Problemas de Otimização em Ambientes de Computação em Nuvem

Nesta aula estudaremos alguns problemas de alocação de recursos em sistemas de Computação em Nuvem procurando modelá-los sob um olhar de otimização combinatória. Nosso principal intuito é exercitar, por meio de problemas e modelos simples, a criação de modelos em ambientes computacionais de larga escala.

1. Maior Receita na Nuvem (BLP)

Neste problema temos um conjunto $\{1...V\}$ de VMs, sendo que cada VM i possui um preço p_i e uma **demanda** de capacidade d_i. Temos também um conjunto de servidores $\{1...M\}$, com cada servidor j possuindo sua respectiva capacidade C_j. Para representar a alocação de uma VM i em um servidor j utilizamos uma **variável de decisão** booleana x_{ij} que assume 1 no caso da máquina virtual i estar alocada no servidor j, e 0 caso contrário. Assim, o que desejamos é a maximização do lucro

$$maximizar \sum_{i=1}^{V} \sum_{j=1}^{M} p_i x_{ij}$$

sujeito às seguintes restrições

$$\sum_{i=1}^{V} d_i x_{ij} \leq C_j, \ para\ todo\ servidor\ j$$

$$\sum_{j=1}^{M} x_{ij} \leq 1, \ para\ toda\ VM\ i$$

$$x_{ij} \in \{0, 1\}, \ para\ todo\ i\ e\ j$$

Este problema é um problema NP-difícil e equivale ao problema das múltiplas mochilas (0-1 *Multiple Knapsack*). Este problema é descrito e diversas formas de solucioná-lo são apresentadas no Capítulo 6 do livro de Martello e Toth [1].

2. Gerência de Energia sem preço (BLP)

Neste problema faremos gerenciamento de energia por meio do desligamento de servidores. Novamente, temos um conjunto $\{1...V\}$ de VMs, sendo que cada VM i possui apenas a sua respectiva demanda de capacidade d_i. Temos também um conjunto de servidores $\{1...M\}$, com cada servidor j possuindo sua respectiva capacidade C_j. Um servidor pode ou não estar ligado, o que é modelado por meio da variável de decisão booleana y_j, com 1 representando o servidor ligado e 0 desligado. A alocação da uma VM i em um servidor j é novamente modelada pela variável de decisão booleana x_{ij} que assume 1 no caso da máquina virtual i estar alocada no servidor j, e 0 caso contrário. Assim, o que desejamos é minimizar o número de servidores ativos.

$$minimizar \sum_{j=1}^{M} y_j$$

sujeito às seguintes restrições

$$\sum_{i=1}^{V} d_i x_{ij} \leq y_j C_j, \ para\ cada\ servidor\ j$$

$$\sum_{j=1}^{M} x_{ij} = 1, \ para\ cada\ VM\ i$$

$$x_{ij} \in \{0, 1\}, \ para\ todo\ i\ e\ j$$

$$y_j \in \{0, 1\}, \ para\ todo\ j$$

Este problema é um problema NP-difícil e equivale ao problema conhecido como Bin-packing. Este problema é descrito e diversas formas de solucioná-lo são apresentadas no Capítulo 8 em [1].

3. Maximizando a Lucro (BLP)

Podemos modificar este problema adicionando preço. Para tanto, basta que adicionemos a variável do preço p_i de cada VM i e um custo energético de cada servidor w_j. Assim, desejamos otimizar a seguinte função objetivo

$$maximizar \left(\sum_{i=1}^{V} \sum_{j=1}^{M} p_i x_{ij} - \sum_{j=1}^{M} w_j y_j \right)$$

sujeita às seguintes restrições

$$\sum_{i=1}^{V} d_i x_{ij} \leq y_j C_j, \ para\ cada\ servidor\ j$$

$$\sum_{j=1}^{M} x_{ij} \leq 1, \ para\ cada\ VM\ i$$

$$x_{ij} \in \{0, 1\}, \ para\ todo\ i\ e\ j$$

$$y_j \in \{0, 1\}, \ para\ todo\ j$$

4. Balanceamento de carga sem preço (BLP)

O objetivo neste problema é equalizar a carga entre os servidores existentes na Nuvem. Assim temos, novamente, um conjunto $\{1...V\}$ de VMs com suas respectivas demandas de capacidade d_i e temos também um conjunto de servidores $\{1...M\}$ com suas respectivas capacidades C_j. Como nos anteriores, a alocação da uma VM i em um servidor j é novamente

modelada pela variável de decisão booleana x_{ij} que assume 1 no caso da VM i estar alocada no servidor j, e 0 caso contrário. Para este problema de seja-se

$$minimizar \max_{j=1\ldots M} \left(\sum_{i=1}^{V} d_i x_{ij} \right)$$

respeitando as seguintes restrições

$$\sum_{i=1}^{V} d_i x_{ij} \leq C_j, \; para\ cada\ servidor\ j$$

$$\sum_{j=1}^{M} x_{ij} = 1, \; para\ cada\ VM\ i$$

$$x_{ij} \in \{0, 1\}, \; para\ todo\ i\ e\ j$$

Este problema é **parecido** com **o problema de partição de um conjunto de inteiros em K subconjuntos disjuntos**, **tal que as somas dos números em cada subconjunto sejam as mais próximas possíveis**. Contudo, o problema difere no fato de que a soma dos inteiros alocados em cada subconjunto não pode ultrapassar um determinado valor, coisa que não ocorre no problema da partição de conjuntos. Uma discussão sobre o problema da partição de conjuntos pode ser encontrada em [2].

5. Proteção de Servidores (LP ou NLP)

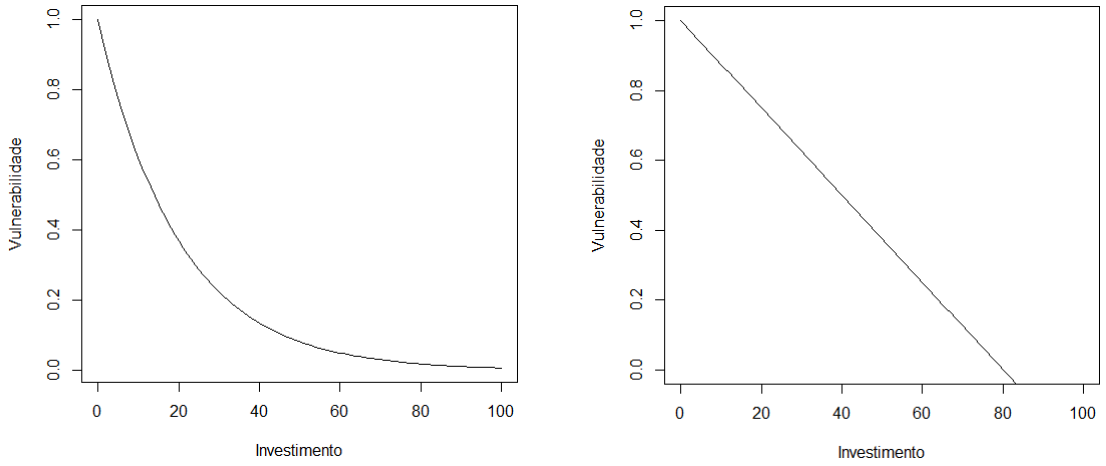
Neste problema introduzimos o conceito de risco. O risco é uma medida da importância de um sistema (ou um de seus componentes) e do quanto este sistema é suscetível a falhas. Assim, todo sistema pode ser visto como tendo uma vulnerabilidade, a qual é medida entre 0 e 1, e uma consequência, valor maior que zero, que indica qual o impacto da perda daquele sistema (ou componente) para seu administrador. Modelamos o risco de um sistema por meio da equação $risco = vulnerabilidade \times consequência$.

Para um sistema formado por **diversos componentes**, temos que o risco do sistema dependerá do risco individual de cada um dos seus componentes (r_i). Considerando a existência de C componentes e que as falhas de cada componente são independentes e não afetam umas às outras temos que o risco do sistema é dado por

$$\sum_{i=1}^{C} r_i = \sum_{i=1}^{C} c_i v_i$$

Contudo, a vulnerabilidade de cada componente pode ser diminuída por meio do investimento na proteção daquele componente. Assim, a vulnerabilidade de um componente é dada pela função $v_i(I_i)$, com I_i correspondendo ao investimento feito no componente i. Esta função é decrescente, porém o formato da curva de decrescimento depende de cada componente. Uma forma corriqueira de descrever esta curva é por meio de uma **função exponencial** da

forma $v_i(I_i) = e^{-\alpha_i I_i}$ como mostrada na Figura 1(a). Esta função é interessante porque reflete o aspecto dos **retornos decrescentes**, i.e., o retorno de investimento inicialmente é alto, mas decresce a medida que investimos mais. Outra função que pode ser utilizada é a **função linear** do tipo $v_i(I_i) = 1 - \frac{I_i}{maxI_i}$ como mostrada na Figura 1(b). Embora a função linear não modele bem a vulnerabilidade de componentes reais, seu estudo pode ser interessante já que: a) permite uma solução simples e elegante para o problema que veremos a seguir; e b) é possível modelar partes de uma função qualquer por meio de uma função linear.



(a) (b)

Figura 1. Exemplo de funções de vulnerabilidade (a) Exponencial e (b) Linear.

No contexto da Computação em Nuvem, podemos aplicar este modelo para decidir como distribuir os investimentos em um **conjunto de servidores** $\{1...M\}$ de modo a minimizar **o risco**, mas sem que o total dos investimentos **ultrapassem um orçamento O dado**. Assim, devemos escolher qual o valor de investimento I_j deve ser destinado para a proteção de cada servidor j de acordo com a **consequência do servidor** (c_j) e da sua **função de vulnerabilidade** $v_j(I_j)$. A consequência do servidor pode ser medida em termos de, por exemplo, o número de máquinas virtuais existentes no servidor, o custo para consertá-lo, a perda financeira caso ele falhe etc. Pode-se ainda usar qualquer combinação destes fatores, desde que $c_j \geq 0$. Já para a vulnerabilidade usaremos a função linear, assim, cada servidor possuirá seu próprio valor $maxI_j$, de forma que $v_j(maxI_j) = 0$.

Considerando estas variáveis, e que a variável de decisão são os valores de investimento I_j em cada servidor, nosso problema é

$$minimizar \sum_{j=1}^{M} c_j v_j(I_j)$$

respeitando a seguinte restrição

$$\sum_{j=1}^{M} I_j \leq O$$

Como utilizamos uma **função linear**, existe uma **solução gananciosa** para este problema que consiste em investir nos servidores com base no retorno de investimento, o qual é dado pelo índice $\frac{c_j}{maxI_j}$. Assim, basta ordenar os servidores com base nesta fração do maior para o menor retorno de investimento e selecionarmos os servidores com maior índice para investir, aplicando a estes servidores um investimento igual ao mínimo entre $maxI_j$ e o restante do orçamento. Assim, uma vez selecionado um servidor para alocação, iremos nos esforçar para zerar a seu risco. Detalhes sobre este problema aplicado a outro contexto e a demonstração de que esta solução leva ao ótimo podem ser encontrados em [3].

A solução do problema **não linear**, i.e., usando a equação $v_i(I_i) = e^{-\alpha_i I_i}$, é obtida por meio de relaxação lagrangiana e pode ser encontrado em [3].

6. Alocação com risco mínimo (BLP)

Podemos introduzir no problema da alocação de máquinas virtuais a ideia de risco de modo que a alocação seja feita para obter o menor risco. Antes de introduzir este problema, introduziremos outro importante problema que utilizaremos para resolver o problema de alocação com risco mínimo.

Fluxo de Custo Mínimo (LP)

O problema de fluxo de custo mínimo é importante por ter aplicações em diversos contextos da computação e engenharias, principalmente em redes de computadores. Neste problema temos um grafo $G = (V, E)$ e dois vértices específicos $s, t \in V$ que serão a origem e destino, respectivamente, do envio de um fluxo de tamanho d. Esta demanda deverá ser enviada através das arestas dos grafo de modo que, para cada enlace $(i, j) \in E$, teremos uma variável $f_{ij} \geq 0$ que mede o quanto do fluxo atravessa o enlace, uma capacidade do enlace c_{ij} e um custo a_{ij} do enlace. Assim, devemos escolher quais enlaces utilizar, de modo que tenhamos o menor custo, sem que o fluxo enviado por meio de uma aresta ultrapasse a capacidade dela.

Assim temos que

$$minimizar \sum_{(i,j)\in E} a_{ij} f_{ij}$$

respeitando as seguintes restrições

$$\sum_{j\in V} f_{ji} - \sum_{j\in V} f_{ij} = \ -d, \ para\, i = s$$

$$\sum_{j\in V} f_{ji} - \sum_{j\in V} f_{ij} = \ 0, \ para\, i \neq s, t$$

$$\sum_{j\in V} f_{ji} - \sum_{j\in V} f_{ij} = \ d, \ para\, i = t$$

$$0 \leq f_{ji} \leq c_{ij} \; para \; todo \; (i,j) \in E$$

Este problema é conhecido por ter uma solução polinomial e para resolvê-lo existem diversos algoritmos. Mais detalhes sobre este problema e suas soluções podem ser encontrados em [4].

■

Retornemos ao problema de alocação com risco mínimo. Considere um ambiente de Computação em Nuvem com $S = \{1...M\}$ servidores, sendo que cada um possui sua capacidade C_j e probabilidade de falha p_j. Diferentemente dos problemas anteriores, as máquinas virtuais $VM = \{1...V\}$ são consideradas todas iguais, não havendo diferença de demanda, contudo cada VM tem uma consequência de perda w_i.

A variável de decisão x_{ij} indica que a VM i está alocada no servidor j. Introduzimos neste problema um parâmetro A_{ij}, que indica a afinidade de uma VM a um servidor. Assim, se a VM i pode ser alocada no servidor j, então $A_{ij} = 1$, caso contrário $A_{ij} = 0$. Note que esta variável permite a modelagem de situações onde uma determinada VM só pode ser alocada em determinados servidores devido a restrições como tipo do sistema operacional, arquitetura do processador, localização geográfica etc.

O problema, portanto, é alocar as VMs minimizando o risco na infraestrutura de nuvem. De modo formal queremos

$$minimizar \sum_{j=1}^{M} p_j \sum_{i=1}^{V} w_i x_{ij}$$

respeitando as seguintes restrições

$$x_{ij} \leq A_{ij}, \; para \; todo \; i \; e \; j$$

$$\sum_{j=1}^{M} x_{ij} = 1, \; para \; cada \; VM \; i$$

$$\sum_{i=1}^{V} x_{ij} \leq C_j, \; para \; cada \; servidor \; j$$

$$x_{ij} \in \{0,1\}, \; para \; todo \; i \; e \; j$$

Este problema pode ser resolvido por meio de sua **redução** ao problema de fluxo de custo mínimo. Para tanto basta criar um problema de fluxo com uma demanda d igual ao número de VMs a ser alocadas e um grafo $G = (V, E)$, onde $V = \{s\} \cup \{t\} \cup \{VM\} \cup \{S\}$ e E será construído da seguinte forma:

a) Haverá um enlace partindo de s para todo vértice em $i \in \{VM\}$, os quais possuirão $a_{si} = 0$ e $c_{si} = 1$;

b) Haverá um enlace partindo de cada vértice $j \in \{S\}$ para o vértice t, os quais possuirão $a_{jt} = 0$ e $c_{jt} = C_j$;
c) Caso $A_{ij} = 1$, então haverá uma aresta do vértice $i \in \{VM\}$ ao vértice $j \in \{S\}$, a qual possuíra $a_{ij} = p_j w_i$ e $c_{ij} = 1$. Caso $A_{ij} = 0$, nenhuma aresta é criada entre o vértice $i \in \{VM\}$ ao vértice $j \in \{S\}$.

Como se pode notar, esta redução leva qualquer instância do problema de alocação com risco mínimo em uma instância do problema de fluxo de custo mínimo, o qual pode ser resolvido em tempo polinomial. Note que problemas de fluxo de custo mínimo podem ter solução fracionária ($f_{ij} \in R$), contudo há um teorema que mostra que a solução será sempre inteira ($f_{ij} \in Z$) para esta redução. Este problema, a redução para fluxo de custo mínimo e o teorema citado são apresentadas em [5] juntamente com avaliações.

7. Referências

[1] Martello, S. e Toth, P. *Knapsack Problems: Algorithms and Computer Implementations*, Wiley & Sons, 1990, Disponível em: http://www.or.deis.unibo.it/knapsack.html
[2] Korf, R. E. *Multi-Way Number Partitioning*, 21st International Joint Conference on Artificial Intelligence, 2009.
[3] Lewis, T. G. *Network Science: Theory and Applications*, Wiley, 2009.
[4] Jungnickel, D. *Graphs, Networks, and Algorithms*, Springer, 3ª Edição, 2008.
[5] Palhares, A. V. A. *Probabilistic Risk Assessment in Clouds: Models and Algorithms*, Dissertação de Mestrado, CIn/UFPE, 2012.