

PUC-Rio
Departamento de Informática
Prof. Marcus Vinicius S. Poggi de Aragão
Período: 2003.2
15 de outubro de 2003
Horário: 2as-feiras e 4as-feiras de 19 às 21 horas

PROJETO E ANÁLISE DE ALGORITMOS (INF 1309)

2ª Lista de Exercícios

1. Considere um mapa rodoviário e um motorista que tem que ir do vértice s ao t . O mapa é representado pelo grafo $G = (V, E)$, não-orientado onde os valores associados às arestas $e \in E$, h_e , correspondem às altitudes das estradas correspondentes aos trechos. O motorista não gosta de altitude e quer fazer o caminho que minimiza a maior altitude que ele vai passar. Proponha um algoritmo que encontre esse caminho (Dica: utilize um algoritmo de árvore geradora mínima). Qual a complexidade do seu algoritmo? Tem que ser $O(n^2)$.
2. Uma floresta é um grafo sem ciclos. Seja um grafo $G = (V, E)$, não-orientado onde os pesos das arestas $e \in E$ são dados por w_e . Proponha um algoritmo para encontrar uma floresta com k arestas, $k \leq n - 1$, de peso total mínimo. Sua complexidade tem que ser $O(n^2)$ ou inferior.
3. Uma 1-árvore geradora mínima pode ser obtida adicionando-se à árvore geradora mínima a menor aresta que não pertence à mesma. Suponha agora que essa aresta adicional precise estar conectada a um dado vértice v , isto é o único ciclo da 1-árvore deve conter o vértice v e após a remoção de uma aresta ligada ao vértice v deve restar uma árvore. Proponha um algoritmo para encontrar uma 1-árvore geradora mínima com esta restrição.
4. Considere que a árvore geradora de peso mínimo (AGM) de $G = (V, E)$, não-orientado onde os pesos das arestas $e \in E$ são dados por w_e , é conhecida. Considere agora que um novo vértice foi acrescentado à G com arestas para todos os vértices em V . Proponha um algoritmo para encontrar a nova AGM. Seu algoritmo deve executar em $O(n \log n)$ ou menos. Tente encontrar um algoritmo $O(n)$, existe.
5. Seja o grafo $G = (V, E)$ onde $V = \{1, 2, 3, 4, 5, 6\}$ e $E = \{(1, 2, 5), (1, 3, 2), (1, 4, 2), (2, 5, 2), (3, 4, 1), (3, 6, 6), (4, 2, 1), (4, 6, 7), (5, 4, 1), (5, 6, 3)\}$, onde os trios (u, v, l) indicam os vértice de partida, de chegada e a distância do arco.
 - (a) Aplique o algoritmo de Ford-Bellman ("Correção de Rótulos") para encontrar os c.m.c.'s do vértice 1 aos demais.
 - (b) Aplique o algoritmo de Dijkstra para encontrar os c.m.c.'s do vértice 1 aos demais.
 - (c) Suponha agora que os arcos não tem orientação, ou seja se existe o arco (u, v, l) , também existe o arco (v, u, l) , e aplique o algoritmo de Kruskal para encontrar a Árvore Geradora Mínima de G .
 - (d) Ainda supondo que os arcos não tem orientação, aplique o algoritmo de Prim para encontrar a Árvore Geradora Mínima de G .

- (e) Ainda supondo que os arcos não tem orientação, aplique o algoritmo de Borůvka para encontrar a Árvore Geradora Mínima de G .
- (f) Ainda supondo que os arcos não tem orientação, aplique o algoritmo de Floyd-Warshall para encontrar os c.m.c.'s entre todos os pares de vértices.

6. Considere o Algoritmo de Dijkstra para encontrar o Caminho-mais-curto (c.m.c.) entre o vértice s e os outros vértices de um grafo $G = (V, E)$, orientado onde as distâncias dos arcos $e \in E$ é dada por l_e .

Algoritmo Dijkstra (s - fonte)

Passo 0: *Inicialização*

Seja S o conjunto de vértices com o caminho mais curto a partir de s já determinado, e \bar{S} seu complemento ($\bar{S} = V - S$).

$S \leftarrow \emptyset$

$d(i) \leftarrow +\infty \forall i \in V$;

$d(s) \leftarrow 0$; $pred(s) \leftarrow 0$;

Passo 1: *Iteração*

Enquanto $S \subset V$ faça

1.1 Encontre $v \in \bar{S}$ t.q. $d(v) = \min_{w \in \bar{S}} d(w)$

1.2 $S \leftarrow S \cup \{v\}$; $\bar{S} \leftarrow \bar{S} - \{v\}$;

1.3 Para todo $w \in \Gamma^+(v)$

Se $d(v) + l_{vw} < d(w)$

então $d(w) \leftarrow d(v) + l_{vw}$; $pred(w) \leftarrow v$;

- (a) Analise a complexidade do algoritmo acima.
- (b) Modifique este algoritmo para que encontre a Árvore Geradora Mínima (no lugar do c.m.c.'s de s aos demais vértices)
- (c) Como você modificaria o algoritmo acima para que sua complexidade fosse $O(m \log n)$? Que passos seriam alterados? Como ficariam estes passos?
- (d) Construa um grafo com alguns arcos e com $l_e < 0$, mas sem ciclos de comprimento negativo, onde o algoritmo de Dijkstra funciona **corretamente**.
- (e) Construa um grafo com alguns arcos e com $l_e < 0$, mas sem ciclos de comprimento negativo, onde o algoritmo de Dijkstra **falha** em encontrar o c.m.c de s aos outros vértices.

7. Prove que é verdade ou que é falso (nesse caso apresentando um contra-exemplo).

- (a) Se todas as distâncias dos arcos são diferentes, então a árvore de c.m.c. (de s aos outros vértices do grafo) é **única**.
- (b) Considere a distância dos c.m.c. de s aos outros vértices do grafo. Se a distância de cada arco é aumentada de k unidades (ou seja $l'_{uv} \leftarrow l_{uv} + k$ para todo arco $e = (u, v) \in E$), as distâncias dos c.m.c. aumentam de um múltiplo de k .
- (c) Se forem retiradas a orientações dos arcos de um grafo orientado G (i.e. passa ser possível passar nos dois sentidos), as distâncias dos c.m.c.'s permanecem as mesmas.

- (d) Entre todos os c.m.c.'s existentes entre dois vértices em um grafo, o algoritmo de Dijkstra sempre acha o c.m.c. que possui o menor número de arestas.
8. Considere o c.m.c. entre um par de vértices em um grafo orientado $G = (V, E)$, s e t por exemplo. Um arco vital com respeito a esse c.m.c. é um arco que, se retirado do grafo, a distância do c.m.c. de s a t aumenta. O arco mais vital é aquele cuja retirada causa o maior aumento.
- Proponha um algoritmo para encontrar o arco mais vital dados $G = (V, E)$, s e t .
 - Analise a complexidade do seu algoritmo.
9. Seja $G = (V, E)$ um grafo orientado e acíclico, com distâncias l_e para $e \in E$, e s um vértice a partir do qual existe caminho para todos os demais vértices do grafo. Proponha um algoritmo com complexidade $O(m)$, $m = |E|$, para encontrar os c.m.c.'s de s aos outros vértices do grafo. (Dica: use ordenação topológica.)
10. (5.16) Suponha que foram obtidos os c.m.c.'s de s aos outros vértices de G e a árvore de c.m.c. é conhecida. Suponha agora que as distâncias de todos os arcos deve ser aumentada de k unidades (ou seja $l'_{uv} \leftarrow l_{uv} + k$ para todo arco $e = (u, v) \in E$). Proponha um algoritmo $O(m)$ para obter os novos c.m.c.'s.