MAC 5711 - Análise de Algoritmos

Departamento de Ciência da Computação
Primeiro semestre de 2004

Lista 6

Devolução dos exercícios 3 e 4: 25 de maio.

1. Suponha dado 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: 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 guloso que resolva o problema em tempo $O(n \log n)$. (Sugestão: comece por escrever um algoritmo recursivo que apenas calcula o número mínimo de envelopes.) Aplique seu algoritmo a um exemplo interessante. Mostre que seu algoritmo está correto.

2. O problema da alocação de atividades pode ser reformulado como o problema de encontrar, numa coleção de intervalos dada, uma subcoleção de tamanho máximo de intervalos dois a dois disjuntos.

   (CLR 17.1-3, CLRS 16.1-4) Nem todo algoritmo guloso resolve o problema da coleção máxima de intervalos disjuntos (conjunto máximo de atividades mutualmente compatíveis). Mostre que nenhuma das três idéias a seguir resolve o problema. Idéia 1: Escolha a atividade de menor duração dentre as que são compatíveis com as atividades já escolhidas. Idéia 2: Escolha uma atividade que seja compatível com as já escolhidas e intercepta o menor número possível de atividades ainda não escolhidas. Idéia 3: Escolha a atividade compatível com as já selecionadas que tenha o menor instante de início.

3. **(parte de CLR 17-3, CLRS 16-4)** 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 $1\,..\,p_t$. 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.

4. A entrada é uma seqüência de números $x_1, x_2, \ldots, x_n$ onde n é par. Projete que particione a entrada em $n/2$ pares da seguinte maneira. Para cada par, computamos a soma de seus números. Denote por $s_1, s_2, \ldots, s_{n/2}$ as $n/2$ somas. O algoritmo deve encontrar uma partição que minimize a máximo das somas e deve ser tão eficiente quanto possível. Explique porque ele funciona e determine a sua complexidade.

5. Use códigos de Huffman para o conjunto de caracteres do enunciado deste exercício. Inclua todos os caracteres. Quantos bits foram economizados no armazenamento do enunciado desse exercício usando códigos de Huffman versus uma codificação onde todos os caracteres são codificados por cadeias de bits do mesmo comprimento?

6. Suponha que um texto foi codificado por meio de códigos de Huffman. A árvore correspondente ao código de Huffman usado foi construída e está disponível para você. As freqüências dos caracteres do texto também estão disponíveis. Suponha agora que o texto foi modificado levemente de maneira que a freqüência de um caracter do texto original x aumentou de 1. Você foi encarregado de atualizar a árvore de maneira que ela seja ótima para o texto modificado. Um amigo seu faz a seguinte sugestão para um algoritmo para modificar a árvore.

   Primeiro, ele observa que uma propriedade importante de uma árvore de Huffman é que as freqüências associadas aos nós estão em ordem não-decrescente conforme os nós estão mais próximos à raiz da árvore. (Em outras palavras, um nó não pode estar num nível acima na árvore que um nó com freqüência maior que a dele.) A freqüência de um nó interno v é a soma de todas as freqüências dos caracteres que aparecem em folhas que sejam descendentes de v na árvore. Conseqüentemente, ele sugere verificar se o nó cuja freqüência foi alterada (a folha que corresponde a x) ainda satisfaz a propriedade acima, comparando a sua freqüência com a freqüência de todos os nós do nível acima do dele. Se não há nenhum nó com freqüência maior que a dele no nível acima, deixe tal nó em seu lugar. Caso contrário, troque x com o caracter numa folha no nível acima que tenha freqüência maior que a freqüência nova de x. Esse algoritmo pode funcionar algumas vezes, mas está em geral errado. Descreva porque ele está errado e como ele pode ser corrigido. Você deve mencionar não apenas o que está faltando no algoritmo mas, mais importante, discutir porque o algoritmo não funciona, como está, em geral. Em outras palavras, ou mostre um contra-exemplo no qual o algoritmo não constrói uma árvore ótima, ou mostre que, caso o algoritmo estivesse certo, teríamos uma contradição (ou alguma implicação maior que seja suspeita) Não é suficiente indicar que o algoritmo não lida com alguns casos. Poderia acontecer que tais casos pudessem ser ignorados. Você precisa mostrar que o algoritmo está definitivamente errado.