CostPlanner: Planejamento de Capacidade de Longo Prazo para Clientes de Provedores IaaS

Caio Galvão[1], **Thiago Emmanuel Pereira**[1], **Francisco Vilar Brasileiro**[1]
Gabriel Gomes[1,2]

[1]Universidade Federal de Campina Grande (UFCG)
Caixa Postal 10.106 – 58.109-970 – Campina Grande – PB – Brazil

[2]VTEX

caio.galvao@ccc.ufcg.edu.br

{temmanuel,fubica}@computacao.ufcg.edu.br

gabriel.cardoso@vtex.com

Abstract. *Cloud provisioning tools currently available help dealing with the trade-offs related to prices, Service Level Agreements (SLAs), and client commitment regarding the provisioning plans offered by cloud providers. Unfortunately, these tools have limitations. In particular, they cannot model the newer plans available in the market. In this paper, we describe a tool that can represent the long-term provisioning plans for the all major cloud providers. We evaluated our tool using data from an industry partner, showing promising results. For example, an allocation strategy that uses a spend-based provisioning (following AWS SavingsPlan) yields a 10% cost reduction in comparison with an strategy that uses the older resource-based commitment (following AWS Reserve).*

Resumo. *Ferramentas de planejamento de capacidade atualmente disponíveis ajudam a ponderar as diferenças entre os planos de provisionamento ofertados por provedores de nuvem. Apesar de sua utilidade, essas ferramentas são limitadas. Em particular, elas não levam em consideração todos os planos ofertados no mercado. Neste artigo, descrevemos uma ferramenta que permite representar os planos de todos os principais provedores de nuvem. Nossa avaliação da ferramenta usando dados de um parceiro da indústria aponta resultados promissores. Por exemplo, considerando os planos mais novos disponíveis, como AWS Savings Plans, a ferramenta permitiu uma redução de custos de* 10% *em comparação com o uso restrito aos planos mais antigos, tais como AWS Reserve.*

1. Introdução

O surgimento da computação na nuvem possibilitou a compra de recursos computacionais sob demanda, de maneira flexível. É uma grande evolução em relação ao modelo antigo, em que era preciso estimar o pico da demanda por recursos com antecedência, para então serem compradas máquinas. Havia o risco de se comprar recursos em excesso, gerando custos adicionais, ou de se comprar menos recursos que o necessário, com consequências mais sérias. De uma forma geral, os provedores de computação na nuvem que ofertam o modelo IaaS (do inglês, *Infrastructure as a Service*) permitem que seus clientes possam rapidamente provisionar e liberar recursos computacionais, pagando apenas pelo que efetivamente foi usado.

Os provedores de IaaS, entretanto, continuam tendo que fazer provisionamento a longo prazo, já que operam a infraestrutura física e precisam garantir que os clientes sejam atendidos. Para reduzir os riscos associados com essa atividade, eles adotam um conjunto de estratégias para facilitar sua própria alocação de recursos, como a definição de um teto de uso para os clientes [Costa et al. 2013]. Uma outra estratégia importante é a definição de planos para a compra de recursos. Os planos apresentam diferentes *trade-offs* entre preço, tempo de compromisso, disponibilidade (*availability*) e estocagem (*obtainability*) dos recursos [Carvalho et al. 2014].

Os planos mais simples são do tipo sob demanda, em que, geralmente, as instâncias são compradas por um período de uma hora, ou mesmo minutos, e o de reservas, em que as instâncias são compradas por períodos de tempos longos, a partir de um ano, sem possibilidade de cancelamento. Ainda existem mercados que utilizam recursos ociosos para oferecer grandes descontos, porém não existe a garantia de disponibilidade, ou seja, os clientes podem perder os recursos a qualquer momento.

Ao longo do tempo, surgiram variações nos planos de reserva para adicionar flexibilidade, em troca de um custo mais elevado. Em particular, nos últimos anos surgiram os mercados de *spend-based commitments*, chamados de *savings plans* na AWS e na Azure e de *committed use discounts* na Google. Nesse tipo de mercado, são feitas reservas de valores monetários, ao invés de quantidade de recursos. Os valores reservados são usados para comprar instâncias de um mesmo grupo, de modo intercambiável.

A existência dos diferentes mercados dificulta o trabalho do profissional responsável por fazer a alocação de recursos na nuvem, já que são várias opções com *trade-offs* diferentes. Para auxiliar neste problema, existem vários tipos de ferramentas, que são adequadas para contextos diferentes. Entras elas, existem os otimizadores, que encontram a alocação com o menor custo, dada uma demanda conhecida.

Embora existam ferramentas no mercado para esse problema, com diversas funcionalidades, não encontramos serviços que implementassem otimizadores. Por outro lado, existem várias sugestões de otimizadores na literatura, mas eles não consideram os mercados de *spend-based commitments*. Este trabalho descreve um projeto conjunto com a empresa VTEX[1] que tenta resolver essas limitações, com alvo na plataforma de comércio eletrônico da VTEX. Para isso, projetamos e desenvolvemos uma ferramenta para auxiliar na alocação de recursos para nuvem, o CostPlanner. Essa ferramenta incorpora um otimizador para os mercados de *spend-based commitments*. O CostPlanner também inclui modos de execução com heurísticas e pode ser facilmente estendido para suportar novas estratégias de alocação.

Acreditamos que os problemas enfrentados com esse projeto são particularmente relevantes para o contexto brasileiro, uma vez que as empresas de TI do país que usam serviços de provedores de nuvem gastam em moeda estrangeira (o que aumenta a incerteza do negócio devido às flutuações da moeda). Ainda, o montante dos gastos com provedores é um custo elevado, podendo chegar a alguns milhões de dólares por mês, dependendo do porte da empresa.

O restante do documento é dividido da seguinte forma. Na Seção 2, descrevemos ferramentas relacionadas disponíveis no mercado, bem como discutimos limitações de

[1] https://vtex.com

modelos de otimização disponíveis na literatura. Na Seção 3, apresentamos uma visão geral do projeto da ferramenta e o nosso processo de avaliação. Por fim, na Seção 4, discutimos perspectivas de continuação do trabalho conjunto com a empresa parceira.

2. Trabalhos relacionados

Existem muitas ferramentas voltadas para planejamento de capacidade na nuvem, com funcionalidades variadas. A funcionalidade mais comum é o monitoramento da infraestrutura para a visualização dos custos, presente em ferramentas como Vantage.sh[2], Spot.io[3], Kubecost[4], Open Cost[5] e Datadog[6]. Algumas ferramentas, como Vantage.sh, Kubecost, e Sedai.io também oferecerem recomendações para diminuir os custos da infraestrutura. Essas recomendações podem incluir identificação de recursos ociosos, *right-sizing* de instâncias ou containers e compra de instâncias nos mercados de reserva ou *spend-based commitments*.

Existem ainda ferramentas que operam automaticamente a infraestrutura para diminuir os custos. O Autopilot do Vantage.sh compra e vende reservas na AWS e o Eco do Spot.io compra, vende e troca reservas ou *spend-based commitments*. Todas essas ferramentas utilizam modelos que não são públicos, nem existem mais detalhes sobre as implementações e testes de desempenho. Na experiência do nosso parceiro na indústria com essas ferramentas, os resultados divergem bastante do esperado, sem maiores explicações por parte das empresas que desenvolveram os serviços.

Na perspectiva acadêmica, existem várias implementações de otimizadores na literatura. Hu *et al.* [Hu et al. 2015] e Jin *et al.* [Jin et al. 2018] propõem otimizadores em programação dinâmica para compra de instâncias nos mercados sob demanda e de reservas. Kim e Jo [Kim and Jo 2017] também sugerem um otimizador para os mesmos mercados, mas implementado em programação linear. Ainda existem implementações com programação estocástica que consideram a incerteza na demanda por recursos [Chaisiri et al. 2012, Mireslami et al. 2021]. Porém, essas implementações não consideram os novos mercados de *spend-based commitments*, que foram adotados nos últimos anos pelos grandes provedores. Elas também não estão integradas em ferramentas voltadas para o uso dos profissionais que fazem a alocação.

3. CostPlanner

O CostPlanner é uma ferramenta de apoio à decisão para profissionais que atuam no processo de planejamento de capacidade de recursos computacionais adquiridos de provedores IaaS. A versão atual implementa tanto estratégias ótimas, quanto heurísticas, e pode ser estendida para incorporar novas estratégias. Esta seção irá descrever a arquitetura do sistema.

3.1. Arquitetura

O CostPlanner foi implementado como uma aplicação web, com o *front-end* desenvolvido em NodeJS e o *back-end* em Python. A arquitetura do *back-end* é apresentada na

[2]https://docs.vantage.sh/cost_reports

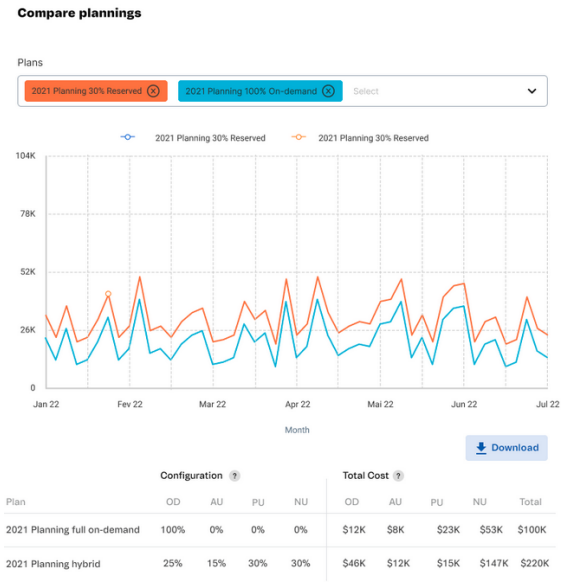
[3]https://spot.io/product/cost-intelligence/

[4]https://docs.kubecost.com/using-kubecost/navigating-the-kubecost-ui/cloud-costs-explorer

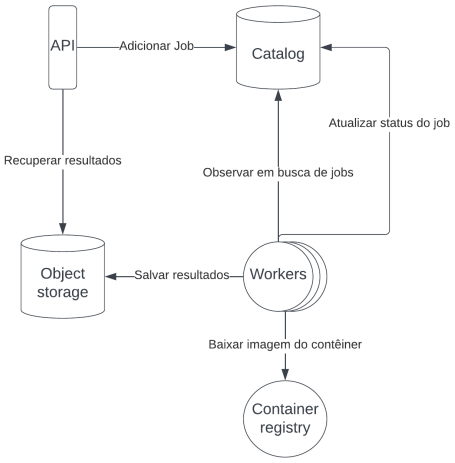
[5]https://github.com/opencost/opencost

[6]https://www.datadoghq.com/product/cloud-cost-management/

Figura 1b. O usuário interage com o sistema no *front-end* (Figura 1a), que se conecta a uma API no *back-end*. Quando o *front-end* requisita uma execução, seja da heurística ou do otimizador, o *back-end* adiciona um novo *job* em um banco de dados, chamado de *catalog*. Os *jobs* podem estar em três estados: *ready*, *running* e *finished*.



(a) Interface web do CostPlanner.



(b) Arquitetura do back-end da ferramenta CostPlanner.

No *back-end* existe um conjunto de *workers* que têm o papel de executar os *jobs*. Os *workers* observam regularmente o *catalog* em buscas de *jobs* que estejam disponíveis (*ready*) e executam o *job* que esteja há mais tempo no *catalog*. Para executar um *job*, o *worker* cria e executa um contêiner a partir de uma imagem que implementa o algoritmo de um determinado tipo de execução (ex. modos diferentes de otimização e heurísticas). O estado do *job* no *catalog* também é atualizado para *running*. Quando a execução termina, o *worker* atualiza o estado do job no *catalog* para *finished* e salva os resultados em *Object Storage* (ex. AWS S3). Esses resultados são recuperados pelo *back-end* quando requisitados.

Essa arquitetura permite que novos modos de execução sejam facilmente adicionados; para isso, é necessário apenas adicionar novas imagens que respeitem o contrato de execução (usando os dados de entrada e gerando dados de saída conforme o padrão definido pela ferramenta).

3.2. Otimizador

O otimizador é um ferramenta separada do CostPlanner e está disponível publicamente [7]. Ele é baseado em um modelo de programação linear desenvolvido como uma extensão e melhoria de um modelo já existente [Kim and Jo 2017], que considera além dos mercados sob demanda e de reservas, o mercado de *spend-based commitments*. A função objetivo do modelo de programação linear é minimizar o custo total com a infraestrutura durante todo o período de tempo, recebendo como variáveis de entrada a demanda por recursos em cada intervalo de tempo e os preços dos mercados para os recursos. Os preços considerados são os vigentes no início do período a ser otimizado.

[7]Otimizador disponível em: https://github.com/ufcg-lsd/AWSome-Savings

Para a sua implementação, foi utilizada a linguagem C++ e, para a otimização, a biblioteca OR-Tools. Esta biblioteca dá suporte a vários *solvers* de programação linear diferentes, permitindo que eles sejam trocados com muita facilidade. Atualmente, está sendo utilizado o *solver* GLOP.

3.3. Avaliação

Para avaliar o método de otimização desenvolvido para o CostPlanner consideramos um estudo de caso com a demanda por máquinas virtuais do nosso parceiro na indústria. Esse conjunto de dados abrange três anos, de 2020 até 2022, e inclui centenas de tipos de instância diferentes. Os dados coletados incluem: a quantidade de instâncias demandadas, o tipo de cada instância demandada bem como o tipo de plano que foi usado para compra.

A Figura 2 mostra os resultados do planejamento obtidos com o CostPlanner, para a demanda de três anos do dataset que consideramos. Consideramos três estratégias: 1) Tudo *OnDemand*, uma estratégia de referência que aloca toda a demanda usando o plano *OnDemand* da AWS; 2) Ótimo com reservas, uma estratégia que pode misturar recursos dos planos *OnDemand* e *Reserve* da AWS; e, finalmente, 3) Ótimo com *Spend-based* que podem misturar recursos de todos os planos da AWS. Como esperado, o modo de alocação de referência, *All OnDemand*, apresenta resultados piores (maior gasto) do que os dois modos otimizados. É importante notar que, para a demanda que consideramos, o modo *Spend-based* permite uma economia significativa (10%) em comparação com o modelo que considera o plano de *Reserve* (o estado-da-arte nos modelos de otimização).

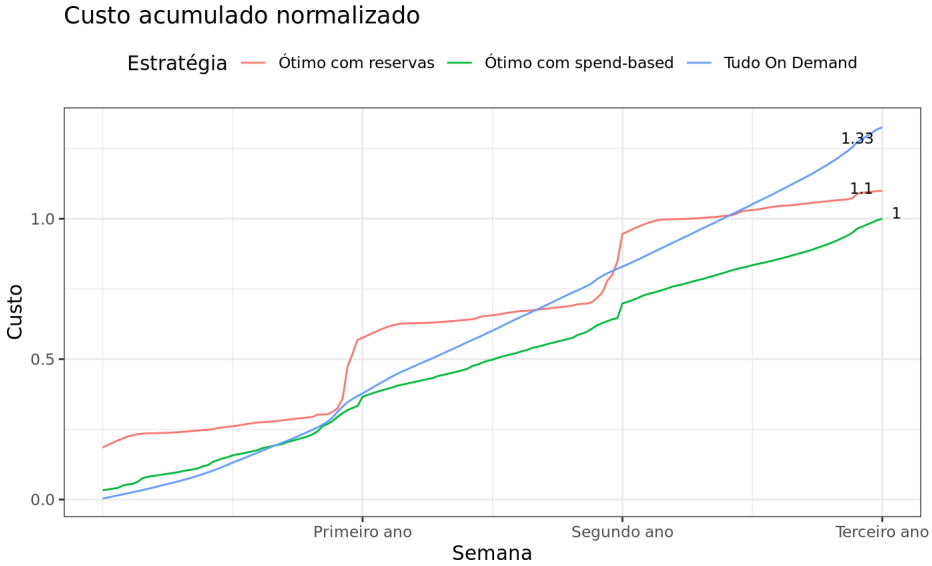


Figura 2. Custo total normalizado para três estratégias de alocação em comparação com a estratégia desenvolvidade de *Spend-based*.

4. Conclusões

Este artigo apresentou uma ferramenta de planejamento de longo prazo para alocação de recursos na nuvem, fruto de uma parceria entre a UFCG e a VTEX. Embora a ferramenta descrita já seja efetiva na sugestão de planos de alocação de recursos, há alguns desafios que consideramos importantes de serem enfrentados no futuro. O primeiro deles diz

respeito ao apoio que precisa ser dado ao tomador de decisão, em face às incertezas de estimativa da demanda; a ferramenta é capaz de encontrar alocações ótimas para uma determinada demanda futura, mas essa demanda é uma estimativa com um erro associado. É importante para o tomador de decisão ter algum suporte para estimar o impacto do erro das estimativas.

Também é importante expandir os modos de execução de heurísticas além dos obtidos com o otimizador. Por exemplo, uma determinada solução ótima pode envolver um número muito grande de decisões de compra (o que pode onerar a operação da empresa). Pode ser interessante ter resultados que não sejam ótimos mas que sejam mais simples de operar.

Por fim, é importante melhorar o tempo de execução do CostPlanner. Atualmente, um planejamento de três anos futuros toma algumas horas para ser executado. Embora o custo de execução da ferramenta seja marginal em relação à economia obtida, soluções mais rápidas podem permitir estudos mais detalhados de planejamento, seguindo estratégias *what-if*.

5. Agradecimentos

Este trabalho foi financiado pelo MCTIC/CNPq-FAPESQ/PB (EDITAL Nº 010/2021) e pela VTEX BRASIL (EMBRAPII PCEE1911.0140).

Referências

Carvalho, M., Cirne, W., Brasileiro, F. V., and Wilkes, J. (2014). Long-term slos for reclaimed cloud computing resources. In *Proceedings of the ACM Symposium on Cloud Computing, November 3-5, 2014*, pages 20:1–20:13. ACM.

Chaisiri, S., Lee, B., and Niyato, D. (2012). Optimization of resource provisioning cost in cloud computing. *IEEE Trans. Serv. Comput.*, 5(2):164–177.

Costa, R., Brasileiro, F. V., de Souza Filho, G. L., and Sousa, D. M. (2013). Analyzing the impact of elasticity on the profit of cloud computing providers. *Future Gener. Comput. Syst.*, 29(7):1777–1785.

Hu, X., Ludwig, A., Richa, A. W., and Schmid, S. (2015). Competitive strategies for online cloud resource allocation with discounts: The 2-dimensional parking permit problem. In *35th IEEE International Conference on Distributed Computing Systems, ICDCS 2015, June 29 - July 2, 2015*, pages 93–102. IEEE Computer Society.

Jin, Y., Hayashi, M., and Tagami, A. (2018). Online algorithms for cost-effective cloud selection with multiple demands. In *30th International Teletraffic Congress, ITC 2018, Vienna, Austria, September 3-7, 2018 - Volume 1*, pages 37–45. IEEE.

Kim, W. and Jo, O. (2017). Cost-optimized configuration of computing instances for large sized cloud systems. *ICT Express*, 3(3):107–110.

Mireslami, S., Rakai, L., Wang, M., and Far, B. H. (2021). Dynamic cloud resource allocation considering demand uncertainty. *IEEE Trans. Cloud Comput.*, 9(3):981–994.