Universidade Federal de Minas Gerais
Programa de Pós-Graduação em Ciência da Computação
Exame de Qualificação 1º Estágio
2º Semestre de 2014

Área: Teoria: Estrutura de Dados, Projeto e Análise de Algoritmos, Técnicas de Programação, Pesquisa e Ordenação

Em 12/08/2014, 10:00 horas

Prova individual sem consulta com duração de 2 horas

Observações:

1. A prova deve ser resolvida no próprio caderno de questões.
2. As questões desta prova estão nas páginas seguintes, as quais estão numeradas de 1 a 12.
3. Faz parte da prova a interpretação das questões. Caso você ache que falta algum detalhe nos enunciados ou nos esclarecimentos, você deverá fazer as suposições que achar necessárias e escrever essas suposições juntamente com as respostas.
4. **Todas** as respostas devem ser justificadas.
5. Somente serão corrigidas respostas legíveis.
6. Não se esqueça de escrever seu nome abaixo.

Desejamos a você uma boa prova!

A COPEQ

Atenção: Esta prova contém um total de 6 (seis) questões, das quais você deve fazer 4 (quatro). Marque abaixo as questões que devem ser consideradas para avaliação:

1 2 3 4 5 6 (selecione até quatro)

Nome: _____

Assinatura: _____

Questão 1

Prove que as propriedades abaixo sobre as notações assintóticas são verdadeiras ou falsas, usando para isso as suas definições e/ou contraexemplos.

a) ω é transitiva.

b) o é reflexiva.

c) O é simétrica.

d) Θ é simétrica.

Questão 2

Dado um vetor com n números inteiros deseja-se reorganizar os elementos de forma que todos os números negativos precedam os não negativos e os zeros fiquem entre eles. Desenvolva um algoritmo que reorganize esse vetor com ordem de complexidade $\Theta(n)$. Forneça um exemplo para ilustrar seu algoritmo.

Detalhes importantes:

1. A reorganização deve ser *in-place*, ou seja, não é permitido usar vetores, listas, etc, para armazenar os elementos temporariamente. Apenas algumas variáveis auxiliares inteiras podem ser utilizadas.

2. A reorganização deve ser feita por comparação de chaves - métodos de contagem não podem ser usados.

Questão 3

Sejam p_1 e p_2 duas permutações de um conjunto de n chaves $S = \{s_1, s_2, \ldots, s_n\}$ e T_1 e T_2 as duas árvores binárias de pesquisa geradas a partir de p_1 e p_2, respectivamente. Cada árvore é gerada a partir da inserção sequencial das chaves de sua permutação iniciando pela chave mais à esquerda.

Para cada afirmação abaixo, diga se é verdadeira ou falsa, provando ou fornecendo um contraexemplo:

a) $T_1 \neq T_2$ **se e somente se** $p_1 \neq p_2$.

b) As ordens de impressão das chaves de T_1 e T_2 obtidas por um caminhamento central (também chamado de caminhamento *in-order*) podem ser distintas.

c) É sempre possível saber em tempo logarítmico se um elemento $s_i \in S$ está na árvore T_1.

d) Seja h_1 a altura da árvore T_1. A complexidade de tempo para se encontrar a chave sucessora de uma chave s_i qualquer em T_1 é $O(h_1)$. (No caso, chave sucessora é a chave seguinte à s_i quando elas estão ordenadas crescentemente)

Questão 4

Você quer viajar de carro de s para t e já decidiu a rota: você vai seguir uma rodovia que passa pelas cidades $u_1, u_2, \ldots, u_n$. Você sabe a distância entre duas cidades vizinhas em sua rota, e sabe também que o tanque do seu carro permite que você viaje no máximo k quilômetros sem reabastecer. Seu objetivo é gastar o menor valor possível em combustível durante a viagem.

Como o preço do combustível varia consideravelmente de uma cidade para outra, você consultou online o preço do combustível na cidade u_i, definido por $p(u_i)$. Para simplificar, assuma que os preços são expressos em reais por quilômetro, ou seja, quanto custa para comprar combustível suficiente para rodar por um quilômetro.

Assuma que você sai de s com o tanque cheio. Dados os preços $p(u_i)$ pesquisados, planeje sua viagem de forma que você chegue a t gastando o mínimo possível em combustível.

a) Dado o exemplo ilustrado na tabela abaixo, e assumindo que um tanque cheio roda no máximo 20 quilômetros, encontre o preço mínimo possível de gastos com combustível para chegar de s a t.

Cidade	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9	t
Preço/$p(u_i)$	3	5	2	1	5	4	9	9	4	1
Distância	7	10	3	3	16	9	18	17	5	4

b) Descreva um algoritmo de programação dinâmica que resolva este problema. O algoritmo deve ser polinomial em n e k. (Dica: expresse o preço do combustível para chegar a t saindo de u_i, usando uma quantidade de combustível c, como uma função de custo que leva em conta o vértice u_j, onde $j > i$.)

Questão 5

Considere um grafo não direcionado $G = (V, E)$ com pesos não-negativos distintos para todas as arestas, i.e., $w_e \geq 0$ para toda $e \in E$. Assuma que você já tenha computados a árvore geradora mínima de G e o caminho mínimo para todos os nós a partir de um nó particular $s \in V$.

Suponha agora que todos os pesos das arestas de G sejam acrescidos de 1, com os novos pesos sendo $w'_e = w_e + 1$, gerando G'.

a) Para o novo grafo G', a árvore geradora mínima é modificada? Dê um exemplo mostrando que isso acontece ou prove que isso não ocorre.

b) Os caminhos mínimos já calculados são modificados em G'? Dê um exemplo mostrando que isso acontece ou prove os caminhos não são modificados.

Questão 6

Prove que o problema de se determinar se um grafo não direcionado $G = (V, E)$ tem um clique de tamanho k, $k \leq |V|$, é NP-Completo.

(Dica: uma possível redução pode ser feita a partir do problema 3-CNF-SAT descrito abaixo).

O problema da satisfazibilidade de fórmulas booleanas em forma normal 3-conjuntiva (3-CNF-SAT) é um caso especial do problema da satisfazibilidade (SAT) no qual as fórmulas booleanas estão na forma CNF. Mais formalmente, um literal em uma fórmula booleana é a ocorrência de uma variável ou sua negação. Uma fórmula booleana está em forma normal conjuntiva (CNF) se é expressa como uma conjunção (AND) de cláusulas, cada uma das quais é uma disunção (OR) de um ou mais literais. Uma fórmula booleana está em forma normal 3-conjuntiva (3-CNF) se cada cláusula tem exatamente 3 literais distintos, como o exemplo a seguir:

$$(x_1 \vee \neg x_3 \vee \neg x_2) \wedge (x_3 \vee x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_3 \vee \neg x_4)$$

O 3-CNF-SAT pergunta se uma determinada fórmula nesse formato é satisfazível. Sabe-se que esse problema é NP-Completo.