Departamento de Ciência de Computadores FCUP

Desenho e Análise de Algoritmos (CC2001) 2017/18

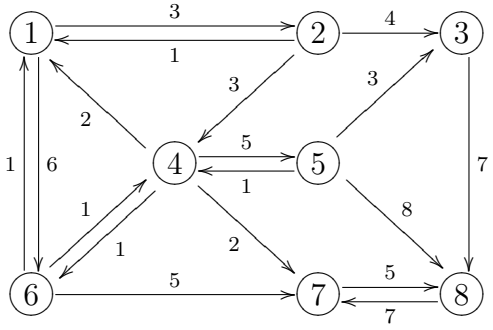
1º Teste (11.11.2017) *duração: 2h30m*

Cotação: $4 \times 2 + 2 + 5 + 5$

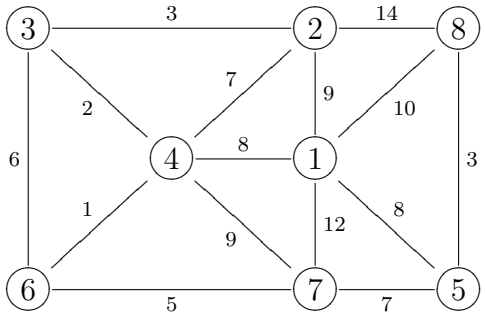
N.º Nome

Das perguntas 1, 2 e 3, deve responder a DUAS (se responder a mais, **3** não será cotada.)

1. Descreva os passos principais do algoritmo de Kosaraju-Sharir e aplique-o para obter o conjunto de nós que definem cada uma das componentes fortemente conexas do grafo G representado. Assuma que a lista dos adjacentes de cada nó em G está ordenada por ordem crescente de identificador. Apresente o conteúdo das estruturas de dados (usadas no algoritmo) no final de cada um dos passos principais.



2. Pretendemos obter uma árvore de suporte com peso mínimo para o grafo representado. Ilustre a aplicação do algoritmo de Prim.



- Deve tomar o **nó 1** como raiz da árvore de suporte e apresentar todos os **valores intermédios** para a distância e o pai de cada nó.
- Deve indicar também o **estado da fila de prioridade** (*heap* de mínimo) no início de cada iteração, desenhando a árvore correspondente com nós etiquetados por pares "identificador-valor". Em casos de empate, opte pelo nó que tem o identificador **menor**.

3. Aplique o algoritmo de Dijkstra ao grafo representado em **2.**, com origem $s = \mathbf{6}$, para obter um vetor p em que $p[v]$ indica o nó que precede o nó v no caminho mínimo de s para v que se encontrou, para cada $v \neq s$. Deve determinar também as distâncias correspondentes. Em casos de empate, opte pelo nó que tiver o identificador **menor**. Indique todos **os valores intermédios** e a ordem pela qual os nós saem da fila de prioridade (mas não desenhe a fila). Assuma que cada ramo do grafo dado é substituído por dois orientados, mas não desenhe o grafo assim (para não sobrecarregar a figura).

---

4. Usando **diretamente** a definição das classes $\Theta(n^2)$, $O(n \log_2 n)$ e $\Omega(n + 3n \log_2 n)$, prove que:

a) $2n^2 - 5n - 2 \in \Theta(n^2)$. Pode assumir que n é suficientemente grande para que $2n^2 - 5n - 2 > 0$.

b) $O(n \log_2 n) \cap \Omega(n + 3n \log_2 n) \neq \emptyset$.

(Continua v.p.f.)

Das perguntas 5 e 6 deve responder a UMA (se responder a mais, **6** não será cotada.)

5. Seja $G = (V, E, d)$ um grafo dirigido com valores nos arcos dados pela função $d : E \to \{0, 1, \varepsilon\}$. A representação do grafo é baseada em *listas de adjacências* (suponha que ε é denotado pelo carater '#'). Os vértices são identificados por inteiros consecutivos, de 1 a $|V|$. Considere um tipo abstrato de dados (ADT) para representação de subconjuntos de $\mathcal{U}_n = \{1, 2, \ldots, n\}$, com funções: MKEMPTYSET(n), que retorna um conjunto vazio, INSERT(x, k), que insere k no conjunto x, UNION(x, y), que obtém em x a união de dois conjuntos x e y, e SET2LIST(x) que retorna a lista (*linked list*) dos elementos do conjunto x.

Dado um conjunto $S \subseteq V$ e dado $a \in \{0, 1\}$, queremos determinar o subconjunto de V formado por todos os nós que são acessíveis de nós em S por percursos com exatamente um símbolo a, sendo os restantes ramos do percurso etiquetados por ε, se existirem. Usando pseudocódigo, apresente um **algoritmo** que resolva o problema e que aplique pesquisa em largura (ou em profundidade) a partir de cada nó $s \in S$. Justifique sucintamente a sua **correção** e **complexidade** temporal (use $|S|$, $|V|$ e $|E|$ para a definir). Assuma que as operações referidas acima têm complexidade $\Theta(n)$, $O(1)$, $\Theta(|y|)$ e $O(n)$, respetivamente.

6. Pretendemos atribuir postos de trabalho a candidatos. Cada candidato está **identificado** por um inteiro (igual ao número de ordem da sua candidatura) e tem **uma classificação** (que é um inteiro positivo que resultou da ponderação de alguns critérios). Os candidatos são colocados nos postos por ordem decrescente de classificação. Não há dois candidatos com a mesma classificação. Cada candidato indicou as suas preferências por postos de trabalho, por ordem decrescente de preferência. Cada posto de trabalho tem disponíveis um certo número de vagas. Há que colocar os candidatos respeitando as suas preferências e classificações. Escreva, em pseudocódigo, uma função para resolver o problema. Suponha que a informação sobre os candidatos está disponível num *array* C, em que $C[i].nota$ dá o valor da classificação do candidato i e $C[i].prefs$ dá a sua lista de preferências (ordenada). O vetor $vagas$ tem em $vagas[j]$ o número de vagas do posto j, para $1 \leq j \leq m$. Existem n candidatos e m postos. O algoritmo deve ser suportado por uma "***heap* de máximo**" de onde se extrai o identificador do candidato que é colocado em cada iteração. A função produz um vetor p em que $p[i]$ indica o posto em que i ficou (e tem -1 se o candidato i não ficar colocado). Justifique sucintamente a **correção** e a **complexidade** temporal do algoritmo que escreveu. Use m, n, e o comprimento total das listas de preferências para definir a complexidade.

---

7. A função INVERTER(v, i, j) inverte o segmento $v[i], v[i+1], \ldots, v[j]$ de um vetor v de n elementos, **quando** $0 \leq i \leq j < n$. Se o estado inicial desse segmento for $(a_i, a_{i+1}, a_{i+2}, \ldots, a_{j-2}, a_{j-1}, a_j)$ então, após a chamada da função, será $(a_j, a_{j-1}, a_{j-2}, \ldots, a_{i+2}, a_{i+1}, a_i)$.

INVERTER(v, i, j)
1. Enquanto $(i < j)$ fazer
2. $\quad aux \leftarrow v[i]; \quad v[i] \leftarrow v[j]; \quad v[j] \leftarrow aux;$
3. $\quad j \leftarrow j - 1; \quad i \leftarrow i + 1;$

a) Indique uma condição sobre o estado das variáveis (quando se está a testar a condição de paragem do ciclo pela k-ésima vez, para $k \geq 1$) que permite concluir que INVERTER(v, i, j) está correta.

b) Usando indução matemática, demonstre a condição que enunciou.

c) Explique como é que, usando a condição que enunciou, se conclui que a função está correta.

d) Assuma que n é par e $n \geq 2$. Considere o bloco de código seguinte.

$i \leftarrow 0;$
Enquanto $(i < n - 1 - i \wedge v[i] < v[n - 1 - i])$ fazer
$\quad$ INVERTER$(v, i, n - 1 - i);$ $i \leftarrow i + 1;$

Justifique que a complexidade temporal assintótica do bloco é $O(1)$ no melhor caso e $\Omega(n^2)$ no pior caso. O que é correto afirmar sobre a complexidade desse bloco: que é $O(n^2)$? $\Theta(n^2)$? $\Omega(n^2)$? Explique detalhadamente, começando por definir um modelo de custos para as instruções elementares e por provar que a complexidade de INVERTER(v, i, j) é $\Theta(j - i + 1)$, para $i < j$. **(Fim)**