Lista oficial de tarefas de grafos e otimização combinatória

Roberto Imbuzeiro Oliveira*

31 de Outubro de 2017

1 Observações

1. Prazo para entrega: 18/11/2017.
2. Como na lista do LHF, vocês devem escrever relatórios contando experimentos e etc.
3. Não esqueça de citar fontes e colegas que o ajudaram.

2 O problema do aluno de graduação

O problema desta seção é um caso típico do problema de ordenar tarefas, que têm as seguintes características:

- Há pares de tarefas (i, j) tais que i deve ser realizada antes de j.
- Tarefas que não são deste tipo podem ser realizadas em paralelo, mas há um limite para quantas podem ser realizadas ao mesmo tempo.

PROBLEMA: uma universidade tem um conjunto V_0 de n disciplinas. Cada uma destas disciplinas dura um semestre e vale $c(i) \in \mathbb{N}$ "créditos". Além disso, há disciplinas j que têm como pré-requisitos outras disciplinas i, que devem ser cursadas em um semestre anterior ao de j. Para facilitar, supomos que todas as disciplinas são oferecidas em todos os semestres e que não há "eletivas"

Queremos um algoritmo que toma como entrada um subconjunto $V \subset V_0$ de disciplinas a serem cursadas e um número L, que é um limite superior do número de créditos por semestre. A saída do algoritmo deve ser:

- Um número $k \in \mathbb{N}$ que diz quantos semestres um aluno precisaria para cursar todas as disciplinas em V.
- Para cada $t = 1, 2, \ldots, k$, o conjunto $D_t \subset V$ das disciplinas a serem cursadas no semestre t. Deve ser satisfeita a restrição do limite do número de créditos, que é:

$$\forall t \in \{1, 2, \ldots, k\} : \sum_{i \in D_t} c(i) \leq L.$$

Obviamente, gostaríamos que k fosse o menor possível, dadas as condições.

*IMPA, Rio de Janeiro, RJ, Brazil, 22430-040.

Exercício 1 Formalize o problema de maneira mais precisa. Que condições L, os números $c(i)$ e o conjunto de "relações de pré-requisito" devem satisfazer para que o problema seja solúvel?

Exercício 2 Prove que, quando $L = \infty$ (ou seja, não há limite de créditos por período), podemos encontrar um algoritmo de tempo polinomial que resolve este problema.

Exercício 3 O problema para L geral é bastante difícil. Procure um algoritmo "ótimo" para ele, ou, se não conseguir, tente uma ou duas heurísticas e veja qual se sai melhor em termos de tempo e número de semestres. *(A única restrição sobre as heurísticas é que elas devem retornar resultados corretos; fora isso, elas não precisam achar o menor k possível.)*

3 Caminhos mínimos e a mudança de uma aresta

São dados um grafo orientado $D = (V, E)$ com $|V| = n$ vértices e valores $c(e) \geq 0$ para cada aresta orientada $e \in E$.

Lembre que o problema de caminhos mínimos com fonte $s \in V$ consiste em encontrar, para cada $v \in V \backslash \{s\}$, o caminho entre s e v que minimiza a soma dos custos sobre suas arestas. Já o problema de caminhos mínimos entre todos os pares é encontrar os caminhos de menor custo entre quaisquer $w, v \in V$.

Agora imagine que um (e apenas um) dos custos das arestas orientadas é reduzido. Não é difícil encontrar exemplos em que os caminhos mínimos são drasticamente alterados por esta mudança e outros exemplos em que os caminhos não são alterados em nada. (Note que falamos de alterações dos caminhos, não de seus custos!)
PROBLEMA: Apresente uma forma "econômica" de recalcular as distâncias mínimas sem refazer todas as operações, no caso de caminhos entre todos os vértices. Mostre que ela leva tempo $O(n^2)$. (Isso pode requerer armazenar mais dados sobre a matriz de distâncias.)

4 Fluxos e emparelhamentos

Considere o problema de Hall: dado um grafo bipartido $G = (V_1 \cup V_2, E)$ com $|V_1| = |V_2| = n > 0$ e $V_1 \cap V_2 = \emptyset$, queremos achar um emparelhamento maximal neste grafo. Vimos em sala que tal emparelhamento existe sempre que para qualquer $A \subset V_1$ temos $|N(A)| \geq |A|$, onde

$$N(A) := \{b \in V_2 \,:\, b \sim a$$

Vimos em sala que podemos reduzir este problema a um problema de fluxo máximo e resolvê-lo via um algoritmo do tipo Ford-Fulkerson. Apresente uma implementação simplificada do algoritmo FF que serve exclusivamente para resolver o problema de Hall. Tente tornar seu algoritmo o mais conceitualmente simples possível e dê uma explicação de porque ele funciona.