有序数组的区间查询

Dezeming Family

2022 年 5 月 16 日

DezemingFamily 系列书和小册子因为是电子书，所以可以很方便地进行修改和重新发布。如果您获得了 DezemingFamily 的系列书，可以从我们的网站 [https://dezeming.top/] 找到最新版。对书的内容建议和出现的错误欢迎在网站留言。

目录

一　算法描述 1

我们有一个从小到大的数组：

```
float data[12] = { 0.0f, 0.1f, 0.15f, 0.21f, 0.35f, 0.44f, 0.58f, 0.71f,
    0.73f, 0.88f, 0.93f, 1.0f };
```

现在给定一个数，我们想计算出它在其中的哪个区间，比如给定 0.26，那么 0.26 大于 0.21 且小于 0.35，因此在索引 3 到索引 4 之间，我们需要靠下的边界，也就是索引为 3。

先写一下我实现的算法：

```
inline int binaryFindsIntervals(const float m, const float arr[], const int
    low, const int high) {
    int mid;
    int l = low, h = high;
    while (l < h-1) {
        mid = (l + h) / 2;
        if (m < arr[mid])
            h = mid;
        else if (m >= arr[mid])
            l = mid;
    }
    return l;
}
```

其中，m 是要搜寻的数据，arr 是数组，low 表示搜寻范围的最小索引，high 表示搜寻范围的最大索引。

测试：

```
    int idx1 = Feimos::binaryFindsIntervals(0.26, data, 0, 11);
    int idx2 = Feimos::binaryFindsIntervals(0.36, data, 0, 11);
```

输出分别是 3 和 4，说明是正确的。其实就是左右互相逼近，直到不能再逼近未知，此时 l 和 h 紧挨着，而 m 恰好落在这个区间内。我们返回这个区间中的低索引，即 l 。

二　算法描述 2

算法 2 来自 PBRT[1]，我进行了一些修改，使得参数列表与我实现的相同：

```
template <typename T, typename U, typename V>
inline T Clamp(T val, U low, V high) {
    if (val < low)
        return low;
    else if (val > high)
        return high;
    else
        return val;
}
inline int bisectFindsIntervals(const float m, const float arr[], const int
    low, const int high) {
    int size = high - low + 1;
    int first = low, len = size;
```

```
    while (len > 0) {
        int half = len >> 1, middle = first + half;
        // Bisect range based on value of _pred_ at _middle_
        if (arr[middle] < m) {
            first = middle + 1;
            len -= half + 1;
        }
        else
            len = half;
    }
    return Clamp(first - 1, low, high);
}
```

区别在于最后返回的是 first-1 的值，这是因为当 m 大于中间值时，first 赋值为 middle+1。

测试代码同上：

```
    int idx1 = Feimos::bisectFindsIntervals(0.26, data, 0, 11);
    int idx2 = Feimos::bisectFindsIntervals(0.36, data, 0, 11);
```

这个实现理论比我实现的要快，因为不会在边缘处重复，但最后还得需要 Clamp 一下。

参考文献

[1] Pharr M, Jakob W, Humphreys G. Physically based rendering: From theory to implementation[M]. Morgan Kaufmann, 2016.