Universidade Federal de Minas Gerais
Programa de Pós-Graduação em Ciência da Computação
Exame de Qualificação 1º Estágio
2º Semestre de 2013

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

Em 13/08/2013, 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

Uma possível melhoria na implementação de um hash com tratamento de colisões por encadeamento (*chaining*) é usar árvores binárias de pesquisa ao invés de listas encadeadas para armazenar os elementos.

a) Mostre qual seria a configuração dessa estrutura após a inserção das chaves: 15, 4, 22, 11, 10, 1, 25, 18, 13. Considere um hash de tamanho $M = 7$ e função hash dada por $f(k) = k\%M$ (resto da divisão inteira de k por M).

b) Considere que as n chaves são distribuídas uniformemente, mas que você não sabe a ordem em que foram inseridas em um hash de tamanho M. Discuta qual seria a complexidade de uma pesquisa com sucesso no melhor caso, pior caso e caso médio.

c) Ainda considerando que as n chaves são distribuídas uniformemente, descreva em linhas gerais um algoritmo $O(nM)$ que imprima as chaves desse hash em ordem crescente. Explique a complexidade do seu algoritmo.

Questão 2

Seja $A[1:n]$ um vetor com n números distintos. O par (i,j) é chamado uma inversão de A se $i < j$ e $A[i] > A[j]$.

a) Que vetor com elementos do conjunto $\{1, 2, \ldots, n\}$ tem o maior número de inversões? Quantas inversões existem?

b) Qual é a relação entre o tempo de execução do algoritmo de Ordenação por Inserção e o número de inversões do vetor de entrada? Justifique sua resposta.

c) Apresente um **algoritmo recursivo** que determine o número de inversões em qualquer permutação de n elementos em tempo $\Theta(nlogn)$.

d) Para demonstrar que o seu algoritmo é $\Theta(nlogn)$, determine e resolva a sua equação de recorrência.

Questão 3

Considere o problema de fornecer um troco de n centavos usando o menor número de moedas. Suponha que o valor de cada moeda seja um inteiro.

a) Descreva um algoritmo guloso para fornecer o troco usando moedas com valores 25, 10, 5 e 1 centavo. O seu algoritmo fornece a solução ótima? Discuta.

b) Dê um conjunto de valores de moedas para o qual o algoritmo guloso não produz uma solução ótima. Seu conjunto deve incluir 1 centavo, de modo que exista uma solução para todo valor de n.

c) Implemente um algoritmo de programação dinâmica de tempo $O(nk)$ que forneça o troco para qualquer conjunto de k valores diferentes de moeda, considerando que uma das moedas é de 1 centavo.

Questão 4

Você deseja traçar um caminho entre dois vértices s e t em um grafo ponderado G onde todos os pesos são não-negativos, mas gostaria de parar em um vértice u se não for inconveniente. Parar em u se torna inconveniente quando o caminho $s-t$ aumentar em mais de 10%. Descreva um algoritmo eficiente que determina o caminho ótimo de s a t dada sua preferência de parar em u se não for inconveniente. O algoritmo deve retornar o caminho mínimo de s a t, ou o caminho mínimo de s a t contendo u. Analise a complexidade do algoritmo proposto.

Questão 5

Responda se as questões abaixo são verdadeiras ou falsas, justificando as suas respostas.

a) Uma solução simples para encontrar uma árvore geradora máxima de um grafo G é utilizar um algoritmo de árvore geradora mínima em um grafo G', onde G' foi gerado multiplicando todos os pesos das arestas de G por -1.

b) A profundidade da árvore da busca em largura construída a partir de um vértice arbitrário v em um grafo não direcionado $G = (V, E)$ é igual ao diâmetro do grafo G.

c) Considere um grafo $G = (V, E, w)$ e X o caminho $s - t$ mínimo em G, onde $s, t, \in V$. Se dobrarmos o peso de cada aresta do grafo, fazendo com que $w'(e) = 2w(e)$ para cada $e \in E$, X continuará sendo o menor caminho s-t em $G' = (V, E, w')$.

d) Dado um grafo $G = (V, E)$ onde todas as arestas tem peso positivo. Os algoritmos de Bellman-Ford e Dijkstra podem gerar caminhos mínimos diferentes mas de mesmo peso.

Questão 6

Um conjunto $A \subseteq V$ de um grafo $G = (V, E)$ é um *conjunto independente* de tamanho k se e somente se $|A| = k$ e para cada dois vértices u e v em A, $(u, v) \notin E$. O problema do conjunto independente IS_k é definido como: *Dado um grafo $G = (V, E)$ e um inteiro positivo k, determine se G tem um conjunto independente de tamanho no mínimo k.*

O problema das n rainhas - nQ - é definido como: *É possível colocar n rainhas em um tabuleiro de xadrez de n linhas por n colunas garantindo que nenhuma rainha possa atacar outra, sabendo que $n \geq 4$?* Note que uma rainha na linha s e coluna t pode atacar todas as demais posições na linha s, na coluna t e nas duas diagonais contendo a posição (s,t).

a) Mostre que o problema nQ é polinomialmente transformável no problema IS_k, ou seja, $nQ \propto IS_k$. Note que para tal você terá que mostrar que existe solução para o problema das n rainhas se e somente se existe solução para o conjunto independente na instância criada.

b) Caso seja encontrada uma solução polinomial para o problema das n rainhas, isto implica que o problema do conjunto independente tem solução polinomial? Justifique sua resposta.