Universidade Federal de Pernambuco
Centro de Informática
Bach. em Ciência da Computação

IF672cc Algoritmos e Estr. de Dados 2017.2
Prof. Paulo Fonseca

Segunda Prova — 05 de Dezembro de 2017

QUESTÃO 1 (1,5pt)

Considere a estrutura de dados ***union-find*** com as heurísticas de ***união ponderada*** e ***compressão de caminhos*** sobre um conjunto $\mathcal{A} = \{0, \ldots, n-1\}$, implementada como um array $P = (p_0, \ldots, p_{n-1})$ onde p_i indica o "nó pai" de i na floresta correspondente ($p_i = i \iff i$ é uma raiz). Nesse caso, quais dos arrays abaixo ***não*** podem ocorrer como consequência de uma sequência de operações ***union/find*** para $n = 10$? Indique claramente a sua resposta e justifique-a representando cada uma das alternativas escolhidas na forma de floresta, circulando a(s) componente(s) que não podem ocorrer.

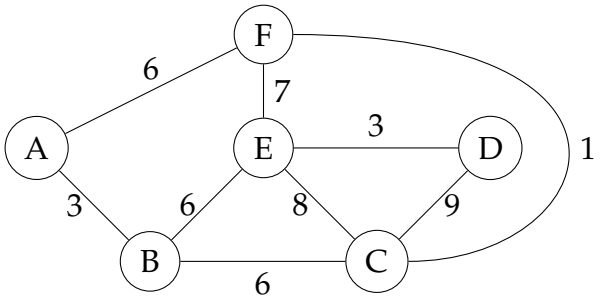
a) (0 1 2 3 4 5 6 7 8 9)
b) (7 3 8 3 4 5 6 8 8 1)
c) (0 9 2 9 4 4 4 1 9 9)
d) (6 3 8 0 4 5 6 9 8 1)
e) (9 6 2 6 1 4 5 8 8 9)

QUESTÃO 2 (3,0 pt)

a) Complete o diagrama

Iter. #	Peso, Precursor A	B	C	D	E	F
0	0, –	∞, ?	∞, ?	∞, ?	∞, ?	∞, ?
1	(0, –)	3, A	...	...	...	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮

correspondente à execução do ***Algoritmo de Dijkstra*** sobre o grafo a seguir.



b) As arestas que ligam cada vértice ao seu precursor formam uma árvore geradora. A árvore geradora resultante dessa execução é uma MST? Justifique através de uma comparação com a a árvore encontrada pelo ***Algoritmo de Prim***.

QUESTÃO 3 (2,0pt)

Considere a codificação do mapa do Brasil a seguir como um grafo não dirigido nos qual cada estado é representado por um nó e dois estados vizinhos são ligados por uma aresta.

Qual o percurso adequado nesse grafo para determinar a menor quantidade de estados $\delta(s,t)$ a serem visitados numa viagem de Pernambuco ($s = 14$) a cada um dos demais estados, $\delta(s = 14, t)$ para $t = 1, \ldots, 27$?

a) Ilustre a execução desse percurso exibindo a árvore correspondente. b) Enumere os estados de acordo com esse percurso e c) Indique o menor número de estados visitados numa viagem de Pernambuco ($s = 14$) ao Paraná ($t = 25$).

■ QUESTÃO 4 (1,5pt)

Uma máquina de vendas automática necessita fornecer um troco T utilizando a menor quantidades possível de moedas de M valores diferentes $V = (v_0, \ldots, v_{M-1})$ supondo, por simplicidade, que a máquina dispõe de uma quantidade ilimitada de moedas de cada valor. Para tanto, o programador da máquina concebeu o seguinte algoritmo baseado na técnica de

_____(a).

Seja $Q(m,t)$ a menor quantidade de moedas necessária para fornecer um troco t usando apenas moedas de valor $V = (v_0, \ldots, v_{m-1})$. Se $t = 0$, temos

$Q(m,t) =$ _____(b) p/ qualquer m.

Se $m = 0$ e $t > 0$, definimos $Q(m,t) = \infty$, pois não é possível oferecer o troco.

Se $m, t > 0$, então a solução ótima $Q(m,t)$ pode ou não incluir uma moeda de valor $V[m-1]$. Se ela ***não inclui*** uma moeda desse valor, então temos

i) $Q(m,t) =$ _____(c).

Essa é a única hipótese se $t < V[m-1]$.

Se, por outra, a solução ótima ***inclui*** uma moeda de valor $V[m-1]$, então teremos

ii) $Q(m,t) =$ _____(d).

Sumarizando os casos (i) e (ii) acima, temos

$Q(m,t) =$ _____(e).

O algoritmo consiste em calcular $Q(m,t)$ par valores crescentes de $m = 0, \ldots, M$ e, para cada um deles, variando $t = 0, \ldots, T$, armazenando os valores calculados numa tabela.

Seguindo esse algoritmo, descobrimos que são necessárias no mínimo cinco moedas para fornecer um troco de $T = 4,80€$, usando moedas de valores $V = (1¢, 2¢, 5¢, 10¢, 20¢, 50¢, 1€ = 100¢, 2€ = 200¢)$ o que, nesse caso, é o mesmo número obtido se adotarmos uma estratégia gulosa.

(f) Esse algoritmo, então, dá sempre o mesmo resultado do algoritmo guloso? Se sim, justifique sucintamente a razão (máx 10 linhas); se não, forneça um contraexemplo $T, V = (v_0, \ldots, v_{n-1})$.

■ QUESTÃO 5 (2,0pt)

No ***Problema das n-rainhas***, temos um tabuleiro de xadrez de tamanho $n \times n$, sobre o qual desejamos posicionar n rainhas de forma que nenhuma delas seja "ameaçada" por nenhuma das demais, ou seja, não pode haver duas peças na mesma linha, coluna ou diagonal. Para um tabuleiro padrão 8×8, existem mais de $4,4$ bilhões de possibilidades de posicionamento das peças, porém a maioria delas não é válida. Entretanto, supondo as linhas do tabuleiro numeradas $0, \ldots, n-1$ de cima para baixo, e as colunas numeradas no mesmo intervalo da esquerda para a direita, esse problema pode ser facilmente resolvido com um algoritmo de ***backtracking***, escolhendo progressivamente as colunas da rainhas das linhas $i = 0, \ldots, n-1$ (cada linha deve ter exatamente uma rainha), tentando cada uma das possibilidades $j = 0, \ldots, n-1$, e retrocedendo assim que um conflito é encontrado.

Determine o menor valor $n_{min} \geq 2$ para o qual o problema tem uma solução, ilustrando a execução do algoritmo acima através da sua árvore de busca para $n = 2, 3, \ldots, n_{min}$

■ ■ ■