Universidade Federal do Ceará
Centro de Ciências
Departamento de Computação
Bacharelado em Ciência da Computação

Construção e Análise de Algoritmos

Lista de exercícios 3

Algoritmos Gulosos

1. Escreva um algoritmo que obtenha um código de Huffman ternário, ou seja, um código de Huffman em que podemos codificar os símbolos com 0, 1 e 2. Escreva agora um algoritmo que obtenha um código de Huffman ao-contrário, ou seja, obtenha o pior código de prefixo possível. Exemplifique esses dois algoritmos e também o código de Huffman normal para o seguinte exemplo: um texto com 100.000 caracteres onde os símbolos são A, B, C, D, E, F e G com frequências 40%, 15%, 11%, 10%, 14%, 7% e 5%. Quantos bits esse texto codificado terá nesses algoritmos? Quantos bits esse texto teria se usássemos uma codificação de tamanho fixo? Compare cada algoritmo: melhorou ou piorou?

2. Considere um conjunto de livros numerados de 1 a n. Suponha que o livro i tem peso p_i e que $0 < p_i < 1$ para cada i. Problema: Dado n e os números $p_1, \ldots, p_n$, acondicionar os livros no menor número possível de envelopes de modo que cada envelope tenha no máximo 2 livros e o peso do conteúdo de cada envelope seja no máximo 1. Escreva um algoritmo eficiente que resolva esse problema. Aplique seu algoritmo a um exemplo interessante. Mostre que seu algoritmo está correto.

3. Escreva um algoritmo eficiente que receba como entrada um conjunto de variáveis $x_1, \ldots, x_n$ e dois conjuntos I e D de pares (x_i, x_j) de variáveis. Os pares (x_i, x_j) em I representam restrições de igualdade $x_i = x_j$ e os pares (x_a, x_b) em D representam restrições de desigualdade $x_a \neq x_b$. Seu algoritmo deve responder se é possível ou não satisfazer todas as restrições em I e em D. Por exemplo, a seguinte entrada não é satisfatível: $I = \{(x_1, x_2), (x_2, x_3), (x_3, x_4)\}$ e $D = \{(x_1, x_4)\}$.

4. Um certo Fulano deseja fazer uma festa e está decidindo quem deve chamar. Ele tem n amigos para convidar e tem uma lista dos pares de amigos que se conhecem. Ele quer que ninguém se sinta deslocado, mas também quer que a festa seja interessante e que pessoas que não se conhecem façam amizade. Assim ele se colocou as seguintes restrições: Para cada convidado, devem existir pelo menos dez pessoas na festa que ele conhece e dez pessoas na festa que ele não conhece. Faça um algoritmo que receba uma lista com n pessoas e uma lista com os pares dessas pessoas que se conhecem e devolva o maior número pessoas que poderão ser convidadas sob estas restrições. (a) Descreva com palavras como será o seu algoritmo. (b) Escreva o algoritmo em pseudo-código.

5. Nesta questão, nós construiremos um novo algoritmo para encontrar a árvore de custo mínimo. Ele é baseado na seguinte propriedade: *Pegue um ciclo do grafo e seja (x, y) a aresta de maior custo nesse ciclo. Então existe uma árvore de custo mínimo que não contém (x, y).*

(a) Prove essa propriedade cuidadosamente

(b) Prove que o seguinte algoritmo está correto: *Ordene as arestas de modo decrescente de acordo com os custos das arestas. Para cada aresta (x, y) nessa ordem: se (x, y) pertence a um ciclo do grafo, remova (x, y) do grafo. Retorne o grafo modificado.*

(c) Em cada iteração, o algoritmo acima deve checar se existe um ciclo contendo uma certa aresta (x, y). Escreva um algoritmo $O(m)$ para essa tarefa e justifique sua correção, onde m é o número de arestas.

(d) Calcule a complexidade de tempo do seu algoritmo em termos de m.

6. Seja $1, \ldots, n$ um conjunto de tarefas. Cada tarefa consome um dia de trabalho; durante um dia de trabalho somente uma das tarefas pode ser executada. Os dias de trabalho são numerados de 1 a n. A cada tarefa T está associado um prazo P_T: a tarefa deveria ser executada em algum dia do intervalo $1, \ldots, P_T$. A cada tarefa T está associada uma multa não-negativa M_T. Se uma dada tarefa T é executada depois do prazo P_T, sou obrigado a pagar a multa M_T (mas a multa não depende do número de dias de atraso). Problema: Programar as tarefas (ou seja, estabelecer uma bijeção entre as tarefas e os dias de trabalho) de modo a minimizar a multa total. Escreva um algoritmo guloso para resolver o problema. Prove que seu algoritmo está correto. Analise o consumo de tempo.