3
Seleção de Links Patrocinados com Restrição de Limite de Exibições

Neste capítulo especificamos o problema de seleção de links patrocinados com o objetivo de maximizar a receita e utilizando como restrição um limite máximo de exibições por anunciante. Em seguida apresentamos duas diferentes estratégias para calcularmos a solução ótima offline. Este cálculo é fundamental para verificarmos o desempenho dos algoritmos online. Por fim discutimos sobre a competitividade dos algoritmos online aplicados a este problema.

3.1
Especificação do Problema

Considere que n anunciantes, $a_1, a_2, ..., a_n$, contratam o serviço de links patrocinados. Dentro de um conjunto Q de consultas possíveis, cada anunciante i faz por uma consulta j uma oferta b_{ij}. Cada anunciante i define também um limite máximo de L_i exibições por dia. Durante um dia de processamento chega uma sequência de m consultas, $q_1, q_2, ..., q_m$, onde $q_j \in Q$. No momento da chegada de cada consulta, podem ser exibidos de 0 a T anunciantes. O objetivo da empresa que presta o serviço é maximizar a receita diária respeitando o limite de exibições de cada anunciante. Consideramos que a receita de cada exibição é exatamente o valor da oferta do anunciante exibido pela consulta.

Para calcular a receita no fim do dia utilizamos uma variável x_{ij} que recebe 1, se o anunciante i é exibido na consulta j, e 0, caso contrário. A receita no fim do dia é: $\sum_{j=1}^{m} \sum_{i=1}^{n} b_{ij}\ x_{ij}$.

Consideramos que os anunciantes e limites, as ofertas e o parâmetro T não se modificam ao longo do dia.

3.2
Solução Ótima

Considere as seguintes constantes e variáveis do problema para um dia específico de processamento:

- n anunciantes fazem ofertas por consultas pré-definidas;
- m é a quantidade de consultas no dia;

- b_{ij} é a oferta de um anunciante i por uma consulta j;
- cada anunciante i paga por até L_i exibições por dia;
- T é o limite de anunciantes exibidos em uma consulta;
- x_{ij} indica se exibe ou não o anunciante i em uma consulta j; se exibe, $x_{ij} = 1$; se não exibe, $x_{ij} = 0$.

O cálculo da solução ótima, fundamental para avaliação do desempenho dos algoritmos online, só é possível no fim do dia, quando todas as consultas ocorridas no dia já são conhecidas.

O caso mais estudado na literatura, quando temos um limite de orçamento por anunciante, pode ser modelado por programação inteira. Apesar de ser NP-completo, podemos atingir uma $(1 + \epsilon)$-aproximação por programação linear, onde ϵ é a diferença máxima que pode ocorrer entre a solução encontrada e a solução ótima. Para atingirmos esta aproximação devemos considerar que as ofertas são bem pequenas em relação aos orçamentos ($b_{ij} << B_i$), o que é bastante aceitável nos casos reais (BHJMQS07).

Uma característica do problema que estudamos, que o diferencia do citado anteriormente, é que pode ser modelado por um fluxo em redes com capacidades inteiras, como veremos ainda neste capítulo, no item 3.2.1. Esta condição garante que a solução ótima encontrada pela programação linear terá uma solução inteira para cada x_{ij} encontrado.

A seguir veremos os dois modelos de solução ótima possíveis e que foram utilizados em nossas avaliações.

3.2.1 Solução por Fluxo em Redes

Para modelar o problema como um fluxo em redes definimos um grafo bipartido tendo como vértices: as consultas e os anunciantes, como vemos na Figura 3.1. Incluímos arestas ligando cada consulta a todos os anunciantes com uma capacidade 1, pois um anunciante pode ser atribuído uma única vez a cada consulta. Criamos um vértice origem s que ligamos a todos os vértices de consultas com capacidade T, para garantir que teremos no máximo T anunciantes por consulta. Criamos finalmente um vértice destino ao qual ligamos todos os anunciantes com capacidade L_i para cada anunciante i, para garantir que o limite de exibições não seja ultrapassado.

O passo seguinte é atribuirmos receitas às arestas que ligam as consultas aos anunciantes que fizeram ofertas por elas para calcularmos o fluxo de maior receita. Vamos entretanto transformar as receitas em custos, como explicamos

a seguir, para utilizarmos um algoritmo para cálculo de fluxo máximo de custo mínimo.

Primeiramente encontramos a maior oferta de um anunciante por uma consulta, a qual chamaremos b_{max}. Atribuímos então um custo c_{ij} a toda aresta que liga uma consulta j a um anunciante i, onde $c_{ij} = b_{max} - b_{ij}$. Caso um anunciante u não tenha feito oferta por uma consulta v, consideraremos esta aresta como de custo máximo, isto é, $c_{uv} = b_{max}$. As arestas que saem do vértice origem e as que chegam no vértice destino não possuem custo. A Figura 3.1 ilustra como ficou a modelagem.

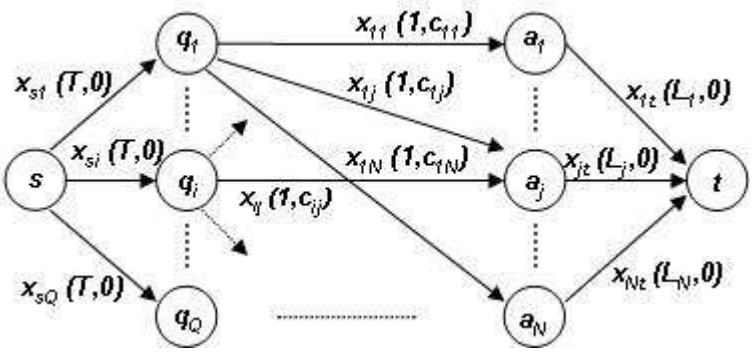

Figura 3.1: Solução ótima por fluxo em redes

Para resolver este problema podemos usar qualquer algoritmo de fluxo máximo de custo mínimo. Em nossa implementação testamos os algoritmos *Successive Shortest Path* e *Cost Scaling*, descritos em (AMO93), e optamos pelo segundo que apresentou um melhor desempenho nos testes iniciais.

Como resultado do algoritmo obtemos os x_{ij} que indicam o fluxo ótimo em cada aresta do grafo. Utilizamos estes valores para calcular a contribuição r_{ij} de cada par de anunciante i e consulta j na receita fazendo $r_{ij} = b_{ij} - c_{ij}$. Depois totalizamos todas as receitas para calcular a receita total ótima fazendo: $R_{otima} = \sum_{i=1}^{n} \sum_{j=1}^{m} r_{ij}$.

3.2.2 Solução por Programação Linear

A outra forma de encontrarmos a solução ótima offline é obtermos o valor da função objetivo do problema de programação inteira descrito na Figura 3.2.

Como este problema pode ser modelado por um fluxo em redes com capacidades inteiras, podemos relaxar a última restrição para transformar o problema em uma programação linear sem perdermos a precisão do resultado. Desta forma a última restrição passa a ser: $\forall i, j : 0 \leq x_{ij} \leq 1$ e o problema pode ser resolvido por qualquer algoritmo de programação linear como o simplex.

$$
\begin{array}{rll}
\text{Maximizar} & \sum_{i=1}^{n} \sum_{j=1}^{m} b_{ij}\ x_{ij} & \\
\text{Sujeito à:} & & \\
\forall j: & \sum_{i=1}^{n} x_{ij} \leq T & \leftarrow \text{restrição por consulta} \\
\forall i: & \sum_{j=1}^{m} x_{ij} \leq L_i & \leftarrow \text{restrição por anunciante} \\
\forall i,j: & x_{ij} \in \{0,1\} & \leftarrow \text{não exibe ou exibe link}
\end{array}
$$

Figura 3.2: Solução ótima por programação inteira

3.3 Competitividade dos Algoritmos Online

Como apresentado no Capítulo 1, em (MSVV05) e (BJN07) são propostos algoritmos que garantem uma competitividade de $1 - 1/e$ para o problema de links patrocinados com restrição de orçamento. Uma premissa importante para provar esta competitividade é que as ofertas sejam bem menores que os orçamentos $b_{ij} << B_i$. Caso não tivéssemos esta restrição, poderia ocorrer a seguinte situação, por exemplo.

Considere que tenhamos apenas um anunciante i e que este faça ofertas pelas consultas j e k, sendo $b_{ij} = 1$ e $b_{ik} = B_i$. Suponha a existência de um adversário que enviasse consultas j até que alguma fosse atribuída ao anunciante i. Logo após a atribuição, este adversário enviaria uma consulta k. Esta seria a última consulta do dia e não poderia ser atribuída ao anunciante. Neste caso, a receita total do algoritmo seria 1, contra B_i da solução ótima. Caso o algoritmo nunca atribuisse a consulta j ao anunciante i, após $B_i - 1$ consultas j, este adversário encerraria a sequência de consultas. Em ambos os casos, o algoritmo ficaria a uma distância de $B_i - 1$ da solução ótima.

Definindo o problema com restrição no limite de exibições, em lugar do orçamento, criamos um problema para o cálculo da competitividade, pois já não temos a limitação no valor das ofertas ($b_{ij} << B_i$). Uma solução para isso, é considerar a maior oferta de um anunciante multiplicada por seu limite de exibições, como sendo seu orçamento, e definir que as ofertas devem ser bem menores que o valor calculado. Evitamos assim que tenhamos ofertas de valores exageradamente grandes, mas não resolvemos o problema do cálculo da competitividade.

Quando o problema possui restrição no orçamento, a maior receita que um algoritmo pode atingir é o somatório dos orçamentos de cada anunciante. Neste caso, quando uma consulta recebida é atribuída a um anunciante, o algoritmo aumenta sua receita do mesmo valor que é abatido do orçamento. No caso do limite de exibições, isto não acontece. O valor da maior receita possível é o somatório dos limites de cada anunciante multiplicado pela maior oferta deste anunciante. Qualquer consulta recebida e atribuída a um anunciante abate o valor do limite de exibições de 1, mas o aumento da receita depende do valor da oferta. Isto significa que, ao recebermos ofertas de baixo valor, gastamos nosso limite da mesma forma que gastamos para uma oferta alta, mas o aumento na receita pode ser bem diferente.

O grande problema é que controlamos exibições, mas medimos a competitividade em receita. Assim, um adversário pode enviar inicialmente muitas consultas com ofertas baixas até esgotar o limite de exibições e depois enviar consultas com ofertas altas e deixar o algoritmo com um desempenho muito distante da solução ótima. Caso o algoritmo não aceite as consultas com ofertas de valor baixo, o adversário não envia outras, deixando o algoritmo com um péssimo desempenho da mesma forma. Assim, qualquer análise de pior caso será sempre muito ruim.