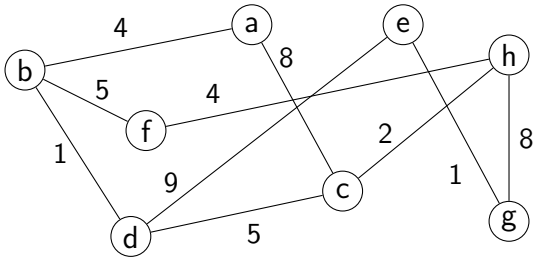
# Estruturas de Dados e Algoritmos II
# Resolução da 2ª Frequência

Departamento de Informática
Universidade de Évora

23 de Maio de 2019

**1.** Seja $G_1$ o grafo seguinte:




(a) [3 valores] Apresente uma ordem pela qual os arcos poderiam ser considerados durante uma aplicação do algoritmo de Kruskal ao grafo, indicando quais seriam incluídos na árvore de cobertura mínima e quais não seriam.

RESPOSTA

Uma ordem possível, escolhendo por ordem lexicográfica os arcos com o mesmo peso, seria:

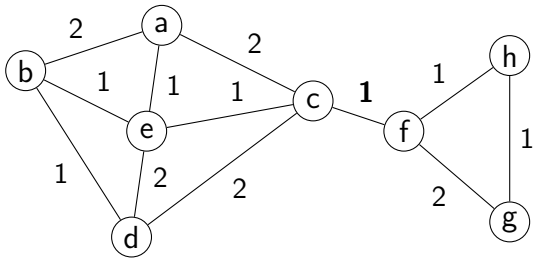
| Ordem | 1º | 2º | 3º | 4º | 5º | 6º | 7º | 8º | 9º |
|---|---|---|---|---|---|---|---|---|---|
| Arco | (b,d) | (e,g) | (c,h) | (a,b) | (f,h) | (b,f) | (c,d) | (a,c) | (g,h) |
| Incluído? | Sim | Sim | Sim | Sim | Sim | Sim | Não | Não | Sim |

(b) [2,5 valores] Apresente um exemplo de um grafo que corresponda ao melhor caso do funcionamento do algoritmo de Kruskal, e um grafo que corresponda ao pior caso. Explique por que razão cada um desses grafos corresponde ao caso respectivo.

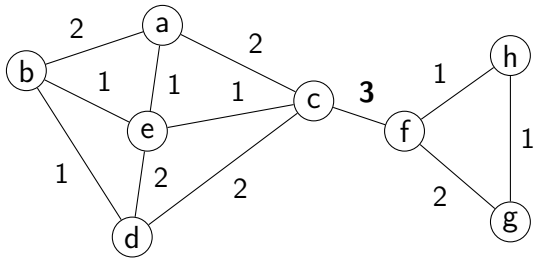
RESPOSTA

Tendo em conta que o algoritmo considera os arcos por ordem não decrescente de peso e termina assim que estão escolhidos os $|V| - 1$ arcos para a árvore de cobertura mínima, o melhor caso dá-se quando esses são os primeiros arcos considerados, por terem pesos inferiores aos restantes, e o pior quando é necessário considerar todos os arcos do grafo, porque o arco com maior peso integrará qualquer árvore de cobertura mínima do grafo.

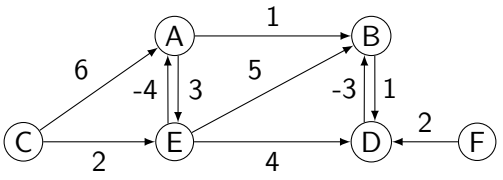
Melhor caso




Pior caso



**2.** Seja $G_2$ o grafo orientado seguinte:




(a) [2 valores] Apresente as componentes fortemente conexas de $G_2$.

RESPOSTA

{A, E} {B, D} {C} {F}

(b) [3 valores] Suponha que o algoritmo de Bellman-Ford é alterado do seguinte modo: a distância (pesada) de um vértice à origem é inicializada a $-\infty$; a distância um vértice à origem é actualizada sempre que se encontra um caminho com maior peso do que o encontrado até essa altura (i.e., a comparação na função RELAX é invertida).

Apresente, para cada vértice do grafo, a sequência de distâncias calculadas quando esta variante do algoritmo é aplicada a $G_2$ a partir do vértice C.

Considere os arcos do grafo por ordem lexicográfica. Nesta ordem, se $(u_1, v_1)$ e $(u_2, v_2)$ são dois arcos do grafo, $(u_1, v_1) < (u_2, v_2)$ sse $u_1 < u_2$ ou $u_1 = u_2 \wedge v_1 < v_2$.

RESPOSTA

| Vértice | Distâncias calculadas/predecessor |
|---|---|
| A | $-\infty$/NIL 6/C |
| B | $-\infty$/NIL 7/E 14/E |
| C | 0/NIL |
| D | $-\infty$/NIL 6/E 8/B 13/E 15/B |
| E | $-\infty$/NIL 2/C 9/A |
| F | $-\infty$/NIL |

(c) [0,5 valores] Os caminhos calculados, na alínea anterior, são os com maior peso a partir do vértice C?

RESPOSTA

Sim.

(d) [1 valor] Para esta variante do algoritmo, há algum tipo de ciclos que seja problemático? Justifique a sua resposta; se respondeu que sim, diga que tipo de ciclos pode causar problemas e porquê.

RESPOSTA

Os ciclos problemáticos são aqueles com peso positivo. Na presença de um destes ciclos, poderá não haver caminhos com maior peso para um ou mais nós, visto que se obtém um caminho com maior peso percorrendo mais uma vez o ciclo.

**3.** [4 valores] Seja $V = (v_1\, v_2\, \ldots\, v_n)$ uma sequência não vazia de naturais positivos. Pretende-se escolher um subconjunto dos valores desta sequência de acordo com as seguintes regras:

- se $v_1$ for escolhido, os $v_1$ elementos seguintes são retirados da sequência;
- se $v_n$ for escolhido, os $v_n$ elementos anteriores são retirados da sequência;
- pode não se escolher nem $v_1$ nem $v_n$ e optar por, simplesmente, retirar ambos da sequência.

Estas regras são aplicadas repetidamente enquanto restar algum valor na sequência. Na aplicação das duas primeiras, se o número de valores na sequência, depois da escolha, for inferior ou igual a $v_1$ ou $v_n$, respectivamente, a sequência ficará vazia.

Como exemplo, seja $V$ a sequência $(3\,5\,4\,2\,1)$. Se for escolhido primeiro o valor 3, restará na sequência só o valor 1. Se for escolhido primeiro o valor 1, a sequência ficará reduzida a $(3\,5\,4)$ e, agora, quer seja o 3 o próximo escolhido, quer seja o 4, a sequência ficará vazia. Depois da escolha do valor 1, também seria possível retirar o 3 e o 4 da sequência, obtendo a sequência $(5)$, na qual só poderia ser escolhido o 5, antes de a sequência ficar vazia. Nas duas últimas escolhas feitas, a soma dos valores escolhidos é $1 + 5 = 6$, que é o maior valor possível para esta sequência.

Apresente *uma função recursiva* que, dada uma sequência não vazia de naturais positivos $V = (v_1\, v_2\, \ldots\, v_n)$, calcula o maior valor possível da soma dos elementos escolhidos de acordo com as regras acima.

Indique claramente o que representa cada uma das variáveis que utilizar e explicite a chamada inicial, que resolve o problema completo. (Note que não é pedido que escreva código.)

RESPOSTA

Sejam

- $n > 0$ um natural positivo;
- $V = (v_1\ v_2\ \ldots\ v_n)$ uma sequência de naturais positivos; e
- $1 \leq i, j \leq n$ os índices de duas posições de $V$.

$S_V[i, j]$, a função que calcula a maior soma dos valores escolhidos, de acordo com as regras do enunciado, na subsequência $(v_i\ \ldots\ v_j)$, é definida como:

$$S_V[i,j] = \begin{cases} 0 & \text{se } i > j \\ \max\left\{v_i + S_V[i+1+v_i, j],\ v_j + S_V[i, j-1-v_j],\ S_V[i+1, j-1]\right\} & \text{se } i \leq j \end{cases}$$

A chamada inicial da função, que corresponde à aplicação à sequência $V$ completa, é $S_V[1, n]$.

**4.** [4 valores] O Grupo Desportivo e Cultural da Gafanha (GDCG) tem em preparação um evento que inclui uma caminhada de vários dias, entre dois pontos já escolhidos, e está com problemas na definição do trajecto a seguir. O GDCG já identificou centenas de pontos por onde é possível passar, assim como trajectos directos entre eles. Como as subidas são as partes das caminhadas em que as pessoas mais se cansam e mais se queixam, o GDCG tem, também, para cada um desses pontos, a sua altitude, em metros, e atribuiu, a cada trajecto entre dois pontos, um "factor subida", que possui um valor inteiro não negativo. O "factor subida" só se aplica quando um trajecto é percorrido de um ponto mais baixo para um ponto mais alto, e não quando ele é percorrido no sentido contrário.

O objectivo do GDCG é utilizar esta informação para desenhar um trajecto para a caminhada que minimize o total do "factor subida", ou seja, para o qual a soma dos "factores subida" dos trajectos directos percorridos é a menor possível. Depois de determinado o trajecto, o GDCG também gostaria de saber quantos metros se subirá, *no total*, ao longo da caminhada.

Na presença de toda a informação disponível, como poderia determinar um trajecto para a caminhada que satisfaça o objectivo do GDCG? E como calcularia o total subido? Descreva detalhadamente como modelaria os dados e como os representaria, e diga que algoritmo(s) utilizaria, porquê, e como obteria o resultado pretendido.

RESPOSTA

## Modelo

Os dados poderão ser modelados através de um grafo orientado pesado, onde:

- os **vértices** correspondem aos pontos por onde é possível passar (incluindo os pontos de partida e de chegada);
- a cada trajecto directo entre dois pontos corresponderão 2 **arcos**, entre os vértices correspondentes, um em cada sentido; e
- o **peso** do arco cuja origem é o vértice correspondente ao ponto com menor altitude é o "factor subida" do trajecto, e o do arco no sentido oposto é zero. (Se os dois pontos estão à mesma altitude, ambos os arcos terão peso zero.)

## Algoritmos

No grafo definido, um caminho (pesado) mais curto será um caminho a que corresponderá a menor soma possível dos "factores subida" dos trajectos directos que o compõem.

Para determinar um trajecto para a caminhada, aplica-se o algoritmo de Dijkstra (o grafo não é acíclico, mas não tem pesos negativos), a partir do vértice correspondente ao ponto de partida. O algoritmo será alterado para terminar assim que é encontrado um caminho mais curto para o vértice correspondente ao ponto de chegada.

O total subido calcula-se percorrendo os predecessores de cada vértice pertencente ao caminho, calculados durante a aplicação do algoritmo de Dijkstra, a partir do ponto de chegada. Para cada vértice visitado, se a altitude do ponto que lhe corresponde é inferior à do correspondente ao vértice que precede, a diferença das altitudes é adicionada ao total subido.

Como o algoritmo usado é o de Dijkstra, o grafo será representado através das suas listas de adjacências.