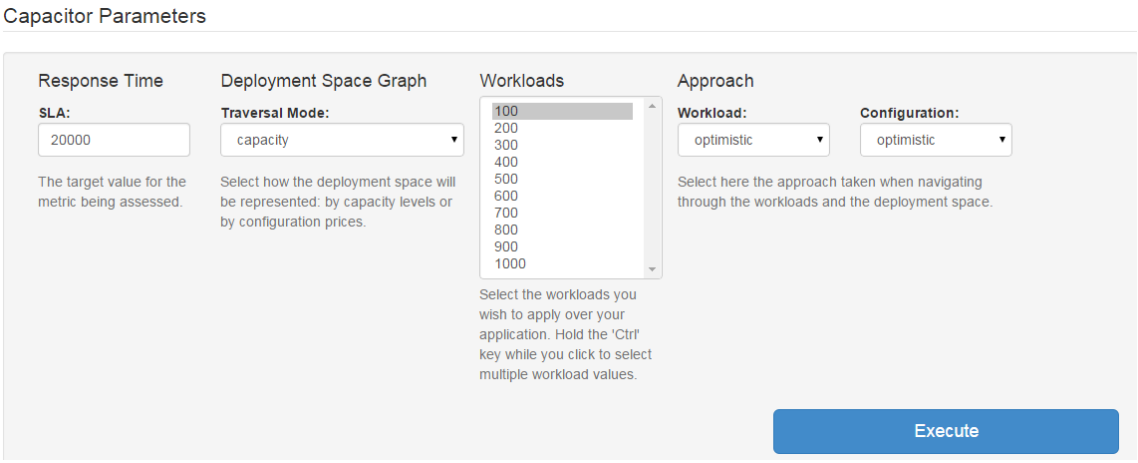


Figura 3. Página inicial do *Capacitor Web*.

3. *Capacitor Web*

O *Capacitor Web* é uma aplicação web implementada utilizando o arcabouço *Ruby on Rails*[3] e que faz uso das funcionalidades oferecidas pela biblioteca *Cloud Capacitor* para a realização de testes de avaliação e inferência de desempenho em ambientes de nuvem. Essa aplicação foi utilizada para apoiar a condução de diversos experimentos de avaliação de capacidade envolvendo a aplicação de *blogging* WordPress[4] na nuvem Amazon EC2. Uma versão do *Capacitor Web*, pré-configurada com os resultados desses experimentos, está disponível em `https://capacitor-web.herokuapp.com/`.

A Figura 3 mostra a tela inicial da aplicação, onde o usuário tem a oportunidade de parametrizar a execução do processo de avaliação. O primeiro campo permite a especificação do SLO que deve ser respeitado pela aplicação para que uma execução seja considerada bem sucedida. Em seguida, apresenta-se o campo para seleção da representação das relações de capacidade entre as configurações do espaço de implantação, com opções de representação por capacidade (tipo de máquina virtual) ou por preço. Mais ao centro da tela, encontra-se uma lista de valores de carga, *Workloads*, essa lista permite que seja selecionada a demanda que será imposta à aplicação alvo e representa o número de usuários simultâneos utilizados nas execuções. Os últimos dois campos apresentam as opções a serem usadas na seleção da demanda e dos níveis de capacidade, respectivamente. As opções aqui são *Optimistic*, *Conservative* e *Pessimistic* para ambos os campos. A combinação desses dois parâmetros constitui a estratégia de avaliação que será utilizada pelo processo durante a exploração do espaço de implantação. Em linhas gerais, uma estratégia otimista seleciona os maiores níveis de demanda e os menores níveis de capacidade; já uma estratégia conservadora seleciona níveis intermediários tanto para a demanda quanto para a capacidade; por fim, uma estratégia pessimista seleciona os menores níveis de demanda e os maiores níveis de capacidade. A escolha da estratégia de avaliação é um ponto crucial da configuração do processo de planejamento de capacidade, uma vez que ela impacta diretamente na quantidade de cenários de implantação que serão efetivamente executados na nuvem [Gonçalves et al. 2015a].

[3] `http://rubyonrails.org/`
[4] `https://wordpress.org/`

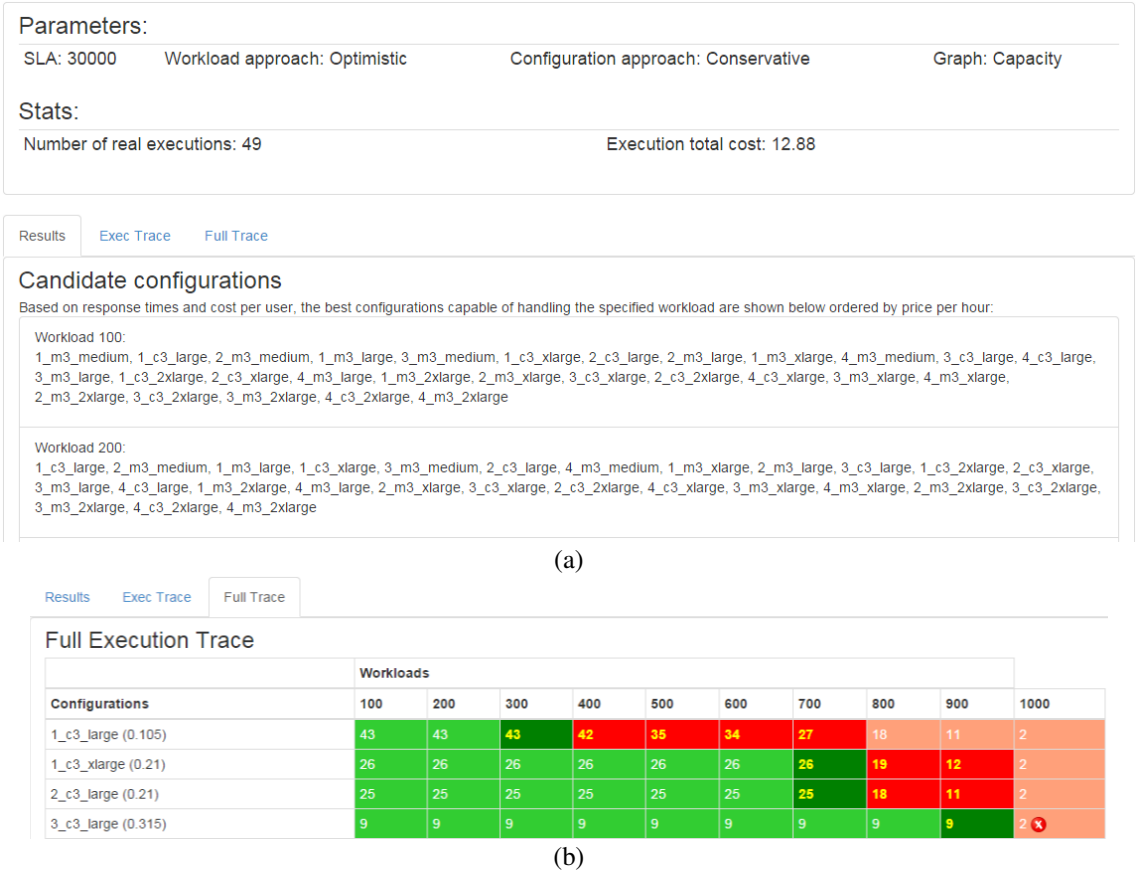


Figura 4. Página de resultados do *Capacitor Web*: (a) dados de execução e configurações candidatas; (b) rastro com os resultados das avaliações.

Após a configuração dos parâmetros da página inicial, o usuário da aplicação pode solicitar a execução do processo acionando o botão "Execute", no canto inferior direito dessa mesma página. A Figura 4 mostra a página de resultados da aplicação, considerando dados obtidos a partir dos experimentos realizados com o WordPress na nuvem Amazon EC2. Essa página se divide em duas grandes áreas: uma superior, fixa, e uma inferior, que, por sua vez, se subdivide em três partes, separadas em abas distintas.

A parte superior mostra os dados passados como parâmetros pelo usuário na tela inicial, para facilitar a análise dos resultados e sua comparação futura. Assim, são exibidos o valor do SLO, as estratégias para seleção da demanda e dos níveis de capacidade das configurações, e a opção selecionada para a geração do espaço de implantação. Logo abaixo, são exibidos o número de execuções reais realizadas no ambiente de nuvem e o custo total (em US$ por hora) dessas execuções, com base no preço estabelecido pelo provedor para cada uma das configurações efetivamente utilizadas nos testes de desempenho da aplicação.

As três abas da parte inferior ativam a visualização das informações complementares resultantes da execução do processo. A primeira aba, com o título "Results", exibe a lista de configurações candidatas identificadas para cada nível de demanda avaliado (ver Figura 4(a)), conforme retornado pelo método *run_for* da classe *Capacitor*.

A segunda aba, com o título "Exec Trace", exibe os dados de rastreamento da

execução do processo. Esses dados permitem visualizar a sequência de execuções da aplicação alvo na nuvem, informando, para cada execução, o valor da demanda e a configuração de recursos utilizados.

Por fim, a terceira aba, com o título "Full Trace", exibe uma tabela cujas colunas e linhas representam, respectivamente, os diferentes valores de demanda e as diferentes configurações de recursos avaliados (ver Figura 4(b)). Dessa forma, cada célula da tabela representa um possível cenário de implantação da aplicação na nuvem, sendo que o número informado na célula indica a ordem em que aquele cenário foi avaliado e a sua cor e tonalidade indicam, respectivamente, o resultado dessa avaliação e o meio através do qual esse resultado foi obtido. Por exemplo, células verdes (vermelhas) indicam que a aplicação atingiu (não atingiu) o SLO desejado, enquanto cores em tons claros (escuros) indicam que os resultados foram obtidos por meio de inferência de desempenho ou de execuções reais da aplicação na nuvem. Dessa forma, uma maior presença de células em tons claros nessa aba indica uma maior redução no número de execuções da aplicação, implicando, portanto, em menores tempo e custo durante o processo de planejamento. Ainda nessa aba, uma célula contendo um círculo vermelho com a letra "x" indica um erro de inferência, ou seja, um resultado obtido via inferência de desempenho mas que não se confirmou na prática. Mais detalhes sobre a acurácia do processo de inferência de desempenho podem ser obtidos em [Gonçalves et al. 2015a].

4. Trabalhos Relacionados

As soluções existentes para avaliar a capacidade de aplicações em nuvens IaaS oferecem alta precisão, quando são baseadas em dados de desempenho obtidos diretamente da execução da aplicação alvo no provedor, como é caso das ferramentas *Expertus* [Jayasinghe et al. 2012], *CloudBench* [Silva et al. 2013] e *Cloud Crawler* [Cunha et al. 2013a, Cunha et al. 2013b]. Além disso, essas soluções oferecem grande flexibilidade de uso, no sentido em que permitem aos usuários avaliar diferentes combinações de componentes da aplicação sob as mais variadas configurações de recursos e demandas. O ponto negativo é a necessidade de executar cada uma das configurações definidas pelo usuário, uma vez que essas ferramentas não oferecem nenhum mecanismo voltado para reduzir a quantidade de execuções da aplicação. Dessa forma, cabe exclusivamente aos usuários dessas soluções definirem as melhores estratégias de explorar o espaço de implantação da aplicação na nuvem.

A ferramenta *Cloud Capacitor*, descrita neste trabalho, oferece uma alternativa mais eficiente às soluções de avaliação de capacidade de aplicações existentes, uma vez que reduz o número total de cenários sob os quais é necessário implantar e executar a aplicação na nuvem. Isso é possível com a utilização da abordagem de inferência de desempenho, também proposta pelos autores deste trabalho [Gonçalves et al. 2015a], a qual permite inferir, com alta precisão, o desempenho esperado de uma aplicação para uma ampla variedade de cenários de implantação, a partir da definição de relações de capacidade entre diferentes configurações de recursos de um dado provedor de nuvem.

5. Conclusão e Trabalhos Futuros

A tarefa de escolher adequadamente os recursos computacionais (em particular, máquinas virtuais) de um provedor de nuvem, de forma a minimizar os custos necessários para

atender diferentes níveis de demanda de uma aplicação, é um desafio importante para o qual ainda não existem soluções plenamente satisfatórias disponíveis. Este trabalho apresentou uma nova ferramenta de apoio ao processo de avaliação de capacidade de aplicações na nuvem, que tem como principal diferencial a utilização de uma abordagem de inferência de desempenho para reduzir o tempo e o custo tipicamente associados com esse tipo de atividade.

Com relação aos trabalhos futuros, algumas possibilidades interessantes para melhoria ou extensão da ferramenta proposta incluem: investigar novos critérios para a geração do espaço de implantação da aplicação, por exemplo, utilizando informações sobre o preço das configurações como fonte para a definição de suas relações de capacidade; implementar novas estratégias de avaliação, por exemplo, considerando informações sobre a utilização dos recursos da nuvem pela aplicação, como consumo de CPU e memória, diante da escolha dos níveis de capacidade e de demanda a serem avaliados; e realizar novos experimentos com outras aplicações e provedores de nuvem, visando não apenas validar a ferramenta mas também melhorá-la como produto.

Referências

Cunha, M. et al. (2013a). A Declarative Environment for Automatic Performance Evaluation in IaaS Clouds. In *IEEE CLOUD 2013*, pages 285–292.

Cunha, M. et al. (2013b). Cloud Crawler: Um Ambiente Programável para Avaliar o Desempenho de Aplicações em Nuvens de Infraestrutura. In *SBRC 2013*.

Fittkau, F. et al. (2012). CDOSim: Simulating cloud deployment options for software migration support. In *IEEE MESOCA 2012*, pages 37–46.

Gonçalves, M. et al. (2015a). Inferência de Desempenho: Uma Nova Abordagem para Planejar a Capacidade de Aplicações na Nuvem. In *SBRC 2015*. Prêmio de Melhor Artigo!

Gonçalves, M. et al. (2015b). Performance Inference: A Novel Approach for Planning the Capacity of IaaS Cloud Applications. In *Proc. of IEEE CLOUD 2015*, pages 813–820.

Jayasinghe, D. et al. (2012). Expertus: A Generator Approach to Automate Performance Testing in IaaS Clouds. In *IEEE CLOUD 2012*, pages 73–80.

Jung, G. et al. (2013). CloudAdvisor: A Recommendation-as-a-Service Platform for Cloud Configuration and Pricing. In *IEEE SERVICES 2013*, pages 456–463.

Li, A. et al. (2011). CloudProphet: Towards Application Performance Prediction in Cloud. In *ACM SIGCOMM 2011*, pages 426–427.

Menascé, D. A. and Ngo, P. (2009). Understanding Cloud Computing: Experimentation and Capacity Planning. In *CMG 2009*.

Silva, M. et al. (2013). CloudBench: Experiment Automation for Cloud Environments. In *IEEE IC2E 2013*, pages 302–311.