EP3 - Ordenando Panquecas

Raphael Ribeiro NUSP-10281601

20 de outubro de 2018

Código

Para as questões a seguir, considere o seguinte algoritmo em discussão

```
void ordenaPanquecas(long int * pan, int tam) {
    /*Ordena o vetor pan[0,...,tam-1], em ordem decrescente,
    utilizando a única operação flip */
    int max;
    int i = 0;

    while (i<tam-1) {
        max = buscaMax(pan, i, tam-1);
        if (i != max) {
            flip(pan,max,tam-1);
            flip(pan,i,tam-1);
        }
        i++;
    }
}
```

Questões

Como funciona seu algoritmo? Qual a ideia geral?

A ídeia básica para ordenar o vetor em ordem decrescente é realizar um flip na posição do maior elemento, movendo-o para o fim do vetor, e depois realizando um flip na posição inicial de forma que o maior elemento seja movido para sua posição final. A seguir, considera-se o vetor[1,...,tam-1] e realiza-se novamente estes passos. Além disso, flips em vetores unitários são desconsiderados. A comparação na linha 8 evita que dois flips aconteçam na mesma posição, uma vez que isso resulta em flips desnecessários. Isto ocorre quando a função buscaMax retorna o índice de um elemento que já está na sua posição final.

Qual sua estimativa de complexidade de tempo de seu algoritmo para ordenar n panquecas?

Sabemos que a complexidade da função buscaMax é $\mathcal{O}(n)$, assim como a complexidade da função flip. Sabemos também que a função buscaMax é executada a cada iteração. Logo, existem n-2 chamadas da função buscaMax, cada uma com complexidade $n - i$, com $i \in [0, n-2]$ Segue portanto, que a complexidade do algoritmo é $\mathcal{O}(n^2)$

Qual o número mínimo e máximo de flips seu algoritmo faz para ordenar uma sequência?

O número de flips é determinado pelo número de vezes que a comparação interna ao laço resulta em verdadeiro. Se max retornar i, nenhum flip é executado. Portanto, o número mínimo de flips ocorre quando max sempre retorna o valor de i, isto ocorre quando o vetor já está ordenado em ordem decrescente. Neste caso, o número de flips é 0.

O número máximo de flips ocorre quando a comparação interna ao laço sempre resulta em verdadeiro. Além disso, como flips em uma vetores unitários não são executados, é necessário que buscaMax retorne um valor diferente de n-1 (ultima posição do vetor). Na última iteração, o vetor tem tamanho dois e buscaMax retorna n-1 (*) ou n-2 (**). No caso (*), apenas um flip é executado. Já no caso (**) nenhum flip é executado pois temos i = max. Sendo assim, o número máximo de flips ocorre quando toda iteração, com exceção da última, resulta em dois flips, e a última resulta em um flip. O número máximo de flips portanto é $2(n-2)+1 = 2n-3$ flips

Mostre sequências em que seu algoritmo realiza o número máximo e mínimo de flips

-> O número mínimo de flips ocorre no caso trivial. Exemplos de sequência são:

1. 13 12 9 8 7 (Número de flips = 0)
2. 7 5 2 1 (Número de flips = 0)
3. 8 8 8 8 (Número de flips = 0)

-> Exemplos de sequências que o algoritmo realiza o número máximo de flips são:

n = 4 (número de flips = 5)

1. 10 17 13 14
2. 13 55 85 53

n = 5 (número de flips = 7)

1. 53 62 84 34 54
2. 45 60 18 53 39

n = 6 (número de flips = 9)

1. 3 2 4 1 3 0

n = 30 (número de flips = 57)

1. 54 44 57 605 197 831 278 223 61 160 831 486 274 692 426 577 401 85 551 883 978 618 625 142 349 474 552 116 934 546

Mostre sequências em que seu algoritmo não realiza o número mínimo possível de flips para ordenar uma sequência.

1. 54 44 57 605 197 831 278 223 61 160 831 486 274 692 426 577 401 85 551 883 978 618 625 142 349 474 552 116 934 546
2. 1. 3 2 4 1 3 0

Você consegue alguma estimativa para a qualidade do seu algoritmo, ou seja, por exemplo, o número de flips dado por seu algoritmo é limitado por uma constante vezes o número ótimo de flips? Ou, o número de flips do seu algoritmo é limitado por n (número de panquecas) vezes o ótimo?

Vamos calcular o numero esperado de flips $E[f]$. Vamos supor que a função buscaMax tem chance igual de retornar um número entre i e n-1.

Na primeira iteração, temos chance de $\frac{1}{n}$ de buscaMax retornar i, nesse caso, nenhum flip é executado. Além disso, existe $\frac{1}{n}$ de chance de buscaMax retornar o ultimo elemento. Nesse caso, o número de flips é 1. Existem n-2 outros possíveis valores que buscaMax pode retornar. Para esses valores, o número de flips é 2.

Sendo assim, na primeira iteração, o número de flips esperado é:

$$E[f_0] = \frac{1}{n}0 + (n-2)\frac{1}{n}2 + \frac{1}{n}1 = (n-2)\frac{2}{n} + \frac{1}{n}$$

Numera iteração i qualquer, temos:

$$E[f_i] = \frac{1}{n-i}0 + (n-2)\frac{1}{n-i}2 + \frac{1}{n-i}1 = (n-2)\frac{2}{n-i} + \frac{1}{n-i}$$

Logo, o número de flips total esperado é:

$$E[f] = \sum_{i=0}^{n-2}(n-2)\frac{2}{n-i} + \frac{1}{n-i}$$

Imagine uma versão do problema em que as panquecas tenham um lado mais queimado que deva ficar voltado para baixo. Seu algoritmo funciona este caso?

Não. Como flips em apenas uma panqueca não são executados pelo programa, quando buscaMax retorna o ultimo elemento, o primeiro flip não é executado e o segundo flip inverte o lado da panqueca. Sendo assim, supondo que o lado mais queimado das panquecas estejam voltados para baixo no ínicio, uma sequência que terminaria com uma panqueca com o lado mais queimado voltado para cima seria:

7 6 5 4 9

Vamos denotar por d_c e d_b a orientação arbitrária da panqueca, onde d é o diâmetro. No ínicio temos:

$$7_c6_c5_c4_c9_c$$

O algoritmo retorna a seguinte sequencias de flips para ordenar a pilha de panquecas: 0 1 Ao realizar o flip na posição 0, temos:

$$9_b4_b5_b6_b7_b$$

Finalmente, ao realizar o flip na posição 1, temos:

$$9_b7_c6_c5_c4_c$$

Como podemos observar, a panqueca de diâmetro 9 encerrou com orientação diferente da inicial.