MAC 5711 - Análise de Algoritmos

Departamento de Ciência da Computação

Segundo semestre de 2005

Lista 6

1. Uma seqüência de n operações é executada em uma estrutura de dados. A i-ésima operação custa i se i é uma potência de 2, e 1 caso contrário. Determine o tempo amortizado por operação.

2. Uma seqüência de operações sobre uma pilha é executada numa pilha cujo tamanho nunca excede k. Depois de cada k operações, uma cópia da pilha toda é feita para propósito de *back-up*. Mostre que o custo de n operações sobre a pilha, incluindo a operação de cópia para *back-up*, é $O(n)$, atribuindo valores adequados de créditos a cada operação.

3. Suponha que desejamos não apenas incrementar um contador mas também algumas vezes reinicializá-lo com zero. Mostre como implementar um contador com um vetor binário de maneira que qualquer seqüência de n operações `incrementa1` e `zera_contador` consuma tempo $O(n)$, desde que o contador esteja inicialmente com zero. (**Dica:** Mantenha um apontador para o 1 mais significativo do contador.)

4. Considere a implementação de lista ligada para representar conjuntos disjuntos. Sugira uma mudança simples da rotina UNION que não necessite do apontador `fim` para o último da lista de cada conjunto. Sua sugestão deve ser tal que, independente de estarmos ou não usando a heurística dos tamanhos (anexe no final a lista menor), o consumo assintótico de tempo de pior caso deve se manter igual.

5. Mostre que $\lg(\lg^* n) = O(\lg^*(\lg n))$.

6. Considere a implementação do union-find por árvores enraizadas. Escreva uma versão não recursiva do FINDSET com compressão de caminhos.

7. Considere a implementação do union-find por árvores enraizadas com compressão de caminhos e heurística dos ranks (a árvore de menor rank é pendurada na de menor rank no union). Considere uma seqüência qualquer (válida) de m operações MAKESET, FINDSET e LINK em que todas as operações LINK aparecem antes das operações FINDSET. Mostre que tal seqüência consome, no pior caso, tempo $O(m)$. O que acontece com o tempo consumido por uma seqüência deste tipo se apenas compressão de caminhos estiver implementada?

8. Calcule a função prefixo π para o padrão *ababbabbababbababbabb* quando o alfabeto é $\Sigma = \{a, b\}$.

9. Mostre que, se todos os caracteres do padrão $P[1\,..\,m]$ são distintos, o algoritmo ingênuo que busca P em um texto $T[1\,..\,n]$ pode ser modificado para consumir tempo $O(n)$.

10. Suponha que o padrão P e o texto T são cadeias de caracteres de comprimentos m e n respectivamente, escolhidas aleatoriamente de um alfabeto $\Sigma_d = \{0, 1, \ldots, d-1\}$, onde $d \geq 2$. Mostre que o número esperado de comparações caractere a caractere feita pelo laço implícito na comparação dentro do laço principal do algoritmo ingênuo é

$$(n - m + 1)\frac{1 - d^{-m}}{1 - d^{-1}} \;\leq\; 2(n - m + 1).$$

    (Suponha que o algoritmo ingênuo pára as comparações assim que encontra um caractere do padrão que não casa com o correspondente do texto, ou quando encontra uma ocorrência do padrão.) Assim sendo, para cadeias de caracteres aleatórias, o algoritmo ingênuo é bastante eficiente.

11. Suponha que o padrão P pode conter ocorrências de um caracter *vão* $\diamondsuit$, que pode ser casado a uma cadeia arbitrária de caracteres (inclusive com a cadeia vazia). Por exemplo, o padrão $ab\diamondsuit ba\diamondsuit c$ ocorre no texto *cabccbacbacab* de duas maneiras diferentes: $c\mathbf{ab}cc\mathbf{ba}cba\mathbf{c}ab$ e $c\mathbf{ab}ccbac\mathbf{ba}\mathbf{c}ab$. Note que o caracter vão pode ocorrer um número arbitrário de vezes no padrão, mas não ocorre no texto nenhuma vez. Descreva um algoritmo polinomial para determinar se um tal padrão ocorre em um texto T, e analise o consumo de tempo do seu algoritmo.

12. Mostre como determinar as ocorrências de um padrão $P[1\,..\,m]$ em um texto $T[1\,..\,n]$ examinando a função prefixo para a cadeia de caracteres PT (concatenação de P com T).

13. Descreva um algoritmo linear para determinar se um texto T é uma rotação cíclica de uma outra cadeia de caracteres T'. Por exemplo, *arco* e *coar* são rotações uma da outra.