Avaliação de Soluções para Alocação de Aplicações Distribuídas em Ambientes de Nuvem

Ana L. R. Herrmann[1], **Guilherme Galante**[1]

[1]Ciência da Computação
Universidade Estadual do Oeste do Paraná (UNIOESTE)
Caixa Postal 711 – 85.819-110 – Cascavel-PR

{ana.hermann, guilherme.galante}@unioeste.br

Abstract. *In IaaS public providers, the user does not have accurate information about the allocation of their virtual machines and how they are connected. This allocation model can lead to significant variations in mean latency between the allocated instances, which can result in degradation of performance of distributed applications. This work evaluates two solutions, Cloudia and Choreo, that aim to improve the allocation of distributed application components among the available VMs considering the latency variability. According to the experiments, both solutions were effective, reducing the execution time of the applications when using the propoded allocations.*

Resumo. *Nos provedores públicos de IaaS, o usuário não possui informação precisa sobre a alocação de suas máquinas virtuais e como estão conectadas. Esse modelo de alocação pode levar a variações significativas na latência média entre as instâncias alocadas, o que pode resultar em degradação significativa do desempenho de aplicações distribuídas, a menos que seja tomado cuidado em como os componentes da aplicação são mapeados para as instâncias. Neste trabalho avalia-se duas soluções, Cloudia e Choreo, que visam melhorar a alocação dos componentes de aplicações distribuídas entre as VMs disponíveis considerando essa questão da latência. De acordo com os experimentos ambas as soluções mostraram-se efetivas, reduzindo o tempo de execução das aplicações ao utilizar as alocações propostas.*

1. Introdução

Na última década, a computação em nuvem ganhou grande popularidade como uma plataforma promissora para a execução de aplicações distribuídas. Seguindo a ideia de Infraestrutura como Serviço (IaaS), algumas nuvens públicas, como o Amazon EC2[1] ou o Google Cloud Platform[2], permitem que seus clientes adquiram conjuntos de recursos de computação por meio de pagamento por uso de curto prazo. De modo geral, os recursos de computação adquiridos são entregues em forma de máquinas virtuais (VMs) hospedadas dentro dos data centers dos provedores.

Atualmente, os provedores de nuvem pública não expõem o escalonamento das instâncias virtuais ou a topologia de rede [Battré et al. 2011]. Os clientes sabem apenas que suas VMs estão em execução em algum lugar na nuvem, sem saber quais VMs

[1]aws.amazon.com/ec2
[2]cloud.google.com

estão hospedadas no mesmo servidor físico, quais servidores compartilham o mesmo rack ou data center, etc. Esse modelo de alocação pode levar a variações significativas na latência média entre as instâncias alocadas pelo cliente, o que pode resultar em degradação significativa do desempenho em aplicações sensíveis à latência, a menos que seja tomado cuidado em como os componentes da aplicação são mapeados para as instâncias [Zou et al. 2015].

Infelizmente, os provedores não disponibilizam serviços que permitam que o cliente mapeie sua aplicação de maneira inteligente, considerando essas questões. Nesse sentido, alguns trabalhos apresentam soluções que visam melhorar a alocação dos componentes de aplicações distribuídas entre as VMs disponíveis considerando a variação da latência.

Neste trabalho duas soluções são avaliadas, Cloudia [Zou et al. 2015] e Choreo [LaCurts et al. 2013], com o objetivo de verificar sua efetividade nesta tarefa. As soluções foram avaliadas em uma nuvem IaaS real, a Google Compute Engine, na qual se realizou um conjunto de experimentos com aplicações com diferentes características. De acordo com os experimentos, ambas as soluções mostram-se efetivas na melhoria da alocação das aplicações em ambientes de nuvem, reduzindo o tempo de execução das aplicações testadas em até 58%.

O restante do trabalho é organizado da seguinte forma. A Seção 2 apresenta os algoritmos de escalonamento avaliados nesse trabalho. Na Seção 3 realiza-se a avaliação experimental. Por fim, a Seção 4 conclui este trabalho.

2. Soluções para Alocação em Ambientes de Nuvem

Nesta seção apresenta-se as duas soluções para a alocação de aplicações distribuídas que serão avaliadas. Ambas são voltadas para em ambiente de nuvem IaaS públicas onde nenhuma informação sobre a alocação das instâncias ou sobre a rede são fornecidas pelos provedores. É importante salientar que as soluções visam oferecer alocações sub-ótimas.

As duas soluções funcionam de maneira semelhante. Como entrada, tem-se (1) o grafo de comunicação da aplicação (representado como uma matriz de adjacências), com os respectivos pesos de cada aresta e (2) a matriz contendo os dados de latência par-a-par. Como saída, obtém-se um conjunto de mapeamentos do componente c para a instância i. Nas seções a seguir, apresenta-se o algoritmo usado por cada uma das soluções. Para mais detalhes sobre os algoritmos recomenda-se a leitura dos artigos originais.

2.1. Cloudia

A alocação dos componentes no Cloudia é realizado conforme apresentado no Algoritmo 1. No algoritmo, $D(x)$ define uma função que retorna a tarefa que está alocada em uma máquina x, enquanto $D^{-1}(v)$ define uma função que retorna a máquina virtual onde está alocada uma tarefa v. Em linhas gerais, o algoritmo parte de uma alocação arbitrária e a partir dela tenta minimizar o custo (latência) das demais alocações.

2.2. Choreo

A alocação dos componentes no Choreo é realizado conforme apresentado no Algoritmo 2. Essa solução baseia-se na alocação de tuplas $< i, j, b >$ que significa que o nó i envia b bytes para j. No Choreo o objetivo é tentar alocar as máquinas que possuem maior comunicação entre si em links com menor latência.

Algoritmo 1: Cloudia()

```
1  begin
2      S → conjunto de instâncias
3      G = (V, E) → grafo de comunicação da aplicação
4      Encontrar o link u_0 → v_0 de menor custo ∈ S × S
5      Encontrar uma aresta arbitrária (x, y) ∈ G
6      D(x)=u_0
7      D(y)=v_0
8      for i=1 até ||V|| − 2 do
9          c_min=∞
10         foreach (u, v) ∈ S × S do
11             if D^{-1}(v) não foi definido e D^{-1}(u) possui vizinhos não alocados then
12                 if latencia(u, v) < c_min then
13                     c_min = latencia(u, v)
14                     u_min = u
15                     v_min = u
16                 end
17             end
18         end
19         w = um dos vizinhos não alocados de D^{-1}(u)
20         D(w) = v_min
21     end
22 end
```

Algoritmo 2: Choreo()

```
1  begin
2      transfers = conjunto de tuplas < i, j, b > em ordem decrescente de b.
3      foreach < i, j, b > em transfers do
4          if i já foi alocada em uma máquina k then
5              P = conjunto de links k → N, ∀ nós N
6          end
7          if j já foi alocada em uma máquina ℓ then
8              P = conjunto de links M → ℓ, ∀ nós M
9          end
10         if i e j não foram alocadas then
11             P = conjunto de links M → N, ∀ nós M e ∀ nós N
12         end
13         foreach link em P do
14             if se a alocação de i em m ou j em n exceder as restrições de CPU de m ou n
                 then
15                 remover m → n de P
16             end
17         end
18         foreach link em P do
19             latência(m,n) = latência do link m → n
20         end
21         Alocar i e j em m e n in latência(m,n) seja minimizada
22     end
23 end
```

3. Experimentos e Resultados

3.1. Ambiente Computacional

O Compute Engine é o serviço de IaaS do Google Cloud Platform. Esse serviço consiste no uso de VMs rodando nos data centers do Google e conectadas à sua rede de fibra ótica. Estes recursos estão distribuídos entre diferentes regiões e zonas. Região é uma localização geográfica específica onde você pode executar os seus recursos. Cada região tem uma ou mais zonas. Por exemplo, a região *us-central1* denota uma região na região central dos Estados Unidos com as zonas *us-central1-a, us-central1-b, us-central1-c e us-central1-f.* A nuvem oferece vários tipos de máquina, que definem uma coleção específica de recursos de hardware virtualizados disponíveis para uma instância de máquina virtual.

Todas as máquinas possuem a mesma configuração: um núcleo de processamento; 4 GB de memória RAM; 40GB de armazenamento não-volátil. O sistema operacional utilizado foi o Ubuntu 16.04, e as configurações gerais são as padrões da plataforma.

3.2. Analisando a Latência

Para comprovar a problemática de heterogeneidade de latências entre um conjunto de máquinas virtuais dentro de um mesmo projeto em nuvem, foi calculada a latência entre 10 VMs instanciadas no Google Cloud Engine, alocadas em três zonas distintas (*us-east1-b, us-west1-a* e *us-central1-c*). O cálculo foi feito com base nas médias dos dados apresentados pelo comando *ping*. Os resultados são apresentados na Figura 1, que apresenta a média das latências entre as zonas.

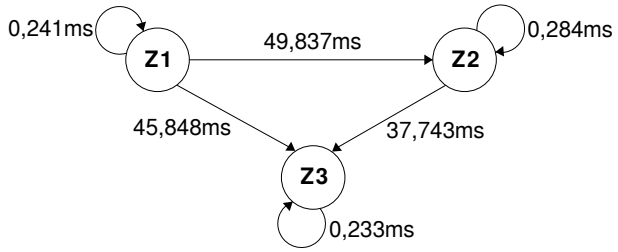

Figura 1. Latência média entre diferentes zonas.

3.3. Aplicação dos Algoritmos

Para a execução dos experimentos deste trabalho, foram utilizadas aplicações com topologias de comunicação Todos-para-Todos e Cliente-Servidor. A aplicação Todos-para-Todos tem como objetivo mimetizar o comportamento de uma aplicação *peer-to-peer*, nas quais há comunicação entre boa parte das tarefas. Por sua vez, na aplicação Cliente-Servidor ocorre o envio de dados das tarefas clientes para a tarefa Servidor, simulando uma operação de redução (todos para um).

Na aplicação Todos-para-Todos, cada uma das 10 tarefas faz o envio de um arquivo para as outras 9 tarefas. Na aplicação Cliente-Servidor as tarefas 2 a 10 enviam dados para a tarefa 1. O tamanho das mensagens enviadas pela aplicação Todos-para-Todos é apresentado na Tabela 1 e pela aplicação Cliente-Servidor é apresentada na Tabela 2. As mensagens foram geradas aleatoriamente e com tamanhos arbitrários.

Tabela 1. Matriz de comunicação para a aplicação Todos para Todos (KB).

	1	2	3	4	5	6	7	8	9	10
1	0	223920	147320	229250	299510	14630	271600	312990	202350	187660
2	59910	0	169960	240000	251200	39250	244890	46170	202890	79980
3	190210	87690	0	205920	80480	293660	44210	84180	212150	292720
4	112130	202900	284430	0	17367	286680	222610	247040	180270	49170
5	242260	78520	42330	67050	0	163700	95270	219520	141580	155080
6	257580	255690	253860	11570	162550	0	315860	254570	290700	266360
7	17960	132990	71820	127300	303880	15300	0	115130	110650	168150
8	189060	43850	52890	214820	304090	291080	15470	0	273130	12650
9	56380	230290	93580	199080	47530	124260	185400	180640	0	269390
10	78500	20440	187710	324710	239100	167280	245280	25410	35110	0

Tabela 2. Matriz de comunicação para a aplicação Cliente-Servidor (KB).

	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	0	0	0	0
2	242300	0	0	0	0	0	0	0	0	0
3	40040	0	0	0	0	0	0	0	0	0
4	206780	0	0	0	0	0	0	0	0	0
5	116440	0	0	0	0	0	0	0	0	0
6	86330	0	0	0	0	0	0	0	0	0
7	96850	0	0	0	0	0	0	0	0	0
8	141370	0	0	0	0	0	0	0	0	0
9	105790	0	0	0	0	0	0	0	0	0
10	86790	0	0	0	0	0	0	0	0	0

Utilizando esses dois cenários, as soluções Cloudia e Choreo são comparadas por uma abordagem *First-Fit*, na qual a tarefa 1 é alocada na máquina 1, a tarefa 2 é alocada na máquina 2 e assim sucessivamente.

Para a aplicação Todos-para-Todos, após aplicar o Cloudia e Choreo, obteve-se a alocação apresentada na Tabela 3 para cada uma das tarefas. Na Tabela 4 apresenta-se o tempo de execução da aplicação utilizando as distintas alocações. Pode-se observar que a melhoria obtida pelo uso das soluções Cloudia e Choreo é superior a 40%, com uma diminuição de aproximadamente 58 segundos no tempo de execução.

Tabela 3. Tarefa da aplicação, e as VMs escolhidas por cada um dos métodos.

Tarefa	VM (First-Fit)	VM (Choreo)	VM (ClouDiA)
1	1	10	10
2	2	1	6
3	3	8	9
4	4	4	4
5	5	2	7
6	6	9	3
7	7	5	2
8	9	3	1
9	9	7	5
10	10	6	8

Tabela 4. Tempo de execução (segundos) para a aplicação Todos-para-Todos.

	ClouDiA	Choreo	First Fit
Tempo Médio	78,5	79,33	136,5
Comparação com o FF	57,51%	58,12%	100%

Para a aplicação Cliente-Servidor, obteve-se uma mesma alocação com os algoritmos do Cloudia e Choreo, conforme apresentada na Tabela 5. Na Tabela 6 apresenta-se o

tempo de execução da aplicação utilizando as distintas alocações. Pode-se observar que a melhoria obtida pelo uso das soluções Cloudia e Choreo é de aproximadamente 58%, com uma diminuição de aproximadamente 16 segundos no tempo de execução.

Tabela 5. Tarefa da aplicação, e as VMs escolhidas por ambos os métodos.

Tarefa	VM (First-Fit)	VM (Choreo e Cloudia)
1	1	2
2	2	6
3	3	3
4	4	1
5	5	4
6	6	8
7	7	7
8	8	9
9	9	10
10	10	5

Tabela 6. Tempo de execução (segundos) para a aplicação Cliente-Servidor.

	ClouDiA	Choreo	First Fit
Tempo Médio	12	12	28
Comparação com o FF	42,85%	42,85%	100%

4. Conclusão

As características da infraestrutura das nuvens públicas, podem causar uma diferença significativa nas latências em comunicações utilizando diferentes pares de máquinas virtuais alocadas para um determinado cliente. Por consequência, aplicações sensíveis à comunicação podem ter seu desempenho afetado negativamente.

Esse impacto pode ser minimizado se forem utilizados técnicas para melhorar a alocação dos componentes de aplicações distribuídas entre as VMs disponíveis considerando essa variação de latência.

Neste trabalho, duas soluções para a alocação de aplicações em nuvens foram avaliadas, Cloudia [Zou et al. 2015] e Choreo [LaCurts et al. 2013]. Ambas as soluções mostraram-se igualmente efetivas na melhoria das alocações das VMs, possibilitando a redução do tempo de execução das aplicações entre 40% e 58%.

Como trabalho futuro, pretende-se estudar outros algoritmos para a alocação de aplicações em nuvens públicas, assim como realizar testes mais completos com outros cenários e utilizando outras classes de aplicações.

Referências

Battré, D., Frejnik, N., Goel, S., Kao, O., e Warneke, D. (2011). Evaluation of network topology inference in opaque compute clouds through end-to-end measurements. In *IEEE CLOUD*, pages 17–24. IEEE Computer Society.

LaCurts, K., Deng, S., Goyal, A., e Balakrishnan, H. (2013). Choreo: Network-aware task placement for cloud applications. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 191–204, New York, NY, USA. ACM.

Zou, T., Bras, R., Salles, M. V., Demers, A., e Gehrke, J. (2015). Cloudia: A deployment advisor for public clouds. *The VLDB Journal*, 24(5):633–653.