Algoritmo para busca binaria em uma matriz de inteiros ordenada

Gustavo Cipriano Mota Sousa

30 de Março de 2005

Ao longo da explicação do algoritmo, os nomes escritos entre parenteses representam variaveis no codigo encontrado logo abaixo. Este é o código da função bsearch_helper(), localizada no arquivo matriz.c encontrado neste mesmo diretório, e é o responsável pela busca do elemento na matriz.

A ideia por trás do algoritmo e encontrar o ponto na diagonal principal no qual o valor anterior na diagonal seja menor que (value) e o proximo valor na diagonal seja maior que (value). Dessa forma, a partir deste ponto os valores a equerda e acima do valor anterior sao menores que (value), e os valores a direita e abaixo do proximo valor sao maiores que (value). Logo sobram dois quadrantes para localizar a busca.

1. O primeiro passo do algoritmo é determinar o tamanho da menor dimensão. e armazená-lo em (size), isso e feito para que a diagonal utilizada durante o algoritmo nao passe os limites da matriz.

2. Logo após, testamos se a matriz possui ao menos um elemento, caso contrario, a função retorna 0

3. Para determinar o ponto que procuramos na diagonal principal, fazemos uma especie de busca binária pela diagonal principal, começamos com um apontador para o ponto inicial (left), e para o final (right) e então calculamos o ponto central (middle). Logo apos, testamos (value) com o elemento do meio, caso sejam iguais, encontramos o elemento, caso (value) seja menor que o elemento mediano, sabemos que (value) não se encontra a direita nem abaixo do elemento mediano, e podemos passar o elemento final (right) para o ponto imediatamente anterior ao elemento do meio na diagonal. O contrário é feito se (value) for maior que o elemento do meio.

4. Após o passo anterior podemos delimitar os quadrantes que ainda precisam ser analisados, e para realizar essa operacao chamamos a função recursivamente.

```
/*
 * value eh o valor a ser buscado
 * sx, sy representam a posicao da matriz a ser analisada, dentro de (matrix)
 * w, h sao as dimensoes da matriz a ser buscada
*/
static int bsearch_helper(int **matrix, int value,
        unsigned int sx, unsigned int sy,
        size_t w, size_t h, point *p) {
    unsigned int left, right, middle;

    size_t size = (w < h ? w : h);

    if (size == 0) {
        return 0;
    }

    left = 0;
    right = size - 1;


    while (left < size && right < size && left <= right) {
        middle = (left + right) / 2;

        if (value == matrix[sx + middle][sy + middle]) {
            if (p != NULL) {
                p->x = sx + middle;
                p->y = sy + middle;
            }
            return 1;
        }

        if (value < matrix[sx + middle][sy + middle]) {
            right = middle - 1;
            middle = right;
        }

        if (value > matrix[sx + middle][sy + middle]) {
```

```
            left = middle + 1;
            middle = left;
        }
    }

    return bsearch_helper(matrix, value, sx + left, sy, w - left, left, p)
        || bsearch_helper(matrix, value, sx, sy + left, left, h - left, p);
}
```