Probabilidade e Grafos

Aluno: Israel Danilo Lacerra, Orientador: José Coelho de Pina
Instituto de Matemática e Estátistica, USP

Algoritmos probabilísticos

Probabilidade tem um papel importante em vários algoritmos[MR96]. Alguns algoritmos determinísticos têm sua eficiência comprovada probabilisticamente. Há ainda algoritmos que fazem escolhas aleatórias durante sua execução, são estes os chamados **algoritmos probabilísticos**. Embora possa parecer que usar um algoritmo que faz escolhas aleatórias não seja uma boa idéia, tais algoritmos são muito usados devido a sua eficiência e simplicidade.

Exemplo: Corte mínimo

Um **corte** em um grafo é um conjunto de arestas que, se removidas, dividem o grafo em dois ou mais componentes. Um corte é **mínimo** se tem o menor número de arestas dentre todos os cortes do grafo. O **problema do corte mínimo** consiste em encontrar um corte mínimo de um dado grafo.

Podemos tentar encontrar o corte mínimo de um grafo usando um algoritmo cujo consumo de tempo esperado é linear[MU05]. A idéia básica é sortearmos a cada iteração uma aresta, que será contraída. Ao contrair uma aresta (u, v), removemos todas as arestas entre u e v, e transformamos u e v em um único vértice. Como a cada iteração diminuímos o tamanho do grafo em um vértice, teremos 2 vértice após $n - 2$ iterações. As arestas entre esses vértices constituem um corte não necessariamente mínimo do grafo.

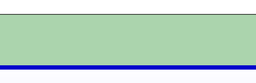

Figura 1: Aresta $(1, 2)$ sendo contraída.

Vejamos dois exemplos de execução do algoritmo para o mesmo grafo. No primeiro exemplo (mostrado na figura 2) o algoritmo não conseguiu encontrar um corte mínimo. No primeiro passo ocorre a contração da aresta $(2, 5)$ fazendo com que os vértices 2 e 5 se tranformem no vértice (ou conjunto de vértices) B. Em seguida ocorre a contração da aresta $(B, 4)$ e o vértice 4 é incorporado ao vértice B. Finalmente o algoritmo contrai a aresta $(1, 3)$ e obtem um corte de tamanho 3.

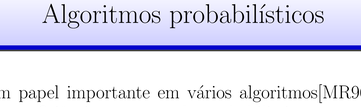

Figura 2: Algoritmo encontra um corte de tamanho 3.

Já no segundo exemplo (figura 3), o algoritmo consegue encontrar um corte mínimo.

Figura 3: Algoritmo encontra um corte mínimo.

E um algoritmo que nem sempre retorna a resposta certa seria útil? Na verdade, podemos provar que o algoritmo retorna a resposta certa com probabilidade maior ou igual a $2/(n(n-1))$ e funciona melhor nos grafos onde o corte mínimo é bem pequeno comparado ao número de arestas (pois nesse caso a probabilidade do algoritmo sortear e contrair arestas que sejam do corte é pequena).

Método Probabilístico

O **método probabilístico** [Spe95, MU05] é usado essencialmente em teoria de grafos como uma ferramenta para provar a existência de determinada configuração em um grafo, ou para provar a existência de um grafo com certas especifidades. A idéia básica é definir um espaço amostral com todas as configurações possíveis e mostrar que a probabilidade de selecionar aleatoriamente a configuração desejada é maior do que zero e, portanto, tal grafo deve existir.

Paul Erdős foi um dos primeiros a usar o método probabilístico em suas provas. Em 1947 ele mostrou, usando o método probabilístico, que se $\binom{n}{2}^{-\binom{k}{2}+1} < 1$ então é possível bicolorir um grafo completo com n vértices de forma que nenhum clique de tamanho menor ou igual a k seja monocromático. Erdős definiu um espaço com todas as possíveis bicolorações do grafo e provou que a probabilidade de escolhermos aleatoriamente uma coloração que cumpra os requisitos é maior do que 0.

Exemplo: Conjuntos de vértices independentes

Um **conjunto de vértices independentes** é um conjunto onde, para quaisquer vértices u e v pertencentes ao conjunto, não existe nenhuma aresta (u, v). Provaremos a seguir, usando uma abordagem que tem como base o método probabilístico, que todo grafo G com n vértices e m arestas possui um conjunto de vértices independentes de tamanho maior ou igual a $n^2/4m$.

Considere o seguinte algoritmo probabilístico que encontra um conjunto de vértices independentes:
Passo 1. Cada vértice é removido, junto com todas as suas arestas, com probabilidade $1 - \frac{1}{2m/n}$.
Passo 2. Para cada aresta restante, remove-se um dos vértices da aresta e a própria aresta.
Passo 3. Retorna os vértices restantes.

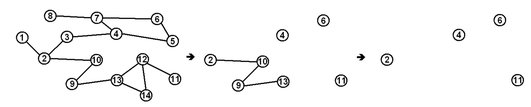

Figura 4: Exemplo de execução do algoritmo

No exemplo da figura 4, os vértices 1, 3, 5, 7, 8, 12 e 14 foram removidos no primeiro passo do algoritmo. Repare que como o segundo passo também não é determinístico, diferentes saídas poderiam ocorrer também nesse passo. Abaixo analisaremos probabilisticamente o algoritmo.

Seja V o número de vértices e A o número de arestas que sobreviveram ao primeiro passo. Repare que a probabilidade de um dado vértice sobreviver é $\frac{1}{2m/n}$ e uma aresta só permanece no grafo se nenhum de seus vértices foi removido.

$$E[V] = n \times \frac{1}{2m/n} = n^2/2m$$

$$E[A] = m \times \frac{1}{(2m/n)^2} = n^2/4m$$

O segundo passo irá remover todas as A arestas e no máximo A vértices. Portanto, ao final do segundo passo, teremos pelo menos $V - A$ arestas. E então:

$$E[V - A] = E[V] - E[A] = n^2/2m - n^2/4m = n^2/4m$$

Mas para qualquer variável aleatória X em um espaço de probabilidades, temos que se $E[X] = x$ então então $\Pr(X \geq x) > 0$. Imagine um espaço amostral de idades de pessoas. Se a esperança, ou seja, a média das idades é 15, com certeza alguém tem 15 anos ou mais. Usando esse fato e a esperança obtida com relação ao algoritmo acima, podemos concluir então que qualquer grafo com n vértices e m arestas possui um conjunto de vértices independentes de tamanho maior ou igual a $n^2/4m$.

Método de Monte Carlo

O método de **Monte Carlo**[MU05] consiste em estimar algum valor usando um espaço amostral e uma simulação nesse espaço. Se quisermos, por exemplo, estimar a quantidade de vértices de cor azul em um grafo bicolorido, podemos sortear um número suficientemente grande de vértices e usar a proporção obtida de vértices de cor azul. A primeira vista isso pode parecer um modo inocente de estimar valores, mas podemos obter algoritmos muito robustos usando o método de Monte Carlo.

Exemplo: Conjuntos de vértices independentes

Podemos encontrar o número de conjuntos de vértices independentes que existem em um grafo usando o Método de Monte Carlo. Seja $e_1, e_2, \ldots e_m$ uma ordenação das arestas do grafo G e seja G_i o grafo formado só com as primeiras i arestas. Seja $\Omega(G)$ o espaço formado por todos os conjuntos independentes de G. Queremos saber $|\Omega(G)|$. Podemos expressar da seguinte forma:

$$|\Omega(G)| = |\Omega(G_m)| = \frac{|\Omega(G_m)|}{|\Omega(G_{m-1})|} \times \frac{|\Omega(G_m - 1)|}{|\Omega(G_{m-2})|} \times \ldots \times \frac{|\Omega(G_1)|}{|\Omega(G_0)|} \times |\Omega(G_0)|$$

Notem que G_0 é um grafo sem arestas e então, qualquer subconjunto de vértices é independente. Portanto $|\Omega(G_0)| = 2^n$. Então temos:

$$|\Omega(G)| = 2^n \prod_{i=1}^{m} \frac{|\Omega(G_i)|}{|\Omega(G_{i-1})|}$$

Usaremos o algoritmo (que é um método de Monte Carlo) abaixo para estimar $\frac{|\Omega(G_i)|}{|\Omega(G_{i-1})|}$ e então teremos uma estimativa de $|\Omega(G)|$.

```
ESTIME (G_i, G_{i-1})
1  x ← 0
2  repita M vezes
3     C ← sorteie um conjunto de vértices independentes de G_{i-1}
4     se C é um conjunto independente de G_i
5        faça x++
5  devolva x/M
```

Para se ter sucesso em nossa estimativa, o valor de M deve ser suficientemente grande. Além disso, o sorteio do conjunto de vértices independentes de G_{i-1} merece atenção pois não pode afetar muito a eficiência do algoritmo e deve ser um sorteio de qualidade. Esse tipo de sorteio é tipicamente implementado com auxílio de uma Cadeia de Markov.

Referências

[MR96] Rejeev Motwani and Prabhakar Raghavan. Randomized algorithms. *ACM Compunting Surveys, Vol 28, No 1*, 1996.

[MU05] Michael Mitzenmacher and Eli Upfal. *Probability and Computing - Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[Spe95] Joel Spencer. Modern probabilistic methods in combinatorics. *British Combinatorial Conference, Stirling*, 1995.