Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Projeto e Análise de Algoritmos (2024-1)

Segunda Avaliação

Prof. Flávio L. C. de Moura

11 de setembro de 2024

Exercício 1. *(2.5 pontos) Mostre como podemos ordenar* n *inteiros contidos no intervalo de 0 a* $n^3 - 1$ *em tempo linear, ou seja, em tempo* $O(n)$.

> *Solução*
>
> *Inicialmente iteramos pela lista dos números, convertendo cada um deles para a base* n. *Depois aplicamos o algoritmo radix-sort sobre a lista, utilizando o counting-sort como o algoritmo de ordenação dos dígitos da lista de números. Dada a nova base, cada número possuirá 3 dígitos, uma vez que* $\log_n(n^3) = 3$, *onde cada dígito estará em um intervalo entre 0 e* $n-1$. *Sabendo disso, vemos que radix-sort ordenará* n *números de 3 dígitos, usando o counting-sort em um intervalo de 0 à* $n-1$, *resultando em uma complexidade de* $O(3(n+n)) = O(n)$.

Exercício 2. *(2.5 pontos) Considere o algoritmo a seguir, onde* $G = (V, E)$ *é um grafo,* $u \in V$ *é um vértice de* G *e* F *uma fila inicialmente vazia:*

```
marque u;
insira u na fila F;
while F ≠ ∅ do
    retire o primeiro elemento v de F;
    for each w ∈ G.Adj[v] do
        if w não está marcado then
            marque w;
            insira w em F;
        end
    end
end
```

Algorithm 1: algoritmo(G, u)

Faça a análise assintótica do tempo de execução deste algoritmo.

> *Solução*
>
> *Este algoritmo visita todos os vértices do grafo* G *que são alcançáveis a partir do vértice* u *utilizando busca em largura. As operações de inserir e retirar um elemento da fila têm tempo constante, de forma que o tempo total destinado a estas operações é* $O(V)$ *no pior caso (*i.e. *quando* G *é conexo). Como a lista de adjacências de um vértice só é visitada quando este vértice é retirado da fila, cada lista de adjacências é visitada somente uma vez. Como a soma do comprimento de todas as listas de adjacências é* $\Theta(E)$, *então concluímos que este algoritmo está em* $O(V + E)$, *ou seja, é linear no tamanho da representação da lista de adjacências de* G.

Exercício 3. *Mostre que 3-SAT* $\leq_p$ *CLIQUE, isto é, mostre que 3-SAT é redutível polinomialmente a CLIQUE considerando a função que transforma instâncias de 3-SAT em instâncias de CLIQUE da seguinte forma:*

Seja φ *uma fórmula contendo* k *cláusulas da seguinte forma:*

$$\varphi = (a_1 \vee b_1 \vee c_1) \wedge (a_2 \vee b_2 \vee c_2) \wedge \ldots \wedge (a_k \vee b_k \vee c_k)$$

A função de redução recebe φ como argumento e constrói o par $\langle G, k \rangle$, onde G é um grafo com a seguinte estrutura: os vértice de G são organizados em k grupos de três vértices cada, digamos $t_1, t_2, \ldots, t_k$. Cada tripla t_i $(1 \leq i \leq k)$ corresponde a uma cláusula de φ, e cada vértice da tripla corresponde a um literal da cláusula associada. Marcamos cada vértice com o nome do literal correspondente. Cada aresta de G conecta todos os vértices de G, excetuando-se os seguintes casos:

- *vértices que pertencem à mesma tripla;*
- *vértices cujos nomes são contraditórios, como x e $\overline{x}$*

A prova de que a função acima é, de fato, uma redução polinomial é dividida em duas partes:

1. *(2.5 pontos) Mostre que se φ é satisfatível então o grafo G possui um k-clique.*

   Solução

   Suponha que φ seja satisfatível, isto é, existe um designação de variáveis que faz com que todas as k cláusulas sejam verdadeiras. Em cada tripla de G, selecione o vértice correspondente ao literal verdadeiro da cláusula. Se mais de um literal for verdadeiro, então pode-se escolher qualquer um deles. Os vértices de G assim selecionados formam um k-clique. De fato, o número de vértices selecionados é igual a k (um em cada tripla), e cada par de vértices selecionados está ligado por uma aresta já que eles não pertencem à mesma tripla e não possuem nomes contraditórios já que ambos são verdadeiros na designação dada. Logo G possui um k-clique.

2. *(2.5 pontos) Mostre que se G possui um k-clique então φ é satisfatível.*

   Solução

   Suponha que G possui um k-clique. Como cada vértice do clique pertence a uma tripla diferente, construiremos uma designação de variáveis fazendo com que o literal correspondente a cada vértice do clique seja verdadeiro. Observe que é possível construir tal designação porque dois literais contraditórios não correspondem a vértices distintos do clique. Por fim, esta designação satisfaz φ porque cada tripla contém um vértice do clique, e portanto cada cláusula de φ contém um literal verdadeiro.